IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 23, 2018, accepted October 7, 2018, date of publication October 10, 2018, date of current version November 8, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2875376

Impact of Approximate Multipliers on VGG Deep Learning Network

ISSAM HAMMAD, (Student Member, IEEE), AND KAMAL EL-SANKARY, (Member, IEEE)
Department of Electrical and Computer Engineering, Dalhousie University, Halifax, NS B3J 1Z1, Canada
Corresponding author: Issam Hammad (issam.hammad@dal.ca)

ABSTRACT This paper presents a study on the applicability of using approximate multipliers to enhance the performance of the VGGNet deep learning network. Approximate multipliers are known to have reduced power, area, and delay with the cost of an inaccuracy in output. Improving the performance of the VGGNet in terms of power, area, and speed can be achieved by replacing exact multipliers with approximate multipliers as demonstrated in this paper. The simulation results show that approximate multiplication has a very little impact on the accuracy of VGGNet. However, using approximate multipliers can achieve significant performance gains. The simulation was completed using different generated error matrices that mimic the inaccuracy that approximate multipliers introduce to the data. The impact of various ranges of the mean relative error and the standard deviation was tested. The well-known data sets CIFAR-10 and CIFAR-100 were used for testing the network's classification accuracy. The impact on the accuracy was assessed by simulating approximate multiplication in all the layers in the first set of tests, and in selective layers in the second set of tests. Using approximate multipliers in all the layers leads to very little impact on the network's accuracy. In addition, an alternative approach is to use a hybrid of exact and approximate multipliers. In the hybrid approach, 39.14% of the deeper layer's multiplications can be approximate while having a reduced negligible impact on the network's accuracy.

INDEX TERMS AI accelerator, approximate computing, approximate multiplier, CNN, deep convolutional network, deep learning, VGGNet.

I. INTRODUCTION

Deep learning using convolutional neural networks (CNNs) has gained increased momentum in recent years. Image classification is one of the primary applications for deep learning using CNN. Several successful CNN architectures were proposed in the literature. One of the most used architectures is the VGGNet proposed by Simonyan and Zisserman [1]. One of the widely used VGG configurations is the VGGNet-16 (referred to as configuration D in [1]), which consists of 13 convolutional layers, and 3 fully connected layers with max pooling applied between the layers. VGGNet has a uniform architecture and uses 3×3 convolutions. Figure (1) depicts the network architecture as proposed in [1] including the number of channels in each stage. According to [1], the VGGNet-16 has 138 million parameters. Since the convolution in a CNN is completed via multiplication and addition, any improvement on the performance or the cost of multiplication will have a significant impact on the overall performance and cost of the entire network. According to [12], the VGGNet-16 required 15.5G multiply-and-accumulates (MACs) operations to complete the classification of one image.

The concept of approximate computing has emerged in recent years to increase systems performance and power efficiency. Usage of approximate computing is promoted for media related systems due to its ability to tolerate error. One of the applications of approximate computing is the approximate multiplier. Compared to an exact multiplier, the approximate multiplier has a reduced power, area, and delay. However, it has a cost of inaccuracy which is usually defined by metrics such as the MRE and the SD.

The research objective is to demonstrate that approximate multipliers can be used to optimize the performance of VGGNet in terms of power, area, and delay. This work assesses the impact of various approximate multiplication error ranges on the VGGNet accuracy and identifies the network layers that are the least impacted by approximate multiplication. Additionally, the research provides a baseline for researchers to apply the proposed simulation methods to

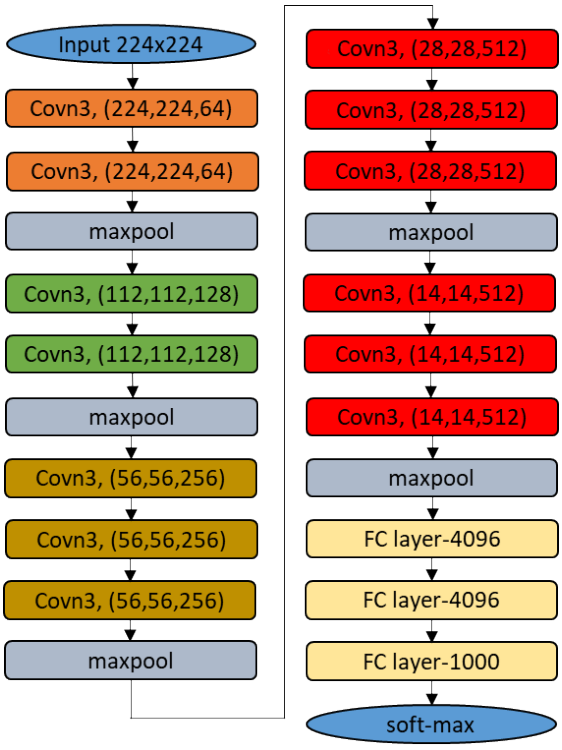

FIGURE 1. VGGNET-16 as proposed by [1].

explore the impact of approximate computing on various deep learning architectures.

VGGNet was selected for this study as it is one of the most popular deep learning architectures for image classification. Additionally, it has a very uniform architecture with 3×3 convolutions and with a modest number of layers comparing to other popular architectures. These features enable a homogeneous evaluation of the impact of approximate multiplication on the network layers.

The focus of this research is on the optimization of pre-trained networks which exclude the training phase. Several systems rely on pre-trained weights especially in low power applications. For these systems the training is done on a central server, then the weights are downloaded to many client devices which usually have limited hardware resources. One example is the internet of things (IoT) hardware accelerator presented in [13] which uses pre-trained weights.

In this paper, a simulation for the impact of approximate multipliers on the accuracy of VGGNet is presented. The simulation included testing the impact of approximate multiplication on all the layers in the first set of tests and on selective layers in the second set of tests. The simulation results of both approaches show that approximate multipliers can be used to achieve significant performance gains with a very minimal cost of added accuracy error.

This paper is organized as follows: Section II provides a background on the approximate multiplier. Section III demonstrates the concept of approximate multipliers simulation by adding error matrices to the network's layers. Section IV presents the results of simulating approximate multiplication in all VGGNet layers. Section V focuses on the impact of applying approximate multiplication in selective network layers and proposes the hybrid approach. Section VI summarizes the research conclusion.

II. BACKGROUND ON THE APPROXIMATE MULTIPLIER

Approximate multipliers can be utilized in applications that are tolerant to inaccuracy. Several different approximate multiplier designs were recently proposed as a replacement for the exact multiplier such as [2]–[6] and [14]. Approximate multipliers can lower the design cost in terms of power, delay and chip size, with the cost of having a certain calculation error. The mean relative error (MRE) is used along with other metrics such as the SD to assess inaccuracy of approximate multipliers. The MRE is the primary common metric used to evaluate the error in approximate multipliers, this can be found in the approximate multiplier publications [2]–[6] and [14]. The MRE is defined as:

$$MRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|Yi' - Yi|}{|Yi|} \tag{1}$$

In equation (1), Yi is the exact value while Yi' is the approximate value. One example of an approximate multiplier is the 16-bit design proposed by Venkatachalam and Ko [2]. The multiplier in [2] has achieved power, area and delay savings of 72%, 56%, and 31% respectively while introducing an MRE of 7.6%. Another example is the approximate multiplier in [5] which introduces an MRE of 1.47% while having a reduction of 59%, 50%, and 47% in the power, area, and delay. Several other implementations for the approximate multiplier can be found in [3], [4], [6], [7], and [14]. The design in [7] showed that a decreased area and power in an approximate multiplier can be achieved by introducing a higher error. Therefore, this allows for a flexibility in the design by balancing the trade-off between the accuracy loss and the achieved savings in power and area.

III. APPROXIMATE MULTIPLIER SIMULATION

This section presents the concept of approximate multiplier error simulation for deep learning networks. To test the impact of approximate multipliers on the accuracy of the network, the multiplication accuracy should contain a certain MRE that an approximate multiplier would have introduced if it was used instead of an exact multiplier. To test the impact of approximate multiplication on the accuracy of the VGGNet, a pre-trained VGGNet was used [8]. This network was built using the python based deep learning platform Keras [9]. This network was modified to add an error matrix prior to completing the convolution at certain layers of the network, depending on whether they are part of the test case or not. The error matrix was applied through element-wise multiplication with the layer input Y. The error matrix was also applied to the fully connected layers in some test cases. In equation (2),

$Y'$is the modified layer input after applying a certain MRE.

$$Y' = Y \odot E \tag{2}$$

Where $\odot$ is element-wise multiplication operator. The matrix E is tuned to apply the required MRE value. For example, to simulate an MRE value of 2.5% using a Uniform probability density function (PDF), the matrix E will contain random values ranging from (0.95-1.05). To simulate the impact of approximate multipliers on the network's accuracy. Several test cases for Uniform and Gaussian PDFs error matrices were applied. These test cases are generic to cover the characteristics of various approximate multipliers in the literature. The MRE and the PDFs of these simulated test cases can be mapped to these published approximate multipliers designs as demonstrated in the next section. Figure (2), shows the histogram of a sample of Uniform and Gaussian error matrices used for one test image in the first layer of the network.

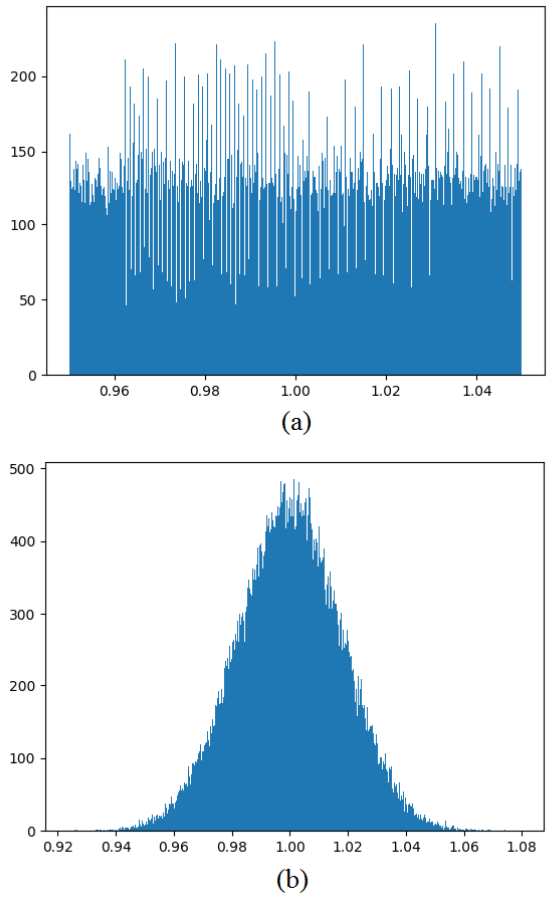

FIGURE 2. A histogram for an error matrix using 500 bins. (a) a sample Uniform error matrix (MRE=∼2.5%). (b) a sample Gaussian error matrix (MRE=∼1.4%).

In this simulation, each test case error range was divided into 10000 unique values. The error matrix E was constructed randomly from these values using Uniform or Gaussian PDFs. Each error matrix for a layer in a test case was generated using a different seed. All the tests were executed using 'float16' precision.

To test the impact on the accuracy, CIFAR-10 and CIFAR-100 datasets were used. CIFAR-10 dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. CIFAR-10 has 50000 training images and 10000 test images [11]. CIFAR-100 has 100 classes containing 600 images each divided as 500 training images and 100 testing images per class. As CIFAR-10 and CIFAR-100 datasets are used, the network architecture in this paper is slightly different than the original VGGNet-16 as per [1]. The network architecture contains changes that were proposed in [10] which handled the design of VGGNet for CIFAR-10 and CIFAR-100 datasets. The network design in [10] is tailored for inputs with size 32 × 32 instead of 224 × 224, consequently, the dimensions of this network's layers are smaller than the original VGGNet as proposed in [1]. The network changes also include using 2 fully connected layers instead of 3 fully connected layers, and changes in other settings such as batch normalization and dropout to reduce overfitting [10]. Figure (3) demonstrates the architecture of this modified VGGNet that is used for the simulation. As the focus of this simulation is to assess the impact of approximate multipliers on a pre-trained network, the error matrices were applied to the test images only.

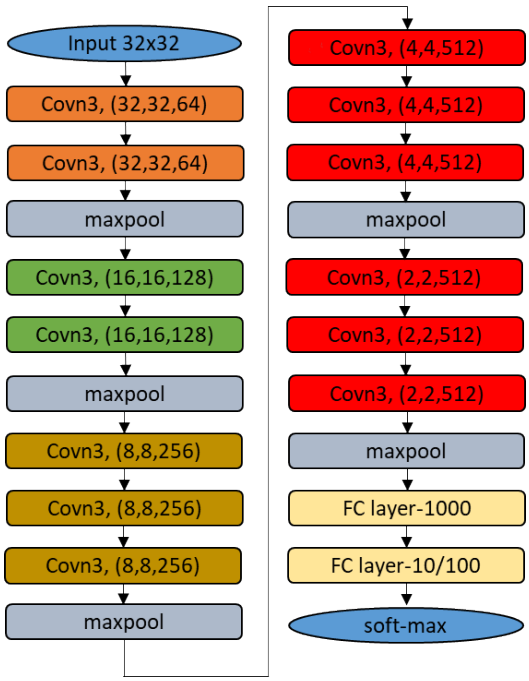

FIGURE 3. Modified VGGNET as per [10].

To evaluate the impact of approximate multipliers, two set of tests were executed. The first set of tests are the "MRE Tests" while the second tests are "Layer Impact Tests". In the "MRE Tests", the error matrix was added to each layer of the network. In each test case, a specific MRE value was simulated to assess its impact on the network accuracy. In the "Layer Impact Tests", the error matrix was applied only to a selective group of layers in each test case. This was done to

evaluate the impact of having approximate multiplication in these particular layers. The next two sections will discuss the simulation results for both the ''MRE Tests'' and the ''Layer Impact Tests''. During the simulation, each test case consisted of 100 loops, in each loop the entire dataset (CIFAR-10 or CIFAR-100) was applied. In each test case, an error matrix with approximately the same MRE and PDF was applied to the tested layers of the network. However, the error matrices had different seeds in each layer of every loop. The tables in this paper list the average network error of the 100 loops. Additionally, due to having different random seeds for the error matrices the listed MRE and SD values in all the tables are approximate.

IV. MRE TESTS

Several Uniform and Gaussian error matrices with different MRE values were simulated to assess their overall impact on the network's accuracy. Table 1 shows the results of the simulation using CIFAR-10 dataset while Table 2 shows the results of the simulation using CIFAR-100 dataset. For each test case, Table 1 and Table 2 list the MRE, the SD, the network error rate, and the error difference compared to an exact multiplier. The network error rate in Table 1 and Table 2 refers to the percentage of incorrect image classifications by the network. The error difference reflects the additional error which resulted from using an approximate multiplier instead of an exact multiplier. In the used architecture for this simulation an execution with an exact multiplier using CIFAR-10 dataset results in a network error rate of 6.4%, while the error is 29.51% for CIFAR-100 dataset. These numbers will be used as a baseline to assess the impact of the approximate multiplier. The network error rate can have minor variations based on the training settings and the used hyperparameters. According to [10], the human rater for CIFAR-10 is 6%, therefore, the used baseline rate is very close to the human classification error rate.

As can be seen from Table 1 and Table 2, the error difference resulting from the added MRE is very minimal and can be considered negligible especially for lower MRE cases. Similar or exceeding error differences can be a result of a tweak to one of the hyperparameters during network training. Figure (4) illustrates the approximate relation between the increase in MRE and the increase in the error difference. While all tests in Tables 1 and 2 were executed using float-16 data type, float-32 data type was also simulated using a subset of tests from Table 1 and Table 2. Using float-32 gave very similar results to float-16, therefore, these simulation results are applicable to both 16 bits and 32 bits approximate multipliers.

The error matrices used in Table 1 and Table 2 are generic to cover a broad spectrum of possible errors resulting from the usage of approximate multipliers. However, these test cases can be mapped to the reported performances of proposed approximate multipliers in the literature as Table 3 presents. Table 3 lists the reported performance enhancements for various approximate multipliers in comparison to exact multipliers. By mapping the simulation results from Table 1 and Table 2 to the approximate multipliers performance result in Table 3, the advantage of using approximate multipliers in VGGNet can be clearly seen. An example is the approximate multiplier DRUM [5], this multiplier has a Gaussian error with MRE of 1.47% and SD of 1.803%. By replacing the VGG's exact multipliers by DRUM's approximate multipliers, the multiplication cost can have approximate savings in power, area, and delay of 59%, 50%, and 47% respectively. One CIFAR image classification using the modified VGGNet as per [10] requires 313.41M MACs. According to the Gaussian simulation results in Table 1 and Table 2, this replacement will cause an approximate additional network error of only 0.038% using CIFAR-10 and 0.064% using CIFAR-100.

TABLE 1. Test results using CIFAR-10.

Matrix E Type	MRE	SD (σ)	Network Error Rate	Error Diff. from Exact Multiplier
No Error	0%	N/A	6.4%	0%
Uniform	~0.75%	~0.087%	6.422%	+0.012%
Uniform	~1.25%	~1.443%	6.43%	+0.03%
Uniform	~2.5%	~2.887%	6.495%	+0.095%
Uniform	~5%	~5.774%	6.598%	+0.198%
Uniform	~7.5%	~8.66%	6.828%	+0.428%
Uniform	~10%	~11.55%	7.358%	+0.958%
Gaussian	~0.6%	~0.75%	6.423%	+0.023%
Gaussian	~1.4%	~1.80%	6.438%	+0.038%
Gaussian	~2.4%	~3.0%	6.477%	+0.077%
Gaussian	~3.6%	~4.5%	6.53%	+0.13%
Gaussian	~4.8%	~6.0%	6.599%	+0.199%

TABLE 2. Test results using CIFAR-100.

Matrix E Type	MRE	SD (σ)	Network Error Rate	Error Diff. from Exact Multiplier
No Error	0%	N/A	29.51%	0%
Uniform	~0.75%	~0.087%	29.556%	+0.046%
Uniform	~1.25%	~1.443%	29.563%	+0.053%
Uniform	~2.5%	~2.887%	29.580%	+0.070%
Uniform	~5%	~5.774%	29.779%	+0.269%
Uniform	~7.5%	~8.66%	30.206%	+0.696%
Uniform	~10%	~11.55%	30.937%	+1.427%
Gaussian	~0.6%	~0.75%	29.563%	+0.053%
Gaussian	~1.4%	~1.80%	29.574%	+0.064%
Gaussian	~2.4%	~3.0%	29.626%	+0.116%
Gaussian	~3.6%	~4.5%	29.718%	+0.208%
Gaussian	~4.8%	~6.0%	29.85%	+0.34%

TABLE 3. Reported performance of approximate multipliers in the literature compared to exact multipliers.

Design	MRE	Power Savings	Area Reduction	Delay Decrease
DRUM [5]	1.47% (G)	59%	50%	47%
Vasileios [3]	3.6% (G)	34.14%	34.17%	11.11%
Suganthi [2]	7.63%	71.7%	55.6%	30.9%
Georgios [6]	2.5% (U)	47%	38%	35%
Tongxin [14]	1.64%	59.9%	50.1%	36.3%

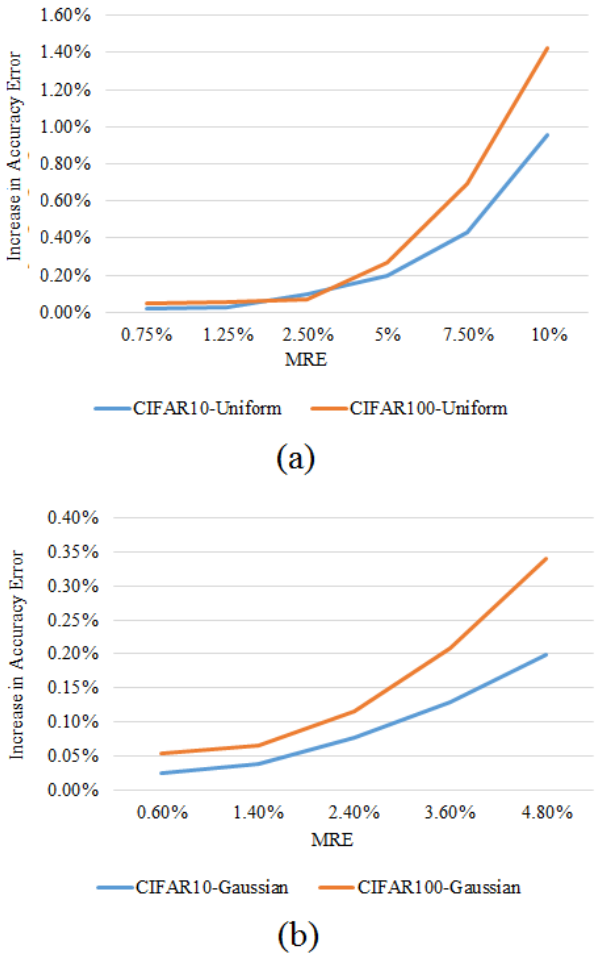

FIGURE 4. An estimated relationship between the MRE of approximate multipliers and the additional error in the accuracy of VGGNet. (a) Uniform MRE impact. (b) Gaussian MRE impact.

This is based on the closest simulation test case with MRE of 1.4% and SD of 1.8%. This additional network error is minimal considering the achieved reduction in power and area and the significant increase in speed. The Speed increase is very critical for deep learning applications, researchers are vigorously trying to speed up deep learning training and testing. More examples can be seen by mapping the performance enhancements of the approximate multipliers in [2], [3], [6], and [14] to the closest simulated test in Table 1 and Table 2. Note that (G) and (U) in Table 3 refers to Gaussian and Uniform distribution, respectively.

The approximate multipliers listed in Table 3 proposes different design methods on the hardware level for the approximate multiplication. For example, the design [3] uses a method of approximation that is performed by rounding the high radix values to their nearest power of two. The design [5] uses a different method which is applied by dynamic range selection scheme and truncation. These different design methods lead to different PDF characteristics for the MRE as specified in Table 3. The simulations presented

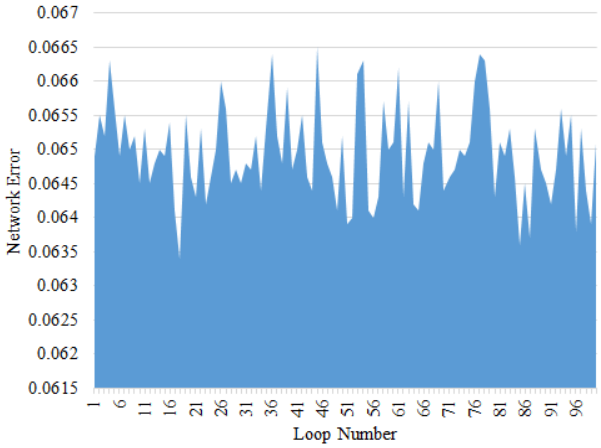

FIGURE 5. Accuracy results for 100 loops of simulation for the Uniform PDF with MRE=∼2.5%.

in this paper are generic as presented in Table 1 and Table 2. These simulation results can be used for approximate mapping between an approximate multiplier's MRE error and the impact on the accuracy of the VGGNet.

Using 8-bits and 12-bits approximate multipliers, [7] has shown that a decrease in the required power and area can be achieved by increasing the error. For example, in the proposed 12-bit approximate multiplier, an increase in the maximum relative error from 1% to 2% has dropped the power from 475 μW to 284 μW and the area from 720.8 μm to 523.8 μm. Also, the power has dropped from 247 μW to 125 μW and the area from 483 μm to 285 μm by increasing the error from 5% to 10%.

As mentioned earlier, Table 1 and Table 2 test cases list the network accuracy by averaging the simulation results of 100 loops. Figure (5) shows the network error for 100 loops of simulation for the Uniform PDF with MRE=∼2.5% test case. The figure shows the loop number and the corresponding network accuracy. Interestingly, some of the loops achieved better results compared to an exact multiplier. The applied error matrices in these cases have randomly reshaped the layer's input for a better classification by the network.

V. LAYER IMPACT TESTS

In this section, the impact of applying approximate multiplication on specific layers is evaluated. Table 4 details the number of parameters per layer, the number of MACs in each layer and their percentage of the total number of MACs in the network. Table 4 information is based on one CIFAR-10 image.

To apply the layer impact testing, several test cases were simulated, where, consecutive layers were grouped together in each test case. A total of 6 sets of layer groups were used to assess the impact of approximate multiplication on these segments of the network. Table 5 illustrates the details of the applied test cases. For all the test cases CIFAR-10 was used with a uniform error matrix (MRE =∼7.5%). The error matrix was applied on the specified layers only.

TABLE 4. Number of MACs per layer.

Layer	Param Count	MACs per Image	(%) of MACs.	Output Size per Image
Conv-64-1	1.79k	1.77M	0.564%	(32,32,64)
Conv-64-2	36.93k	37.75M	12.042%	(32,32,64)
Conv-128-1	73.86k	18.87M	6.021%	(16,16,128)
Conv-128-2	147.58k	37.75M	12.042%	(16,16,128)
Conv-256-1	295.17k	18.87M	6.021%	(8,8,256)
Conv-256-2	590.08k	37.75M	12.042%	(8,8,256)
Conv-256-3	590.08k	37.75M	12.042%	(8,8,256)
Conv-512-1	1.18M	18.87M	6.021%	(4,4,512)
Conv-512-2	2.36M	37.75M	12.042%	(4,4,512)
Conv-512-3	2.36M	37.75M	12.042%	(4,4,512)
Conv-512-4	2.36M	9.44M	3.011%	(2,2,512)
Conv-512-5	2.36M	9.44M	3.011%	(2,2,512)
Conv-512-6	2.36M	9.44M	3.011%	(2,2,512)
FC-1	262.66k	262.14k	0.084%	(512)
FC-2	5.13k	5.12k	0.002%	(10)
Total	15M	313.46M	100%	N/A

TABLE 5. Layers testing results using Uniform error matrix (MRE=∼7.5%).

Group ID	Error Matrix Location	Network Err. Rate	Error Diff. from Exact Multiplier
L64	The 2 conv-64 layers	6.715%	+0.315%
L128	The 2 conv-128 layers	6.484%	+0.084%
L256	The 3 conv-256 layers	6.513%	+0.113%
L512-1	The first 3 conv-512 layers	6.445%	+0.045%
L512-2	The second 3 conv-512 layers	6.42%	+0.02%
LFC	The fully-connected layers	6.612%	+0.212%

TABLE 6. Tests results for the hybrid approach.

Matrix E Type	MRE	Network Error Rate	Error Diff. from Exact Multiplier	Err. Diff from All Layer Appr. Multiplier
Uniform	∼1.25%	6.414%	+0.014%	-0.016%
Uniform	∼2.5%	6.419%	+0.019%	-0.076%
Uniform	∼5%	6.437%	+0.037%	-0.161%
Uniform	∼7.5%	6.488%	+0.088%	-0.340%
Uniform	∼10%	6.502%	+0.102%	-0.856%
Gaussian	∼1.4%	6.417%	+0.017%	-0.021%
Gaussian	∼2.4%	6.420%	+0.020%	-0.057%
Gaussian	∼3.6%	6.432%	+0.032%	-0.098%
Gaussian	∼4.8%	6.436%	+0.036%	-0.163%

As can be seen from Table 5, having an approximate multiplier on groups "L512-1" and "L512-2" have the least impact on the overall accuracy. These layers count for 39.14% of the total multiplications required as per Table 4. Therefore, a hybrid approach can be used in which the approximate multiplication is used in only these deeper layers.

The hybrid approach simulation results are presented in Table 6. In this approach, the approximate multiplication was applied only on the 512 channel layers of the network. A subset of tests from Table 1 were repeated in Table 6 to test this hybrid approach. Table 6 lists the reduction in network error achieved compared to the "all layers" approach in the previous section as per Table 1.

As can be seen from Table 6, the hybrid approach has a negligible impact on the accuracy compared to an exact multiplier, for the case of Gaussian PDF with MRE=∼1.4% which is similar to the approximate multiplier proposed in [5], the added network error rate is only 0.017%. Additionally, this approach has a reduced additional network error compared to the "all layers" approximate multiplier approach which was presented in the previous section. In summary, using an approximate multiplier similar to [5], 39.14% of the network's MACs can have approximately half the power, area and delay with the cost of 0.017% of an added network error.

VI. CONCLUSION

This research work demonstrates that deep learning can be optimized using approximate computing. Approximate computing can reduce the chip power and area while increasing the speed. The paper started by providing a background on the approximate multiplier, then the concept of approximate multiplier simulation was introduced. The primary contribution was to simulate the impact of the approximate multipliers on the accuracy of image classification using VGGNet. The simulation included applying several error matrices with various MRE values which cover the impact of several proposed approximate multipliers in the literature. The simulation covered both Uniform and Gaussian PDFs and assessed their impact on the network's accuracy. The simulation results show that approximate multipliers have very little impact on the network's accuracy. This comes with the advantage of significant reductions in power, area, and delay. Additionally, an alternative hybrid approach was proposed which uses a mix of exact and approximate multipliers. In the hybrid approach, the approximate multiplication can be used in deeper layers of the network which has the least impact on the accuracy. The hybrid approach simulation leads to a reduced negligible impact on the accuracy while having significant savings in power, area, and delay on a large portion of the network.

REFERENCES

[1] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[2] S. Venkatachalam and S.-B. Ko, "Design of power and area efficient approximate multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 5, pp. 1782–1786, May 2017.

[3] V. Leon, G. Zervakis, D. Soudris, and K. Pekmestzi, "Approximate hybrid high radix encoding for energy-efficient inexact multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 3, pp. 421–430, Mar. 2018.

[4] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, and M. Pedram, "RoBA multiplier: A rounding-based approximate multiplier for high-speed yet energy-efficient digital signal processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 2, pp. 393–401, Feb. 2017.

[5] S. Hashemi, R. Bahar, and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2015, pp. 418–425.

[6] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 10, pp. 3105–3117, Oct. 2016.

[7] V. Mrazek, S. S. Sarwar, L. Sekanina, Z. Vasicek, and K. Roy, "Design of power-efficient approximate multipliers for approximate artificial neural networks," in *Proc. 35th Int. Conf. Comput.-Aided Design*, 2016, p. 81.

[8] Y. Geifman. *VGG16 Models for CIFAR-10 and CIFAR-100 Using Keras*. Accessed: May 2017. [Online]. Available: https://github.com/geifmany/cifar-vgg

[9] F. Chollet. (2015). *Deep Learning for Humans*. [Online]. Available: https://github.com/fchollet/keras
[10] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *Proc. 3rd IAPR Asian Conf. Pattern Recognit. (ACPR)*, Nov. 2015, pp. 730–734.
[11] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009. [Online]. Available: https://scholar.google.ca/scholar?hl=en&as_sdt=0%2C5&q=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.222.9220%26rep%3Drep1%26type%3Dpdf&btnG=
[12] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.
[13] L. Du *et al.*, "A reconfigurable streaming deep convolutional neural network accelerator for Internet of Things," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 1, pp. 198–208, Jan. 2018.
[14] T. Yang, T. Ukezono, and T. Sato, "Low-power and high-speed approximate multiplier design with a tree compressor," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Nov. 2017, pp. 89–96.

ISSAM HAMMAD received the B.Sc. degree in computer engineering from Princess Sumaya University for Technology, Amman, Jordan, in 2008, and the M.A.Sc. degree in electrical and computer engineering from Dalhousie University, NS, Canada, in 2010, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. He is currently a Software Consultant with the Energy Department, Alithya Digital Technology Corporation, Toronto, ON, Canada. He has more than eight years of industrial experience in machine learning, video compression, cryptography, and nuclear software systems. His current research interests include machine learning, deep learning, AI hardware acceleration, and AI embedded systems.

KAMAL EL-SANKARY (M'07) received the B.Eng. degree from Lebanese University, Tripoli, Lebanon, in 1997, the M.A.Sc. degree in electrical engineering from the University of Quebec, Montreal, QC, Canada, in 2002, and the Ph.D. degree in electrical engineering from École Polytechnique, Université de Montréal, Montreal, Canada, in 2006. He joined the Department of Electrical and Computer Engineering, Dalhousie University, Halifax, NS, Canada, in 2006, where he is currently an Associate Professor. His current research interests include mixed-signal, analog, digital, and RF integrated circuits design, and embedded systems. He is the Chair of the Atlantic Canada IEEE Circuits and Systems and the Solid State Circuits Joint Chapter.