# Reconfigurable FET Approximate Computing-based Accelerator for Deep Learning Applications

Raghul Saravanan*, Sathwika Bavikadi*, Shubham Rai†, Akash Kumar† and Sai Manoj Pudukotai Dinakarrao*
* Department of Electrical and Computer Engineering, George Mason University, Fairfax, VA, USA
† Department of Computer Science, Technische Universität Dresden, Dresden, Germany
*{rsaravan, sbavikad, spudukot}@gmu.edu, †{shubham.rai, akash.kumar}@tu-dresden.de

***Abstract*—Reconfigurable nanotechnologies such as Silicon Nanowire Field Effect Transistors (FETs) serve as a promising technology that not only facilitates lower power consumption but also supports multi-functionality through reconfigurability. It enables reconfigurability and supports multiple functionalities per computational unit. These features motivate us to design a novel state-of-the-art energy-efficient hardware accelerator for implementing memory-intensive applications including convolutional neural networks (CNNs) and deep neural networks (DNNs). To accelerate the computations, we design Multiply and Accumulate (MAC) units to perform the computations. For the design of MACs , we employ Silicon nanowire reconfigurable FETs (RFETs). The use of RFETs leads to nearly 70% power reduction compared to the traditional CMOS implementation and also reduced latency in performing the computations. Further to optimize the overheads and improve memory efficiency, we introduce a novel approximation technique for RFETs. The RFET-based approximate adders lead to reduced power, area, and delay while having a minimal impact on the accuracy of the DNN/CNN. In addition, we carry out a detailed study of varied combinations of architectures involving CMOS, RFETs, accurate adders, and approximate adders to demonstrate the benefits of the proposed RFET-based approximate acclerator. The proposed RFET-based accelerator achieves an accuracy of 94% on MNIST datasets with 93% and 73%reduction in the area, power and delay metrics respectively compared to the state-of-the-art hardware accelerator architectures.**

## I. Introduction

Machine learning (ML) techniques such as Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) are widely adopted in numerous applications including computer vision, text recognition, speech recognition, and cybersecurity [1]–[3] due to their superior performance compared to the heuristic techniques. However, DNNs and CNNs are often encountered with a requirement to perform a plethora of multiply-and-accumulate (MAC) operations [4]. Performing such a massive amount of MAC operations in real-time requires voluminous resources at disposal [5].

FPGA- and ASIC-based hardware accelerators are introduced to meet the computational and real-time performance demands along with overcoming the limitations of traditional CPU and GPU architectures [6], [7]. FPGA architecture facilitates reconfigurability and programmability along with accelerating the MAC operations in the CNNs to meet performance requirements. However, the energy efficiency and overheads of the FPGAs impose challenges. On the other hand, Traditional ASIC-based accelerators provide high performance and energy-efficiency compared to their counterparts.

Traditional CMOS-based ASIC accelerators though have shown superior performance compared to CPUs and GPUs, suffer from area and reconfiguration overheads, latency, and leakage currents. To address such challenges, emerging transistor technology devices such as FinFETs, RFETs, and Memristors are adopted in designing accelerators [7]. Further, some of these devices such as Memristors also support storage along with computations *a.k.a* in-memory computing [8].

Among multiple emerging devices, the recent research on nanotechnologies such as silicon nanowires (SiNW) or germanium-based nanowires-based reconfigurable field effect transistors [9] have proven to exhibit multiple functionalities per computational unit with minimal reconfigurability overheads. The SiNW RFET devices exhibit ambipolar conditions due to inherent materials used for construction, which in turn help to reconfigure the operating regions (i.e., p and n regions) of the transistor. Thus, through programming the gate terminals, the different functionalities can be obtained [9]. This motivates us to employ SiNW-based circuits for the design of neural network accelerators, making them reconfigurable, energy, area and performance efficient.

In addition to architectural design choices, computational optimization is widely explored to meet real-time demands along with performance. Among multiple techniques, the approximation is a computational paradigm widely explored in DNNs and CNNs for computational complexity optimization [7]. Several approximate computing-based hardware accelerators [7] have been shown to optimize area, power, latency, and PDP (Power Delay Product) on varied datasets.

In this work, we orchestrate the benefits of integrating SiNW RFET (SiNW) with approximate computing for an ML accelerator design. The novelty of this work can be outlined in a three-fold manner as follows:

- A novel ML accelerator architecture aiming at CNN/DNN acceleration optimizing the resource utilization by leveraging the reconfigurability of the transistors through SiNW RFET that supports approximate computing paradigm as well. To the best of our knowledge, this is the first SiNW RFET-based ML accelerator that supports

This work was supported in part by the US National Science Foundation (NSF) Grants CNS-2228239, and CNS-2213404. This research was supported in part by the German Research Foundation (DFG), project SecuReFET (Project Number: 439891087).

approximate computing.

- A generic approximate adder architecture following a simple carry chain technique to aid in both computational and architectural optimizations is introduced for ML acceleration. Furthermore, the control over the tunable approximation bits aids in achieving negligible accuracy loss during the training and testing of CNN.
- An emerging RFET design flow using SiNW exploiting the reconfigurability through one of the gate terminals to enable efficient computations per unit area.

## II. Related Work and Background

We present the basics of the SiNW reconfigurable FETs (RFETs) and the existing works on hardware accelerators here.

### A. SiNW Reconfigurable FETs

The schematic of SiNW-based RFET [9] is shown in Figure 1. The RFET encompasses a Control Gate (i.e., A or CG) which is responsible for carrying the charges, and a Program Gate (i.e., P or PG) which controls the functionality of the gate. Depending upon the voltage applied to the PG, the p-type or the n-type region gets activated. If P=0 (red line), then the p-type functionality is activated, and when P=1 (blue line), then the n-type functionality is activated. These devices when combined can perform multiple computations through programming/reconfiguration with less area occupied [9]. An essential requirement for the practical use of RFET is that both p-type and n-type programs have to deliver the same output current at an identical geometry exhibiting fully symmetrical *I-V* characteristics as shown in Figure 1. As the SiNW RFETs are recently introduced, they have been explored only in the design of basic logic gates and 1-bit ALU [9].

A well-functionally enhanced combinational logic gates was designed using the RFETs [9] consumes 20% less area when compared to the CMOS and a 1-bit ALU designed using RFET technology occupies 30% less area and 34% less delay than the design which is mapped to the traditional CMOS technology.

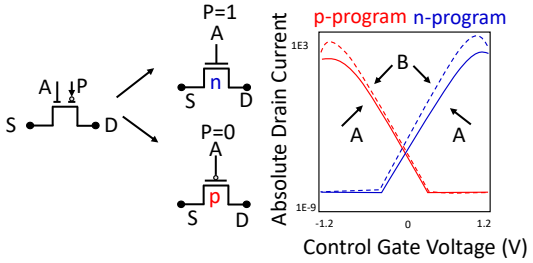


Fig. 1. Construction of RFET and its I-V characteristics.

### B. State-of-the-Art

Here, we discuss some of the most relevant CMOS and emerging devices-based ASIC accelerators. DianNao is an ASIC-based accelerator operating at 65nm node [10] proposed by Chen *et. al.* focuses on minimal memory transfers for efficient computation. This was later extended to PuDianNao which can accelerate at 1.20× faster than NVIDIA K20M GPU [7]. A further extension of the DianNao family was ShiDianNao which targeted high performance and low power consumption by optimizing the data movements resulting in 60× less energy compared to DianNao. On the other hand, AXNet was proposed where approximate computing is deployed on the software to predict the approximate results of the training and testing process [11]. The approximate adders [12] and multipliers [13] have exhibited results close to the accurate computing methods in various CNN/DNN accelerators.

Similarly, work in Pj-AxMTJ [14] deployed on non-volatile memory (memristive RAM (MRAM)) based approximate adders to perform approximation for efficient magnetization switching in magnetic tunnel junction (MTJ). In addition, approaches such as [15], [16] introduce system-level and circuit-level approximation by bit trimming and Memristance scaling. Along with the two approximate approaches, the hybrid approximate PIM in [16] supports dynamic approximation, where the operations can be modified on the fly by the controller, and a tunable approximation of the bit-trimming can be done dynamically.

Despite their superior performances, these ASIC-based accelerators lack programmability and reconfigurability, which is pivotal when applied to diverse applications. Reconfigurability of the accelerator design amalgamated with approximate computing can lead to improved resource utilization and superior performance with minimal degradation in accuracy.

## III. Proposed SiNW-based ML Accelerator

### A. System Architecture

The ML accelerator proposed in this work is designed to accelerate data-intensive applications such as CNNs and DNNs. Figure 2 depicts the hierarchical view of the proposed ML accelerator design that includes the cluster of computational cores, the arrangement of cores in the cluster, and the interconnection between the multipliers and the adders inside the core. Each core is a Processing Element (PE) that is capable of performing MAC operations efficiently. The weight and the input channel information stored in the memory establish direct communication with the cores to enable faster computation. The channels can establish parallel communication with each of the cores such that multiple operations are computed in a single clock cycle. Each core's (intermediate) outputs are accessible during the operation and facilitate tracing back the intermediate outputs of the cores. Design details of the system, approximate computing, and SiNW implementation are discussed below.

### B. Design of Cores with Approximate MAC Unit

As shown in Figure 2 each cluster encompassing cores performs MAC operations on a matrix grid. We choose each core to operate on two 4-bit operands, 4-bit operands meet requirements for a majority of applications [17]. Higher precision computations can be performed by reconfiguring the interconnections among the cores [17]. To support 3×3 convolutions and MAC operations in parallel, each core encompasses 9 multipliers and 8 adders as shown in Figure 2. The input and weight channels comprise the 36-bit line which disintegrates into nine pairs of 4-bit lines into the multipliers performing nine 4-bit multiplication per core. These data lines holding the intermediate product results are further routed to

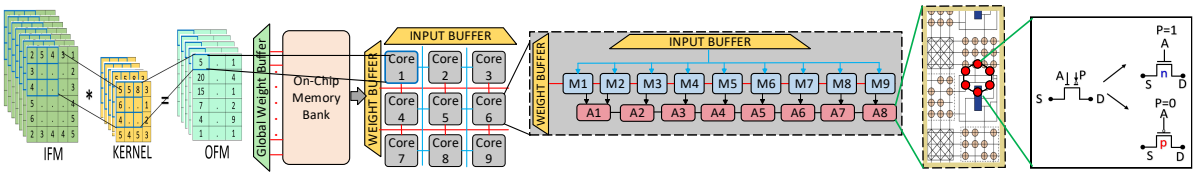


Fig. 2. Proposed Accurate and Approximate ML Accelerators Implementation

the adders which complete the MAC operations, as shown in Figure 2. Each core can compute 3×3 convolution in parallel, thereby facilitating high throughput.

The accurate architecture performs the MAC operations with accurate adders, whereas the approximate architecture performs the with approximate adders, as discussed below.

*1) Multiplier:* It is cardinal to perform the MAC operations in an efficient manner since CNN/DNN uses MAC operations very often. Each PE has nine multiplication units for which the data are fed from the weight and input channels. The multiplier core is capable of performing multiplication operations between 4-bit operands, followed by a series of addition operations in several stages. However, the approximate multiplier can also be deployed to perform MAC operations. In our proposed design the PE for both accurate and approximate computing follows the traditional accurate multiplication algorithm. Since matrix multiplication can be performed in parallel but the addition process gets slogged back during the accumulation phase of the operation we optimize the addition process by sliding in the approximate adder.

*2) Approximate Adder:* Our proposed nine-core formation allows a perfect mapping of one complete 4-bit multiplication for eight different adders. Each PE unit has eight adders which account for seventy-two adders in each cluster. The results of the multiplier are loaded into the adder in a sequential fashion. To optimize the computational process, the simple carry chain truncation technique is used for the approximate adder. Since our proposed design is being mapped with RFET, it aids in architectural optimization as well. For the adders, we eliminate the carry chain by equating the $C_{in}$ of a 1-bit full adder (FA) for the given stage with one of the inputs (i.e., A in our design) of the previous stage. The sum and the carryout for the approximate 1-bit FA can be obtained by S= (Ã)(B+C)+(B.C) and C=A respectively. This will result in 6 correct computational results of the 8 possible cases. Furthermore, the 1-bit FA adder can be reinstated to design a multi-bit adder depending on the application. These bunch of adders can be formulated to logic gates which when mapped with RFET translate into global data lines that reconfigure the logic gates which can perform approximation efficiently per unit leading to optimized resource consumption. In our design four 8-bit, two 9-bit, one 10-bit, and one 11-bit adder are used for the flexibility to control the number of approximation bits of the operands. This aids in achieving results not far from the results of the traditional accurate adder.

### C. Application Mapping

The proposed cluster can be programmed to perform the different layers such as the Convolutional layer, and Fully-connected Layers. However, the convolutional layer accounts for the majority of the MAC operations required for the training and inference process. The cluster architecture follows the wight stationary approach wherein the weights are preloaded onto the on-chip memory while the input orders are fed sequentially to obtain the outputs. Our design can process all the convolutional processes required by the network. The cluster can process the varied Input Feature Map (IFM) sizes and kernel window of size n×n resulting in an Output Feature Map (OFM). This results in efficient feature extraction.

The proposed core has a dedicated input channel and a common weight bus that spans over the architecture supporting weight stationary method. The CNN/DNN architecture mapped with a CMOS design flow operates by switching between n and p-type regions keeping either the pull-up or pull-down network active. This results in single function per CMOS device. However, in the case of the RFET the logic of the architecture is optimized by exhibiting the same symmetrical output at both regions of the network. The functionality of the network can be programmed by applying a control signal over the program terminal of the network. This helps in deploying multiple functionalities for a single network. The MAC units are translated to optimized networks exhibiting multiple functions per device leading to a better performance than CMOS.

## IV. Experimental Results

In this section, we evaluate the proposed accelerator in terms of power, area, delay, and accuracy for ML applications.

### A. Experimental Setup

The accelerator was designed using RTL (VHDL) technique. The delay, power, and area for the proposed accelerator are obtained from Synopsys Design Compiler using CMOS 28 nm standard cell libraries and 26 nm for RFET. The power consumption of the cluster is that of all the cores, on-chip memory, and the channels communicating between the memory and the cluster. The results obtained for the architecture are for one cluster that is designed for 3×3 convolution operation. The accelerator was tested against the MNIST and CIFAR-10 datasets.

### B. Performance Evaluation

We have also evaluated our proposed work by mapping to different neural networks like LeNet, 3C(3 Convolutional)+2P(Pooling)+2F(Fully Connected Layer) etc, as shown in Figure 3. The accuracy of the networks for approximate resulted in a minimal decrease by 3% for the MNIST dataset. For the CIFAR-10 convolution matrix multiplication, the approximate adder yields approximately 4% error when compared to the accurate matrix multiplication.

A comparative study was made among CMOS and RFET both computing in accurate and approximate paradigms. As

shown in Figure 4 the power consumed per unit area (i.e., power density) by RFET is reduced by 88% and 94% when compared with CMOS accurate and approximate designs respectively. Furthermore, the proposed RFET approximation is more efficient than RFET-accurate by 48% in terms of power density. However, between the CMOS architectures, CMOS approximate architecture has a mild increase in power density. This is due to the fact that the RFET design flow optimizes the approximate paradigm more efficiently than the CMOS design flow. Moreover, there is a steep decrease in the power of the CMOS approximate architecture as well.

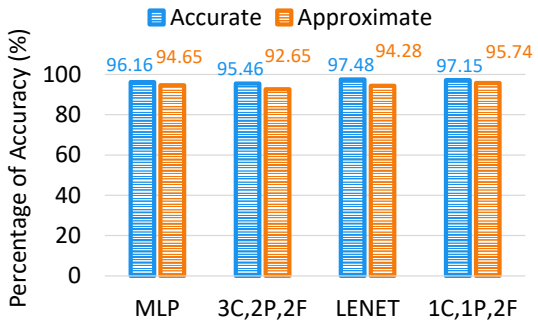


Fig. 3. Accuracy of Proposed Accurate and Approximate Networks

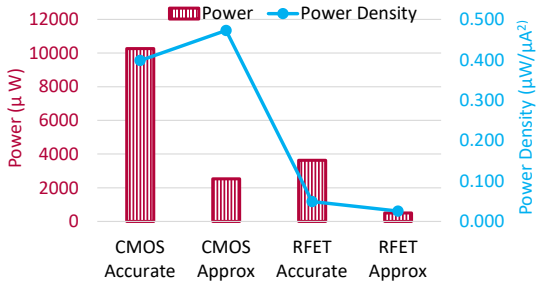


Fig. 4. Power Consumption of CMOS and RFET Design

We also measure the critical path delay and the Power Delay Product (PDP) for the above four architectures as shown in Figure 5. The critical path delay for RFET accuracy and RFET approximate approach is increased when compared to the CMOS flow. This is due to the fact the RFET flow operates at 92 MHz whereas CMOS operates at 1 GHz. The factor that limits the operating frequency of the RFET is due to the innate low drive strength cells. However, in terms of power consumed per unit area is significantly less than the CMOS design flow.

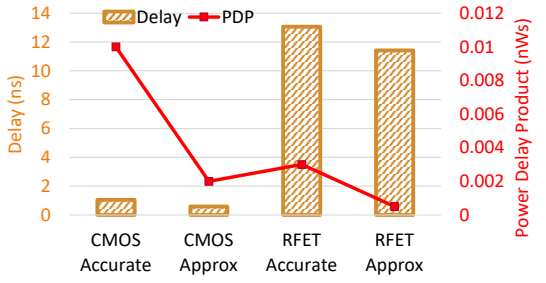


Fig. 5. Critical Path Delay of CMOS and RFET Design

### C. Performance Comparison

To evaluate our performance of the design we have compared it with the ASIC and emerging technology accelerators. For a fair comparison, we match the MAC operations in our design with respect to the previous designs. We scale up our design to 11 clusters containing 891 multipliers and 792 adders performing 1683 MAC operations. DianNao [10] and PuDianNao [18] are ASIC-based accelerators designed using TSMC 65 nm library operating at 0.98 GHz and 1GHz respectively. The proposed CMOS approximate approach operates at 1 GHz and RFET approximate operating at 90 MHz can compute 1683 MAC operations which consume less power approximately by 94% and 98% respectively.

The proposed design can handle 70% more MAC operations than [10] and [18] consuming less power and area. However, the adders are excess in the proposed cluster which can be used further for deep networks or future computing with the same resources. Moreover, the proposed RFET approximate design consumes less power per area by 87%. While comparing with the emerging technologies [19] and [20] which use FeFET crossbar arrays for CNNs operating at 400 MHz and 200 MHz respectively. Both [19] and [20] are designed using 22nm technology. RFET approximately operates at low power by 77% performing 64% increased MAC operations. There is a huge decrease in power density by the RFET when compared with RFET. On a whole, the proposed RFET approximate is one of the best designs to perform similar state-of-the-heart operations with efficient resource consumption.

TABLE I
SIMULATION RESULTS FOR CMOS AND EMERGING TECHNOLOGIES.

| | Dian Nao [10] | Pu Dian Nao [18] | FeFET In-memory [19] | FeFET Binary [20] | Proposed CMOS Ap-prox. | Proposed RFET Ap-prox. |
|---|---|---|---|---|---|---|
| Freq. (GHz) | 0.98 | 1 | 0.4 | 0.2 | 1 | 0.09 |
| # Mul. | 272 | 272 | 512 | - | 891 | 891 |
| # Adder | 256 | 784 | 512 | - | 792 | 792 |
| # MAC | 528 | 1056 | 1024 | 256 | 1683 | 1683 |
| Power ($\mu$W) | 485000 | 596000 | 500000 | 22000 | 27775 | 5526 |
| Power Den-sity | 0.160 | 0.169 | 328.94 | 846.15 | 0.47 | 0.0254 |

## V. CONCLUSION

In this paper, we present the design of a novel state-of-the-art RFET-based CNN/DNN accelerator. The hardware accelerator performance for the CNN/DNN with an accurate and approximate paradigm has been presented. We also present the advantages of approximate computing over accurate algorithms both in CMOS and RFET technology. In addition, we also prove that approximate computing is supported by the RFETs.The proposed architecture achieves a reduction in power (93%) and area (73%), The proposed CNN/DNN model was tested against MNIST datasets with 97% accuracy in accurate and 94% accuracy in approximate computing.

## REFERENCES


[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[3] A. Al-Furas, M. El-Dosuky, and T. Hamza, "Multi-column deep neural network based on particle swarm optimization," in *2019 First International Conference of Intelligent Computing and Engineering (ICOICE)*, 2019, pp. 1–5.

[4] M. Lechner, A. Jantsch, and S. M. P. Dinakarrao, "Resconn: Resource-efficient fpga-accelerated cnn for traffic sign classification," in *2019 Tenth International Green and Sustainable Computing Conference (IGSC)*, 2019, pp. 1–6.

[5] A. Dhavlle and S. M. Pudukotai Dinakarrao, "A comprehensive review of ml-based time-series and signal processing techniques and their hardware implementations," in *2020 11th International Green and Sustainable Computing Workshops (IGSC)*, 2020, pp. 1–8.

[6] A. Ganguly, R. Muralidhar, and V. Singh, "Towards energy efficient non-von neumann architectures for deep learning," in *20th International Symposium on Quality Electronic Design (ISQED)*, 2019, pp. 335–342.

[7] S. Bavikadi, A. Dhavlle, A. Ganguly, A. Haridass, H. Hendy, C. Merkel, V. J. Reddi, P. R. Sutradhar, A. Joseph, and S. M. Pudukotai Dinakarrao, "A survey on machine learning accelerators and evolutionary hardware platforms," *IEEE Design & Test*, vol. 39, no. 3, pp. 91–116, 2022.

[8] C. Yakopcic, M. Z. Alom, and T. M. Taha, "Memristor crossbar deep network implementation based on a convolutional neural network," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 963–970.

[9] S. Rai, J. Trommer, M. Raitza, T. Mikolajick, W. M. Weber, and A. Kumar, "Designing efficient circuits based on runtime-reconfigurable field-effect transistors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 3, pp. 560–572, 2019.

[10] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, "Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," *SIGARCH Comput. Archit. News*, vol. 42, no. 1, p. 269–284, feb 2014. [Online]. Available: https://doi.org/10.1145/2654822.2541967

[11] Z. Peng, X. Chen, C. Xu, N. Jing, X. Liang, C. Lu, and L. Jiang, "Axnet: Approximate computing using an end-to-end trainable neural network," in *Proceedings of the International Conference on Computer-Aided Design*, ser. ICCAD '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3240765.3240783

[12] B. S. Prabakaran, S. Rehman, M. A. Hanif, S. Ullah, G. Mazaheri, A. Kumar, and M. Shafique, "Demas: An efficient design methodology for building approximate adders for fpga-based systems," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 917–920.

[13] Z. Ebrahimi, S. Ullah, and A. Kumar, "Simdive: Approximate simd soft multiplier-divider for fpgas with tunable accuracy," in *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, ser. GLSVLSI '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 151–156. [Online]. Available: https://doi.org/10.1145/3386263.3406907

[14] H. Cai, H. Jiang, M. Han, Z. Wang, Y. Wang, J. Yang, J. Han, L. Liu, and W. Zhao, "Pj-axmtj: Process-in-memory with joint magnetization switching for approximate computing in magnetic tunnel junction," in *2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2019, pp. 111–115.

[15] H. E. Yantir, A. M. Eltawil, and F. J. Kurdahi, "Approximate memristive in-memory computing," *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 5s, 2017.

[16] H. E. Yantır, A. M. Eltawil, and F. J. Kurdahi, "A hybrid approximate computing approach for associative in-memory processors," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 4, pp. 758–769, 2018.

[17] G. Yuan, X. Ma, S.-F. Lin, Z. Li, and C. Ding, "A sot-mram-based processing-in-memory engine for highly compressed dnn implementation," *ArXiv*, vol. abs/1912.05416, 2019.

[18] D. Liu, T. Chen, S. Liu, J. Zhou, S. Zhou, O. Teman, X. Feng, X. Zhou, and Y. Chen, "Pudiannao: A polyvalent machine learning accelerator," ser. ASPLOS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 369–381. [Online]. Available: https://doi.org/10.1145/2694344.2694358

[19] T. Soliman, R. Olivo†, T. Kirchner, M. Lederer, T. Kämpfe, A. Guntoro, and N. Wehn, "A ferroelectric fet based in-memory architecture for multi-precision neural networks," in *2020 IEEE 33rd International System-on-Chip Conference (SOCC)*, 2020, pp. 96–101.

[20] T. Soliman, R. Olivo, T. Kirchner, C. D. l. Parra, M. Lederer, T. Kämpfe, A. Guntoro, and N. Wehn, "Efficient fefet crossbar accelerator for binary neural networks," in *2020 IEEE 31st International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2020, pp. 109–112.