ISSN No:-2456-2165

High Performance FPGA Based CNN Accelerator

Pratiksha B. Dange
Department of Electronics & Telecommunication Engineering
J D College of Engineering & Management
Nagpur, India

Dr. S.L. Haridas
Department of Electronics & Telecommunication Engineering
J D College of Engineering & Management
Nagpur, India

Abstract:- Over the years, convolutional neural networks have been used in different applications, due to their ability to perform tasks using a reduced number of parameters compared to other in-depth learning methods. However, the use of power and memory constraints, which are often marginal and portable, often conflict with the requirements of accuracy and latency. For these reasons, commercial commercial accelerators have become popular and their design is built on the tendencies of the overall convolutional network models. However, the layout of the gate-mounted gateway represents an attractive view because it offers the opportunity to use a hardware design designed for a particular model of a convolutional network, with promising results in terms of latency and power consumption. In this article, we propose a complete accelerator for chip-programmable gate array hardware for convolutional neural network partition, designed for a keyword recognition system.

Keywords:- *CNN, Accelerator, FPGA.*

I. INTRODUCTION

As communication systems evolve and power levels increase, the spectrum is pushed up to higher waves to deal with the bulk of the information. With the introduction of 5G mobile technology, these assumptions are thought to be as high as between 3 and 27 GHz [10] which will go far beyond the standard radar spectra, especially with the K-band radar. With this comes the need to improve spectrum sensitivity and signal identification algorithms that allow sensors and radios to detect and identify spectrum users and participants. These algorithms have traditionally been the result of hand-crafted masterpieces. With the recent practice of using a machine to learn to process powerful results, neural networks have been shown to do well in the problem of radio signal recognition. The culture of neural networks, and especially deep neural networks, has been applied to graphics processing units (GPUs). Today's GPUs are very powerful and have many similar computer features that are well suited for deep applications. Unfortunately GPUs work poorly with power to make them unsuitable for power-limited applications. Radar works with a large amount of data and requires high throughput and low latency, if the radar is installed in an object without external power resources, it should also use as much power as possible. For this reason, it may be necessary to use these algorithms on customized hardware to meet these requirements. Field-based gate layout (FPGAs) has a good balance between cost, energy efficiency and computational resources that make it a good fit for this application. In recent years we have had a huge data boom in many fields. To address the expansion of this "big data" the answer is found in Artificial Intelligence (AI). We can define it as an AI software or hardware application that thinks and solves problems as a human being can. Problems ranging from language translation to internal image separation to understand and understand different faces and people. What we have found so far is small AI, which uses some algorithms and techniques to solve some specific problems.

Since neural networks are naturally similar, they can draw a significant amount in comparison to FPGAs (Field Programmable Gate Arrays). Performance in FPGAs has been shown to have significantly lower power consumption per function than equivalent in GPU (Graphical Processing Unit), which is a requirement for embedded systems. However, implementation is no small feat because the development of FPGAs is often done in hardware that describes the hardware, e.g. VHDL.

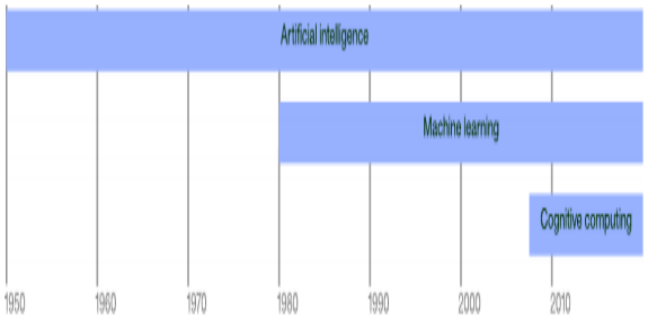

Figure 1 Humans and AI

Within the AI field, we find Machine Learning (ML), which consists of using a large set of data and a number of partition algorithms to change the standard method we are accustomed to making a program. With our standard programming method, we create many algorithms that are complex, but we are well versed in each of them. The basic idea of creating a division is to get big data and simply perform tasks to understand which of the data we need and thus improve hands-on results and get a system that without writing the whole algorithm, is able to make decisions based on available data. Some of these categories follow mathematical methods we know, such as straight lines, polynomial functions, mathematical functions, and so on. Some of these are very good at predicting a particular type of behavior that is very difficult to record in an algorithm such as guessing the price of a house based on a history series. Further, as an ML branch, we find a specific data learning process, known as Deep learning (DL).

ISSN No:-2456-2165

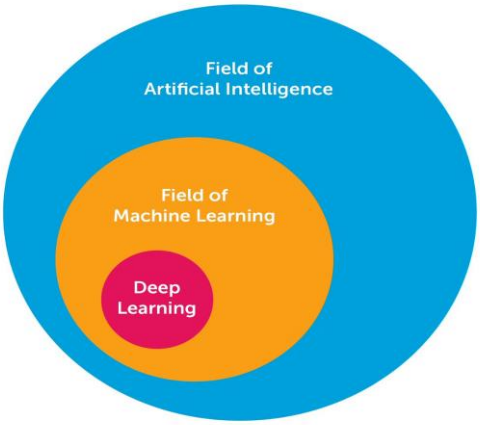

Figure 1.2: AI world

The development of DL occurred in a similar manner to a study mainly of neural networks. It is characterized by efforts to create a learning model with multiple levels of automation, where deeper levels take into account the effects of previous levels, transforming them and further amusing. This intuition at the reading level gives a name to the whole field and is inspired by how the brain of a mammal processes information and learns, responding to external factors.

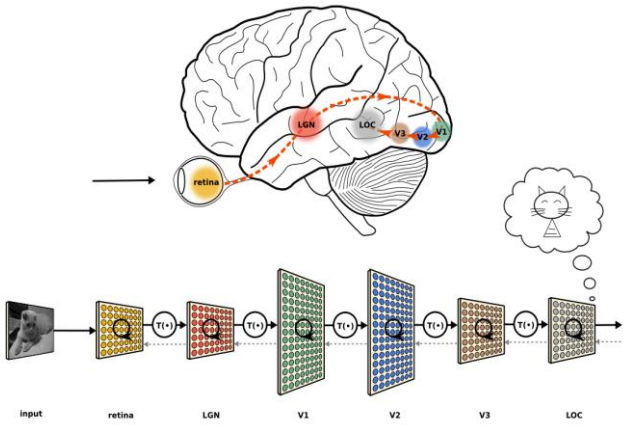

Figure 1.3: Mammal brain and Convolutional process

Over the years, convolutional neural networks (CNNs) have found application in many different fields such as object detection [1, 2], object recognition [3, 4], depending on memory and energy consumption, which often contradicts the requirements of delay and accuracy. In particular, with standard purpose-based solutions based on the use of a microcontroller, limited available memory limits network complexity, with a potential impact on t accuracy.[7]

In the same way, microcontroller based systems feature the worst trade-off between power consumption and timing performances [8]. For this reason, commercial hardware accelerators for CNNs such as Neural Compute Stick (NCS) [9], Neural Compute Stick 2 (NCS2) [9], and Google Coral [10] were produced. Such products feature optimized hardware architectures that allow to realize inferences of CNN models with low latency and reduced power consumption. Standard communication protocols, such as Universal Serial Bus (USB) 3.0., are generally exploited for communication purposes. Nevertheless, since they were designed for the implementation of generic CNNs, their architectures are extremely flexible at the expense of the optimization of the single model.

For such a reason, hardware accelerators customized for a specific application might offer an interesting alternative for accelerating CNNs. In particular, field-programmable gate arrays (FPGAs) represent an interesting trade-off between cost, flexibility, and performances especially for applications whose architectures have been changing too rapidly to rely on application-specific integrated circuits (ASICs) and whose production volumes might be not sufficient. FPGAs offer high flexibility at the same time, which permits the implementation of different models with a high degree of parallelism and the possibility of customizing the architecture for a specific application.

II. LITERATURE SURVEY

A well-executed block of floating point-point (BFP) was adopted in our accelerator to determine the functional tendency of deep neural networks in this paper. Feature maps and model parameters are represented in 16-bit and 8-bit formats, respectively, for off-chip memory, which can reduce memory and band-band requirements by 50% and 75% compared to 32-bit FP colleague. The proposed 8-bit BFP figure with optimized advantages and flexibility of performance-based schemes improves energy efficiency and triple hardware. The FPGA-based CNN accelerator is distributed on the Xilinx VC709 test board [1].

The functional design of the hardware is introduced based on FFT performance due to the radix-2 frequency decimation algorithm (R2DIF) and a distributed method that allows data to be shared efficiently by keeping registry changes. A well-rounded method / design uses a rotating computer algorithm for converting connections (m-CORDIC) and Radix-2r according to a coding system to replace complex multiplication such as FFT. The m-CORDIC algorithm improves computer integration, while Radix-2r allows logarithmic reduction of adder steps. Suggested design does not require large memory blocks used to retain a feature like twiddle [2].

CNN accelerator based on FPGA. The most effective accelerator function is designed to build a flexible neural network and memory optimization with the use of low-cost resources. The results show performance gains and power compared to the Core i5 CPU and GTX 960 GPU [3].

The hardware model is designed for CNN's advanced step-by-step use of hardware definition language, including CNN's computer architecture, multi-layer use, weight loading system, and data interference [4].

The possibility that existing existing low-power register (RTL) strategies could serve as a low-power design scheme to accelerate an CNN-based object recognition system in contrast to conventional strategies. Many of the most effective design strategies for CNN acceleration focus on

High-level Synthesis (HLS) features, such as memory bandwidth usage, network architecture, data reuse, and batch editing [5]

The CNN accelerator on the Xilinx ZYNQ 7100 hardware platform accelerates both standard resolution and depth of cleverly divided convolution. Taking the design of the MobileNet + SSD network as a speed, the accelerator simulated the measurement of the entire computer network under the ZYNQ 7100 roof model. chip on the chip using the data streaming interface and set the ping-pong buffer mode. [6]

III. MOTIVATION

A major problem with the implementation of the CNN-based model in the FPGA regarding the limitations in terms of hardware resources (combinations, sequences, Digital Signal Processors (DSPs), ram blocks, etc.) of such devices. CNN algorithms are based on Multiply-and-Accumulate (MAC) operations that require a large number of logical objects or DSPs. In addition, CNNs are characterized by a large number of parameters such as resource requirements, number of work per second (Gops), Density Efficiency (DE), time required to create layers of CONVs, FC and Softmax, and Power Efficiency (PEff). For these reasons, the hardware speed of the hardware was carefully designed taking into account the trade between the measurement period and the available resources.

IV. PROBLEM STATEMENT

Hardware accelerator performing convolution function, the most critical function of ConvNet. To give an idea of the computer load of ConvNet, in the AlexNet model, for example, 90% of the processing time is spent on convolution tasks. Moreover, the complexity of this network is strongly linked to their depth and one of the major problems is that this type of work requires a lot of memory. We will try to use the structures that perform this function in a structured way, and to use strategies to reduce the amount of access to external DRAM linked to FPGA during the calculation of convolution. The purpose of both is to exploit as much as.

V. HARDWARE ACCELERATORS

A hardware accelerator is a specialized hardware unit that performs a set of tasks with higher performance or better energy efficiency than a general-purpose CPU. Example of common accelerators are GPUs, digital signal processors (DSPs) and fixed-function application specific integrated circuits (ASICs) like video decoders [11]. To understand why accelerators have become so important, the history of the semiconductor industry has to be taken into account. The semiconductor industry has historically been driven by two scaling laws: Moore's Law and Dennard scaling. It is these two scaling trends that have made CMOS technology so popular in the computing industry. Moore's law states that the number of transistors that can economically be fit onto an integrated circuit doubles every two years. It is more than just shrinking transistors to yield better integration capabilities, it is fundamentally a cost-scaling law. Moore was interested in shrinking transistor costs. Moore observed that the cost of transistors depends on two factors. One is the density of transistors that can be crammed in onto a single chip, and the second is the yield of fabrication. To maintain Moore's law, two factors are critical:

1) transiztor size - the smaller the better.
2) wafer size - the larger the better since more chips can be produced from a fixed number of processing steps.

VI. DESIGN ARCHETECTURE

Accelerator architectures for neural nets There exists two major architectures of hardware accelerators for neural networks, single computation unit accelerators or streaming accelerators. Single computation unit (SCU) accelerators have a similar construction to a RISC CPU that executes instructions with a fixed datapath. Instead of an ALU or FPU, the SCU accelerator has a dedicated matrix multiplier tailored for big matricies or a systolic array of computation elements. When a network is to be accelerated on an SCU accelerator, instructions are generated for that specific network. The accelerator can then execute these instructions from memory. These types of accelerators are very flexible since the only network specific data that has to be stored are the instructions and the parameters. This enables networks with different architectures to be executed on the same accelerator. These accelerators suit systems that execute several different networks since the accelerator can be shared between the tasks and the overhead to execute a new network is not that big. Even though SCU accelerators are often fixed for all types of networks, they can be tailored to specific networks with regard to the width of datapath and size of matrix multiplier/systolic array to better match the layers sizes in the network and yield a higher resource utilization. This semi tailoring of the SCU accelerators tend to reach higher performance on CNN's with a uniform structure. This is because the utilization of the shared compute unit increases if the kernel sizes between layers are similar. It is also true if the size ratio between different layers is a power of two. When the deployment of neural networks onto these types of accelerator are automated, the automation framework generates the instructions and quantizes the weights as needed

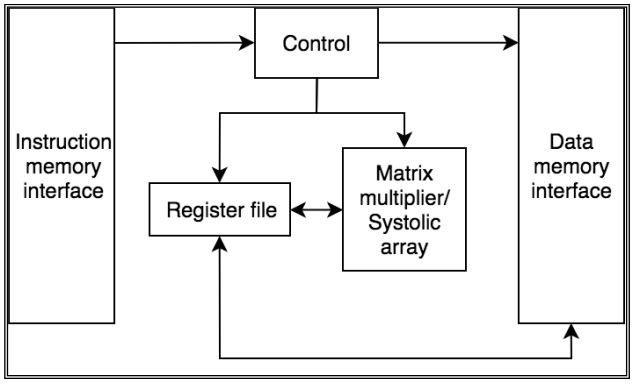

Figure 3: Basic structure of a single compute unit accelerator

Accelerators with the streaming architecture always tailor the hardware with respect to the target network. The layers are often directly implemented in hardware and its

ISSN No:-2456-2165

possible to get a very high level of parallelism and utilization. The intermediate results between layers can be stored in registers, memory or directly pipelined into the next layer. This architecture is better suited for smaller networks since a direct mapping can consume a lot of resources. One way to circumvent this resource constraint issue is by using a method called folding. With folding, one layer at a time is executed on the FPGA and the FPGA is reconfigured between each layer. Since the FPGA needs to be reconfigured between each layer, batches of data has to be executed to yield a sensible throughput. Folding can generally yield a very high throughput since the level of parallelism in each layer tend to be high but the latency is often big since large batches of data has to be executed. Figure 4 shows a block diagram of a simple streaming accelerator.

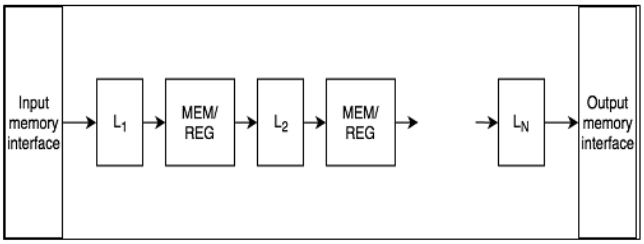

Figure 4: Basic structure of an streaming accelerator

VII. CONCLUSION

The design of cnn accelerator for improving performance of the system is very much important. The design of accelerator which makes reduce of load over CPU/GPU. The design improves the efficiency of a system. An hardware accelerator able to merge the demands in terms of speed and power through a careful analysis of the possible parallelization inside the CNN algorithm.

REFERENCES

[1]. Xiaocong Lian , Member, IEEE, Zhenyu Liu, Member, IEEE, Zhourui Song, Jiwu Dai, Wei Zhou , Member, IEEE, and Xiangyang Ji , Member, IEEE, "High-Performance FPGA-Based CNN Accelerator With Block-Floating-Point Arithmetic" ,IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 27, NO. 8, AUGUST 2019.

[2]. M. S. Kavitha1 · P. Rangarajan2, "An Efficient FPGA Architecture for Reconfigurable FFT Processor Incorporating an Integration of an Improved CORDIC and Radix-2r Algorithm", Circuits, Systems, and Signal Processing. https://doi.org/10.1007/s00034-020-01436-4

[3]. Sheping Zhai, Cheng Qiu, Yuanyuan Yang, Jing Li and Yiming Cui,Sheping Zhai,, Cheng Qiu1, Yuanyuan Yang1, Jing Li1 and Yiming Cui1, "Design of Convolutional Neural Network Based on FPGA". CISAT 2018

[4]. Xiaofeng Chen1, Jingyu Ji1, Shaohui Mei1, Yifan Zhang1, Manli Han2, Qian Du, "FPGA BASED IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK FOR HYPERSPECTRAL CLASSIFICATION", IGARSS ©2018 IEEE.

[5]. Heekyung Kim and Ken Choi, "Low Power FPGA-SoC Design Techniques for CNN-based Object Detection Accelerator", ©2019 IEEE.

[6]. Bing Liu , Danyin Zou, Lei Feng, Shou Feng, Ping Fu and Junbao Li, " An FPGA-Based CNN Accelerator Integrating Depthwise Separable Convolution", Electronics 2019, 8, 281; doi:10.3390/electronics8030281

[7]. Yuchi Tian et al. "Deep Test: Automated Testing of Deep-neural network- driven Autonomous Cars". In: Proceedings of the 40th International Conference on Software Engineering. ICSE '18. 2018, pp. 303–314.

[8]. C. Zhang et al., "Optimizing fpga-based accelerator design for deep convolutional neural networks," in Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays, Feb. 2015, pp. 161–170.

[9]. J. Qiu et al., "Going deeper with embedded fpga platform for convolutional neural network," in Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays, pp. 26–35, 2016.

[10]. K. Guo et al., "Angel-Eye: A complete design flow for mapping CNN onto embedded FPGA," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 37, no. 1, pp. 35–47, Jan. 2018.

[11]. H. Li, X. Fan, L. Jiao, W. Cao, X. Zhou, and L. Wang, "A high performance FPGA-based accelerator for large-scale convolutional neural networks," in Proc. 26th Int. Conf. Field Program. Logic Appl., Aug./Sep. 2016, pp. 1–9.