

CNN inference acceleration on limited resources FPGA platforms_epilepsy detection case study

Afef Saidi, Slim Ben Othman, Meriam Dhouibi, Slim Ben Saoud

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

Article Info

Article history:

Received Aug 15, 2022

Revised Nov 28, 2022

Accepted Dec 30, 2022

Keywords:

Acceleration

Classification

Convolutional neural network

Epilepsy

Field programmable gate array

Xilinx

ABSTRACT

The use of a convolutional neural network (CNN) to analyze and classify electroencephalogram (EEG) signals has recently attracted the interest of researchers to identify epileptic seizures. This success has come with an enormous increase in the computational complexity and memory requirements of CNNs. For the sake of boosting the performance of CNN inference, several hardware accelerators have been proposed. The high performance and flexibility of the field programmable gate array (FPGA) make it an efficient accelerator for CNNs. Nevertheless, for resource-limited platforms, the deployment of CNN models poses significant challenges. For an ease of CNN implementation on such platforms, several tools and frameworks have been made available by the research community along with different optimization techniques. In this paper, we proposed an FPGA implementation for an automatic seizure detection approach using two CNN models, namely VGG-16 and ResNet-50. To reduce the model size and computation cost, we exploited two optimization approaches: pruning and quantization. Furthermore, we presented the results and discussed the advantages and limitations of two implementation alternatives for the inference acceleration of quantized CNNs on Zynq-7000: an advanced RISC machine (ARM) software implementation-based ARM, NN, software development kit (SDK) and a software/hardware implementation-based deep learning processor unit (DPU) accelerator and DNNDK toolkit.

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



Corresponding Author:

Afef Saidi

Laboratory of Advanced Systems, Tunisia Polytechnic School, University of Carthage

Sidi Bou Said, Av. de la République, Carthage 1054, La Marsa, Tunisia

Email: afef.saidi@ept.rnu.tn

1. INTRODUCTION

Epilepsy is a nervous system disorder affecting more than 50 million people worldwide, according to the World Health Organization. The commonly used technique for the detection of epileptic seizures relies on the visual interpretation of electroencephalogram (EEG) by experts. Hence, several studies were conducted to build an automatic approach for the detection of epileptic seizures [1]. There are two major approaches in the literature for the development of an automatic epileptic seizure detection system: software-based and hardware-based methods.

With the development of machine learning algorithms, the classification accuracy of automated techniques for epilepsy detection is improving. According to Akyol [2], a new stacking ensemble technique based on deep neural network (DNN) for the detection of epileptic seizures was proposed with an accuracy of 97.17%. To improve the detection accuracy of seizure, Choubey and Pandey [3] used the combination of statistical parameters and two classification algorithms, k-nearest neighbor (KNN) and artificial neural network (ANN), for the classification of EEG signals into healthy, inter-ictal and ictal states. For KNN and

ANN classifiers, the proposed technique achieved an accuracy of 98% and 94%, respectively. Research work on epilepsy detection has been fruitful, with certain epilepsy detection algorithms achieving 100% accuracy [3].

The computational cost of training and inference in these software-based solutions, as well as the need for real-time detection of epileptic seizures, has prompted several studies into hardware acceleration of these computationally intensive algorithms while taking into consideration the detection latency, hardware cost and power consumption. Bahr *et al.* [4] suggested a convolutional neural network (CNN) implementation of an epileptic seizure detection system on an ultra-low-power GAP8 microprocessor with the reduced instruction set computer-five (RISC-V) architecture. This work reached a sensitivity of 85%. In terms of power consumption, the proposed approach outperforms state-of-the-art work by a factor of 6. Elhosary *et al.* [5] proposed a seizure detection system based on a support vector machine (SVM) classifier. The system is implemented and evaluated on two different platforms: field programmable gate array (FPGA) (Xilinx Virtex-7 board) and application-specific integrated circuit (ASIC) (UMC 65 nm CMOS technology). The proposed seizure detection system achieved a sensitivity of 98.38%. Using the dynamic partial reconfiguration (DPR) technique, the authors proposed two optimized designs that resulted in a 64% reduction in power consumption. To reduce the computational complexity of the epilepsy detection system, a systolic array architecture using an SVM classifier was proposed in [6]. A fixed-point arithmetic unit was used to implement the proposed technique on the Xilinx Virtex 7. Compared to the existing system on chip and FPGA seizure detection systems, this work showed efficient results with an average accuracy of 97.2%.

Due to the tremendous performance that deep learning algorithms have enabled, CNN has been used to seek solutions in a variety of medical applications. CNN-based techniques have emerged as the most promising alternative for seizure detection. This widespread comes at the cost of their increasing demands in computation and storage. For example, visual geometry group-19 (VGG-19) model needs up to 39 billion floating point operations for the classification of an input image with a size of 224×224 and more than 500 MB of model parameters [7]. The complexity of these models increases the challenges of their hardware implementation, which must be performed with high speed (i.e., high throughput and low latency) and with minimal energy consumption. The need for this balance has grown critical in a number of areas, including real-time applications. To overcome these issues, many efforts have been dedicated to reducing the model size and the computation cost without affecting the accuracy by suggesting several optimization techniques. The most commonly used techniques are quantization [8] and pruning [9].

Taking advantage of these optimization methods, several research works have been carried out for the development of suitable hardware accelerators for CNN inference. For an efficiency in terms of power consumption and performance (runtime), several hardware CNN accelerators have been suggested in the literature using graphics processing units (GPUs) [10], ASICs [11] or FPGAs [12]. Other works have investigated CNN implementation on microcontrollers to gain lower power and cost [13]. Among these different proposed solutions, FPGAs played an important role and have known a great success [14] due to their high energy efficiency compared to GPUs and their flexibility compared to ASICs and the provided space for design exploration compared to microcontrollers.

For the sake of achieving state-of-the-art classification accuracy on epileptic seizure detection, the size of CNN models gets deeper, which implies an increase in the computation and storage complexity as well as the inference time. For resource-constrained platforms, this is challenging by virtue of the limited resources and frequency. Hence, obtaining good performance and high energy efficiency from a straight-forward FPGA based design implementation of CNN model should not be expected. Thus, there is a critical need for a flexible hardware accelerator that can handle different CNN architectures while achieving higher resource efficiency and optimum performance. In this regard, various tools have been introduced to decrease the complexity of CNN models and to enhance the implementation performance of FPGA-based CNN accelerators. For a further improvement in the performance of FPGA-based accelerators for CNN inference, Xilinx has provided an intellectual property (IP) core named deep learning processor unit (DPU) [15], which supports many basic functions of deep learning and can be used for any topology of CNN in contrast to other hardware accelerators based on FPGA.

The main contributions of this work are: i) we explore an FPGA-based epileptic seizure detection architectures using VGG-16 and ResNet-50 models, ii) we validate our design model approach through a software implementation, iii) we apply filter and weight pruning on the convolution layers of VGG-16 and we evaluate the results in terms of accuracy loss and reduction in model parameters, iv) we use the advanced RISC machine (ARM) NN software development kit (SDK) toolkit for the execution of quantized ResNet-50 and Inception-V4 models on the ARM Cortex-A9 of ZedBoard, and v) we use the Xilinx deep neural network development kit (DNNDK) to quantize, compile and deploy ResNet-50 and Inception-V4 models on the DPU IP implemented on ZedBoard. The remainder of this paper is organized as: section 2 presents our proposed method, the implementation results and discussion are provided in section 3. Finally, conclusions are presented in section 4.

2. METHOD

For this study, to validate the hardware implementation of our proposed epilepsy detection system on ZedBoard, we investigated some of the available tools and frameworks in the research community to reduce the complexity of CNN models and facilitate their implementation on low performance FPGA boards. To reduce the CNN model size and its complexity, we will use different pruning techniques. Additionally, we will explore the software implementation of some complex CNN models using the ARM neural network (NN) SDK toolkit. For a further exploration of the CNN implementation on these platforms, we will introduce a hardware acceleration on Zynq FPGA using the DPU IP accelerator.

2.1. Dataset description

The dataset used for this work is the Children's Hospital Boston-Massachusetts Institute of Technology (CHB-MIT), which is an open-access dataset available on PhysioNet [16]. The dataset consists of scalp EEG (sEEG) recordings of 23 pediatric subjects. The sampling frequency of the dataset used was 256 Hz. EEG recordings consist of 23 channels and contain multiple seizure occurrences. A preprocessing step based on the short-time fourier transform (STFT). This generates time-frequency spectrum images at the input of the CNN model.

2.2. Development board

The target board is a development kit for the Xilinx Zynq-7000 all-programmable system on chip (SoC). To enable a large variety of applications, this board includes all the essential interfaces and supporting functions. It has a Zynq7020 SoC and a dual-core ARM Cortex-A9 processing system (PS), on-chip DDR3 memory (512 MB), quad serial peripheral interface (QSPI) flash memory (256 MB), LUTs (53,200) and flip-flops (106,400) [17].

2.3. Convolutional neural network models

We conducted experiments using three well-known CNN networks: VGG-16, ResNet-50 and Inception-V4. Table 1 shows the different properties of these networks. VGG-16 was trained using the Canadian Institute For Advanced Research (CIFAR-10) dataset. This dataset consists of sixty thousand images (50,000 images for the training and 10,000 images for the test) with 32×32 resolution and it contains ten classes. ResNet-50 and Inception-V4 were trained using ImageNet dataset. It is a well-known dataset that has been used for CNN benchmarks in many applications, such as object detection and object classification. This dataset contains about 10 million images scaled to 224×224 with 1,000 classes.

Table 1. Properties of the used CNN models

CNN model	ResNet-50	Inception-V4	VGG-16
Layers	50	81	16
Number of parameters (million)	25.6	33	138.3
Operation (GFLOPs)	7.7	24.5	30.97

2.4. Application of pruning techniques

At first, to explore the reduction of CNN model size, we targeted two approaches: filter pruning and weight pruning. The evaluation of these approaches has been done on the VGG-16 model. The implementation of both pruning techniques is based on PyTorch. In these experiments, we focused on the convolutional layers, which are the most computationally intensive layers in CNNs. Thus, in our weight pruning method, only the weights in the convolutional layers were pruned. After the network training, the unimportant connections (i.e., whose weight is lower than a given threshold) are pruned and at last the network is retrained in order to fine tune the remaining connections weights. Considering the filter pruning method, for the selection of unimportant filters we used L1-norm, which calculates the sum of the absolute weights of a filter.

2.5. Application of quantization technique

For the evaluation and test of the quantization technique, we exploited two approaches. The first approach is based on the ARM NN SDK for the software implementation of a TensorFlow lite quantized CNN model on the ARM Cortex-A9 of the target board. The second relies on the employment of the Xilinx DNNDK for the hardware implementation of quantized CNN models on the FPGA target.

2.5.1. Software implementation with ARM NN SDK

Lately, ARM revealed the NN SDK and ARM compute library [18] as a set of open-source machine learning software, libraries, and tools enabling existing high-level neural network frameworks

(i.e., TensorFlow, TensorFlow lite, caffe and ONNX) to run on the ARM Cortex-A family of CPU processors and the ARM Mali family of GPUs. The deployment of deep learning application inference on ARM-based processors is done through the functions from the NN SDK API, which translate neural networks to the internal ARM NN format using the optimized functions provided by the compute library. The general workflow is illustrated in Figure 1.

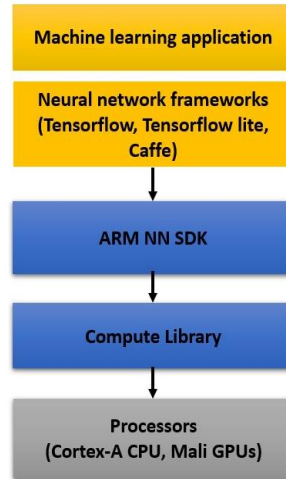


Figure 1. Deployment of deep learning application using ARM NN SDK

We took advantage of the ARM NN SDK toolkit for the execution of CNN models on the ARM Cortex-A9 of ZedBoard. As mentioned above ARM NN SDK supports TensorFlow lite which is an open-source framework that was developed to enable the deployment of machine learning models inference in resource constrained devices such as mobile, embedded and IoT devices. This framework supports model optimization through quantization. It provides several options (different lower precision formats) for model quantization. In our work, we used a TensorFlow lite model with INT8 precision. For the set up and the build of the ARM NN environment on our board, more than 4 GB of storage space is needed for the dependencies of some large libraries such as TensorFlow and other required tools. Given the limited memory of our board, we used another alternative recommended by ARM to target the Arm Cortex CPU of ZedBoard, which is the cross-compilation of ARM NN using an x86_64 system.

2.5.2. Hardware acceleration using xilinx DPU IP

In this part, we have opted to switch to a strictly hardware acceleration of CNNs on ZedBoard using the Xilinx DPU. We used the suggested pipeline shown in Figure 2 to deploy a CNN model on an FPGA utilizing DPU IP. The model is initially trained on the host PC using the TensorFlow framework. Then, we quantized, compiled, and deployed this model on DPU IP using the Xilinx DNNDK.

DNNDK is an integrated framework provided by Xilinx [19], that enables the development and deployment of CNNs on DPUs. It was designed especially for the acceleration of CNN inference on Xilinx FPGA platforms, edge devices (i.e., Xilinx Zynq MPSoC), as well as cloud-based data center systems (i.e., Xilinx Alveo accelerator cards). For an efficient mapping of CNN inference on FPGAs integrated with hard CPU cores, DNNDK offers a set of toolchains including compression, compilation, deployment and profiling. It provides high-level user-space APIs in C++. DNNDK takes in CNN models generated in caffe, TensorFlow or Darknet. The main steps of the deployment of deep learning applications into Xilinx DPU platforms using DNNDK flow are as (Figure 3):

- Model compression through quantization with a tool named DECENT, which converts a floating point (FP 32) CNN model to a fixed-point (INT8) model without a loss in accuracy.
- Compilation of the quantized fixed-point model into DPU kernels using a deep neural network compiler (DNNC). The DPU kernels are ELF format object files containing the DPU instructions and parameters for the network model.
- Hybrid compilation to produce the executable for the target DPU platform.
- Run the executable on the target DPU platform to see the results.

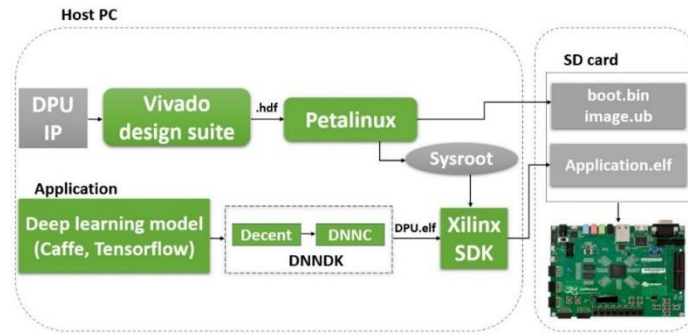


Figure 2. Methodology flow of CNN inference on ZedBoard using Xilinx DPU IP

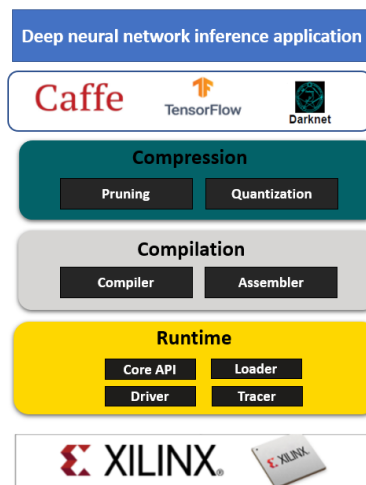


Figure 3. DNNDK workflow

The DPU IP is a soft-processor implemented in the programmable logic (PL) with direct connections to the PS and the external memory. It includes a complex module of computing responsible for the computational tasks of CNN, such as convolutions, and pooling. This module contains a certain number of processing elements (PEs). In order to control the computing complex operations, the DPU retrieves instructions from offchip memory. For high throughput and efficiency, the on-chip memory is utilized to buffer input, intermediate and output data. Figure 4 illustrates the DPU hardware architecture.

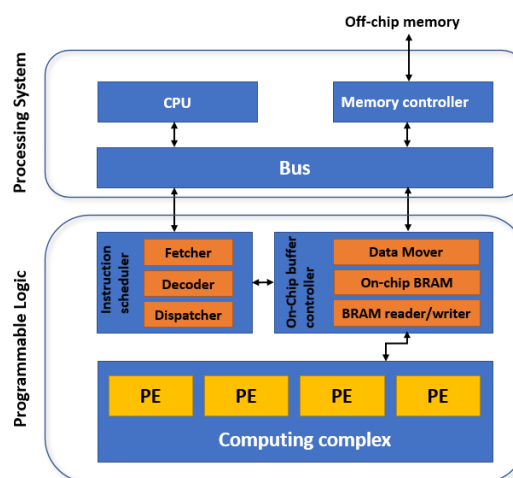


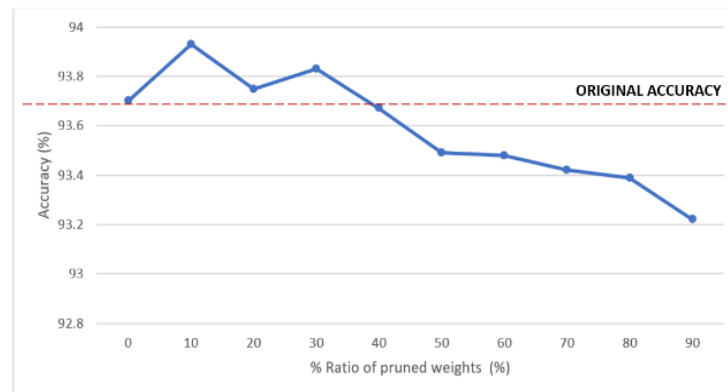
Figure 4. Hardware architecture of DPU

After integrating the DPU IP into the hardware design using Vivado, a bitstream file is generated and the hardware description file (.hdf) is exported to be used for the software build. The board-specific components are built using Petalinux, where the required settings for DPU are included. To deploy CNN models on the DPU, the Xilinx DNNDK is used. This tool enables the model compression using a deep compression tool (DECENT). Then, it generates through the DNNC compiler a DPU.elf file that contains the DPU instructions and the parameters of the model. This file will be compiled in a hybrid compilation together with C/C++ program instructions for the deployment of the CNN application on ZedBoard. Finally, all the needed files (the generated executables, i.e., boot.bin, image.ub and application.elf) are gathered on the SD card to be loaded in ZedBoard. For a comprehensive analysis and evaluation of these two approaches of quantization, we chose ResNet-50 and Inception-V4 models, which have different numbers of layers. The parameter size of these models alters from a few MBs to hundreds of MBs, as presented in Table 1.

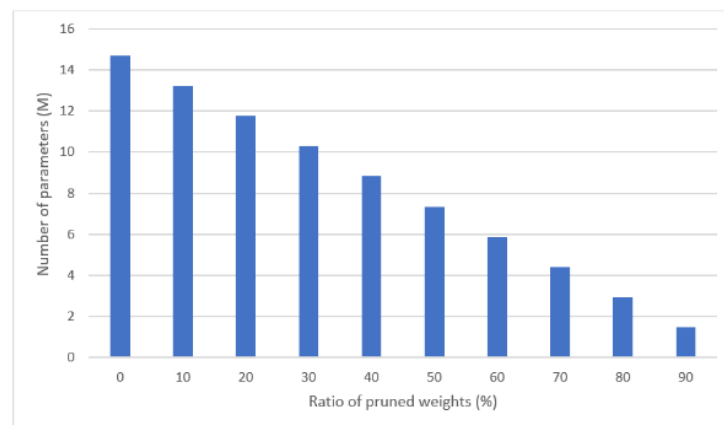
3. RESULTS AND DISCUSSION

To evaluate the performance of the proposed approach, software experiments were conducted on a laptop computer with a NVIDIA GeForce MX 150 and a 64-bit operating system. The achieved accuracy for epileptic seizure detection with the VGG-16 model is about 97.75%, which outperforms ResNet-50 (96.17%). The focus of this paper is the implementation of an epileptic seizure detection system on resource-limited platforms using two large CNN models (VGG-16 and ResNet-50). To ensure the proposed approach, we first leverage the different available solutions for the implementation of these CNN models on Zynq-7,000 platforms and investigate some of them.

We first explored two different techniques of pruning to reduce CNN model size with a minimal accuracy drop namely filter and weight pruning. The performance of the pruning techniques used in our experiments is evaluated in terms of accuracy loss and model parameters reduction. The results of the weight pruning of VGG-16 on the CIFAR-10 dataset is presented in Figures 5(a) and (b).



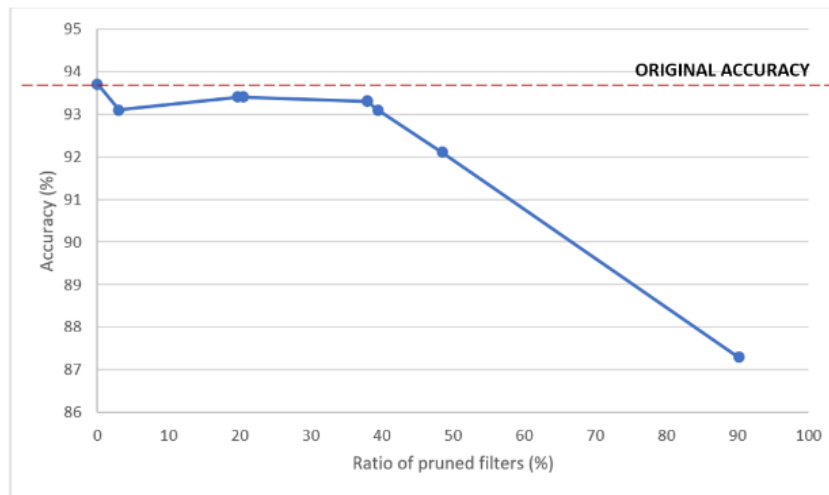
(a)



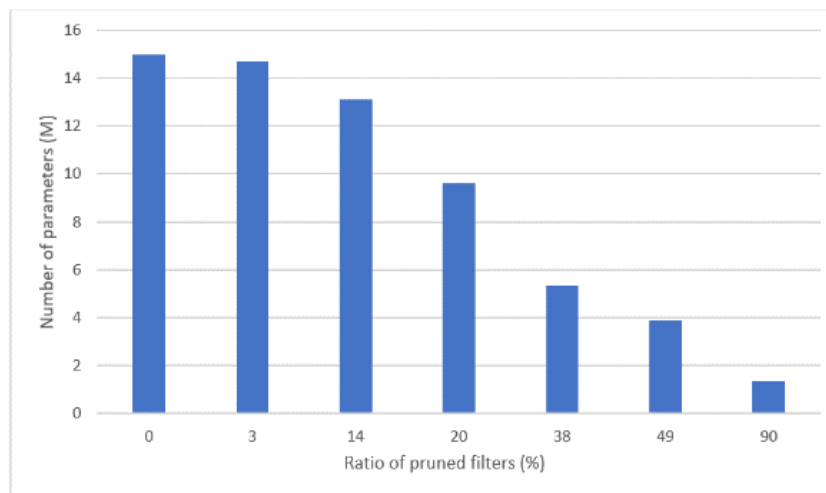
(b)

Figure 5. Weight pruning results (a) weight pruning results of VGG-16 on CIFAR-10 in terms of accuracy and (b) number of remaining parameters in the convolution layers for each pruning ratio of weights

Figures 6(a) and (b) show the results of the filter pruning approach on VGG-16 model in terms of accuracy, the remaining parameters in convolution layers and accuracy loss in different pruning ratio of filters. With a pruning ratio of 90% of filters there is an accuracy loss of 6.4%. This can be explained by the sensitivity of some layers and when pruning them it may be difficult to recover accuracy. While for the weight pruning, a considerable reduction of weight parameters with small accuracy loss has been achieved. As depicted in Figure 5(a), with a pruning ratio of 90%, an accuracy loss of 0.48% is obtained. However, such non-structured method where arbitrary weights are pruned leads to an irregular structure of the network. In contrast, filter pruning as a structured pruning technique induces sparsity in a hardware-friendly way which make it a suitable choice for the acceleration of CNNs.



(a)



(b)

Figure 6. Filter pruning results (a) filter pruning results of VGG-16 on CIFAR-10 in terms of accuracy and (b) number of remaining parameters in the convolution layers for each pruning ratio of filters

Moreover, we investigated the quantization technique by evaluating the implementation of quantized CNN models through two different approaches. The software implementation using the ARM NN SDK and the software/hardware implementation using the DPU accelerator. As shown in Table 2, for the software implementation on the ARM Cortex-A9 of ZedBoard by using TensorFlow lite quantized ResNet-50 and Inception-V4 models, a speed-up of about 1.75 \times was achieved compared to the floating-point models.

In spite of that, the quantization of ResNet-50 and Inception-V4 models through the Xilinx DNNDK (DECENT) tool and their hardware implementation using the DPU IP provides better results in terms of execution time with 85 \times for ResNet-50 and 79 \times for Inception-V4 compared to the software implementation

approach (Table 3). In addition, a performance evaluation of the inference acceleration of ResNet-50 and Inception-V4 models on ZedBoard using Xilinx DPU was done in terms of throughput and runtime as presented in Table 4. The performance evaluation of Xilinx DPU IP has proved the feasibility of running large CNN models with complex architectures (ResNet-50 and Inception-V4) on ZedBoard without the need to develop a custom accelerator.

Table 2. Execution time comparison of INT8 and FP32 ResNet-50 and Inception-V4 models implementation on the ZedBoard ARM Cortex-A9

Execution time (ms)		
Precision	FP32	INT8
ResNet-50	15000	8571.4
Inception-V4	44000	25882.3

Table 3. Implementation results of quantized CNN models using the two approaches in terms of execution time

Target	ZedBoard with DPU accelerator		ARM Cortex-A9 of ZedBoard	
Frequency	90 (MHz)		33.333 (MHz)	
Tool	Xilinx DNNDK		ARM NN SDK	
Models	ResNet-50	Inception-V4	ResNet-50	Inception-V4
Execution time (ms)	100.636	327.162	8571.1	25882.3

Table 4. Implementation results of CNN models on ZedBoard using DPU IP

Board	ZedBoard with DPU accelerator	
Dataset	ImageNet	
CNN model	ResNet-50	Inception-V4
Wokload (GOPS)	7.71	24.5
Execution time (ms)	100.636	327.162
Throughput (GOPS/s)	76.6127	74.9155

Nevertheless, running more complex CNN models, such as VGG-16, on Zynq-7,000 platforms using DPU IP remains a challenge. Even with the quantization of the VGG-16 model with Xilinx DNNDK, the model size is 132 MB [20]. To complete the large amount of calculation needed for VGG-16, a DPU core with the maximum size realized on the ZCU102 board was used, which is the B4096 core [20]. Compared to the available resources on Zynq-7,000 platforms, this DPU core uses over 3.2× of DSP, 1.12× of LUT, and 1.9× of BRAM. Moreover, for Zynq-7,000 platforms only one DPU core can be used. The main goal of this work was to implement on ZedBoard our epilepsy detection system with the CNN model that gives the best accuracy, which is VGG-16. However, as mentioned in the discussion above, the implementation of VGG-16 on platforms with limited resources, such as ZedBoard is a challenging task. This makes the implementation of our approach for epilepsy detection using VGG-16 on ZedBoard not possible.

At last, after our exploration of the proposed tools and techniques, we have encountered some difficulties. During our experiments of pruning methods which were conducted on host PC (intel core I5-8250U), we faced some issues including the computational complexity associated with pruning and the retraining step which demands a lot of time. Also, the choice of hyperparameters like the sparsity of the network pruning is a tedious and time-consuming task.

In addition, after our use of the Xilinx DNNDK tool, we have noticed that it has some limitations. A limited number of pre-trained CNN models can be successfully run on ZedBoard using DNNDK. In addition, the DECENT pruning tool requires a license and only the Xilinx pruning tool is supported. If the network is pruned using third-party pruning tools, DNNDK will not support it. Even when using the same technique adopted by the Xilinx pruning tool, which is coarse-grained pruning (iterative channel pruning). Thus, it is not possible to implement an optimized model on DPU without going through Xilinx tools. This has been concluded after our different attempts to implement pruned CNN models on ZedBoard using DPU, which are:

- We applied channel pruning to the ResNet-50 model based on the caffe framework. However, we were not able to quantize this model using the DECENT tool.
- We applied channel pruning to a simple CNN model (4 convolution layers and 2 fully connected layers).
- We managed to get the model quantized with DECENT, unfortunately the execution on the board stops at the last layer of convolution.
- When trying to implement the provided pruned models by the Xilinx model zoo repository (SSD, RefineDet, VPGNet, OpenPose), we encountered some problems due to the need for additional DNNDK.
- Binaries and libraries, which are for arm-64 and they don't support Zynq-7,000.

4. CONCLUSION





This work presented an FPGA implementation of epileptic seizure detection using CNN model. This paper has examined the efficiency of two pruning methods, filter and weight pruning, applied on the convolution layers of VGG-16 model in the reduction of the model parameters and in keeping the accuracy. In this paper the inference implementation of large CNN models on resource constrained platforms, as a case ZedBoard, has been investigated from two aspects, a software implementation on the ZedBoard ARM Cortex-A9 and a hardware implementation using Xilinx DPU IP accelerator. A complete flow for such an implementation has been presented, including the implementation flow, the development frameworks used in these implementations as well as the hardware architecture design.

REFERENCES




- [1] S. Saminu *et al.*, "Applications of artificial intelligence in automatic detection of epileptic seizures using EEG signals: a review," *Artificial Intelligence and Applications*, vol. 1, no. 1, pp. 1–15, 2022.
- [2] K. Akyol, "Stacking ensemble based deep neural networks modeling for effective epileptic seizure detection," *Expert Systems with Applications*, vol. 148, p. 113239, 2020, doi: 10.1016/j.eswa.2020.113239.
- [3] H. Choubey and A. Pandey, "A combination of statistical parameters for the detection of epilepsy and EEG classification using ANN and KNN classifier," *Signal, Image and Video Processing*, vol. 15, no. 3, pp. 475–483, 2021, doi: 10.1007/s11760-020-01767-4.
- [4] A. Bahr *et al.*, "Epileptic seizure detection on an ultra-low-power embedded risc-v processor using a convolutional neural network," *Biosensors*, vol. 11, no. 7, pp. 1–18, 2021, doi: 10.3390/bios11070203.
- [5] H. Elhosary *et al.*, "Hardware acceleration of high sensitivity power-aware epileptic seizure detection system using dynamic partial reconfiguration," *IEEE Access*, vol. 9, pp. 75071–75081, 2021, doi: 10.1109/ACCESS.2021.3079155.
- [6] S. Shanmugam and S. Dharmar, "Very large scale integration implementation of seizure detection system with on-chip support vector machine classifier," *IET Circuits, Devices and Systems*, vol. 16, no. 1, pp. 1–12, 2022, doi: 10.1049/cds2.12077.
- [7] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "A survey of FPGA-based neural network inference accelerators," *Arxiv-Computer Science*, vol. 3, pp. 1–26, 2018, doi: 10.48550/arXiv.1712.08934.
- [8] M. P. Vestias, R. P. Duarte, J. T. D. Sousa, and H. C. Neto, "A configurable architecture for running hybrid convolutional neural networks in low-density FPGAs," *IEEE Access*, vol. 8, pp. 107229–107243, 2020, doi: 10.1109/ACCESS.2020.3000444.
- [9] X. Sui *et al.*, "A hardware-friendly high-precision CNN pruning method and its FPGA implementation," *Sensors*, vol. 23, no. 2, pp. 1–22, 2023, doi: 10.3390/s23020824.
- [10] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: an extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.
- [11] H. Sharma *et al.*, "Bit fusion: bit-Level dynamically composable architecture for accelerating deep neural networks," in *Proceedings-International Symposium on Computer Architecture*, 2018, pp. 764–775, doi: 10.1109/ISCA.2018.00069.
- [12] H. Sharma *et al.*, "From high-level deep neural models to FPGAs," in *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, 2016, pp. 1–12, doi: 10.1109/MICRO.2016.7783720.
- [13] H. A. Shiddieqy, F. I. Hariadi, and W. Adijarto, "Plug-load classification based on CNN from V-I trajectory image using STM32," in *2021 International Symposium on Electronics and Smart Devices (ISESD)*, 2021, pp. 1–5, doi: 10.1109/ISESD53023.2021.9501919.
- [14] S. Mittal, "A survey of FPGA-based accelerators for convolutional neural networks," *Neural computing and applications*, vol. 32, pp. 1109–1139, 2020.
- [15] *DPU for convolutional neural network v3.0 DPU IP product guide*. San Jose, USA: Xilinx, 2019.
- [16] "CHB-MIT scalp EEG database," *PhysioNet*, 2010. <https://physionet.org/content/chbmit/1.0.0/> (accessed Apr. 17, 2021).
- [17] *ZedBoard (zynq™ evaluation and development) hardware user's guide*. Phoenix, USA: Avnet Electronics Marketing, 2012.
- [18] "IP products|arm NN–arm developer," *Arm Developer*, 2018. <https://developer.arm.com/ip-products/processors/machine-learning/arm-nn> (accessed Dec. 21, 2020).
- [19] *DNNDK user guide*. San Jose, USA: Xilinx, 2019.
- [20] Y. Lei, Q. Deng, S. Long, S. Liu, and S. Oh, "An effective design to improve the efficiency of DPUs on FPGA," in *Proceedings of the International Conference on Parallel and Distributed Systems-ICPADS*, 2020, pp. 206–213, doi: 10.1109/ICPADS51040.2020.00036.

BIOGRAPHIES OF AUTHORS






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 at 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 Othman    received the Dipl.-Ing and M.S degree in electrical engineering from the National Institute of Applied Science and Technology (INSAT), Tunisia, in 2004. In 2011, he obtained his Ph. D in industrial data processing from INSAT. Currently, he is a research member at Laboratoire des Systemes Advances (Laboratory of Advanced Systems) in the Polytechnic School of Tunisia. Since 2006, he has been a professor assistant in the Department of Electrical Engineering, Higher Institute of Medical Technologies, Tunisia. His research interests include embedded systems design methodologies and architectures as well as artificial intelligence algorithms acceleration through reconfigurable computing on FPGA. He can be contacted at email: slim.benothman@ept.rnu.tn.



Meriam Dhouibi    Ph.D student at the Department of Electrical Engineering at the National Engineering School of Carthage. She is doing her research work at the Advanced Systems Lab (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 Equipment. 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.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 SpaceWire modules. He can be contacted at email: slim.bensaoud@gmail.com.