# Cross-Layer Exploration and Chip Demonstration of In-Sensor Computing for Large-Area Applications with Differential-Frame ROM-Based Compute-In-Memory

Jialong Liu[1], Wenjun Tang[1], Deyun Chen[1,2], Chen Jiang[1], Huazhong Yang[1] and Xueqing Li[1]
[1]: Department of Electronic Engineering, BNRist, Tsinghua University, Beijing, China
[2]: Key Laboratory of Advanced Sensor and Integrated System, Tsinghua Shenzhen International Graduate School
{liujl19,twj21,chendy22}@mails.tsinghua.edu.cn,{chenjiang,yanghz,xueqingli}@tsinghua.edu.cn

## ABSTRACT

In-sensor computing has emerged as a promising approach to mitigating huge data transmission costs between sensors and processing units. Recently, the emerging application scenarios have raised more demands of sensory technology for large-area and flexible integration. However, with thin-film technologies that are capable of providing flexible and large-area integration support, the implementation of in-sensor computing can be strongly restricted due to the low device performance, large-area integration variation, and costly interface between sensors and CMOS processors. To address this challenge, we propose an in-sensor computing architecture to facilitate high-parallelism NN pre-processing and effective data compression. The boundaries of computing parallelism are expanded by adopting compact ROM-based compute-in-memory scheme next to sensing array. Differential-frame computing provides not only excellent robustness, but also high data sparsity. A bio-inspired data compression method with residual recovery caches and zero-skip circuits further enhances output sparsity without accumulated error. Based on the proposed cross-layer design optimization, an LTPS TFT-based ROM CiM chip has been fabricated and experimentally measured. The system-level evaluation demonstrates 3.85× speedup and 5.10× energy efficiency improvement compared with traditional architecture with separated sensors and processors, outperforming existing in-sensor computing works in large-area thin-film technology scenarios.

## KEYWORDS

Compute-in-memory, in-sensor computing, thin-film transistors

**ACM Reference Format:**
Jialong Liu[1], Wenjun Tang[1], Deyun Chen[1,2], Chen Jiang[1], Huazhong Yang[1] and Xueqing Li[1]. 2024. Cross-Layer Exploration and Chip Demonstration of In-Sensor Computing for Large-Area Applications with Differential-Frame ROM-Based Compute-In-Memory. In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3657324


*DAC '24, June 23–27, 2024, San Francisco, CA, USA*

ACM ISBN 979-8-4007-0601-1/24/06.
https://doi.org/10.1145/3649329.3657324

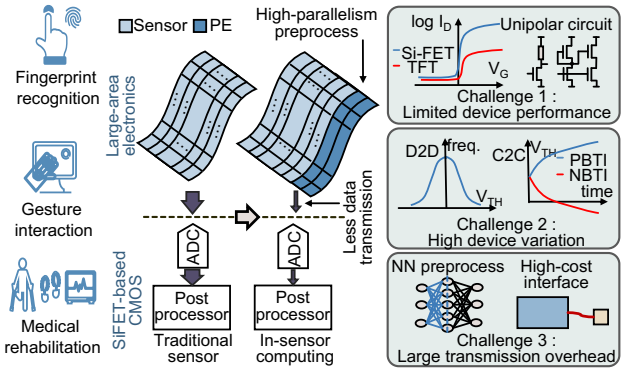


**Figure 1: The opportunities and challenges of large-area in-sensor computing based on thin-film technology.**

## 1 INTRODUCTION

With development of internet-of-things (IoT), smart edge devices have surged to process and transmit massive data, resulting in huge energy consumption and long delays. To tackle this problem, in-sensor computing, i.e., processing raw sensing data instantly in (or near) the sensors with reduced transmitted data, has become a promising solution for data-intensive smart edge scenarios.

Recently, thin-film transistors (TFTs) have enabled abundant large-area flexible sensing tasks, including medical monitoring [1], flexible e-skins [2] and large-area tactile sensing [3]. These tasks also benefit from in-sensor computing, as shown in Fig.1. However, previous in-sensor computing works with TFT technologies suffer from low on-state current, difficult bipolar integration, as well as high device-to-device (D2D) and cycle-to-cycle (C2C) variations. Moreover, the general-purpose neural networks (NNs) usually bring larger data volume after the first layer [4], which is especially harmful when considering the long wiring and high transmission cost between TFT and CMOS chips.

To address these challenges, this work proposes an in-sensor computing architecture with TFT technology. This work is capable of high-parallel and robust data processing and compression for large-area smart sensing applications. The architecture adopts ultra-dense TFT ROM-based compute-in-memory (CiM) to accelerate the process of the first NN layer, compensating for the low performance of TFT devices. A differential-frame (diff-frame) computing method is proposed, which not only eliminates the impact of D2D and C2C variation of $V_{th}$, but also enables high output data sparsity. A bio-inspired compression method supported by residual recovery caches and zero-skip shift-registers is proposed to further compress data with no more accumulated error. We fabricate and

measure a ROM-based CiM chip with 4$\mu m$ low-temperature poly-Si (LTPS) TFT, based on which comprehensive circuit- and system-level evaluations are carried out. The contributions of this paper include:

- First exploration of utilizing compact ROM-based CiM to break the parallelism restriction for large-area sensing; simultaneous computing of more than 20 output channels for each sensing row is achieved with <10% area overhead.
- Diff-frame computing not only enabling robust analog-field computing with 16× less output variation, but also achieving high data sparsity in continuous sensing applications;
- A bio-inspired diff-frame data compression method to enhance output sparsity with no more accumulated error; more than 30% average sparsity improvement is achieved among three different tasks.
- Chip demonstration of LTPS-TFT CiM, as well as the system-level evaluations based on that, exhibit 3.85× speedup and 5.10× energy efficiency improvement on average, compared with traditional architectures.

The rest of this paper is organized as the following order: Section 2 introduces the backgrounds of in-sensor computing and diff-frame processors. Section 3 describes the proposed in-sensor computing architecture in detail. Section 4 reports the measurement and performance evaluation results of the proposed architecture. Section 5 gives discussion and conclusion of this paper.

## 2 PRELIMINARIES

### 2.1 The Opportunities and Challenges of In-Sensor Computing

By taking the similar concepts of in-memory computing [5–7] into sensors, the emerging in-sensor computing also shows its potential in smart sensing tasks [4, 8–14]. Recently, adopting in-sensor computing to accelerate the first several layers of neural networks (NNs) has been verified by a series of smart sensing works [8–10]. Although these inspiring works demonstrate the great potential of in-sensor computing, directly transferring these works to large-area sensing scenarios may not work due to much higher cost for necessary operations in thin-film technology. Moreover, these works do not consider the large-area attribution, which may bring opportunities to break the limit of in-sensor computing parallelism.

### 2.2 Promote Smart Large-Area Sensing with In-Sensor Computing

TFT devices, which are the basis of large-area sensing and display industry, are becoming the ideal candidate for large-area in-sensor computing [15]. Far beyond traditional sensing and displaying applications, recent researches have shown new technology paths of TFT devices for processors [16], low-power long-retention memory[17] and versatile analog circuits [18]. These works prove that TFT can achieve all the functions of CMOS circuits despite its relatively low performance and high device variation. There are also several TFT-based works aiming at in-sensor pre-processing [11, 12] and data compression [13, 14]. Although most of these works are at the early research stage with only simple functions, they have shown great potential of large-area in-sensor computing paradigm.

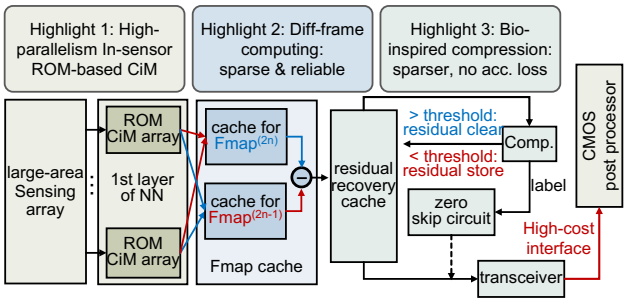


**Figure 2: The framework of proposed TFT-based in-sensor computing architecture.**

However, in most of neural network architectures, the size of intermediate feature maps are larger than input data [4]. This data transmission overhead becomes unacceptable since the interface of TFT and CMOS chips is time- and energy-consuming [10]. On the other hand, the works aiming at compressed sensing give up the advantage of high-parallelism computing and need an extra decoder process [13, 14].

### 2.3 Time-Domain Solution: Diff-Frame Sparse Processors

A possible solution for the dilemma of pre-processing and data compression is to conduct diff-frame sparse computing to reduce the computing budget of continuous inference of CNN. The computing procedure of convolution layer can be expressed in (1):

$$y = \omega * x + b \tag{1}$$

where $*$ stands for the 2-D convolution operation. If diff-frame operation is adopted, the output can be re-written as (2):

$$y^{(t)} = \omega * (x^{(t-1)} + \delta x) + b = y^{(t-1)} + \omega * \delta x \tag{2}$$

With diff-frame inputs, diff-frame feature can be computed by performing $\omega * \delta x$. Accumulating this feature will equivalently derive normal CNN computing results. This method benefits from higher sparsity of diff-frame input with no mathematically approximation, which is utilized to accelerate CNN in several works[19–21].

The existing diff-frame processor works demonstrate the potential to transmit diff-frame data from sensors. However, they are mainly focusing on the processor-side architecture design. To generate and pre-process diff-frame data, the architecture of sensor-side needs to be designed carefully.

## 3 PROPOSED TFT-BASED IN-SENSOR COMPUTING ARCHITECTURE

The proposed TFT-based in-sensor computing architecture is described in Fig. 2. The architecture consists of a large-area sensing array and three highlighted modules: **near-sensor ROM-based CiM arrays**, **diff-frame computing module**, and **bio-inspired compression module**. The raw data from sensors will go through these modules sequentially, after which only the processed and compressed data will be transmitted through the high-cost signal channel connecting TFT and CMOS chips. The efficient pre-processing and data compression improve the latency and energy efficiency of the smart sensing system. The detailed descriptions of the three highlight modules are in Section 3.1, 3.2 and 3.3, respectively.

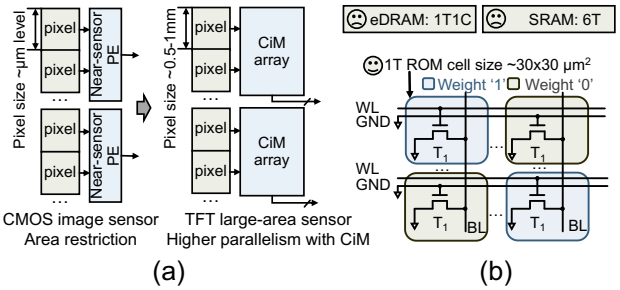


**Figure 3: CiM for large-area sensing: (a) opportunities of high-parallelism in-sensor computing; (b) the schematic of the compact 1T ROM CiM.**

## 3.1 The Near-Sensor ROM-Based CiM Arrays

Different from most CMOS-based in-sensor computing works which only output one data per sensing row, the larger area applications make it possible to place CiM arrays nearby to improve parallelism, as shown in Fig. 3(a). With the compact 1T structure, ROM structure becomes an optimal candidate of CiM, enabling highest density to implement NN's first layer within the width of several rows of sensing pixels. Moreover, some works have shown the capability of NNs to tackle various tasks with the weights of first layer fixed [22], compensating for the inflexibility of ROM CiM arrays.

The schematic of the proposed ROM CiM is shown in Fig. 3(b). Weights are represented by the connection way of transistors. When computing, the input data are sent through WLs while BLs are clamped to a certain voltage. The cells storing logic '0' (gate connected to gnd) keep in the cutoff region, while the cells storing logic '1' (gate connected to WL) are set into linear region with relatively high WL voltage. As a result, the BL current can be represented in (3):

$$I_{BL} = \sum_{i=1}^{N} k[(V_{gs,i} - V_{th,i})V_{ds} - \frac{1}{2}V_{ds}^2]w_i \tag{3}$$

where $w_i = 0/1$ is the binary weight storage.

The proposed ROM CiM-based near-sensor processor is shown in Fig. 4. Multiple BLs are implemented in a single row to achieve 2D readout kernel. It is cost-effective to achieve higher parallelism and simpler scheduling at the cost of area. First, we duplicate multiple CiM arrays to store all weights with different permutations according to each output connection case, avoiding complicated scheduling. Second, we further expand the readout kernel to increase the computing parallelism. The ROM CiM array operates analog-binary MAC computing of all output channels simultaneously, while a local bit-combination module is used to combine the split computing results in analog field. All the output data are then sent into the readout circuit and analog buffer for successive operations.

## 3.2 Diff-Frame Computing Scheme

The high $V_{th}$ variation of TFT device becomes the greatest challenge to implement reliable in-memory computing in analog field. Although there are lots of solutions to compensate for $V_{th}$ variation [23], it is crucial for the ROM CiM cells to be compact without such extra structures.

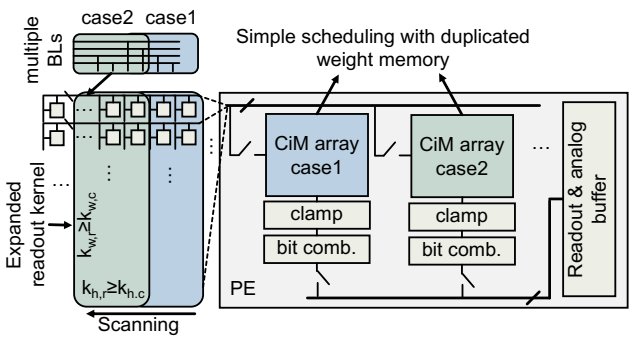


**Figure 4: The structure of near-sensor processor based on TFT ROM CiM.**

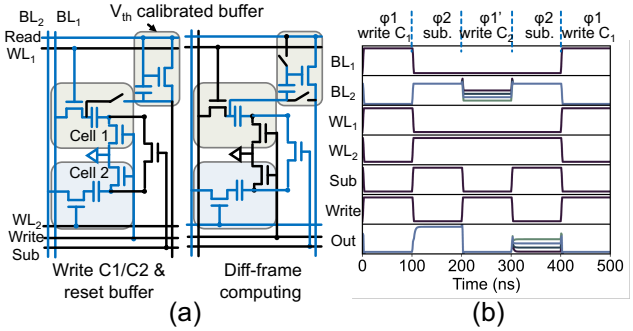


**Figure 5: The illustration of the proposed diff-frame computing module: (a) schematic; (b) simulated waveform.**

We propose a diff-frame computing method to achieve high robustness without additional structures. The diff-frame computing subtracts the computing results of two consecutive frames, and the differential output can be written in (4):

$$I_{BL}^{(n)} - I_{BL}^{(n-1)} = \sum_{i=1}^{N} kV_{ds}(V_{gs,i}^{(n)} - V_{gs,i}^{(n-1)})w_i \tag{4}$$

After diff-frame computing, the output current represents the multiplication of differential input and weight. Moreover, the term with $V_{th}$ is eliminated in the formula, which means the impact of high device variation and severe BTI effect can be significantly reduced.

The diff-frame computing is realized with specially-designed caches consisting of two 1T1C DRAM cells (C1 and C2) and a $V_{th}$ calibrated source follower, which is shown in Fig. 5(a). The processed results of one frame are stored in C1 and C2 alternatively. When one frame is processed, the switch T3 turns on and connects the two capacitors in serial, achieving the diff-frame computing. The BL2 is set to a high voltage to ensure the output source follower works in the correct region. The $V_{th}$ calibration structure can be reset during the writing phase of C1 and C2, without taking extra clock cycles. The simulated waveform is shown in Fig. 5(b).

## 3.3 Bio-Inspired Data Compression Method

The sparsity of diff-frame data is often less than 40%, which cannot be directly used to optimize data transmission. However, directly pruning small data to zero will cause severe accumulated error. This dilemma results in challenges to further compress output data.

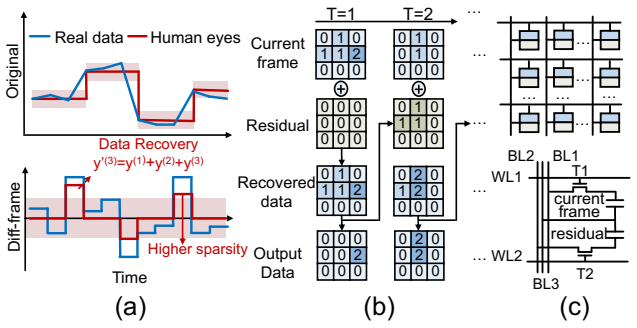


**Figure 6: The proposed bio-inspired compression method: (a) the illustration of filtering function of human eyes; (b) the procedure of the proposed method; (c) the schematic of the proposed residual recovery cache.**

Inspired by the mechanism of human eyes, we implement a data compression method to enhance the diff-frame data sparsity. The mechanism, which is concluded by Weber's law (shown in Fig. 6(a)), describes the filtering function of human eyes [24]. This law demonstrates that ignoring slight fluctuations will not influence the vision information collection. Changing the perspective from raw input data to diff-frame data, this law illustrates the possibility to prune the diff-frame data under a threshold if the caused error can be recovered in the successive frames. Therefore, the proposed method imitates this procedure by preserving pruned data, i.e., residuals, and add them to the next frame, as shown in Fig. 6(b).

The residual recovery is achieved with the specially-designed cache, which is shown in Fig. 6(c). The proposed cache cell consists of two eDRAM cells to store both current diff-frame data and the residuals. A third BL is connected to the common node. When reading data into the transmission module, BL3 is set to be floating, combining two cells to directly read the summation of diff-frame data and residuals. Before the final transmission, zero-skip shift registers with bypass branches [25] are adopted to skip unexpected outputs without cost of clock cycles.

# 4 THE PERFORMANCE EVALUATION

## 4.1 Chip Measurement and Circuit-Level Simulation

We have fabricated large-area LTPS TFT chips consisting of 3×5 ROM-based CiM arrays, with each CiM array containing 32×128 cells. The 30×30 $\mu m^2$ 1T ROM CiM cell enables ultra-high memory density with the 4$\mu m$ LTPS technology. The chip and test platform are shown in Fig. 7(a)(b). In pursuit of high energy efficiency, the proposed TFT ROM CiM adopts a BL discharging scheme to sense the CiM output current with low operation voltages, which is friendly for TFT-based control circuitry. In the chip measurement, the input voltage is set to a range from 2.4V to 3.0V to ensure high linearity, while the precharge voltage of output BL is 1V. The measurement results are shown in Fig. 7(c)(d)(e), which demonstrate high computing linearity, verifying the feasibility of TFT ROM-based CiM scheme.

Based on the measurement of TFT-based ROM CiM arrays, device model is extracted and utilized in further circuit- and system-level simulation. By extracting parameters from the measurement results.

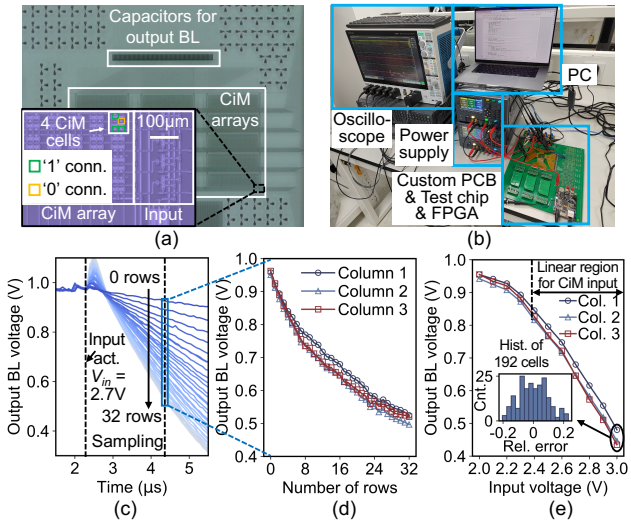


**Figure 7: The chip photograph and measurement results. (a) The fabricated TFT-ROM-based CiM chip; (b) test platform; (c) the readout waveform with different number of activated rows and (d) sampled output; (e) output v.s. input voltage.**

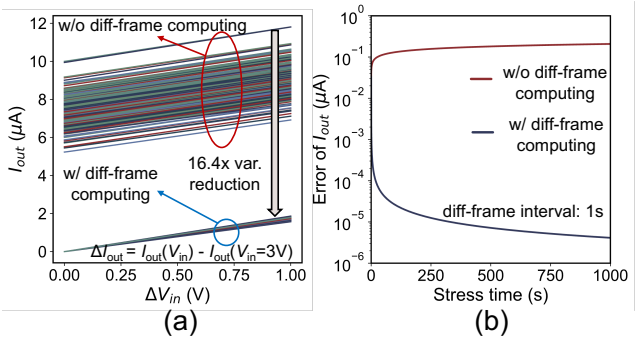


**Figure 8: The robustness analysis of proposed diff-frame computing with: (a) D2D variations; (b) C2C variations.**

Fig. 8 describes the Monte Carlo simulation results of diff-frame computing module with D2D and C2C $V_{th}$ variations. According to the measurement results, $\sigma_{V_{th}}$ is set to be 0.3V, while a variation of device mobility with $\sigma_{\mu_0}/\mu_0 = 2\%$ is also considered. A BTI model for TFT [26] is adopted for C2C variation evaluation. With such high device variations, traditional CiM scheme without diff-frame computing can hardly generate correct results. On the contrary, the diff-frame computing almost eliminates the impact of $V_{th}$ variation, achieving high-precision computing by reducing the output variation by more than one order of magnitude.

## 4.2 System-Level Evaluation

***Design Space Exploration***: We assume the size of readout kernel and convolution kernel to be $k_{h,r} \times k_{w,r}$ and $k_{h,c} \times k_{w,c}$, respectively. The output channel and weight precision of the first layer are set to be $c_o$ and $b$ bits. The PE is required to be placed within the width of $k_{w,r} - k_{w,c} + 1$ rows to generate outputs in one-shot. This restriction can be described by (5):

$$w_{pixel} \geq \frac{k_{h,r} k_{w,r} w_{ROM} + w_{peri}}{k_{w,r} - k_{w,c} + 1} + (k_{h,r} - k_{h,c} + 1) c_o w_{wire} \quad (5)$$

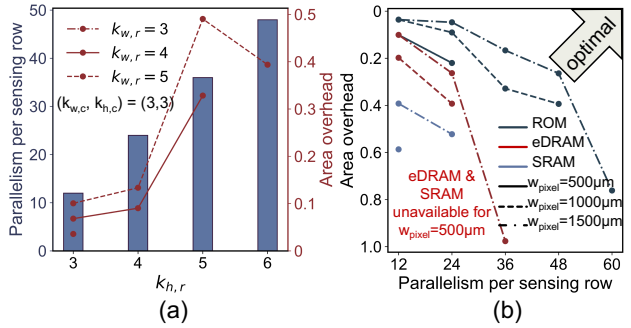


**Figure 9: The design space exploration to optimize parallelism and area overhead: (a) The design space with different $(k_{h,r}, k_{w,r})$ for ROM CiM; (b) the Pareto frontier of different CiM technologies and pixel sizes.**

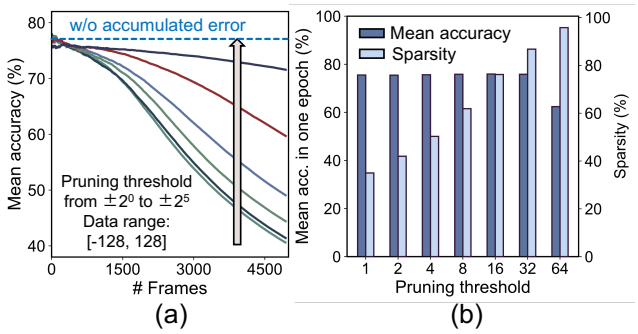


**Figure 10: The evaluation of the proposed data compression method with different pruning threshold: (a) The continuous inference simulation results; (b) the sparsity and accuracy results.**

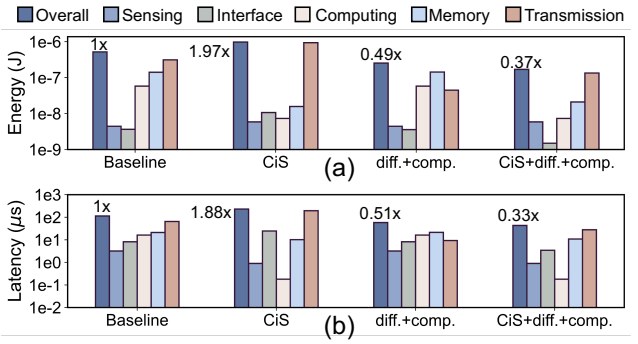


**Figure 11: The breakdown of proposed architecture w/ or w/o in-sensor computing and compression modules: (a) energy breakdown; (b) latency breakdown.**

where $w_{pixel}$, $w_{ROM}$, $w_{peri}$ and $w_{wire}$ represent the width of different parts in the layout. We investigate the output parallelism and area overhead (defined by (6) and (7)) under different configurations.

$$Parallelism = c_o(k_{h,r} - k_{h,c} + 1) \tag{6}$$

$$AreaOverhead = \frac{A_{PE}}{A_{sensor}} \tag{7}$$

The design space for ROM CiM with different $(k_{w,r}, k_{h,r})$ is explored and shown in Fig. 9(a). Only the valid configurations satisfying restriction (5) are depicted in the figure. The Pareto frontier of the design space is extracted and compared to other CiM technologies in Fig. 9(b). Thanks to the ultra-high memory density, The ROM-based CiM outperforms the other two CiM technologies by supporting higher parallelism with lower area overhead.

***Setup for system-level evaluation***: A system-level simulator is then built, taking the sensors, TFT-based CiM arrays, transceivers and other involved circuits into consideration. We adopt the $4\mu m$ LTPS TFT model based on our measurement results. The performance of circuit modules is derived by either circuit-level simulation or referring to other works [10, 23]. The Monte Carlo simulation results in Fig. 8 are considered in error modelling of both CiM arrays and analog caches.

Two relevant tactile sensing datasets (STAG [27] and HAART [28]) and a well-known video dataset (UCF-101 [29]) are evaluated in this work. The first layers of the chosen networks are implemented on the near-sensor CiM array, optimized with 12 output channels and $stride = 2$ convolution for less output data.

***Verification of bio-inspired data compression method***: The adopted neural network achieves accuracy of 76.58% and 75.24% with software and hardware implementation on STAG dataset, as is shown in Fig. 10. We quantize both the weights and transmitted data to 8bit. No significant accuracy loss is observed as the threshold increases up to 32 (quarter of the data range). Moreover, the narrower transmission range can even improve the immunity of accumulated error caused by the ADC quantization on the processor-side. By utilizing the proposed compression method with <1% accuracy loss, the sparsity is enhanced over 50% for STAG and HAART datasets. For more difficult UCF-101 dataset, over 15% improvement of sparsity is still achieved. The contribution of the data compression method on three tasks are listed in Table 1.

***Performance simulation results***: The entire procedure from sensing to storing feature maps of the first layer into memory is evaluated. The baseline is set to be a traditional architecture consisting of separated sensors, memory and processors. The processor is evaluated with 65nm TSMC technology by Synopsys Design Compiler. The evaluation results of ablation studies are listed in Table 1. We denote our three methods: in-sensor computing, diff-frame computing and bio-inspired compression as *CiS*, *diff.* and *comp.*, respectively. With all three methods, our proposed architecture achieves 3.85× speedup and 5.10× energy efficiency improvement on average compared with baseline architecture.

The latency and energy consumption breakdown of the proposed architecture on the STAG dataset is shown in Fig. 11. We also analyse the contribution of both in-sensor computing (*CiS*) and data compression (*diff. + comp.*). Since the costs of computing and data transmission are comparable in this scenario, the improvement of performing either in-sensor computing or data compression is limited. With our proposed architecture, the efficient data processing and compression can be both conducted, achieving significant performance improvement.

***Benchmarking***: Fig. 12 shows the benchmarking results of the recent in-sensor computing works. To have a fair comparison getting rid of the impact of diverse tasks and various technologies, we evaluate these works using our simulator under the same thin-film technology. For the works adopting data compressing algorithms ([4, 13, 14]), extra decoding computation is considered. The same configuration of the first NN layer is applied in the works with in-sensor NN processing ([8, 9]). The data are obtained by averaging

**Table 1: The Performance Improvement with the Proposed Methods**

| Dataset | Accuracy | | Latency per frame ($\mu s$) | | | | Energy consumption per frame ($\mu J$) | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | SW | HW | baseline | CiS | CiS+diff. | **CiS+diff.+comp.** | baseline | CiS | CiS+diff. | **CiS+diff.+comp.** |
| STAG | 76.58% | 75.24% | 117 | 231 | 150 | **43.3** | 0.520 | 0.977 | 0.630 | **0.170** |
| HAART | 82.23% | 80.28% | 181 | 231 | 82.8 | **32.9** | 0.913 | 0.991 | 0.353 | **0.138** |
| UCF101 | 94.19% | 92.56% | 3.70e4 | 2.22e4 | 1.38e4 | **1.10e4** | 332.1 | 106.9 | 70.86 | **58.94** |

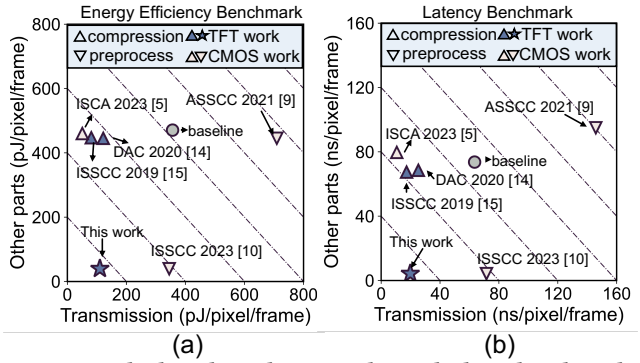


**Figure 12: The benchmarking results with the related works on: (a) energy efficiency; (b) latency. All results are evaluated under unified TFT technology and application scenario.**

the performance on the three datasets [27–29]. We split the cost into two parts, i.e., the transmission part and other parts including sensing, computing and memory fetching, to show the tradeoff between data compression and NN processing. The benchmarking results show the superiority of this work in large-area smart sensing scenarios, achieved by not only supporting simultaneous in-sensor processing and data compression, but also making specific optimizations for large-area applications.

## 5 CONCLUSION

This work proposes a TFT-based in-sensor computing architecture adopting ROM-based diff-frame computing and data compression. The cross-layer exploration with chip demonstration, circuit-level simulation and behavior-level evaluation demonstrates 3.85× speedup and 5.10× energy efficiency improvement, proving the great potential of the proposed architecture. This sensor-side optimization can be further integrated with other energy-efficient processors to achieve even better performance. With the development of large-area technology, the proposed in-sensor computing architecture is expected to become more versatile, fast and energy efficient in the future.

## ACKNOWLEDGEMENTS

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFA0706100; in part by NSFC under Grant U21B2030, Grant 92264204 and Grant 82151305.

## REFERENCES

[1] X. Huang et al. A 256-channel actively-multiplexed $\mu$ecog implant with column-parallel incremental $\Delta\Sigma$ ADCs employing Bulk-DACs in 22-nm FDSOI technology. In *ISSCC*, volume 65, pages 200–202. IEEE, 2022.
[2] K. Takei et al. Physical and chemical sensing with electronic skin. *Proceedings of the IEEE*, 107(10):2155–2167, 2019.
[3] Y. Tang et al. Flexible, transparent, active-matrix tactile sensor interface enabled by solution-processed oxide TFTs. In *IEDM*, pages 24–3. IEEE, 2022.
[4] T. Ma et al. LeCA: In-sensor learned compressive acquisition for efficient machine vision on the edge. In *ISCA*, pages 1–14, 2023.
[5] C. Zhou et al. Ml-hw co-design of noise-robust tinyml models and always-on analog compute-in-memory edge accelerator. *IEEE Micro*, 42(6):76–87, 2022.
[6] F. Tu et al. MuITCIM: A 28nm 2.24$\mu j$ /token attention-token-bit hybrid sparse digital cim-based accelerator for multimodal transformers. In *ISSCC*, pages 248–250. IEEE, 2023.
[7] P. Chen et al. A 22nm delta-sigma computing-in-memory ($\Delta\Sigma$ CIM) SRAM macro with near-zero-mean outputs and LSB-first ADCs achieving 21.38 tops/w for 8b-mac edge AI processing. In *ISSCC*, pages 140–142. IEEE, 2023.
[8] H. Xu et al. A 4.57 $\mu$w@ 120fps vision system of sensing with computing for bnn-based perception applications. In *ASSCC*, pages 1–3. IEEE, 2021.
[9] T. H. Hsu et al. A 0.8v intelligent vision sensor with tiny convolutional neural network and programmable weights using mixed-mode processing-in-sensor technique for image classification. In *ISSCC*, pages 1–3. IEEE, 2022.
[10] S. Sadasivuni et al. In-sensor neural network for high energy efficiency analog-to-information conversion. *Scientific reports*, 12(1):18253, 2022.
[11] W. Rieutort-Louis et al. A large-area image sensing and detection system based on embedded thin-film classifiers. *IEEE JSSC*, 51(1):281–290, 2015.
[12] T. Ohmaru et al. A 25. 3 $\mu$ w at 60 fps 240 × 160 pixel vision sensor for motion capturing with in-pixel nonvolatile analog memory using CAAC-IGZO FET. *IEEE JSSC*, 51(9):2168–2179, 2016.
[13] L. Shao et al. Robust design of large area flexible electronics via compressed sensing. In *DAC*, pages 1–6. IEEE, 2020.
[14] L. E. Aygun et al. Hybrid system for efficient LAE-CMOS interfacing in large-scale tactile-sensing skins via TFT-based compressed sensing. In *ISSCC*, pages 280–282. IEEE, 2019.
[15] W. Tang et al. Computing-in-memory with thin-film transistors: challenges and opportunities. *Flexible and Printed Electronics*, 7(2):024001, 2022.
[16] J. Biggs et al. A natively flexible 32-bit arm microprocessor. *Nature*, 595(7868):532–536, 2021.
[17] A. Belmonte et al. Capacitor-less, long-retention (> 400s) DRAM cell paving the way towards low-power and high-density monolithic 3d dram. In *IEDM*, pages 28–2. IEEE, 2020.
[18] G. Cantarella et al. Review of recent trends in flexible metal oxide thin-film transistors for analog applications. *Flexible and Printed Electronics*, 5(3):033001, 2020.
[19] Y. Yang et al. A 65-nm energy-efficient interframe data reuse neural network accelerator for video applications. *IEEE JSSC*, 57(8):2574–2585, 2021.
[20] M. Riera et al. Computation reuse in DNNs by exploiting input similarity. In *ISCA*, pages 57–68. IEEE, 2018.
[21] B. Pan et al. Recurrent residual module for fast inference in videos. In *CVPR*, pages 1536–1545, 2018.
[22] Y. Chen et al. Yoloc: deploy large-scale neural network by rom-based computing-in-memory using residual branch on a chip. In *DAC*, pages 1093–1098, 2022.
[23] C. Pai and Y. Tai. P-44: A new analogue buffer using poly-Si TFTs with deviation less dependent on the gray level for active matrix displays. In *SID Symposium Digest of Technical Papers*, volume 36, pages 438–441. Wiley Online Library, 2005.
[24] E. Steinbach et al. Haptic data compression and communication. *IEEE Signal Processing Magazine*, 28(1):87–96, 2010.
[25] M. Guo et al. A 3-wafer-stacked hybrid 15mpixel cis+ 1 mpixel evs with 4.6 gevent/s readout, in-pixel TDC and on-chip ISP and ESP function. In *ISSCC*, pages 90–92. IEEE, 2023.
[26] C. Chen et al. A reliability model for low-temperature polycrystalline silicon thin-film transistors. *IEEE EDL*, 28(5):392–394, 2007.
[27] S. Sundaram et al. Learning the signatures of the human grasp using a scalable tactile glove. *Nature*, 569(7758):698–702, 2019.
[28] X. L. Cang et al. Different strokes and different folks: Economical dynamic surface sensing and affect-related touch recognition. In *International Conference on Multimodal Interaction*, pages 147–154, 2015.
[29] K. Soomro et al. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.