ICIC Express Letters
Volume **16**, Number **9**, September **2022**

ICIC International ©2022 ISSN 1881-803X
pp. 951–955

APPROXIMATE MULTIPLIER DESIGNS FOR DEEP NEURAL NETWORK ACCELERATORS

Raghuram Srinivasan*, Shashank Narayanaswamy, Rahul Bhat
Vandana P Manjunath and Reshma Bhat

Department of Electronics and Communication Engineering
M. S. Ramaiah Institute of Technology
MSRIT Post, Bengaluru 560054, India
{ 1ms17ec101; 1ms17ec083; 1ms17ec122; 1ms17ec086 }@msrit.edu
*Corresponding author: raghuram@msrit.edu

Received October 2021; accepted January 2022

Abstract. *With the rapid evolution of Deep Neural Networks (DNN) models for various classification tasks across diverse domains, hardware accelerators for these models provide additional advantages: they improve the efficiency of the classification process, while also increasing the penetration of this revolutionary technology. However, the inherently complex nature of DNN architectures as they are non von Neumann type architectures and require multiple computing units operating in parallel, poses serious implementation challenges for accelerators. This paper investigates the feasibility of using approximate computing as an approach to reducing the hardware complexity of accelerator implementations. Approximate multipliers and adders are used to realize a floating-point Multiply and Accumulate (MAC) unit. A novel approximate multiplier design, following the Karatsuba algorithm, has been proposed which outperforms the state-of-the-art approximate designs by 40%, 23% and 20% improvement in power, delay, and area, respectively. The proposed design with the approximate multipliers and approximate adders was tested in three popular CNN architectures – LeNet, VGG and ResNet. It is shown that our approximate design produces the best results in terms of accuracy metrics in all these three DNN implementations.*

Keywords: Hardware accelerators, Approximate computing, Karatsuba algorithm, Deep neural networks

1. **Introduction.**

Approximate computing can be used in applications that are known to be error tolerant, and the results in [1] show that machine learning algorithms fall in this class. The approximation allows for certain optimizations in the system, thereby improving overall performance characteristics. In this work, the focus is on approximate computing units, which are in turn used to build a floating point MAC unit. An approximate multiplier design, based on the Karatsuba algorithm, is proposed and it is shown that the design has better power and delay metrics as compared to existing approximate designs. Along with performance analysis, the error analysis of approximate designs was also carried out. The standard error metrics like Error Rate (ER), Normalized Mean Error Distance (NMED), and Mean Relative Error Distance (MRED) of our design and published approximate designs are compared. Finally, these approximate units are used to realize popular DNN architectures, and their accuracy degradation is compared.

Compressors are popular options for implementing approximate multipliers. In [2], the authors have presented an array multiplier design which uses approximate compressors to generate fewer partial products. Varied levels of accuracy are obtained by using different combinations of approximate compressors in the reduction stages. Among all the discussed designs in the paper, 1-step full (ACM-1F), 1-step truncation (ACM-1T) and

DOI: 10.24507/icicel.16.09.951

2-step full (ACM-2F) designs provide the best trade-offs for an 8-bit multiplier. The designs are tested on two applications: Gaussian smoothing image filtering and adaptive LMS filtering. The best results obtained for the smoothing image filtering are a reduction of 77% in power savings with a 8% degradation of the image quality. In [3], an approximate adder is designed, to realize the summation of the partial products, thereby making the multiplier also an approximate design. In comparison with exact Wallace tree based architectures, the authors have achieved a reduction in power and timing metrics for their approximate designs.

In the next section, we discuss our approximate multiplier design, following which we elaborate on the experimental setup to test DNNs with our design. Then, we present results of the implementation and its accuracy in DNN architectures. Finally, we summarize the paper and extend future work.

2. Karatsuba Approximate Multiplier Design.

The Karatsuba algorithm uses a divide and conquer approach making it faster than a classical multiplier [4]. For an 8-bit design, the algorithm implementation also gives an improvement in terms of area when compared with other multiplication algorithms [5]. x_H and y_H are the upper $n/2$ bits of x and y respectively and x_L and y_L are the lower $n/2$ bits of x and y respectively, with

$$x = x_H 2^{n/2} + x_L$$
$$y = y_H 2^{n/2} + y_L$$

a, b, and c are as follows:

$$a = x_H y_H$$
$$b = x_L y_L$$
$$c = (x_H y_L + x_L y_H) = (x_H + x_L)(y_H + y_L) - a - b$$

Using these, the exact Karatsuba multiplication may be realized as in (1)

$$xy = a2^n + c2^{n/2} + b \tag{1}$$

The approximation proposed in this work is to drop the term b from (1): the term b accounts for the lower bits of the final product. Hence, the margin of error introduced will be small. This approximation will completely eliminate one $n/2$ bit multiplication and decrease the number of full-adders required, thereby improving the performance. The approximate Karatsuba operation will be as in (2). The reduction in the number of full adders and the elimination of one partial product is seen in Figures 1 and 2.

$$xy = a2^n + c2^{n/2} \tag{2}$$

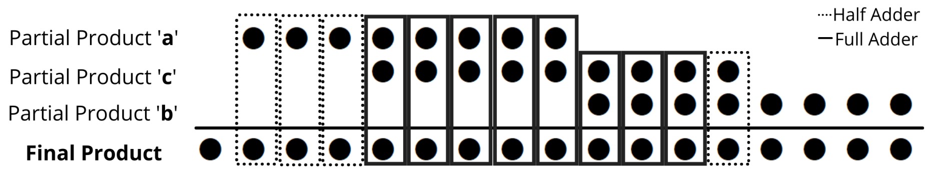

FIGURE 1. Accurate Karatsuba design

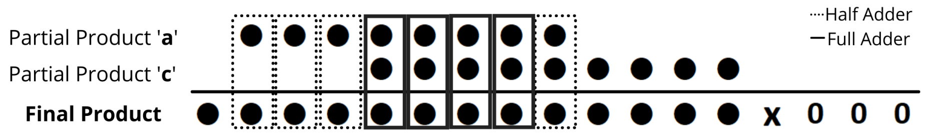

FIGURE 2. Approximate Karatsuba design

3. **Experimental Setup.** MAC units with the exact and approximate computing units have been realized in Verilog and the synthesis reports are generated using a generic 90 nm technology and the Cadence design tools. The MAC units are then realized in Python, for the approximate designs. Next, Python models of the DNNs are created, using these exact and approximate MAC units. Hence, the exact and approximate units are indirectly used to realize the DNN functionality. In this way, the level of accuracy of the DNN, when using the approximate units, was evaluated. Finally, since the focus is on investigating the effect of the approximate units, rather than the overall DNN accuracy, it is assumed that the result of using the exact floating point unit is always correct, and any deviation from this classification is erroneous.

4. Results & Discussion.

4.1. **Error analysis.** Standardized parameters such as Error Rate (ER), Normalized Mean Error Distance (NMED), and Mean Relative Error Distance (MRED) are used to measure the quality of the approximate design [6]. Table 1 compares our design, the Karatsuba Approximate Multuplier (KAP), with the best performing designs in [2] – ACM-1F, ACM-2F, and ACM-1T, and the 11-bit error recovery (LPER-11) and 4-bit error recovery (LPER-4) from [3]. Our KAP design has the second highest error rate, only ACM-1T [2] being higher. ACM-1T [2], ACM-2F [2] and LPER-4 [3] have a much higher MRED compared to our design, meaning that the amount by which it is erroneous is higher. This could be catastrophic for application like DNN accelerators, where the error due to many levels of sequential computations produces drastically different results. While LPER-11 [3] and ACM-1F [2] have a lower MRED compared to our design, their NMED is higher. Among all the designs, only ACM-1F has better properties. However, when digital implementations are considered, it is seen that our design has better performance properties.

Table 1. Error metrics comparison

Design	ER	MRED	NMED
LPER-4 [3]	81.06	10.3	0.0185
LPER-11 [3]	31.59	0.6	0.0020
ACM-1F [2]	19.90	0.2	0.0003
ACM-2F [2]	48.67	1.9	0.0054
ACM-1T [2]	96.50	5.2	0.0032
KAP (Our design)	86.33	1.5	0.0007

4.2. **Performance metrics of the KAP approximate multiplier.** In this section, the performance metrics of digital implementations of the approximate designs is compared in Table 2. The first row for each design gives the actual values, and the second row shows the comparison with the exact counterparts. LPER-4 [3] and ACM-1T [2] have the best performance metrics compared to the exact multipliers; however, these designs have a relatively high MRED and NMED, leading to increased error compared to our KAP design, as is shown in the next section.

4.3. **Approximate units in DNN accelerators.** The classification results of DNNs when approximate units are used in the MAC unit are discussed in this section. For this study, three popular image classification/recognition DNNs, LeNet [7], VGG [6] and ResNetv1 [8], were trained with MNIST [7], Fashion MNIST [9] and CIFAR-10 [8] datasets, respectively. The DNNs are trained offline and with those weights, 1000 images from the test set are used for comparing the performance with the exact and approximate

Table 2. Performance comparison

Design	Area (um^2)	Power (nW)	Delay (ps)
Exact	1154	47102.45	2836
LPER-4 [3]	727	14506.18	1085
	−37%	−69%	−61%
LPER-11 [3]	1074	29477.81	2504
	−7%	−37%	−12%
ACM-1F [2]	940	39565.65	2381
	−18%	−16%	−16%
ACM-2F [2]	801	24281.11	2610
	−30%	−48%	−8%
ACM-1T [2]	663	18190.30	2158
	−42%	−61%	−24%
KAP (Our design)	919	28093.25	2170
	−20%	−40%	−23%

Table 3. DNN architecture details

	Number of layers			Total MAC units
Network	Convolution	Fully connected	Total	
LeNet	2	3	5	446.5K
VGG	4	2	6	19.9M
ResNetV1	15	1	16	31.3M

Table 4. DNN accuracy with approximate units

Design	LeNet	VGG	ResNetv1
LPER-04 [3]	97.8	95	78
LPER-11 [3]	100.0	100	97
ACM-1F [2]	99.9	100	99
ACM-2F [2]	99.4	98	90
ACM-1T [2]	100.0	99	92
KAP (Our design)	100.0	100	99

computing units. Table 3 shows the total layers and the MAC units in each of the DNNs. While LeNet and VGG have a similar number of layers, the input size and the number of filters are much higher in the more modern VGG architectures, thereby leading to a large increase in the number of MAC units.

Table 4 compares the accuracy of the classification results when the approximate units are used in the DNN implementations. In the MAC unit implementation, among the approximate designs for the adders and multipliers, only one of them is used at a time, to analyze the individual effect of the approximation. We see that our design has the highest levels of accuracy, across all the three DNNs, with a slight drop of 1% for the largest of the architectures. While ACM-1F [2] has comparable accuracy, the digital implementation reduces the power and delay by 16%, while our design reduces the same metrics by 40% and 23% respectively, making it a better option for accelerator implementation.

5. **Conclusion.** In this paper, a novel design for an approximate multiplier, based on the Karatsuba algorithm, is presented. It is shown that the design is appropriate for use in DNN accelerators. To extend this study, the use of these approximate units in reducing overfitting in DNN predictions can be investigated, thereby increasing the overall quality of the classification result.

REFERENCES

[1] Z. Du, K. Palem, A. Lingamneni, O. Temam, Y. Chen and C. Wu, Leveraging the error resilience of machine-learning applications for designing highly energy efficient accelerators, *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp.201-206, 2014.

[2] D. Esposito, A. G. M. Strollo, E. Napoli, D. De Caro and N. Petra, Approximate multipliers based on new approximate compressors, *IEEE Trans. Circuits and Systems I: Regular Papers*, vol.65, no.12, pp.4169-4182, 2018.

[3] C. Liu, J. Han and F. Lombardi, A low-power, high-performance approximate multiplier with configurable partial error recovery, *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp.1-4, 2014.

[4] P. L. Montgomery, Five, six, and seven-term Karatsuba-like formulae, *IEEE Trans. Computers*, vol.54, no.3, pp.362-369, 2005.

[5] F. W. Wibowo, Comparison of multiplication algorithms based on FPGA, *2018 2nd Borneo International Conference on Applied Mathematics and Engineering (BICAME)*, pp.326-331, 2018.

[6] M. Masadeh, O. Hasan and S. Tahar, Comparative study of approximate multipliers, *Proc. of the 2018 on Great Lakes Symposium on VLSI*, pp.415-418, 2018.

[7] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, vol.86, no.11, pp.2278-2324, 1998.

[8] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *arXiv Preprint*, arXiv: 1409.1556, 2015.

[9] H. Xiao, K. Rasul and R. Vollgraf, Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, *arXiv Preprint*, arXiv: 1708.07747, 2017.