INSPIRE: In-Sensor Compressed Weight Retrieval for Enhancing ViT Efficiency at Edge

Sabbir Ahmed[1,*], Deniz Najafi[2,*], Mohaiminul Al Nahian[1,*], Navid Khoshavi[3], Abdullah Al Arafat[4], Mamshad Nayeem Rizve[5], Mahdi Nikdast[6], Adnan Siraj Rakin[1], Shaahin Angizi[2]
[1]State University of New York at Binghamton, USA [2]New Jersey Institute of Technology, USA [3]AMD, USA
[4]Florida International University, USA [5]Adobe Inc., USA [6]Colorado State University, USA
sahmed9@binghamton.edu, dn339@njit.edu, arakin@binghamton.edu, shaahin.angizi@njit.edu
[*] Equal Contributions

***Abstract*—Deploying Vision Transformer (ViT) models on edge devices poses significant challenges due to the high bandwidth, energy demands, and latency associated with transmitting large weight parameter sets to the sensing unit, along with limited on-chip memory resources, which are often insufficient for storing these parameters. To address these constraints, we present a software-hardware co-design framework that incorporates a novel in-sensor Compressed Weight Retrieval mechanism within an intelligent vision sensor. This framework offers two key contributions. First, we propose an innovative hardware-friendly weight compression algorithm that substantially reduces bandwidth and power consumption by optimizing on-chip memory usage for storing weight parameters. Second, we leverage the exceptional efficiency of Silicon Photonic (SiPh) devices and design a novel in-sensor accelerator called INSPIRE for the first time to perform in-sensor retrieval of the compressed weights and parallel fine-grained convolution operations next to the pixel array, enabling low-power adaptable ViT inference on resource-constrained edge platforms. Our extensive simulation results show that INSPIRE can remarkably reduce the memory footprint of ViT results with favorable accuracy. Besides, INSPIRE significantly reduces the bandwidth and power requirements associated with storing weight parameters in on-chip memory. INSPIRE achieves up to 245.4 Kilo FPS/W and reduces the data transfer energy by a factor of ∼11× on average compared with 4-bit quantized ViTs.**

I. Introduction

Most Internet of Things (IoT) vision sensors still rely predominantly on cloud-based decision-making due to limited on-device resources, mainly memory, resulting in high power consumption data transmission as well as compute-intensive algorithms and inefficient data usage. It has been reported that in edge vision sensors, over 96% of the total power consumption is attributed to the processes of pixel value conversion and storage [1], [2].

On the hardware implementation front, recent advancements in CMOS image sensor technology have focused on enhancing Deep Neural Network (DNN) processing capabilities. A notable strategy involves integrating CMOS image sensors and processors onto a single chip, a configuration known as Processing-Near-Sensor (PNS) technology [3], [4], illustrated in Fig. 1(a). An even more advanced technique, termed Processing-In-Sensor (PIS), incorporates computational units within each pixel [2], facilitating data processing at the pre-Analog-to-Digital Converter (pre-ADC) stage before transmission to either on-chip or off-chip processors. Despite these innovations, several challenges persist. High energy consumption in components such as Analog-to-Digital Converters

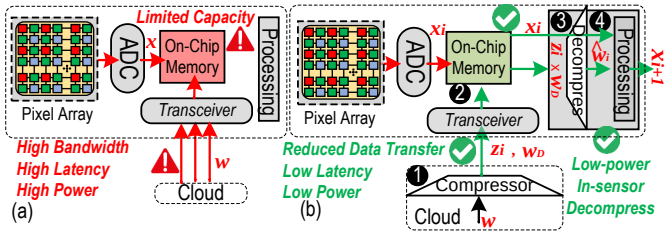

Fig. 1: *Overview of (a) Existing processing-near-sensor platforms and challenges, (b) our proposed system.*

(ADCs) and Digital-to-Analog Converters (DACs) within PIS and PNS architectures, limited on-chip weight storage capacity, and the substantial power and bandwidth required for data transfer to the cloud impose significant constraints, as depicted in Fig. 1(a), limiting the feasibility of deploying DNNs adjacent to pixel arrays [1], [5]. Besides, data transmission on most advanced 4G and 5G networks to edge devices involves critical trade-offs in terms of power consumption and latency performance, where the power requirements to maintain high data rates and ensure signal reliability are substantial.

On the software implementation front, sophisticated models have continued to grow in size and complexity, demanding substantial memory and energy resources. The Vision Transformer (ViT) architecture has ushered in a new era in machine learning, demonstrating exceptional performance and even surpassing traditional Convolutional Neural Networks (CNNs) in many cases. While ViT has achieved remarkable success in image classification tasks, it presents additional challenges for edge device deployment due to its significant memory and energy demands. Finding the response to sensors' resource constraints has driven significant research into model compression techniques. Strategies such as pruning [6], quantization [7], and neural architecture search [8], together with developments like MobileNets [9], have allowed for the effective deployment of DNN without major sacrifices in accuracy. While existing compression techniques can reduce model size with minimal accuracy loss, they remain impractical for deploying ViTs on edge devices. For instance, DeiT-Small has a parameter size of approximately 85 MB. Parameter compression through existing methods to accommodate on-chip deployment of ViT (Typical memory of about 8 MB) would require more than 10× weight compression, which results in substantial accuracy degradation. The performance degradation arises because aggressive weight compression techniques may result in substantial information loss within the compressed weights.

Consequently, facilitating on-chip development of ViTs necessitates not only more aggressive weight compression strategies but also effective on-chip weight decompression methods to recover lost information and preserve accuracy.

This paper addresses the bandwidth and memory bottlenecks of deploying ViTs on edge devices by proposing a software–hardware co-design that combines in-sensor weight retrieval with an efficient in-sensor Silicon Photonics (SiPh) accelerator. As shown in Fig. 1(b), the proposed framework reduces power, computation, and latency under tight on-chip memory constraints. Our primary contributions in this work are as follows. (1) We introduce INSPIRE as a novel *Encoded Weight Compression & Compressed Weight Retrieval algorithm* designed to significantly reduce the bandwidth and power requirements associated with transmitting and storing weight parameters to on-chip memory. (2) We co-design INSPIRE architecture, as a novel SiPh in-sensor accelerator for enhancing ViT efficiency at edge, for the first time to perform weight decompression and parallel Multiply-and-Accumulate (MAC) operation next to the pixel array; (3) We develop a bottom-up evaluation framework from SiPh device fabrication to application study to demonstrate the effectiveness of proposed technique in deploying ViTs on vision sensors.

II. Background

Vision Transformers Compression. ViT [10] has shown exceptional performance in image classification when trained on large-scale datasets. This led to the emergence of several transformer-based vision models. For instance, DeiT [11] reduced the dependency on large datasets, Pyramid Vision Transformer (PVT) [12] and Swin Transformer [13] incorporated hierarchical feature representations, broadening ViTs' applicability across diverse computer vision tasks. However, ViTs demand substantial memory, limiting their use on edge devices and prompting research into compression techniques.

Compression techniques, such as unstructured pruning [14], removes redundant parameters, but the irregular sparsity patterns are less hardware-efficient. More hardware-friendly structured pruning, for instance, [15] uses binary masks based on parameter importance, while another method [16] combines pruning, layer skipping, and knowledge distillation to compactly represent transformers. Quantization, another common technique, reduces memory use by converting parameters to lower precision [17]. Tailored quantization methods for ViTs include PTQ4ViT [18], which applies twin uniform quantization with Hessian-guided scaling, FQ-ViT [19] utilizes powers-of-two scale quantization, RepQ-ViT [20] decouples quantization from inference to manage extreme activation distributions, and [21] enables data-free quantization. *Despite these advancements, current pruning and quantization approaches still struggle to compress ViTs to meet edge devices' stringent memory, latency, and energy constraints without compromising performance.*

MicroRing Resonators and SiPh Acceleration. SiPh-based accelerators, with high operational bandwidth and solutions for fan-in/fan-out issues, enhance DNN and machine vision tasks

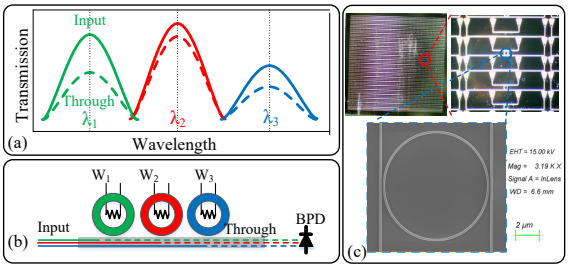

Fig. 2: *(a) MR input and through ports' spectra after imprinting a parameter. By adjusting the MR's resonant wavelength, part of the input signal drops into the ring while the remaining propagates towards the through port, hence imprinting any parameter in the transmitted signals. (b) Multiple MRs in a single arm imprint weight values onto the input signal at different wavelengths. (c) SEM image of our fabricated SiPh integrated MR array.*

[22], [23], [24]. These accelerators are categorized into coherent designs, using a single wavelength for encoding parameters in the optical signal [25], and non-coherent designs, using multiple wavelengths for parallel computations [22], [26]. Microring Resonators (MRs) dynamically manage wavelengths and modulate light intensity in non-coherent systems, acting as input/weights [22], [26] or only weights [27]. MRs enhance Multiply-and-Accumulate (MAC) operations by modifying the transmission spectrum (as shown in Fig. 2) and tuning resonant wavelengths to overlap with input light. The resonant wavelength is calculated as $\lambda_{res} = \frac{n_{eff} \times L}{m}$, where n_{eff} is the effective refractive index, L is the MR's circumference, and m is the resonant mode order [28]. Previous research has explored MR-based photonic technologies for DNN acceleration. LightBulb [24] accelerates binarized Convolutional Neural Networks (CNNs) with photonic XNOR operations but has high power consumption due to ADCs. ROBIN [26] and CrossLight [22] use low-bit-width weights for convolutional layers, relying on DACs/ADCs, which increase footprint and power. Lightator [27] is aimed to realize a near-sensor DNN accelerator enabling compressive acquisition of input frames and fine-grained convolution. *Yet, to our knowledge there is no SiPh acceleration technique has been developed for ViTs.*

III. Proposed INSPIRE Method

We propose INSPIRE algorithm, which is capable of aggressively compressing the weights of a given ViT model through weight encoding to a lower dimension. However, to preserve model performance, it incorporates a decompressor unit that reconstructs the weights back to their original decompressed version at inference, as shown in Fig. 3. Proposed INSPIRE algorithm consists of two components: i) *Encoded Weight Compression* algorithm, which trains a unified compressor-decompressor model end-to-end with the base ViT model, and ii) After training, we only retain the decompressor and compressed weights and perform *Compressed Weight Retrieval* at inference. In section IV, we elaborate on efficiently performing the decompression step using the INSPIRE accelerator.

Encoded Weight Compression. We propose a unified compression-decompression training strategy to train compressor and decompressor networks jointly. The compressor network, denoted as $\mathbf{F}$ with parameters $\mathbf{W}_C$, learns to compress the original model weights $\mathbf{w}$ from $\mathbb{R}^d$ to $\mathbb{R}^r$, where

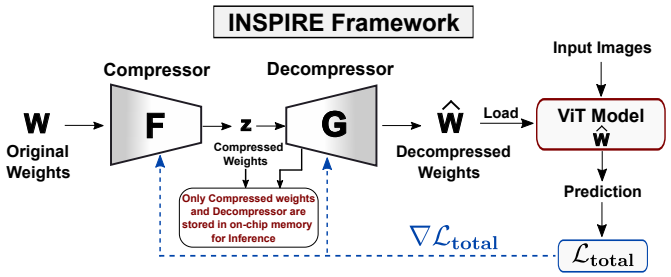

Fig. 3: *Proposed INSPIRE algorithm. The Compressor and Decompressor network parameters are optimized jointly to minimize the loss* $\mathcal{L}_{\text{total}}$ *for training. After training, only compressed weights and the decompressor network are retained for In-Sensor Decompression.*

$r \ll d$. Simultaneously, the decompressor network, $\mathbf{G}$, learns to decompress the compressed weights back to the original weight dimension $\mathbb{R}^d$. For our purposes, we use a single fully connected layer as our decompressor network, $\mathbf{G}$ with weight $\mathbf{W}_D \in \mathbb{R}^{r\times d}$ to reduce computation associated with the decompression step. After training, we no longer need the compressor network and only the compressed weights and the decompressor network are retained for inference. Given a high compression ratio, the compressed weights and the decompressor network $\mathbf{G}$ have the potential to be stored in the on-chip memory of edge devices, effectively eliminating the significant latency and energy consumption associated with off-chip memory access. Additionally, to reduce memory requirements during inference, we decompress each layer sequentially, i.e., after decompressing weights of a given layer, we perform computations on the given layer and store results temporarily in on-chip memory after freeing up any previously saved result. This temporary stored value helps perform inference on the next layer. This sequential decompression strategy ensures that we remain within the budget of on-chip memory.

Consider a ViT model $\mathbf{H}_{\mathbf{W}}(\cdot)$ with L layers, where the weights of the layers are represented by the set $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_L\}$. During training, each layer's weight $\mathbf{w}_i$ is first passed through the compressor network $\mathbf{F}$ to compress them to a lower dimension representation, $\mathbf{z}_i = F(\mathbf{w}_i)$. And the decompressor $\mathbf{G}$ is used to decompress them back to their original dimensionality. So, the reconstructed weight can be expressed as $\hat{\mathbf{W}} = \{\hat{\mathbf{w}}_1, \hat{\mathbf{w}}_2, \ldots, \hat{\mathbf{w}}_L\}$, where $\hat{\mathbf{w}}_i = G(\mathbf{z}_i)$. The compressor parameters $\mathbf{W}_C$ and decompressor parameters $\mathbf{W}_D$ are jointly optimized by minimizing the Mean Squared Error (MSE) between the original and the reconstructed weights. The MSE-based reconstruction loss is defined as:

$$\mathcal{L}_{\mathbf{mse}} = \frac{1}{L}\sum_{i=1}^{L} \|\mathbf{w}_i - \hat{\mathbf{w}}_i\|^2 \tag{1}$$

However, due to the nature of MSE loss optimization, the distribution of reconstructed weight may become significantly different from the original weights even after the optimization. So, relying solely on MSE-based weight reconstruction does not necessarily improve task performance [29]. Hence, we proposed incorporating the base ViT model in the training loop. To do so, we will utilize a Cross-Entropy (CE) and Knowledge Distillation (KD) loss from a pre-trained model during training. CE ensures the ViT model is optimized w.r.t. the ground truth, while KD prevents the decompressed weight parameters from deviating significantly from the pre-trained ViT model's weight distribution.

Note that we do not update the ViT model's weight directly, instead, after obtaining $\hat{\mathbf{W}}$ from the decompressor, the forward pass of the end-to-end training is expressed as $\mathbf{H}_{\mathbf{W}}(\cdot) \rightarrow \mathbf{H}_{\hat{\mathbf{W}}}(\cdot)$, implying the ViT model only uses the decompressed weights for inference. This forward pass computes predictions as $\hat{\mathbf{y}} = \mathbf{H}_{\hat{\mathbf{W}}}(\mathbf{x})$, where $\mathbf{x}$ are input images and $\hat{\mathbf{y}}$ are the predictions. The CE and KD losses are then computed as:

$$\mathcal{L}_{\mathbf{CE}} = -\sum_i \mathbf{y}_i \log(\hat{\mathbf{y}}_i); \quad \mathcal{L}_{\mathbf{KD}} = \sum_i \hat{\mathbf{y}}_{\mathrm{T},i} \log\left(\frac{\hat{\mathbf{y}}_{\mathrm{T},i}}{\hat{\mathbf{y}}_i}\right), \tag{2}$$

where $\hat{\mathbf{y}}_{\mathrm{T}}$ are the predictions from the pre-trained teacher model and KL is the Kullback-Leibler divergence. The total loss function is:

$$\mathcal{L}_{\mathbf{total}} = \mathcal{L}_{\mathbf{mse}} + \mu_1 \cdot \mathcal{L}_{\mathbf{CE}} + \mu_2 \cdot \mathcal{L}_{\mathbf{KD}}, \tag{3}$$

where μ_1 and μ_2 control the contributions of CE and KD losses. Finally, the overall loss is minimized by jointly optimizing the compressor and decompressor parameters, $\mathbf{W}_C, \mathbf{W}_D$.

In summary, this algorithm ensures that the decompressor network not only reconstructs the original weights with minimum reconstruction error but also improves task-relevant performance. After the training is complete, it is sufficient to keep only the decompressor parameters, $\mathbf{W}_D$ and compressed weight representations, $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_L\}$ for reconstructing the original weights.

Compressed Weight Retrieval. During inference, we decode the compressed representation of the weights in a layer-by-layer fashion to stay within the limited on-chip memory constraints of edge devices. Let, $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_L\}$ are the set of compressed weights saved in memory. Consider the forward pass of an input activation $\mathbf{x}_i \in \mathbb{R}^{N\times C}$ in the network. The weight $\mathbf{z}_i$ is matrix multiplied by $\mathbf{w}_D$ to get the decompressed weight, $\hat{\mathbf{w}}_i = (\mathbf{z}_i \times \mathbf{w}_D) \in \mathbb{R}^{C\times d}$. The input activation $\mathbf{x}_i$ goes through matrix multiplication with the resulting decompressed weight to get $\mathbf{x}'_i = \mathbf{x}_i \times \hat{\mathbf{w}}_i \in \mathbb{R}^{N\times d}$. Let, $\mathcal{A}[\cdot]$ be an activation function. Then the output activation for the layer can be defined as $\mathbf{x}_{i+1} = \mathcal{A}[\mathbf{x}'_i]$. This output activation $\mathbf{x}_{i+1}$ is stored in memory, rewriting any previously saved activations and used for inference of the next layer. Typically, the value of the compressed dimension r is chosen so that the compression follows the strict memory budget of a given device.

IV. INSPIRE on Silicon Photonics Substrate

Architecture and Flow. To demonstrate the practicality of INSPIRE algorithm, we introduce an in-sensor SiPh accelerator designed to perform compressed weight retrieval leveraging parallel MAC operations efficiently during inference. Fig. 4 illustrates the proposed architecture with its electronic and photonic building blocks, all of which are co-designed and co-optimized to have a standalone reconfigurable, energy-aware optical vision sensor node to integrate sensing and processing

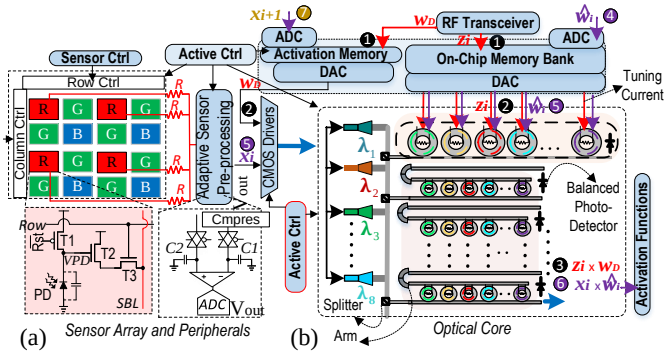

Fig. 4: *Proposed INSPIRE architecture: (a) Sensing component, (b) Processing component enabling in-sensor compressed weight retrieval and computation.*

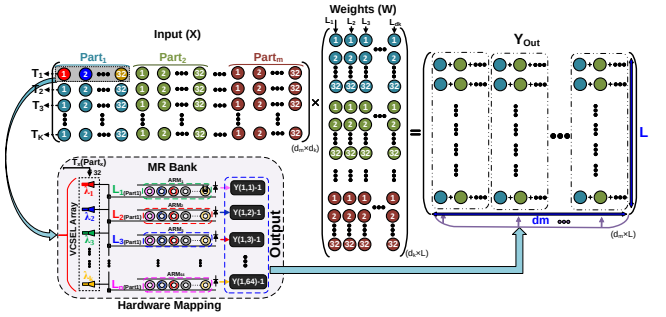

Fig. 5: *Matrix splitting and mapping methodology.*

phases during inference. Such a design can be manufactured leveraging 2.5D and 3D heterogeneous integration. INSPIRE architecture consists of two key components, i.e., the sensor array in Fig. 4(a) and the in-sensor optical core in Fig. 4(b). INSPIRE intrinsically implements a granularity-configurable convolution operation required in a wide variety of vision processing tasks, such as data compression and complex classification as well as processing the layers in ViT in a low-bit-width fashion, to tailor the trade-offs between power consumption and accuracy. A power-aware Active Controller (see Fig. 4(a)) is devised to account for the sensor node's power state while orchestrating different components to maximize the overall performance and efficiency.

During step ❶, the compressed ViT parameters ($\mathbf{z}_i$ and $\mathbf{w}_D$) are received from the cloud using an RF transceiver and written to the on-chip memory banks. In step ❷, the compressed weight values $\mathbf{z}_i$ will be converted to analog signals and applied layer-by-layer to adjust the underlying MR devices within the optical core for weight modulation and decompression. At the same time, $\mathbf{w}_D$ that is initially written to the activation memory will pass through the CMOS drivers designed with integrated Vertical-Cavity Surface-Emitting Laser (VCSEL) sources, to optically feed the values to the optical core. To execute MAC-based decompression operation, a light signal generated by VCSELs, modulated at different wavelengths to carry all necessary $\mathbf{w}_D$ values, travels through an arm embedded with MRs where $\mathbf{z}_i$-weights are mapped. As the light progresses along the arm, each MR modulates the intensity of light at the wavelength linked to that particular MR. At the end of each arm, a Balanced PhotoDetector (BPD) performs the accumulation, enabling MAC operations ($\hat{\mathbf{w}}_i = \mathbf{z}_i \times \mathbf{w}_D$) across each arm within the optical core (❸). In step ❹, $\hat{\mathbf{w}}_i$ is written back to the on-chip weight memory bank. In step ❺, the input image is captured by a power-aware RGB image sensor pixels and pre-processed in an ADC-less fashion following the method presented in [30], co-designed with VCSEL light sources, to optically feed $\mathbf{x}_i$ to the optical core. The CMOS driver is also implemented to support applying previous-layer activations coming from the activation memory. Simultaneously, $\hat{\mathbf{w}}_i$ is imprinted on MRs in the optical core to enable layer-wise MAC operation. The results generated at step ❻ ($\mathbf{x}_{i+1} = \mathcal{A}[\mathbf{x}_i \times \hat{\mathbf{w}}_i]$) can be readily activated and/or written back to the activation memory (step ❼) for the next layer processing. Implementing the activation function at the end of each layer is more efficient in the electronic domain than in the optical domain [22], [26]. Consequently, the electronic component is designated to handle the *GELU* and *Softmax* activation functions.

Photonic-Friendly Hardware Mapping. ViT's large matrices ($d_m \times d_k$) exceed MR capacity. We propose matrix splitting and mapping (Fig. 5), where each optical core has 64 arms and 32 VCSEL-driven wavelengths. Thus, 32 vector elements are injected per cycle, multiplied with pre-tuned weights, and accumulated over slots to reconstruct full computation. This ensures efficient wavelength/arm utilization without oversized arrays. Finally, we embed attention scaling factor $\frac{1}{\sqrt{d_k}}$ into MR tuning. Rather than dividing outputs post-computation, the MR bank is tuned with $\frac{W_K^T}{\sqrt{d_k}}$, eliminating extra division, reducing latency, and avoiding ADC/DAC conversions. Together, decomposition, splitting, and embedded scaling provide hardware-aware mapping that maximizes throughput while minimizing photonic tuning overhead.

V. Experimental Results

Setup. We present a bottom-up evaluation framework, illustrated in Fig. 6, for the comprehensive implementation of INSPIRE. At the device level, MR components were fabricated and meticulously calibrated to achieve up to 8-bit precision (SEM image of the MR array is shown in Fig. 2(c)). The measured data were modeled and co-simulated with interface CMOS circuits using the Synopsys HSPICE tool. At the circuit level, the pixel array, optical core, and peripheral circuits were initially designed with the 45nm Predictive Technology Model (PTM) library in Synopsys HSPICE, allowing for output voltages and currents to be obtained. All other components, excluding the memories, were developed in Synopsys Design Compiler [31]. The memory component was implemented in Cacti [32]. At the application level, we extensively modified the PiPSim [33]. PiPSim serves as a flexible platform, offering a broad sensor array configuration option that allows precise efficiency and accuracy tuning within DNN structures. We designed the INSPIRE with a pixel array size of 600×600 and a 3-transistor 1-photo-diode pixel structure as shown in Fig. 4(a). We let ADC precision vary between 4 and 8 and consider a parallelism level of 3, which means three 3×3 filters are read and computed at the time. The INSPIRE model's peak activation memory is limited to 512kB, while the total on-chip weight memory is capped at three configurations (2MB,

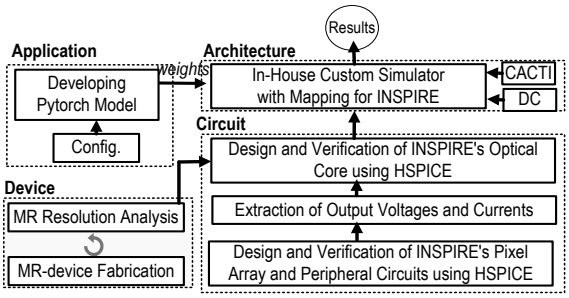

Fig. 6: *Proposed evaluation framework.*

4MB, 8MB). These constraints are configured to align with the optimized memory capabilities of the Cepton Vista-P60 [34] and Espressif ESP32-S3 [35] vision sensors. We compare our INSPIRE performance with a highly-pruned model (termed as P) in [15] and a 4-bit and 8-bit quantized models (Q4 and Q8) [36] deployed onto the same INSPIRE architecture.

The compressor and decompressor networks are trained for 200 epochs with a batch size of 256 and a base learning rate of 1×10^{-4}, using a cosine learning rate scheduler without a warm-up phase. For data augmentation and KD, we apply the standard augmentation techniques and pre-trained distilled DeiT models used in [11], and set $\mu_1 = 1$ and $\mu_2 = 2 \times 10^3$. As a next step, we quantize the compressed weights and the decompressor parameters to 4 bits and perform Quantization Aware Training of compressed weights and the decompressor parameters for an additional 20 epochs without the MSE loss. Similar to [17], we quantize both the patch embedding (first) layer and the classification (last) layer with 8-bit precision. The weight parameters are extracted, quantized, fine-tuned, and scaled to preserve performance before being mapped into the INSPIRE's optical core. The data is then transmitted to the MRs via VCSELs. The modeled VCSEL exhibits a delay of 10 ns and consumes approximately 0.66 mW of power, as reported in [39].

Accuracy Analysis. In Fig. 7, we summarize the DeiT-Small model's compression performance across different pruning and quantization methods on the ImageNet-1K dataset. Among the methods, INSPIRE achieves the highest compression ratio of 15×, reducing the model size to 5.63 MB while maintaining a top-1 accuracy of 78.03%, which is only 1.69% lower than the uncompressed DeiT-S baseline (79.72%). In contrast, WDPruning [15] achieves much lower accuracies of 60.51% with small compression gain of 2.5×, which is not suitable for resource-constrained devices. LSQ [37] and Q-ViT [15] achieve 68% and 71.90% accuracy, respectively, under a 13.6× compression on a 2-bit compression, falling behind INSPIRE accuracy by ∼10% and ∼6%, respectively. Although 4-bit LSQ retains 79.6% accuracy, the 7.46× compression is not competitive. This makes INSPIRE the only method suitable for effective deployment on edge devices with aggressive compression requirements.

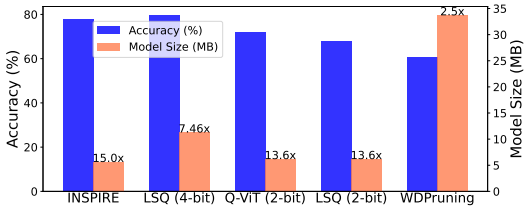

Fig. 7: *Comparison of INSPIRE with SOTA ViT compression methods [15], [37], [17] for compressing DeiT-S [11] trained on Imagenet-1k [38].*

Multi-Objective Performance Optimization. To evaluate the INSPIRE platform performance for edge deployment, it is essential to establish strict constraints on peak activation memory and on-chip weight storage. The DeiT-S model under analysis presents an initial memory footprint of 84.12 MB, which is incompatible with the on-chip memory limitations. Applying our custom INSPIRE algorithm reduces this footprint to 5.63 MB, thereby freeing approximately 2.37 MB on-chip for decompression processes given an 8MB on-chip memory size. However, given a smaller memory size, transferring the complete set of weight parameters to the edge sensor in a single batch is infeasible due to the model's memory profile. Furthermore, energy and latency associated with data transfer, weight loading, and decompression significantly impact the system's overall energy efficiency and performance. These factors must be carefully optimized to meet the INSPIRE's operational constraints and maintain feasible latency and energy profiles. We can cast the problem as a multi-objective optimization problem, where we aim to minimize an overall design cost $C(x)$ while satisfying both intra-level and inter-level constraints. Let x represent the design parameters, and let the cost function combine memory $M(x)$, timing $T(x)$, and energy $E(x)$:

$$\min C(x) = \alpha \cdot M(x) + \beta \cdot T(x) + \gamma \cdot E(x), \qquad (4)$$

where α, β, and γ are coefficients indicating the priority of each term in the objective function. These coefficients can be properly calculated to meet the edge device constraints.

Storage Requirement Analysis. A thorough evaluation of the DeiT-S model's memory requirements is conducted by calculating its peak memory consumption. Using TFLite-Micro as the interpreter, we assess memory usage by tracking the execution sequence of model operations to pinpoint the peak memory utilization. This precise approach ensures that the model adheres to defined limits, aligning with INSPIRE's hardware specifications. We assume a cloud server with a ViT weight parser that transmits weight parameters in variable batch sizes to the edge device. Fig. 8(a) displays the memory footprint breakdown across different configurations: INSPIRE, the 32-bit baseline, Q8, Q4, and P models. Given an 8MB on-chip memory, INSPIRE stands out as the only configuration that does not require batching, allowing the entire model to fit within the chip. The next best fit is the Q4 model, with a footprint of 10.51 MB with batching. To better understand the impact of memory demands, we define a Memory Bottleneck Ratio (MBR), representing the proportion of time computation is delayed while waiting for data transfers between on-chip and off-chip memory, which impacts overall performance and leads to the memory wall effect. This analysis accounts for peak performance and empirically derived data for each platform, considering the number of memory accesses across bit-width configurations. As shown in Fig. 8(b), INSPIRE experiences an MBR of around 14%, while the next closest option, Q4,

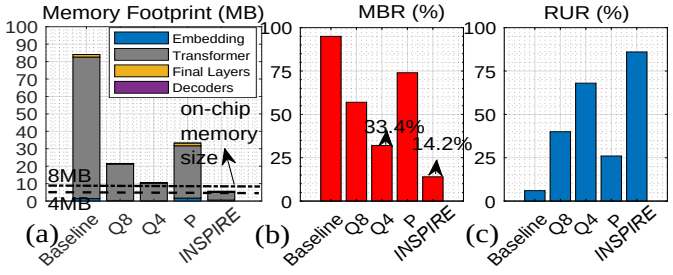

Fig. 8: *(a) Memory footprint in MB, (b) Memory bottleneck ratio, (c) Resource utilization ratio.*

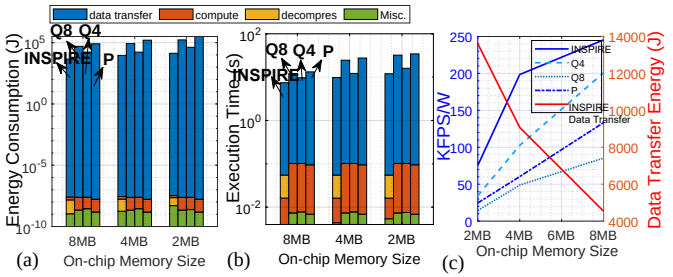

Fig. 9: *Breakdown of (a) Energy consumption, (b) Execution time of INSPIRE with various compression mechanisms vs. on-chip memory size. (c) INSPIRE power efficiency (KFPS/W) vs. data transfer energy.*

shows an MBR of approximately 33.2%. A lower memory wall ratio corresponds to a higher Resource Utilization Ratio (RUR) for the compression algorithm, as illustrated in Fig. 8(c), reinforcing the findings from Fig. 8(b).

Energy Consumption and Execution Time. Fig. 9(a)-(b) illustrates the energy consumption and execution time breakdown for INSPIRE running various compression algorithms across different on-chip memory sizes. We factor in the data transfer component using a 5G protocol with ultra-low latency and high bandwidth (1 Gbps, $\sim 10^{-4}$ J/bit). In cases where the on-chip memory capacity is exceeded by the compression algorithms under test, multiple layers are batched before transmission. In addition, the performance of the INSPIRE optical core in both computation and decompression scenarios is evaluated, as well as transceiver and controller performance included under miscellaneous.

Key observations are as follows: (i) INSPIRE demonstrates the lowest energy consumption for processing DeiT-S with 12 transformer blocks. It achieves $\sim$18.9×, 10.6×, and 4.7× energy efficiency compared to P, Q8, and Q4 mechanisms, respectively. As on-chip memory size decreases, INSPIRE shows even greater efficiency. (ii) In the 8MB configuration, INSPIRE reduces data transfer energy by approximately 17.7×, 11.1×, and 3.8× compared to P, Q8, and Q4 mechanisms, respectively. With further reductions in on-chip memory size, these energy savings become more pronounced, reaching around 32.4×, 19.3×, and 11×, respectively. (iii) In terms of total MAC energy consumption, the INSPIRE mechanism incurs approximately 40% higher energy costs for decompression compared to the Q4 quantization scheme. Nonetheless, INSPIRE demonstrates lower overall power consumption than Q8 in the evaluated cases. (iv) Regarding execution time, the INSPIRE architecture achieves significant improvements, yielding approximately 12.1×, 8.9×, and 2.3× speedups relative to P, Q8, and Q4 quantization mechanisms. This performance gain is primarily attributed to reduced data transfer overhead. Fig. 9(c) provides a performance guideline for the INSPIRE, illustrating the correlation between power efficiency, data transfer energy, and on-chip memory size for the INSPIRE compared to other methods. As shown, INSPIRE can achieve performance levels of 75.4, 198.5, 245.4 KFPS/W (kilo frames per second per Watt) with on-chip memory sizes of 2MB, 4MB, and 8MB, respectively. This efficiency translates to substantial reductions in data transfer energy. These design exploration results can be utilized to optimize the model according to the objective function previously discussed.

Performance Comparison Vs. SiPh Accelerators. Table I compares the efficiency of various MR-based optical accelerators, including LightBulb [24], HolyLight [23], HQNNA [40], Robin [26], CrossLight [22], and Lightator [27] as described in the background section. For objective comparison, we reconstructed each design from scratch to closely match the original, leveraging our evaluation framework and proprietary simulator, and ensured a consistent area constraint across all accelerators (approximately 20-60mm^2). We observe that INSPIRE significantly outperforms comparable designs, especially with an 8MB on-chip memory size, it improves power efficiency by 30.3%, 363.8%, 427.7%, and 609.2% over Lightator [27], CrossLight [22], Robin [26], and HQNNA [40].

TABLE I: Comparison with SOTA SiPh accelerators.

Designs	LightBulb [24]	HolyLight [23]	HQNNA [40]	Robin [26]	CrossLight [22]	Lightator [27]	INSPIRE
Node (nm)	32	32	45	45	*	45	45
KFPS/W	57.75	3.3	34.6	46.5	10.78-52.59	61.61-188.24	75.4-245.4
Improv.	324.9%(↑)	7336.3%(↑)	609.2%(↑)	427.7%(↑)	363.8%(↑)	30.3%(↑)	ref

*Data is not reported/not achievable from the paper [22].

Vs. Common Computing Platforms. The efficiency of the INSPIRE platform on the same algorithm is further evaluated and compared with state-of-the-art FPGA and GPU platforms commonly utilized for ViT inference [41], including Xilinx VCK190 and TensorRT implementations on NVIDIA A100 GPUs. The findings underscore the outstanding energy efficiency of INSPIRE, which achieves two to three orders of magnitude greater efficiency. Specifically, INSPIRE attains a peak performance of 245.4 KFPS/W with an 8MB on-chip memory, while Xilinx VCK190 and NVIDIA A100 deliver 1.42 and 0.86 KFPS/W, respectively.

VI. Conclusion

This work introduced the INSPIRE as a novel compressed weight retrieval mechanism co-developed with a high-performance and energy-efficient in-sensor SiPh ViT accelerator. This combination could effectively minimize the on-chip memory footprint, data transfer energy, and bandwidth requirements for ViTs for the first time without compromising model accuracy. Our results validate the framework's potential for low-power, high-performance ViT inference on resource-constrained edge platforms. INSPIRE achieves up to 245.4 Kilo FPS/W and reduces the data transfer energy by a factor of $\sim$11× on average compared with 4-bit quantized ViTs.

Acknowledgments

This work is supported in part by Semiconductor Research Corporation (SRC) and the National Science Foundation (NSF) under grant no. 2216772, 2228028, and 2046226.

REFERENCES

[1] R. Song, K. Huang, Z. Wang, and H. Shen, "A reconfigurable convolution-in-pixel cmos image sensor architecture," *IEEE TCSVT*, 2022.

[2] H. Xu, N. Lin, L. Luo, Q. Wei, R. Wang, C. Zhuo, X. Yin, F. Qiao, and H. Yang, "Senputing: An ultra-low-power always-on vision perception chip featuring the deep fusion of sensing and computing," *IEEE TCASI*, 2021.

[3] S. J. Carey, A. Lopich, D. R. Barr, B. Wang, and P. Dudek, "A 100,000 fps vision sensor with embedded 535gops/w 256× 256 simd processor array," in *Symposium on VLSI*. IEEE, 2013.

[4] T.-H. Hsu, Y.-R. Chen, R.-S. Liu, C.-C. Lo, K.-T. Tang, M.-F. Chang, and C.-C. Hsieh, "A 0.5-v real-time computational cmos image sensor with programmable kernel for feature extraction," *IEEE JSSC*, vol. 56, pp. 1588–1596, 2020.

[5] R. LiKamWa, Y. Hou, J. Gao, M. Polansky, and L. Zhong, "Redeye: analog convnet image sensor architecture for continuous mobile vision," *ACM SIGARCH Computer Architecture News*, vol. 44, pp. 255–266, 2016.

[6] J. Yu, A. Lukefahr, D. Palframan, G. Dasika, R. Das, and S. Mahlke, "Scalpel: Customizing dnn pruning to the underlying hardware parallelism," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 2, pp. 548–560, 2017.

[7] D. Zhang, J. Yang, D. Ye, and G. Hua, "Lq-nets: Learned quantization for highly accurate and compact deep neural networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 365–382.

[8] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *Journal of Machine Learning Research*, vol. 20, no. 55, pp. 1–21, 2019.

[9] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[11] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[12] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 568–578.

[13] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[14] V. Sanh, T. Wolf, and A. Rush, "Movement pruning: Adaptive sparsity by fine-tuning," *Advances in neural information processing systems*, vol. 33, pp. 20 378–20 389, 2020.

[15] F. Yu, K. Huang, M. Wang, Y. Cheng, W. Chu, and L. Cui, "Width & depth pruning for vision transformers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 3, 2022, pp. 3143–3151.

[16] S. Yu, T. Chen, J. Shen, H. Yuan, J. Tan, S. Yang, J. Liu, and Z. Wang, "Unified visual transformer compression," *arXiv preprint arXiv:2203.08243*, 2022.

[17] Y. Li, S. Xu, B. Zhang, X. Cao, P. Gao, and G. Guo, "Q-vit: Accurate and fully quantized low-bit vision transformer," *Advances in neural information processing systems*, vol. 35, pp. 34 451–34 463, 2022.

[18] Z. Liu, Y. Wang, K. Han, W. Zhang, S. Ma, and W. Gao, "Post-training quantization for vision transformer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 092–28 103, 2021.

[19] Y. Lin, T. Zhang, P. Sun, Z. Li, and S. Zhou, "Fq-vit: Post-training quantization for fully quantized vision transformer," *arXiv preprint arXiv:2111.13824*, 2021.

[20] Z. Li, J. Xiao, L. Yang, and Q. Gu, "Repq-vit: Scale reparameterization for post-training quantization of vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 17 227–17 236.

[21] Z. Li, L. Ma, M. Chen, J. Xiao, and Q. Gu, "Patch similarity aware data-free quantization for vision transformers," in *European conference on computer vision*. Springer, 2022, pp. 154–170.

[22] F. Sunny, A. Mirza, M. Nikdast, and S. Pasricha, "Crosslight: A cross-layer optimized silicon photonic neural network accelerator," in *DAC*. IEEE, 2021, pp. 1069–1074.

[23] W. Liu, W. Liu, Y. Ye, Q. Lou, Y. Xie, and L. Jiang, "Holylight: A nanophotonic accelerator for deep learning in data centers," in *DATE*. IEEE, 2019, pp. 1483–1488.

[24] F. Zokaee, Q. Lou, N. Youngblood, W. Liu, Y. Xie, and L. Jiang, "Lightbulb: A photonic-nonvolatile-memory-based accelerator for binarized convolutional neural networks," in *DATE*. IEEE, 2020, pp. 1438–1443.

[25] Z. Zhao, D. Liu, M. Li, Z. Ying, L. Zhang, B. Xu, B. Yu, R. T. Chen, and D. Z. Pan, "Hardware-software co-design of slimmed optical neural networks," in *ASP-DAC*. IEEE, 2019, pp. 705–710.

[26] F. P. Sunny, A. Mirza, M. Nikdast, and S. Pasricha, "Robin: A robust optical binary neural network accelerator," *ACM TECS*, no. 5s, pp. 1–24, 2021.

[27] M. Morsali, B. Reidy, D. Najafi, S. Tabrizchi, M. Imani, M. Nikdast, A. Roohi, R. Zand, and S. Angizi, "Lightator: An optical near-sensor accelerator with compressive acquisition enabling versatile image processing," *arXiv preprint arXiv:2403.05037*, 2024.

[28] W. Bogaerts, P. De Heyn, T. Van Vaerenbergh, K. De Vos, S. Kumar Selvaraja, T. Claes, P. Dumon, P. Bienstman, D. Van Thourhout, and R. Baets, "Silicon microring resonators," *Laser & Photonics Reviews*, pp. 47–73, 2012.

[29] S. Ahmed, U. Kamal, and M. K. Hasan, "Dfr-tsd: A deep learning based framework for robust traffic sign detection under challenging weather conditions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5150–5162, 2021.

[30] M. Morsali, S. Tabrizchi, D. Najafi, M. Imani, M. Nikdast, A. Roohi, and S. Angizi, "Oisa: Architecting an optical in-sensor accelerator for efficient visual computing," in *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2024, pp. 1–6.

[31] Synopsys, Inc., "Synopsys design compiler, product version 14.9.2014," 2014.

[32] K. Chen *et al.*, "Cacti-3dd: Architecture-level modeling for 3d die-stacked dram main memory," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*. IEEE, 2012, pp. 33–38.

[33] A. Roohi, S. Tabrizchi, M. Morsali, D. Z. Pan, and S. Angizi, "Pipsim: A behavior-level modeling tool for cnn processing-in-pixel accelerators," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.

[34] (2022) Cepton 3d lidar for smart machines. [Online]. Available: www.dataspeedinc.com/app/uploads/2019/10/Copy-of-VISTA-P60.pdf

[35] (2022) Esp32-s3 designed for aiot applications. [Online]. Available: https://www.espressif.com/en/products/socs/esp32-s3

[36] S. Zhou *et al.*, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv:1606.06160*, 2016.

[37] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," *arXiv preprint arXiv:1902.08153*, 2019.

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[39] Z. Ruan, Y. Zhu, P. Chen, Y. Shi, S. He, X. Cai, and L. Liu, "Efficient hybrid integration of long-wavelength vcsels on silicon photonic circuits," *Journal of Lightwave Technology*, vol. 38, no. 18, pp. 5100–5106, 2020.

[40] F. Sunny, M. Nikdast, and S. Pasricha, "A silicon photonic accelerator for convolutional neural networks with heterogeneous quantization," in *GLSVLSI*, 2022, pp. 367–371.

[41] P. Dong, J. Zhuang, Z. Yang, S. Ji, Y. Li, D. Xu, H. Huang, J. Hu, A. K. Jones, Y. Shi *et al.*, "Eq-vit: Algorithm-hardware co-design for end-to-end acceleration of real-time vision transformer inference on versal acap architecture," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 11, pp. 3949–3960, 2024.