SenTer: A Reconfigurable Processing-in-Sensor Architecture Enabling Efficient Ternary MLP

Sepehr Tabrizchi
School of Computing,
University of Nebraska-Lincoln,
Lincoln, Nebraska, USA
stabrizchi2@unl.edu

Rebati Gaire
School of Computing,
University of Nebraska-Lincoln,
Lincoln, Nebraska, USA
rgaire2@unl.edu

Shaahin Angizi
Department of Electrical and Computer Engineering,
New Jersey Institute of Technology,
Newark, New Jersey, USA
shaahin.angizi@njit.edu

Arman Roohi
School of Computing,
University of Nebraska-Lincoln,
Lincoln, Nebraska, USA
aroohi@unl.edu

ABSTRACT

Recently, Intelligent IoT (IIoT), including various sensors, has gained significant attention due to its capability of sensing, deciding, and acting by leveraging artificial neural networks (ANN). Nevertheless, to achieve acceptable accuracy and high performance in visual systems, a power-delay-efficient architecture is required. In this paper, we propose an ultra-low-power processing in-sensor architecture, namely SenTer, realizing low-precision ternary multi-layer perceptron networks, which can operate in detection and classification modes. Moreover, SenTer supports two activation functions based on user needs and the desired accuracy-energy trade-off. SenTer is capable of performing all the required computations for the MLP's first layer in the analog domain and then submitting its results to a co-processor. Therefore, SenTer significantly reduces the overhead of analog buffers, data conversion, and transmission power consumption by using only one ADC. Additionally, our simulation results demonstrate acceptable accuracy on various datasets compared to the full precision models.

CCS CONCEPTS

• **Computer systems organization** → *Special purpose systems*; *Sensors and actuators*; • **Computing methodologies** → *Neural networks.*

KEYWORDS

processing in-sensor, multi-layer perceptron, low-power CMOS imager

GLSVLSI '23, June 5–7, 2023, Knoxville, TN, USA

ACM ISBN 979-8-4007-0125-2/23/06…$15.00
https://doi.org/10.1145/3583781.3590225

ACM Reference Format:
Sepehr Tabrizchi, Rebati Gaire, Shaahin Angizi, and Arman Roohi. 2023. SenTer: A Reconfigurable Processing-in-Sensor Architecture Enabling Efficient Ternary MLP. In *Proceedings of the Great Lakes Symposium on VLSI 2023 (GLSVLSI '23), June 5–7, 2023, Knoxville, TN, USA.* ACM, Knoxville, TN, USA, 6 pages. https://doi.org/10.1145/3583781.3590225

1 INTRODUCTION

Cloud-based communication and computation pose several serious challenges today, including high latency (a large number of nodes and their geographical distribution), questionable scalability (unbalanced load among the computers), quality of service (QoS) (it is difficult to guarantee the same level of service quality), privacy, and security (data must be securely transferred and stored). It may be possible to address a number of these issues by shifting computing architecture from a cloud-centric perspective to a thing-centric one as the Internet of Things (IoT) or Internet of Everything (IoE) advances. A global network of 75+ billion IoT devices, including smart homes, smart cities, smart industries, wearables, and implantable systems for healthcare, is expected to reach $1100 billion by 2025. Recently, Intelligent IoT (IIoT) has gained significant attention due to its capability of sensing, deciding, and acting by leveraging artificial neural networks (ANN). Through various sensors, such as CMOS image sensors (imagers), IIoT nodes collect and process data. Nevertheless, ANNs are extremely storage and computation intensive in order to achieve high accuracy and acceptable performance in visual systems, making them difficult to implement on edge devices with limited resources. Additionally, many vision applications require continuous monitoring or detection of anomalies by sensory systems, while low information density wastes bandwidth, storage, and computing resources. Towards addressing these challenges, IoT nodes can process sensed data by incorporating Edge Intelligence (EI) devices into a thing-centric computing architecture, where data transferring and data density are reduced by using local computing at the sensing units. In recent years, researchers have studied the development of imagers capable of accelerating ANNs. Pixels' digital outputs can be accelerated using an on-chip processor in the vicinity of the sensor, forming a paradigm, namely Processing near-Sensor (PNS) [4, 6, 15, 16]. It is also possible to process pre-Analog-to-Digital converter data, an image, via a Processing

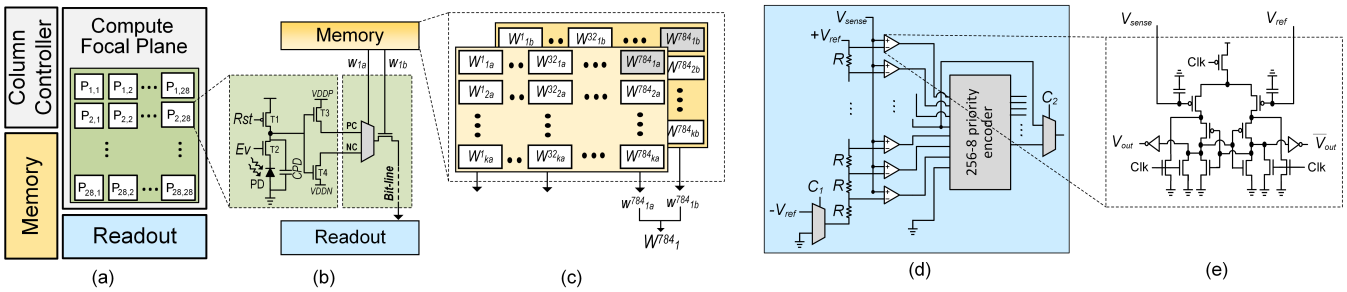

Figure 1: (a) Proposed SenTer architecture, including an array of (b) pixels and compute add-ons, (c) memory unit consisting of two buffers, (d) the readout peripherals, and (e) Circuit-level comparator.

in-Sensor (PIS) platform that eliminates redundant output data. Utilizing the PIS unit within an imager (1) reduces power consumption in converting photocurrents into pixel values, (2) speeds up data processing, (3) alleviates memory bottlenecks, and (4) simultaneously senses and computes. With limited computing and storage resources, deploying all layers into a pixel array is inefficient and complicated. Most PIS studies perform some essential computations like edge detection or, at best, accelerate the first layer and utilize digital neural network accelerators for the remainder.

In this paper, to make real-time decision-making IIoT devices a reality, advances are made from both an algorithmic and hardware architecture perspective. Herein, we propose an efficient PIS scheme with different capabilities to alleviate the power costs of data conversion and transmission at the cost of accuracy degradation. Our architecture, **SenTer**, allows analog convolution, which supports different activation functions. SenTer supports ternary weight neural networks that provide energy efficiency by mitigating the overhead of analog buffers with comparable accuracy to the floating-point (FP) baseline.

2 BACKGROUND

Data transmission off-chip and ADC bandwidth can be reduced by systematizing the integration of computing and sensor arrays. This integration allows for improved power efficiency, higher sampling rates, and better resolution in data acquisition. Additionally, the integrated system can collect more data points and also reduce data processing time. Near-sensor and in-sensor processing architectures are two efficient approaches for enabling embedded signal/image processing and computer vision algorithms to be executed directly on-chip, prior to data transfer off-chip.

PNS architecture is a type of architecture for image processing where the processing is done close to the sensor, either on the same chip or near the sensor, prior to data transfer to an external processor. The PNS unit receives the raw data from the sensor and performs any necessary calculations or processing before sending the data to the main system. This type of architecture is typically used in applications where data needs to be collected from multiple sources and different types of sensors. Near-sensor processing can also be used in applications where data needs to be collected from multiple locations. PNS reduces the amount of data that needs to be transferred off-chip and processed, allowing for higher signal processing and vision performance. According to [7], the CMOS image sensor includes dual-mode delta-sigma ADCs designed to process the first convolutional (conv.) layer of binary-weight neural networks (BWNNs). RedEye [10] implements convolution using charge-sharing tunable capacitors. By sacrificing accuracy in favor of energy savings, this design reduces energy consumption. This system utilizes a custom image sensor integrated with a low-power digital signal processor to perform image processing tasks. In[16], vertically stacked column-parallel ADCs and processing elements are implemented and utilized to run spatiotemporal image processing. To reduce the amount of power consumed by the ADC, [6] converts photocurrents into pulse-width modulated signals, which are then processed by a dedicated analog processor.

PIS, on the other hand, involves the integration of processing capabilities directly into the sensor itself. This approach allows sensors to perform necessary calculations or processing before sending the data to the main system. This includes algorithms running on multiple cores and algorithms adapted for ultra-low-power operation. More precise and accurate vision results can be achieved by enabling complex image processing functions to be performed directly within the sensor itself. PIS also enables on-chip memory, allowing data to be stored and processed on-chip. This type of architecture is typically used in applications where data is collected quickly, such as industrial automation and security systems. As a PIS platform, MACSen [15] integrates MAC (multiply-accumulate) operations directly into the image sensor and leverages double sampling to process the first conv. layer of BWNNs. As a result, visual data may be processed efficiently in real-time at the point of acquisition without requiring additional power-hungry devices. However, this method suffers from high power consumption and huge overhead due to the SRAM-based design. Although PNS can provide more flexibility and scalability, it can also be more complex and expensive to implement [6, 9]. PIS can be more compact and efficient but may be more limited in processing capabilities [2, 3, 12, 13]. PNS is an ideal choice for lower-end processing needs, while PIS is a better choice for applications requiring more sophisticated algorithms. In CMOS image sensors, either of these architectures can reduce data transfer and processing overhead and enhance computing efficiency.

3 PROPOSED ARCHITECTURE

We propose **SenTer** as an ultra-low-power sensor for event detection and classification targeting TinyML applications, shown in Fig.1. SenTer consists of a compute focal plane (CFP) (Fig. 1(b)), a

Table 1: Ternary Values regarding the stored weights.

Buffer_0 (W_a)	Buffer_1 (W_b)	Represented Weight	Output Current
×	0	0	0
0	1	-1	$-I_{CPD}$
1	1	1	I_{CPD}

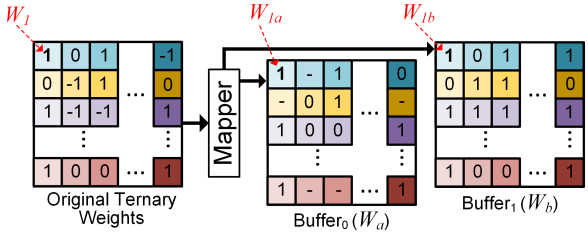

Figure 2: Mapping the original ternary weights into two dedicated buffers.

column controller (CC), memory components (Fig. 1(c)), and a readout circuit (Fig. 1(d)). The proposed architecture operates in two event detection and object classification modes. Since the SenTer architecture is able to implement the first layer of a ternary multilayer perceptron within a sensor, there is no need for a row decoder to read pixels in a row-wise manner. Removing the row decoder compared with conventional convolution-in-pixel structures leads to more power saving at the cost of application generality. The CFP comprises a $m \times n$ pixel array, and $m \times n$ computes add-ons (CPs), i.e., 28×28, as depicted in Fig. 1(b). Depending on a stored weight read from the memory unit, a CP generates a current on a shared bit-line. Finally, in the Readout module, this value converts to 1-to-8 bits according to the configuration of the proposed ADC. All the components and their functionalities are outlined in the following sections, which consume ~0.0255 μW power in total.

3.1 SenTer Components

3.1.1 Memory Unit. SenTer includes efficient non-volatile memory (NVM) components to store the first layer weights of an MLP network. For each ternary weight, two bits (W_a) and (W_b), weight representatives, are required that are stored in the memory components, buffer_0 and buffer_1, respectively. The combinations of these bits to generate the ternary weights are summarized in Table 1. In MLP networks, every node is connected to all the next layer's nodes, where each buffer size should be $28 \times 28 \times k$, where k is the number of nodes in the first hidden layer. Because the SenTer design utilized 28×28 memory elements for event detection tasks, the actual size of the memory unit is $2 \times [28 \times 28 \times (k+1)]$. There is a row selector positioned within the memory unit to select 28×28 weights and connect them to pixels' CAs.

3.1.2 Compute Focal Plane (CFP). The CFP module is the core of the SenTer architecture, which performs sensing and analog computation within a sensor. A pixel observes scenes from an environment, and a compute add-on (CA) produces proper currents in accordance with the specific ternary weights, i.e., $\in \{-1, 0, +1\}$. As shown in Fig. 3(a), first, all the pixel capacitors (CPD) are charged

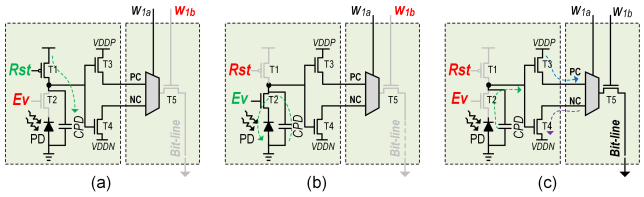

Figure 3: The proposed sensor and its CA in (a) pre-charge, (b) evaluation, and (c) reading phases.

to V_{DD} through T_1. Then CPD starts to discharge depending on the light intensity of the environment through T_2 and photo-diode (PD) in Fig. 3(b). In the next step, T_1 and T_2 disconnect, and the voltage level of CPD remains unchanged. Thus, in order to generate positive and negative weights, CPD is connected to positive and negative voltages through T_3 and T_4, respectively, to generate appropriate currents on the bit-line (BL). As depicted in Fig. 3(c), one of these two generated currents is chosen using a multiplexer with W_{1a} selector. Another weight representative (W_{1b}) connects to a transistor located after the multiplexer (T_5 in Fig. 3) to enable the pixel by passing the generated negative (-1) or positive (+1) currents or disconnects/disables the pixel to denote zero weight (0). Another T_5 responsibility is operating as a row controller in the event detection mode, where inactive pixels should be disconnected to save power. The functionality of one CA is validated by HSpice with 45nm CMOS technology, and the transient simulation results are depicted in Fig. 4. In ❶, W_{1a} and W_{1b} are equal to 0 and 1, respectively; therefore, based on Table 1, the generated current on BL is negative. By setting different weight representatives, two other ternary weights are achieved, shown in ❷ and ❸ cycles. As it can clearly be perceived, In ❹, regardless of changing W_{1a}, because W_{1b} is 0, BL remains unchanged, considered as inactive pixel.

3.1.3 Readout Circuit. The ADCs in conventional image sensors measure only the value of the activated row at the end of each column, which means for a $m \times n$ sensory array, the design requires m ADCs and n clock cycles to read the full array. While in an MLP network, there is no need for each pixel's exact value; therefore, the SenTer architecture utilizes only one ADC without having a row selector. The proposed readout module is depicted in Fig.1(d), including additional components that are added to the conventional ADC, realizing various activation functions. The transistor-level schematic of one comparator is shown in Fig. 1(e).

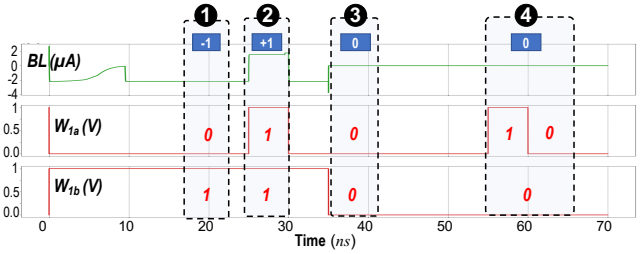

Figure 4: Transient simulation of one pixel and its CA, w.r.t weight W_1.

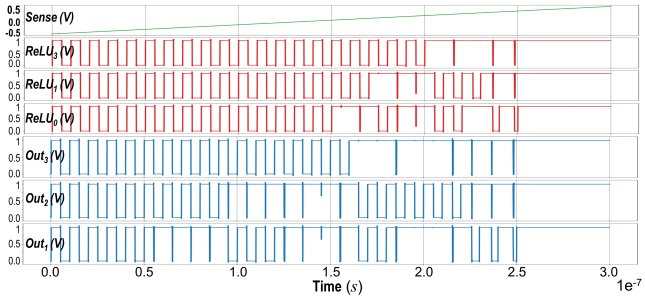

Figure 5: Transient vectors for the proposed ADC in the normal, ReLU, and sign modes.

In the proposed ADC, by changing the control signals C_1 and C_2, SenTer is able to switch between the normal operation and the activation functions, including ReLU and Sign, to save more power at the cost of accuracy degradation. By changing the $-V_{ref}$ to 0, the ADC's functionality switches from the normal to the ReLU operation. In this case, to have the same accuracy as normal mode $[-2^7, 2^7-1]$, SenTer turns off approximately half of its comparators and encoder $[0, 2^7-1]$; otherwise, more accuracy is achieved using ReLU, $[0, 2^8-1]$. Furthermore, the sign function equals the most significant bit in the normal mode. The transient results of the proposed ADC are shown in Fig. 5, where V_{sense} alters from $-0.5V$ to $+0.5V$ while $\pm V_{ref}$ sets to ± 0.3. For simplicity, a 3-bit ADC is considered; however, their functionalities are identical. The red and blue signals indicate ReLU and normal outputs for a 3-bit ADC, in which red signals change only in the positive direction of V_{sense}. Moreover, Out_3 acts similarly to the sign function. It should be noted that for the sign function, we can turn off irrelevant circuits.

3.1.4 Column controller. SenTer is designed for ultra-low-power operation. Therefore, the column controller is designed in a way to a further reduction in power consumption. In SenTer architecture, each column of the pixel array is connected to a separated V_{DD} line. In this case, the column controller is responsible for disconnecting or connecting columns of pixels to a power supply to save more power. This feature is useful in event detection mode.

3.2 SenTer Modes

As mentioned earlier, SenTer operates in two modes: event detection and object classification, elaborated in Algorithm 1.

3.2.1 Event Detection. The proposed low-power event detection approach is designed to detect small-to-moderate differences between frames n and $n+t$, where $t \in \mathbb{N}$, namely the difference margin and can be justified by users' needs. Smaller t provides better accuracy at the cost of power overhead. By setting $t = 1$, SenTer checks every two consecutive frames to detect changes/events. The primary steps of the event detection phase are summarized in lines **1** to **8**. First, in line **2**, the last row of the memory weight (k+1) is chosen. All the pixels' values are read (line **4**), and the produced voltage based on the summation of all currents is measured using an ADC. Herein, designers can determine which pixels are important for comparison. One simple arrangement is expressed by equation 1, where the pixel array is partitioned into 3-pixel boxes, 3×3. Thus,

Algorithm 1 SenTer Mode:

1: **procedure** EVENT-DETECTION
2: `Choose_Weights(`$k+1$`)` ▷ Read the last row of the buffers.
3: **while** (true)
4: pixel_values←`read_pixels_value()` ▷ Only read ON pixels.
5: **if** $pixel\_values < old\_pixel\_value - threshold$ **OR** $pixel\_values > old\_pixel\_value + threshold$ ▷ Check the margin
6: `Break`
7: `Enable` (Object-Classification)
8: **end procedure**
9: **procedure** OBJECT-CLASSIFICATION
10: pixel_values←`read_pixels_value()`
11: **for** i **in_range** k ▷ k is number of nodes in the first hidden layer.
12: `read_weights` (i) ▷ $m \times n$ weights are applied to CA.
13: `calculate_node` (i) ▷ Calculate MACs regarding the i^{th} node.
14: `Enable` (Event-Detection)
15: **end procedure**

each box includes nine pixels, including only one ON pixel and eight inactive ones. In this strategy, SenTer disconnects 2/3 of columns from V_{DD}, and only 1/9 pixels remain connected to the ADC by setting $W_b = 1$. Then the measured voltage compares with the previous one (line **5**) to ensure the difference is lower than a threshold. Lines **4** and **5** repeat until the difference between old_pixel_value and $pixel\_value$ exceeds the threshold. Once it happens, the mode changes to the object classification mode. It should be mentioned that the frequency of the loop execution is dependent on the application, but generally, the frame rate is lower than object classification. To exemplify the event detection mode, two inputs from the MNIST dataset are considered, 7 and 8. Using the proposed algorithm, the obtained values for the $pixel\_values$ variable are approximately equal to 10 and 19 for 7 and 8, respectively. Therefore, by shifting the images vertically and/or horizontally, these values vary between ± 1, which is a good estimate for the $threshold$ parameter.

$$W^x_{k+1b} = \begin{cases} 1 \text{ if } mod\ (x,3) = 2\ \&\ x \in \left(i \times 28, (i+1) \times 28\right] \\ 0\ otherwise \end{cases} \quad (1)$$
$$W^x_{k+1a} = 1, \quad \text{where } i \in \{3\mathbb{N} - 1\}$$

3.2.2 Object Classification. The object classification mode is designed to classify the images using the MLP networks. Herein, all the pixels are active (ON), and their values are evaluated and held unchanged in the sensor's capacitors (line **10**). Then the stored weights (28×28) corresponding to sensors are applied to CAs in line **12**. In the next step, using the proposed reconfigurable ADC, the result of node i^{th} is measured (line **13**). These two steps are repeated for all the hidden layer's nodes. In this mode, SenTer performs all the required operations of the first layer, including the fully-connected and activation layers, within the sensor in the analog domain. The output of this mode is passed to a microprocessor as the next layer's input to compute the remaining layers in the digital domain. Then SenTer changes its mode back to the event detection mode.

4 EVALUATION RESULTS

In addition to the functionalities of SenTer's components in the previous sections, our evaluation phase consists of accuracy and qualitative comparison with the state-of-the-art PIS-based MLP accelerators.

Datasets: We evaluated our models on three publicly available datasets: MNIST[8], FashionMNIST[14], and CBCL FACE[1], shown

Figure 6: Samples of three examined datasets, including (a) MNIST, (b) Fashion-MNIST, and (c) CBCL Face.

Table 2: Network specifications and parameters

Network	# of Layers	# of params in 1^{st}-Layer	Total params in the network	# of Zeros in 1^{st}-Layer
MLP5	5	401920	575050	176208
MLP4	4		567434	187249
MLP3	3		535818	194119
MLP2	2		407050	229006

in Figs. 6 (a), (b), and (c), respectively. MNIST[8] consists of 70000 images of hand-written digits with respective labels for each image. Each image is a 28×28 grayscale image with ten classes. Similarly, FashionMNIST[14] has 70000 grayscale images of size 28×28, associated with a label from 10 categories. Lastly, the CBCL FACE[1] dataset contains 19×19 grayscale images of two classes: face and non-face. For each dataset, we resized the images to 28×28 grayscale images before feeding them to the networks. We randomly sampled 10% of the training samples as a validation set.

NN Architecture: We demonstrate the advantages of SenTer through an image classification task by designing four different MLP architectures with listed specifications in Table 2. Each network architecture has 512 nodes in the first hidden layer and is reduced by half until the last layer. The number of parameters in the table includes both weights and biases.

We implemented the entire networks, training, and testing pipeline in the PyTorch framework. For optimization, we employ Stochastic Gradient Descent(SGD)[11] Optimizer. We initialized the learning rate to 0.001 and scaled it by 0.1 after 60 epochs. We set the batch size to 32 and trained all the networks to 100 epochs. The best-performing checkpoint on the validation set is saved, and the performance on the test set is reported.

Accuracy: We trained each network with full-precision weights and the ReLU activation function in each layer. Then we evaluated the performance of the trained model in the following four settings: (a) using full-precision weights with the ReLU activation function in the first layer. Then (b) replace the ReLU activation function with the sign function in the above setting. We then (c) used a quantized model with ReLU activation in the first layer and finally (d) the quantized model with the sign activation function in the first layer. We used a range-based linear quantization [5] to ternarize the weights of the first layer with values in the range {-1, 0, 1}. In this quantization technique, we multiply the float value with a numeric constant, the scale factor. The scale factor (q_x) is computed using the following equation, $q_x = \frac{2^n - 1}{max_{xf} - min_{xf}}$, where n is the number of bits to encode, which in our case is 2. To minimize the effects of outliers, we take the 99^{th} and 1^{st} percentiles of full-precision weights(x_f) as max_{xf} and min_{xf}, respectively. We then divide the values into three sections and replace them with the integers {-1, 0, 1}. Table 3 of quantitative performance on test datasets reveals that using ReLU activation with full-precision weights does not significantly lower performance when the network size is reduced. In fact, a smaller network size performs better than larger ones when the first layer uses quantized weights with ReLU activation. On average, the accuracy decrease due to quantization is less than 3%, with some exceptions. The change in the activation function of the first layer to the sign function leads to a notable decrease in accuracy but also results in a significant decrease in power consumption.

Performance: As different designs are developed for specific domains, for an impartial comparison, we summarized some of the state-of-the-art PIS-based accelerators when all units execute the similar task of processing the 1^{st}-layer of MLP. Table 4 compares the structure of selective near/in -sensor processing designs that target MLP implementations. MACSen [15] and PISA [3] architectures target binary weight neural networks and utilize $m \times n \times k$, computing elements, where $m \times n$ is spatial dimension of sensor arrays, and k is the number of nodes in the first hidden layer. By leveraging Tizbin [13] and SenTer, the number of computing elements is reduced by a factor of k, while they support ternary weight neural networks. PISA and TizBin designs accommodate both processing and sensing functionalities. All the compared PIS architectures measure every pixel's value row by row, defining the sensing scheme. While SenTer can support processing only, it calculates the summation of all the pixels' values simultaneously, leading to a considerable reduction in power consumption. The main advantages of SenTer over the previous designs include the ReLU activation and usage of one ADC, which leads to better accuracy and power saving.

5 CONCLUSION

This paper proposed SenTer, a low-power intelligent visual perception architecture, to enable a processing in-sensor scheme with event detection and object classification capabilities. SenTer performs low-precision ternary MLP in the analog domain to mitigate the overhead of ADCs. Once an event is detected, it switches to the high-power object classification mode to classify the input. The obtained results exhibit acceptable accuracy compared to the full-precision baseline on three data sets, while SenTer consumes 0.0255 μW.

ACKNOWLEDGMENTS

This work is partly supported by the National Science Foundation under Grant No. 2216772 and 2216773.

REFERENCES

[1] [n. d.]. CBCL FACE DATABASE. http://www.ai.mit.edu/projects/cbcl.old/software-datasets/FaceData2.html. Accessed: 2022-09-30.
[2] Minhaz Abedin, Arman Roohi, Maximilian Liehr, Nathaniel Cady, and Shaahin Angizi. 2022. MR-PIPA: An Integrated Multilevel RRAM (HfO x)-Based Processing-In-Pixel Accelerator. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits* 8, 2 (2022), 59–67.
[3] Shaahin Angizi, Sepehr Tabrizchi, and Arman Roohi. 2022. Pisa: A binary-weight processing-in-sensor accelerator for edge image processing. *arXiv preprint arXiv:2202.09035* (2022).
[4] Stephen J Carey, Alexey Lopich, David RW Barr, Bin Wang, and Piotr Dudek. 2013. A 100,000 fps vision sensor with embedded 535GOPS/W 256× 256 SIMD processor array. In *2013 symposium on VLSI circuits*. IEEE, C182–C183.
[5] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2021. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630* (2021).
[6] Tzu-Hsiang Hsu, Yi-Ren Chen, Ren-Shuo Liu, Chung-Chuan Lo, Kea-Tiong Tang, Meng-Fan Chang, and Chih-Cheng Hsieh. 2020. A 0.5-V real-time computational

Table 3: Summary of quantitative performance comparison of different models on three test sets.

Method	Activation	MNIST				FashionMNIST				CBCL Face			
		MLP5	MLP4	MLP3	MLP2	MLP5	MLP4	MLP3	MLP2	MLP5	MLP4	MLP3	MLP2
Full Precision	*ReLU*	97.77	97.92	97.78	97.90	89.64	89.67	89.68	89.41	97.85	97.75	97.74	97.83
	Sign	64.84	53.39	50.45	50.55	48.26	50.66	65.01	55.75	63.58	62.57	70.34	56.75
Ternarized	*ReLU*	95.59	96.37	96.41	96.14	82.59	80.95	81.19	86.57	95.97	94.39	95.87	96.14
	Sign	64.27	55.17	53.98	52.84	30.99	41.48	57.6	53.99	60.74	52.48	60.74	35.6

Table 4: Qualitative comparison of various PIS-based MLP accelerators.

Designs	Technology (*nm*)	Array Size $m \times n$	Precision	Functionality	Activation	#Compute Add-on	#ADC	Accuracy %		
								MNIST	Fashion-MNIST	CBCL
MACSen [15]	180	32×32	Binary	Sensing† Processing‡	Sign	$m \times n \times k^{\star}$	m	96.0	83.17	90.67
PISA [3]	65	128×128	Binary	Sensing Processing	Sign	$m \times n \times k$	m	95.12	-	-
TizBin [13]	45	600×600	Ternary	Sensing Processing	Sign	$m \times n$	m	97.38	85.68	92.30
SenTer	45	28×28	Ternary	Processing	Sign ReLU	$m \times n$	1	96.41	86.57	96.14

†Receives raw images similar to conventional imagers. ‡Performs event detection and classification computations. *Number of nodes in the first layer.

CMOS image sensor with programmable kernel for feature extraction. *IEEE Journal of Solid-State Circuits* 56, 5 (2020), 1588–1596.

[7] Woo-Tae Kim, Hyunkeun Lee, Jung-Gyun Kim, and Byung-Geun Lee. 2020. An on-chip binary-weight convolution CMOS image sensor for neural networks. *IEEE Transactions on Industrial Electronics* 68, 8 (2020), 7567–7576.

[8] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[9] Qin Li, Changlu Liu, Peiyan Dong, Yanming Zhang, Tong Li, Sheng Lin, Minda Yang, Fei Qiao, Yanzhi Wang, Li Luo, et al. 2021. NS-FDN: Near-sensor processing architecture of feature-configurable distributed network for beyond-real-time always-on keyword spotting. *IEEE Transactions on Circuits and Systems I: Regular Papers* 68, 5 (2021), 1892–1905.

[10] Robert LiKamWa, Yunhui Hou, Julian Gao, Mia Polansky, and Lin Zhong. 2016. Redeye: analog convnet image sensor architecture for continuous mobile vision. *ACM SIGARCH Computer Architecture News* 44, 3 (2016), 255–266.

[11] Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics* (1951), 400–407.

[12] Sepehr Tabrizchi et al. 2023. AppCiP: Energy-Efficient Approximate Convolution-in-Pixel Scheme for Neural Network Acceleration. *IEEE JETCAS* 13, 1 (2023), 225–236.

[13] Sepehr Tabrizchi, Shaahin Angizi, and Arman Roohi. 2022. TizBin: A Low-Power Image Sensor with Event and Object Detection Using Efficient Processing-in-Pixel Schemes. In *2022 IEEE 40th International Conference on Computer Design (ICCD)*. IEEE, 770–777.

[14] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[15] Han Xu, Ziru Li, Ningchao Lin, Qi Wei, Fei Qiao, Xunzhao Yin, and Huazhong Yang. 2020. Macsen: A processing-in-sensor architecture integrating mac operations into image sensor for ultra-low-power bnn-based intelligent visual perception. *IEEE Transactions on Circuits and Systems II: Express Briefs* 68, 2 (2020), 627–631.

[16] Tomohiro Yamazaki, Hironobu Katayama, Shuji Uehara, Atsushi Nose, Masatsugu Kobayashi, Sayaka Shida, Masaki Odahara, Kenichi Takamiya, Yasuaki Hisamatsu, Shizunori Matsumoto, et al. 2017. 4.9 A 1ms high-speed vision chip with 3D-stacked 140GOPS column-parallel PEs for spatio-temporal image processing. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 82–83.