International Journal of Industrial Engineering
2020;4(2):44-49.
ISSN: 2456-8449

Low-Complexity Deep Neural Network Accelerator on VLSI Using Stochastic Computing

Anantha Raman Rathinam[1*], Yuvaraj S[2], K Umapathy[3]

[1]*Department of Computer Science and Engineering, Malla Reddy Institute of Engineering and Technology, Hyderabad, Telangana, India.*
[2]*Department of Electronics and Communication Engineering, Faculty of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamil Nadu, India.*
[3]*Department of Electronics and Communications Engineering, Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya University (Deemed University), Kanchipuram, Tamil Nadu, India.*

**Corresponding author: granantha.raman@gmail.com*

Abstract. Convolutional Neural Networks continue to be dominant research inside the areas of GPU acceleration using Programmable Arrays, demonstrating their efficacy in a variety of technical vision tasks such as extraction of features, image analysis, person detection, and rear bridge alert, among many others. Nonetheless, there are other constraints to deploying the Deep network on FPGA, such as limited on-chip recall, DNN dimensionality, and parameters. This study proposes Tv commercial, a powerful DNN prototype based on the baseline Alexnet prototype. The proposed framework makes use of a Commercial engine, which is a developed version of the insight different and independent randomization unit. Designers also propose a GPU integration model that supports the initiation characteristics Mish and also Rectified linear. Regardless of the fact that only a minimal quantity of computer equipment was employed, the practical results were attained. When compared to state-of-the-art techniques, the given scheme offers a comparatively high detection accuracy while requiring fewer computer system resources. Furthermore, when compared to the estimate technique, the proposed framework helps to reduce system resources even more than others.

Keywords: Optimization, CNN, Algorithm, Machine learning, Deep learning, DNN, Decoder.

INTRODUCTION

Convolutional neural networks (CNNs) are widely used in research, particularly in image processing applications such as facial recognition, image classification, meteorology, and object identification. CNNs built recently have become deeper [1,] as has the area of study as a whole. The more complicated the CNN, the more challenging it is to build as a silicon acceleration on an FPGA-based combined board while keeping comparable computation speed and accuracy. Shallow and lightweight CNN models, such as MobileNet [2] and ShuffleNet, are appealing options for FPGA implementation. Nonetheless, everyone is concerned about maintaining the greatest precise and computational capability possible[3]. To address this issue, Ad-MobileNet is a system based on the conventional MobileNet design that achieves significant identification accuracy that exceeds numerous leading FPGA-based variants while reducing the processing complexity of physical infrastructure. Ad-MobileNet is not the same as the conventional MobileNet paradigm. In this approach, designers propose an Ad-depth unit to replace the traditional depth-wise separable Fourier. This unit is best suited for embedded devices because to its higher precision.

Convolutional neural networks (CNNs) requires costly CPU operations and memory capacity, leading standard CPUs to fall short of the requisite efficiency requirements. As a result, to increase CNN throughput, hardware accelerates based on application-specific computer devices (ASICs) and field-programmable gate arrays (FPGAs) have already been used [4]. In truth, FPGAs provide flexibility, high speed, fine-grained parallelism, and energy efficiency. Nonetheless, due to ram resources, computing capacity, and power use limits, installing electronics CNNs at the edge is a tough task. Many innovative ways for increasing the acceleration efficacy of CNNs on FPGAs have been emerged in recent years [5]. FPGAs have the potential to deliver enough functionality while using less power. FPGAs, from the other hand, have a limited amount of

Received: 14.08.2020 Revised: 08.09.2020 Accepted: 19.09.2020

memory and computing power. As a result, several methodologies that propose foundations for CNNs highlight the trade-offs between resolution, computational power, and infrastructure components. In [6,] the researchers proposed a processing element (PE) in which the multiplier accumulation (MAC) was altered by a Wallace Tree-based Multiplication. By combining FIFO and ponging storage, the authors of [7] demonstrated an efficient virtual memory using a Xilinx ZCU102 device. The authors of [8] proposed a pipelined layered architecture with a conventional ReLU as an input signal to improve computing performance. Analyses have been carried out at the coarse grain level to boost the speed of CNN reasoning. [9] proposed a SqueezeNet architecture with 23 layers, with each layer of the CNN optimised and constructed individually on a Xilinx Experience. An FPGA board is included. However, the investigations demonstrated a low recognition rate of 79 percent for the top-five accuracy. In [10], the authors suggested a decreased CNN model, which reduced the network model's size. As a result, to fit smaller convolutional kernels, they employed a Xilinx. In particular, [11] has two separate processing engines for representations and depthwise convolution, Conv and Dwc [12].

The CNN has a faceted framework that includes a convolutions that extracts wavelet transforms from an image pixels, a variational activation that determines which functionality are conveyed between neurons, a grouping surface which thus reduces the dimension of something like the feature of convolution operation, thus decrease the occurrence of criteria and computational stockpile (it also prevents fitting problems during the learning experience), and fully connected layers that categorise the extracted features[13]. There are a variety of CNN architectures available, each with its own size and shape.

The convolution procedure employs a set of learnable filters to detect basic characteristics in the image. Convolution is a linear method in which a double array called a filter is multiplied by an input array. The smaller-than-input-data filtering is used several times throughout the input array, resulting in a two-dimensional array known as a feature map[14]. The component multiplication is performed via a multiply-accumulate (MACC) arithmetic operator, which returns a single value. The filter, kernel, or weight matrix multiplication of a receptivity region of the inputs is depicted in Figure 1. A bias value is multiplied by the sum of the weighted inputs. The convolution technique for a three-dimensional input is depicted in the diagram[15].

An activating approach creates a weight matrix and then adds a bias to it to determine if a neuron should be stimulated. It's like a gate that verifies if a value entering it is greater than a specific threshold. The fundamental goal of the activation stored process is to generate nonlinear behaviour in a photoreceptor output. To summarise, the artificial neuron improves the ability of the neural network to learn and accomplish complex tasks[16]. Almost all researchers have used fundamental nonlinear functions that need little computational resources for practical applications.

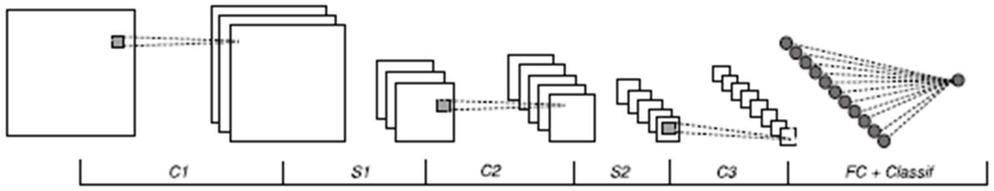

FIGURE 1. CNN Structure with Three Convolutional Layers, Two Pool Layers

CNNs have such a high computational burden as well as a high memory capacity need. For systems with everyday operation and quasi-data accesses, the cost of decoding and utilising each on application platforms is substantial, and caching design for memory locations is typically unneeded and inferior to efficient buffer structures. Because the cost of CNN calculation is highly predictable, performing this work and using very simple buffering methods can considerably reduce the number of DRAM accesses necessary.Any information needed in the previous matrices would be published, according to the announcement[17]. The next step is to evaluate it. At times, some characteristics may not have values and outliers matrix. Each of these qualities needs a contribution to the data collection process. You must've been willing to accomplish the three requirements stated above to avoid losing access to examples. At this level, a variety of strategies are possible. Linear, periodic, exponentially, or exponential extension are two methods for filling in the gaps with values in the range of the mean, median, or zero[18]. Different VLSI archietctures are discussed in [19].

The factorial approach, it was said, now delivers a more diversified data depiction than the GAF method. Diverse visual trends make it easier for the CNN model to differentiate between normal and erroneous patterns, allowing it to apply additional filtering techniques. The learning, validation, and fine-tuning of the CNN model are explained in this section. The selection of proper hyper-parameters for the ML job is critical. During training, the hyperparameters indicated above can be used. After the prototype has gone through the testing steps, it should be reviewed. Various tactics can be used, such as gullible searching or a search algorithm. The CNN model's searching optimization considers and evaluates an infinite number of hyperparameter pairs, whereas the random search considers and evaluates a random number of alternatives. The optimal CNN model for addressing the challenge of predictive analytics. The max-pooling layer works in the same way as the convolution layers. The importance of the convolution characteristic should be diminished. It also aids in the reduction of feature sizes while gathering the most important qualities in order to achieve a high level of identification accuracy. Pooling is accomplished using two well-known strategies: average pooling and maximum pooling. Max-pooling reduces the feature dimension while denoising the input. The pooling layer, on the other hand, serves as a wavelet transform denoising approach.

The pooling layers approach resulted in a single compressed vector of data. Each value is the probability that the traits and labels are equivalent. The convolution layer converts the images into labels using the characteristics generated by the pooling layers approach. Each neuron prioritises the matching label for each incoming weight. Following that, all neurons will vote on which label will be the winner of the classification.Over the years, several unique network topologies have been proposed for deployment on support systems with limited storage and processing capabilities, such as ShuffleNet and Part of an extended network. The success of these models is due to the use of the depth-wise separable convolutional (DSC) unit. The effectiveness of this unit has been demonstrated by the reduction in the number of variables and network computations. The accuracy of the network was affected by reducing the amount of variables. As a result, we saw DSC development as a cost-versus-accuracy trade-off. For each input port used by the convolutional network, the multiple outputs are summed to form one output channel in conventional inversion.

However, owing of the large number of parameters and calculations required by the networks, as well as the limited memory available on the hardware, the optimised form of CNNs is widely used in real-world applications. As a consequence, we experimented with making a small and thin model with high accuracy while using a minimal amount of underlying technology. The system that has been proposed is based on shallow water.

The result is a matrix including all of the individual matrix components, with the feature value equal to the sum of their values. Following the filter's application, it continues to cross over the input vector and do each iteration of the 3x3-sized inner product multiplied for each other area combination until the extracted features are produced shown in Figure 2. The generated feature maps are concatenated in the completed product of a convolutional layer for many filters utilised for one input. A Laplacian filter is an edge detector that is used to estimate the derivative of an image by identifying the rate at which the first derivatives change.

This determines whether nearby pixels are adjusted in a continuous route or along an edge. Low characteristics are commonly seen in bridges or array centres. Angles may be both favourable and negative. The arithmetic mean supplied by Pooling layers is the largest value of elements in the part of the picture covered by the filtering. When it comes to detecting dominant characteristics, maximum pooling is regarded to be more effective.The more layers that are connected, the higher the degree of categorisation. The entering matrix is flattened to a row vector and then processed through a series of declared completely connected layers to accommodate a variety of output dimensions.

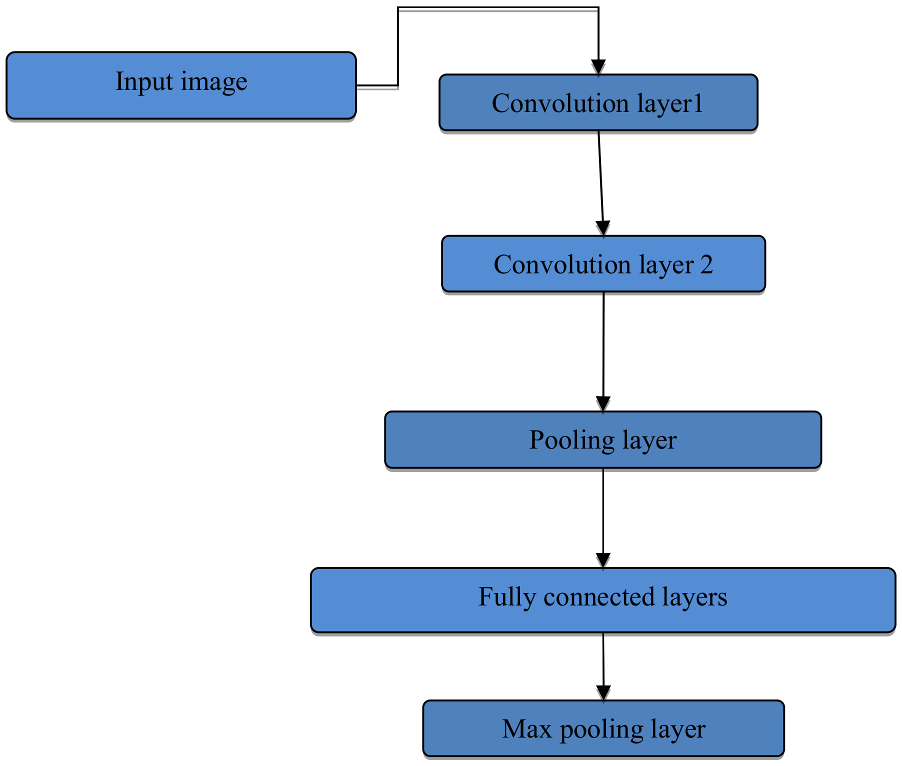

FIGURE 2. Proposed CNN Layers

RESULTS

After all of the predictions have been produced, the model uses the verification data set to evaluate itself. The product makes use of this tool to perform hyperparameter modifications. After the model has been properly trained, the dataset will be used to test it. The fitting problem occurs when a model includes unwanted variance in the dependent variable. It comes to mind automatically, which explains why the facts fit so well. If, on the other hand, the model doesn't precisely fit the facts, it might suggest a misunderstanding of the long-term pattern in the data.

On new datasets, bad estimates are found when both are matched. By presenting the number of exercises in a batch, the batch size may be inferred. As input, an image was merged with the Laplacian filter - The image's dimensions are as follows: (256X256). A 256X256 matrix is created by combining it with the Wavelet coefficients Filter. The Test Pattern is convolutioned with the Laplacian Filter. The size of the pattern to be discovered is (16X8). An 8X6 matrix is created by combining it with the Laplacian Filter. In the parallelizing version, identically numbered rows are supplied into the multiplication block, which generates six outputs, and addition is also pipelined. A overview of the CNN layer's LUTs usage is shown in Table1.

TABLE 1: Comparison with Existing Method

Parameter	*Existing method*	*Proposed method*
LUTs	4856	4226

This element of both the algorithm, which consists of simple arithmetic logic activities, is suitable for implementation on an FPGA as indicated in Table 1. These findings indicate that this acceleration is compatible with current FPGA and GPU technologies. These studies assessed the accelerator's field. A tweak like this might improve both the area and the energy efficiency by using the CNN Laplacian filter.

CONCLUSIONS

Object detection uses a technique to count the number of times a pattern occurs in the source photos. The recommended system can detect the design and the couple of times it occurs in the test picture using a test image and just a test pattern. The framework will be built using a Convolutional Neural Network (CNN). The convolution operation and the Pooling layer are two CNN layers. Object detection utilises a method to locate a recurring pattern in the input picture. Using the Laplacian filter in the training pictures, the system can easily distinguish the patterns and the few occurrences in which they appear based on the test image and the test pattern. This decreases the overall amount of multiplication necessary for convolution, i.e. the amount of hardware required. These findings indicate that this acceleration is compatible with current FPGA and GPU technologies. These studies assessed the accelerator's field. A tweak like this might improve both the area and the energy efficiency by using the CNN Laplacian filter.

REFERENCES

[1]. L. Chen, K. Preston, S. Manipatruni, and M. Lipson, 2009, "Integrated GHzsilicon photonic interconnect with micrometer-scale modulators and de-tectors,"*Opt.Express*,**17(17)**, pp.15248–15256.

[2]. S. Stathopoulos, A Khiat, M Trapatseli, S Cortese, A Serb, I Valov and T Prodromakis, 2017, "Multibit memory operation of metal-oxide bi-layer memristors,"*Sci.Rep.,*7(1), pp. 1-7

[3]. B. Xu, Y.Zhou, and Y.Chiu, 2017,"A23-mw24-gs/s6-bitvoltage-timehybrid time-interleaved ADC in 28-nm CMOS,"*IEEE J. Solid-State Circuits*,**52(4)**, pp.1091–1100.

[4]. M Ziebell, D Marris-Morini, G Rasigade, P Crozat, JM Fédéli, P Grosse, E Cassan, L Vivien, 2011, "Ten Gbit/sring resonator silicon modulator based on inter digitated PN junctions,"*Opt.Express*,**19(15)**, pp.14690–14695.

[5]. FY Gardes, A Brimont, P Sanchis, G Rasigade, D Marris-Morini, L O'Faolain, F Dong, JM Fedeli, P Dumon, L Vivien, TF Krauss, 2009, "High-speed modulation of a compact silicon ring resonator based on are verse-biased PN diode,"*Opt.Express*,**17(24)**, pp.21986–21991.

[6]. T Baba, S Akiyama, M Imai, N Hirayama, H Takahashi, Y Noguchi, T Horikawa, T Usuki, 2013,"50-Gb/s ring-resonator-based silicon modulator,"*Opt.Express*,**21(10)**, pp.11869–11876.

[7]. P Dong, S Liao, D Feng, H Liang, D Zheng, R Shafiiha, CC Kung, W Qian, G Li, X Zheng, AV Krishnamoorthy,2009,"Low V_pp,ultra low-energy,compact, high-speed silicon electro-optic modulator,"*Opt. Express*, **17(25)**, pp. 22484-22490.

[8]. N Kirubanandasarathy and KKM Rajmohan, 2010, Study and Survey about Mixed Radix FFT Processor in MIMO OFDM, *Int. J. of MC Square Sci. Res.*, **2**, pp. 47-52.

[9]. H Nakahara, H. Yonekawa, T Fujii, and S Sato, 2018, "A lightweight YOLOv2: A binarized CNN with a parallel support vector regression for an FPGA," In *Proc. of the 2018 ACM/SIGDA Int. Symposium on field-programmable gate arrays,* pp. 31-40.

[10]. Y. Ma, Y. Cao, S Vrudhula, and JS Seo, 2019, "Performance modeling for CNN inference accelerators on FPGA," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, ***39*(4)**, pp.843-856.

[11]. S Liu, H Fan, X Niu, HC Ng, Y Chu, and W Luk, 2018, "Optimizing CNN-based segmentation with deeply customized convolutional and deconvolutional architectures on FPGA," *ACM Trans. on Reconfigurable Tech. and Systems (TRETS)*, ***11*(3)**, pp.1-22.

[12]. M Ali, PA Rad, and D Göhringer, 2020, "April. RISC-V based MPSoC design exploration for FPGAs: area, power and performance," *Int. Symposium on Applied Reconfigurable Computing,* pp. 193-207.

[13]. M A Manivasagam, 2017, "An Efficient Self-Reconfiguration and Route Selection for Wireless Sensor Networks." *Int. J. of MC Square Sci. Res.* **9(2)**, pp. 192-199.

[14]. N Jeebaratnam, SS Nayak, and N Patra, 2020. FPGA Design for Implementation of the 2 D Discrete Cosine Transformation for Image Processing, *Test Eng. and Management*, pp. 17220 – 17224.

[15]. S Murugan, S. Mohan Kumar, and T.R. Ganesh Babu, 2020, "Convolutional Neural Network-based MRI Brain tumor classification system," *Int. J. of MC Square Scientific Res.* **12(3),** pp. 1-8.

[16]. MM Ismail, M Subbiah and S Chelliah, 2018, "Design of pipelined radix-2, 4 and 8 based multipath delay commutator (MDC) FFT, *Indian J. of Public Health Res. and Development*, **9(3)**, pp. 765–768.

[17]. V Jaiganesh and S. Murugan, 2005, "PC based heart rate monitor implemented in xilinx fpga and analysing the heart rate,"*Proc. of the Third IASTED Int. Conf. on Circuits, Signals, and Systems, CSS 2005*, pp. 319–323.

[18]. R. Nusullapalli and N. Vaishnavi, 2018, "Design of FFT Architecture Using Kogge Stone Adder," *Int. J. Adv. Sig. Img. Sci*, **4(2)**, pp. 8–15.

[19]. V. Ellapan and J. S. Alaric, 2019, "A Parallel and Pipelined Architecture for Cordic Algorithm," *Int. J. Adv. Sig. Img. Sci*, **5(2)**, pp. 23–31.