H4H: Hybrid Convolution-Transformer Architecture Search for NPU-CIM Heterogeneous Systems for AR/VR Applications

Yiwei Zhao[1], Jinhui Chen[2], Sai Qian Zhang[3], Syed Shakib Sarwar[2], Kleber Hugo Stangherlin[2], Jorge Tomas Gomez[2], Jae-Sun Seo[2], Barbara De Salvo[2], Chiao Liu[2], Phillip B. Gibbons[1], Ziyun Li[2]
[1]Carnegie Mellon University, [2]Reality Labs Research, Meta, [3]New York University
{yiweiz3,pgibbons}@andrew.cmu.edu;sai.zhang@nyu.edu;js3528@cornell.edu
{jinhuic1,shakib7,kbr,jtgomez,barbarads,chiaoliu,liziyun}@meta.com

ABSTRACT

Low-latency and low-power edge AI is crucial for Augmented/Virtual Reality applications. Recent advances demonstrate that hybrid models, combining convolution layers (CNN) and transformers (ViT), often achieve a superior accuracy/performance tradeoff on various computer vision and machine learning (ML) tasks. However, hybrid ML models can present system challenges for latency and energy efficiency due to their diverse nature in dataflow and memory access patterns. In this work, we leverage architecture heterogeneity from Neural Processing Units (NPU) and Compute-In-Memory (CIM) and explore diverse execution schemas for efficient hybrid model executions. We introduce H4H-NAS, a two-stage Neural Architecture Search (NAS) framework to automate the design of hybrid CNN/ViT models for heterogeneous edge systems featuring both NPU and CIM. We propose a two-phase incremental supernet training in our NAS to resolve gradient conflicts between sampled subnets caused by different block types in a hybrid model search space. Our H4H-NAS approach is also powered by a performance estimator built with NPU performance results measured on real silicon, and CIM performance based on industry IPs. H4H-NAS searches hybrid CNN-ViT models with fine granularity and achieves significant (up to 1.34%) top-1 accuracy improvement on ImageNet-1k. Moreover, results from our algorithm/hardware co-design reveal up to 56.08% overall latency and 41.72% energy improvements by introducing heterogeneous computing over baseline solutions. Overall, our framework guides the design of hybrid network architectures and system architectures for NPU+CIM heterogeneous systems.

KEYWORDS

neural architecture search, neural processing unit, compute-in-memory, edge AI inference, algorithm-hardware co-design

ACM Reference Format:
Yiwei Zhao, Jinhui Chen, Sai Qian Zhang, Syed Shakib Sarwar, Kleber Hugo Stangherlin, Jorge Tomas Gomez, Jae-Sun Seo, Barbara De Salvo, Chiao Liu, Phillip B. Gibbons, Ziyun Li. 2025. H4H: Hybrid Convolution-Transformer Architecture Search for NPU-CIM Heterogeneous Systems for AR/VR Applications. In *30th Asia and South Pacific Design Automation Conference (ASPDAC '25), January 20–23, 2025, Tokyo, Japan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3658617.3697627

1 INTRODUCTION

Augmented/Virtual Reality (AR/VR) are increasingly prevailing as key next-generation human-oriented computing platforms [1]. Recent artificial intelligence (AI) advances further power AR/VR applications, revolutionizing how people communicate with each other, improving productivity and changing human interactions with the world. These applications typically run Deep Neural Network (DNN) inferences for various tasks, such as hand/eye tracking [20, 48], object detection [16], photorealistic avatars [64], etc.

Typically, to meet the low latency requirements of AR/VR applications (such as hand tracking and detection) and to preserve user privacy, most DNN inferences need to be processed locally on AR/VR devices. Moreover, given the limited compute, memory capacity, and power budget on these devices (AR/VR glasses) , as well as the recent emergence of smart cameras [37, 53], on-device processing is heavily distributed between the main SoC and multiple intelligent sensors. This setup allows a portion of the processing to reside locally on intelligent sensors [12, 17].

These intelligent sensors, although limited in compute and memory capacity due to area constraints, are required to achieve high energy efficiency in ML tasks with ultra-low latency. Meanwhile, DNN models for these applications are becoming increasingly diverse to improve task performance, even when targeting similar classes of workloads. For instance, in computer vision (CV), ResNet [21], MobileNet-v2 [51] and vision transformers (ViT) [13, 40] have vastly different basic block structures, requiring increasingly flexible execution schemas on hardware. This diversity poses challenges in designing general-purpose accelerators that are efficient across various models: An accelerator heavily optimized for one generation of models may become less efficient as new models are introduced.

Various edge AI acceleration designs have emerged to address these challenges and meet the stringent energy/latency requirements for edge AI. Among these, Neural Processing Units (NPUs) have shown great promise, with the technology recently maturing into widespread adoption in commercial products [2, 52]. Many state-of-the-art NPUs demonstrate high efficiency in compute-intensive workloads. For instance, ARM Ethos-U55/U65 [2, 3] are particularly efficient in handling convolution layers (CNN).

As the compute capacity increases, however, the frequent data movement between memory and processor dominates energy/latency costs. To mitigate this, compute-in-memory (CIM) has re-emerged to effectively reduce data movement. In CIM, computing elements are close to (near-memory computing (NMC) [7–9, 28, 29, 45, 67]) or even merged with (in-memory computing (IMC) [23–25, 32, 61, 65]) memory, thereby enhancing latency/energy efficiency. CIM triggers the design of related AI accelerators [39, 57, 58]. Please refer to Section 2.3 for more details.

With all these diverse advances in both ML algorithm/model and edge hardware acceleration, the current design spaces for edge AI/ML systems become exceedingly complicated. Consequently, two questions naturally arise, which we hope to solve in this paper:

(1) Can we design a **heterogeneous system** with multiple hardware acceleration features, that can *generalize* itself to accelerate various models with different execution schemas, or even a single **hybrid model** with different block types?

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
ASPDAC '25, January 20–23, 2025, Tokyo, Japan
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0635-6/25/01.
https://doi.org/10.1145/3658617.3697627

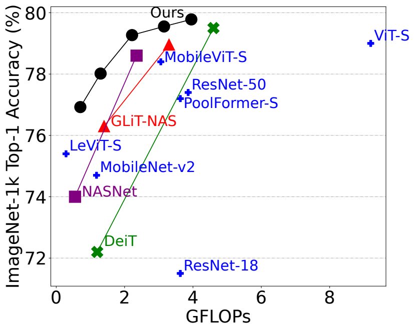


Figure 1: End-to-end comparison of model accuracy.

(2) If such design proves to be beneficial, can we **automate** the co-design of the model and the system, even if enormous heterogeneity on both sides will complicate such co-design?

In this work, we propose a generic design that combines both NPUs and CIMs, leveraging the architectural heterogeneity from NPUs and CIMs to accelerate edge AI with diverse dataflows arising from our hybrid CNN/ViT models. We also introduce an automated design workflow, with neural architecture search (NAS) as its core, to co-design hybrid CNN/ViT models to achieve the best accuracy/performance trade-offs for heterogeneous architectures.

Our key contributions and novel aspects are as follows:

- We present H4H-NAS: A neural architecture search framework to seamlessly automate the design and search of efficient Hybrid CNN/ViT models for usage on Heterogeneous edge compute featuring NPU and CIM.
- We modify the supernet training recipe in our H4H-NAS—using a two-phase incremental training—to improve the NAS training result of a hybrid model space.
- We build a system modeling tool in the workflow, using post-silicon results for NPU and industry IP-based results for CIM to guide the efficient model development process.
- We propose system-level improvements on current CIM-based designs, including adding multiple compute-units in CIMs and multiple macros in the system, to further improve system performance on ML workloads.

Our workflow produces hybrid models with better accuracy than state-of-the-art ones (Figure 1); meanwhile, our hardware accelerations achieves up to 56.08% latency and 41.72% energy improvements compared to single-device systems.

2 BACKGROUND

In this section, we provide backgrounds and motivations behind our methods. We discuss recent advances in AI/ML models utilizing different basic blocks, the latest development of neural architecture search, and the heterogeneity in different edge accelerators, such as NPUs and CIMs. These backgrounds motivate our approach of Algorithm/Hardware co-design for efficient hybrid models and their acceleration using a heterogeneous system within edge AI devices.

2.1 State-of-the-art ML Models

Various types of basic CV blocks have emerged as efficient alternatives to traditional ones such as VGG/ResNet [21, 38].

CNN. Convolution neural networks (CNN) continue to dominate the landscape of CV models. Specifically, the inverted residue bottleneck block (IRB) from MobileNet-v2 [51] emerged as one of the most widely used basic blocks for memory efficient, low-latency edge AI inferences. Other recent CNNs include EfficientNet [55, 56, 62], ConvNeXt [41, 63] and YOLO [26, 59].

ViT. As a byproduct of the advancements in language models, vision transformers (ViT) [13, 36, 40, 43] have recently emerged, showcasing superior performance as model size scales up. A ViT block integrates diverse operations together, including Q/K/V generators, head-level multiplication, layer norms, softmax, positional encoding and multilayer perceptrons (MLP).

Hybrid Models. In addition to diversity resulting from different components within a single block, some recent models employ multiple types of blocks in their networks. For instance, SAM [33], LeViT [19] and AlterNet [46, 70] combine both CNNs and ViTs.

2.2 Neural Architecture Search

Neural Architecture Search (NAS) is an efficient method that automates the design of a vast number of DNNs to discover memory/compute-efficient solutions for mobile deployment. Conventional NAS approaches, utilizing evolutionary search [50] or reinforcement learning [72], often require extensive training due to the large number of models trained in a single experiment. Recent advances in NAS have decoupled model training and architecture search into two separate stages [69], significantly reducing training costs. More recent NAS practices incorporate weight sharing into the supernet training stage [4, 10], which greatly alleviates the heavy computational burden of training all candidate networks from scratch.

While current NAS paradigms enjoy high efficiency in designing CNNs [4, 60], transformers [6, 18] and graph networks [15, 71], two fundamental problems persist when designing edge AI/ML models:

Inflexible Search Space. Although NAS enables flexible exploration over a vast number of subnets, in most cases, it primarily adjusts "network configurations"—such as feature map width, kernel sizes and channel numbers. The topology of the basic blocks is not significantly modified, which hinders NAS from capitalizing on new block structures such as ViTs. More recent research has investigated hybrid search spaces incorporating different block types (CNN and ViT). However, these approaches constrain the flexible placement of CNN and ViT blocks, thus limiting the full exploration of hybrid model space. For example, NASViT [18] sticks to a "first CNN then ViT" structure similar to LeViT [19].

Gradient Conflicts. During supernet training for hybrid NAS, different sampled subnets often exhibit conflicting/misaligned gradient directions, leading to degraded training quality [18, 42]. Please see Section 4.1 for details of the problem and our solution.

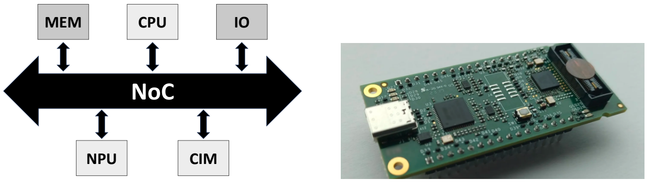
2.3 Edge AI Computing Hardware

NPU. The neural processing unit (NPU) has emerged as a prevalent solution for accelerating edge AI under the stringent resource constraints of edge devices. Typically, an NPU adopts a systolic array as its core component for efficient computation of matrix multiplications. Since its first emergence in the 1970s [34], NPU has remained an attractive design for the latest compute-intensive workloads, both in the cloud [27] and at the edge [2, 3, 52]. In this work, we specifically focus on edge NPU use cases.

CIM. Traditional architectures (both Von Neumann and accelerator-based) separate computation from storage, necessitating data movements between memory and compute and resulting in significant energy consumption and latency. To mitigate this, computing-in-memory (CIM) recently emerges which brings computing elements close to memory or even merges them with memory.

Both SRAM [5, 14, 44] and DRAM [65] has been employed for CIM. However, their volatility leads to efficiency degradation in mostly-off scenarios and latency issues during initialization. To address this, non-volatile memory such as resistive RAM (ReRAM) [23–25, 61], phase-change RAM (PCRAM) [30–32], and magnetic RAM (MRAM) [7–9, 67] are proposed. This paper focuses on *MRAM-based NMC* designs, given its potential advantages.

Recent work [11] shows that there remains a significant ***gap*** between the macro-level and processor-level energy efficiency. The key bottleneck is in non-trivial surrounding modules. This aligns with our modeling results (in Section 3.2), and prevents energy efficiency to be significantly improved by just introducing CIM.



(a) Target system to optimize. **(b) ARM Ethos-U55 NPU silicon.**

Figure 2: System and hardware overview.

Heterogeneous Edge Systems. NPU and CIM appear to be complementary components in edge AI/ML acceleration, tailored to accelerating compute-intensive and memory-intensive tasks, respectively. They might synergistically work together to achieve efficient processing. However, studies on designing heterogeneous systems consisting of both components are still in the early stages. Existing works focusing on such heterogeneous systems [47, 66] are highly specialized in their target workloads. This motivates us to explore the possibility and benefits of designing a heterogeneous NPU+CIM system for more general use cases.

3 PROPOSED DESIGN

Although hybrid models can achieve higher accuracy (Section 2.1), their diverse layers often require hardware support for various execution schemas. Additionally, different layers may impose different requirements on compute and memory access patterns, placing varying pressure on compute units and memory devices.

Meanwhile, previous works and our profiling results in Section 3.2 indicate that the heterogeneity between NPU and CIM provides potential solutions for hybrid model executions. NPUs excel in compute-intensive workloads, while CIMs are highly efficient on memory-intensive executions.

In this work, we focus on co-designing hybrid models and heterogeneous systems comprising both NPU and CIM, offering a potential solution for efficient edge AI/ML. We have chosen the ARM Ethos-U55 NPU and digital-based NMC MRAM CIM as representatives of the hardware components, as we believe they are adaptable to multiple workloads. However, our methods can also be applied to other hardware designs, such as analog-based IMCs.

3.1 Hybrid Models with CNN and ViT

Our target ML model architectures are hybrid models comprising both CNN and ViT. We perceive CNN blocks as local information extractors in CV applications and ViT blocks as global information comprehenders. We anticipate that these two types of blocks, each with distinct roles, could complement each other and enhance the overall performance, insipred by previous insights [46].

Moreover, we aim to automate the design of such models and develop a workflow that supports various hybrid CNN+ViT model variants. This necessitates a flexible model search space, as outlined in Table 1, along with several techniques to seamlessly automate the design within a NAS framework (as will be discussed in Section 4).

3.2 Heterogeneous NPU+CIM Platform

In this paper, we delve into the design of an example target system that integrates CIM components and an NPU on the same commodity network-on-chip (NoC) (Figure 2a). CIM and NPU share the same NoC bandwidth, set at 4-8 GB/s. The NoC incurs a warm-up latency of tens of cycles and CIM utilizes streaming processing over IFMP to hide its latency via pipelining. Workflow partition ensures that NPU/CIM will not occupy the NoC bandwidth simultaneously.

To architect AI edge systems incorporating both NPU and CIM, we begin by collecting and analyzing performance data from real-world silicon of NPUs and SPICE-simulated industrial CIM IPs. These collected data points offer an accurate modeling of the energy and performance of a heterogeneous system for our framework.

NPU. We use the ARM Ethos-U55 [2] as a typical example of an NPU on edge devices. Our test silicon, shown in Figure 2b, is fabricated and measured using 7nm FinFET technology.

We test different DNN model layers—regular / depth-wise / point-wise CNNs and fully-connected layers—using the NPU with ARM ethos-u-vela toolchain. All experiments are performed with a batch size of 1, which is common in edge inference applications. System metrics measured are execution latency and energy consumption.

Figure 3 shows the throughput and energy cost of typical layers executed on U55, both normalized by its theoretical best performance. In short, different layer types illustrate different execution efficiencies, but they all follow a trend of "**increasing then saturating**" as data sizes increase.

CIM. We acquire our CIM data on a digital-based NMC MRAM-based CIM macro. The non-volatility of MRAM helps reduce wake-up overhead on edge AR/VR applications. The MRAM macro is evaluated in 7nm technology (projected from 16nm designs) for fair comparison with the NPU. It is implemented based on production designs [35, 54] with read optimization for lower read energy. Each MRAM macro has 10Mb memory capacity and can compute the 16 accumulations of 9 products between 8-bit input and 8-bit weight. The memory and the computation peripheral occupy approximately 0.9mm^2 and 0.15mm^2, respectively. Figure 4 shows the overall architecture of our MRAM CIM macro.

We focus on performance of CIM executing memory-bounded layers, such as depthwise convolutions and fully-connected layers, as NPU performs suboptimally on these workloads. We also acquire CIM performance on pointwise convolution, as there is potential in leveraging this workload over NPU results.

Figure 5 shows the comparative ratio of throughput and energy efficiency between eight MRAM CIM macros and one U55 NPU. The results show that a system with multiple CIM macros working together can potentially outperform the NPU on memory-bounded DNN layers in both throughput and energy efficiency, for practical layer configurations from existing models.

4 METHODOLOGY: NEURAL ARCHITECTURE SEARCH FOR HYBRID MODELS

Workflow Overview. We have developed a workflow–H4H–to automate the co-design of algorithms and hardware for efficient inference with hybrid CNN+ViT models on heterogeneous edge systems featuring NPUs and CIM. This workflow targets **CV tasks** in AR/VR applications and integrates real-world resource constraints, such as those found in intelligent cameras.

We develop our H4H-NAS based on the two-stage NAS framework — with a first stage of supernet training (Section 4.1) and a second stage of subnet searching (Section 4.2). Our focus is on enabling a flexible search space of hybrid models and deploying them on heterogeneous architectures built from industrial IPs.

Search Space. We summarize our search space in Table 1. For inverted residual bottleneck blocks (IRB), we search for the number of output channels (width), the number of layers in a single block (depth), and the expansion ratio of depthwise convolutions. Stride=2 only applies to the first layer in each block. For vision transformer encoders [36], we search for the Q/K/V dimension (width), the number of layers in a single block (depth), and the expansion ratio of MLP. We fix the number of input channels and output channels of a transformer block to be equal to enable unchanged residues to bypass transformer blocks. We use (3,3)-sized kernels in all convolution layers and 8-dimension heads in all transformers.

We construct our supernet structure using repeated "convolution + transformer" blocks. It is worth noting that our supernet can be flexibly reduced to either an IRB-only model, a ViT-only model, or a "first CNN then ViT" structure similar to LeViT [19], as shown in Figure 6. This design ensures superior flexibility in the supernet architecture, allowing it to be reduced to a best model pattern among various model types during subnet search.

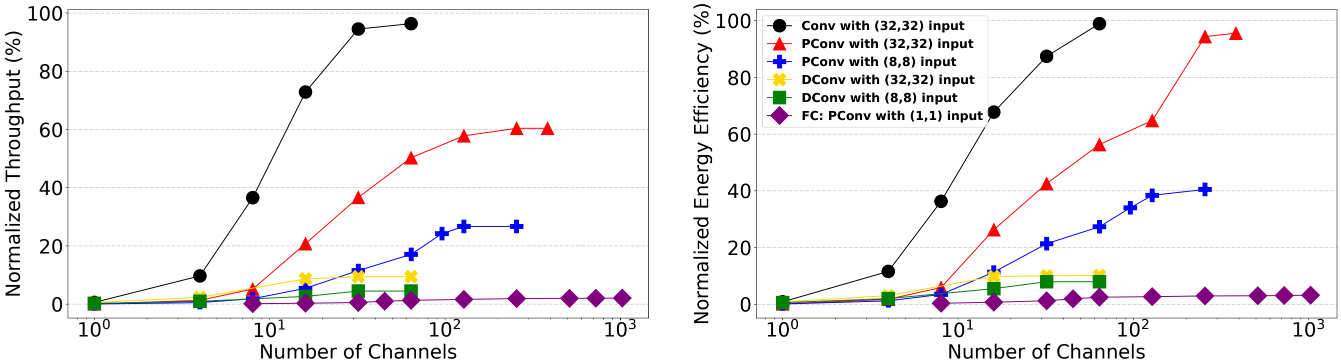


Figure 3: Throughput and energy efficiency of Ethos-U55 NPU execution of different layers, normalized by the theoretical best performance on U55. Conv and Dconv respectively stands for regular convolution and depthwise convolution with (3,3)-kernels. PConv represents pointwise convolution with (1,1)-kernel. FC refers to fully-connected layers.

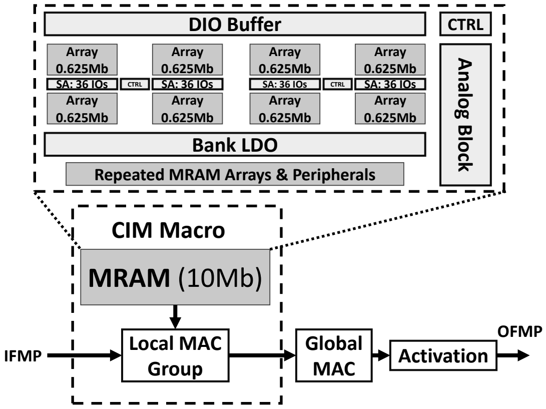


Figure 4: Architecture layout of our MRAM CIM macro. IFMP/OFMP stand for input/output feature maps.

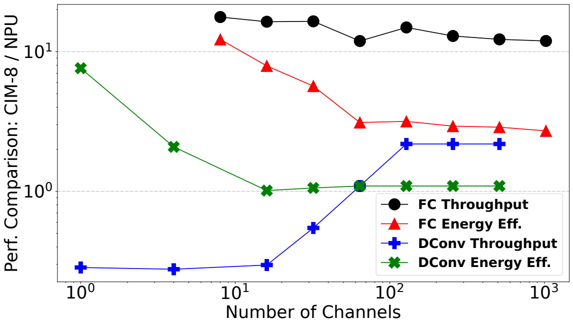


Figure 5: The comparative ratio of throughput and energy efficiency between a system with 8 CIM macros and a U55-only system when executing fully-connected layers and depthwise convolution with (3,3)-kernel and (32,32)-input.

4.1 Two-Phase Incremental Supernet Training

Baseline: Vanilla Supernet Training. We initially examine a vanilla supernet training as a baseline, which spans 360 epochs on ImageNet-1k. Recommended by LeViT [19] and NASViT [18], we utilize the AdamW optimizer when training. We employ the sandwich sampling rule [68], using an averaged gradient over the four sampled subnets for weight updates. We set both the dropout and drop-connect rates to be 0.2 and utilize AutoAugment.

Block	Width	Depth	Exp. Ratio	Stride
Conv-0	16 ~ 32	1	-	2
MBConv-1	16 ~ 32	1 ~ 2	1	1
MBConv-2	32 ~ 64	2 ~ 6	4 ~ 6	2
MBConv-3	32 ~ 64	2 ~ 6	4 ~ 6	2
ViT-3	24 ~ 64	0 ~ 1	1.0 ~ 2.0	-
MBConv-4-1	64 ~ 96	1 ~ 3	4 ~ 6	2
ViT-4-1	48 ~ 96	0 ~ 2	1.0 ~ 2.0	-
MBConv-4-2	64 ~ 96	0 ~ 3	4 ~ 6	1
ViT-4-2	48 ~ 96	0 ~ 2	1.0 ~ 2.0	-
MBConv-5-1	96 ~ 128	3 ~ 4	4 ~ 6	1
ViT-5-1	64 ~ 128	0 ~ 2	1.0 ~ 2.0	-
MBConv-5-2	96 ~ 128	0 ~ 4	4 ~ 6	1
ViT-5-2	64 ~ 128	0 ~ 2	1.0 ~ 2.0	-
MBConv-6-1	192 ~ 224	2 ~ 4	4 ~ 6	2
ViT-6-1	144 ~ 224	0 ~ 2	1.0 ~ 2.0	-
MBConv-6-2	192 ~ 224	0 ~ 4	4 ~ 6	1
ViT-6-2	144 ~ 224	0 ~ 2	1.0 ~ 2.0	-
MBConv-7	224 ~ 240	1 ~ 2	6	1
ViT-7	176 ~ 240	0 ~ 3	1.0 ~ 2.0	-
MBPool	1792 ~ 1984	1	6	-
Input Resolution	{192, 224, 256, 288}			

Table 1: H4H-NAS search space. MBConv refers to IRB [51]. ViT is from [36]. MBPool is an efficient last stage [22].

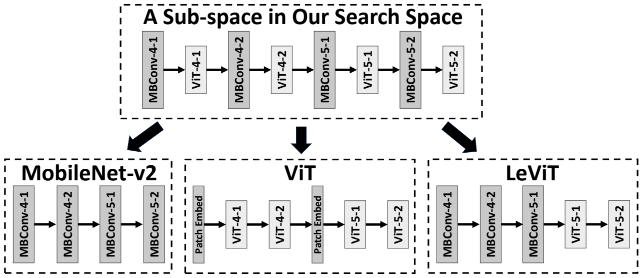


Figure 6: An example of how our search space can be flexibly reduced to basic blocks of different existing model types.

Additionally, we train a CNN-only supernet as a competitor using the same vanilla training recipe. The CNN-only supernet removes all ViT components from Table 1 and retains only the remaining CNN parts. In other words, the CNN-only supernet is a subset of the original hybrid supernet.

Gradient Conflict. Figure 8 (red and green parts) and Table 2 (row 1-2) depict results on an NPU-only system after vanilla supernet training. Note that the CNN-only supernet is a subset of the hybrid supernet. Thus theoretically speaking, after ideal training, the best subnet in every latency/energy bucket in the hybrid search space should perform no worse than the one in the CNN-only search space. Consequently, all green dots in Figure 8 should lie above the red frontier, and all values in the second row in Table 2 should be no smaller than the corresponding ones in the first row. However, it is observed that in both presented results, small-sized subnets in the CNN-only search space outperform those in the hybrid search space. This indicates that vanilla supernet training is non-ideal and leads to accuracy degradation in these small subnets.

Phenomena with similar causes have been observed in previous works [18, 42]. A common inference is that such degradation results from the non-alignment of gradients in different sampled subnets during training. We further deduce that such gradient conflict/non-alignment is amplified by the different block types in a hybrid search space based on our observations on different supernet training.

Two-Phase Incremental Supernet Training. To address this gradient conflict problem, we propose a new training recipe called *two-phase incremental* supernet training. Our supernet training stage is divided into two phases. In the first phase, we remove all the ViT blocks from the supernet and solely train a partial supernet with all remaining CNN components (i.e., train a CNN-only supernet first). In the second phase, we load all the pre-trained CNN weights from the first stage into the complete hybrid supernet and continue the training. The underlying idea is that, during each phase, only the blocks belonging to the same type are trained together. Therefore, the gradient conflicts should be mitigated within each phase.

Similar to vanilla training, during both phases we use Sandwich sampling, AdamW, AutoAugment, and dropout and drop-connect rates of 0.2. During the second phase, we only train the ViT blocks and batch normalizations, and do not update the already-trained CNN weights. This partial training not only enhances accuracy (see Section 6.2) but also reduces training costs.

4.2 Subnet Search and Performance Modeling

Once supernet training is completed, we employ evolutionary search [49] to find the optimal subnets, considering stringent system constraints on energy/latency.

We model subnets running on heterogeneous AI edge devices with both NPU and CIM macros. Our system model breaks down model inferences into fine granularity. For convolution layers, it partitions the execution of different channels onto different devices. Similarly, for transformer layers, the generation of Q/K/V and the execution of different heads in attention layers can be partitioned.

The system modeling tool combines measurement results using custom silicon and simulation results from industrial CIM IPs (Section 3.2). In addition, latency/energy caused by the data transfer between NPU and CIM over NoC are also modeled. As a result, we obtain accurate latency and energy estimations for target subnets.

5 EVALUATION RESULTS

In this section, we present the results of co-designing hybrid CNN/ViT models on the heterogeneous NPU+CIM system, followed by an ablation study in Section 6.

Heterogeneous Systems Reduce Hybrid Model Latency. Figure 7 illustrates the results of latency-constrained search for our hybrid models in systems with different numbers of CIM macros. It highlights that introducing heterogeneity into AI edge hardware significantly reduces inference latency. Given the same latency requirement, a system with 8 CIM macros can support a hybrid model with a 1.341% higher top-1 accuracy compared to an NPU-only system. Meanwhile, when acquiring models with the same accuracy, a system with 8 CIM macros can perform inference with an average latency reduction of 21.99% and up to 56.08%.

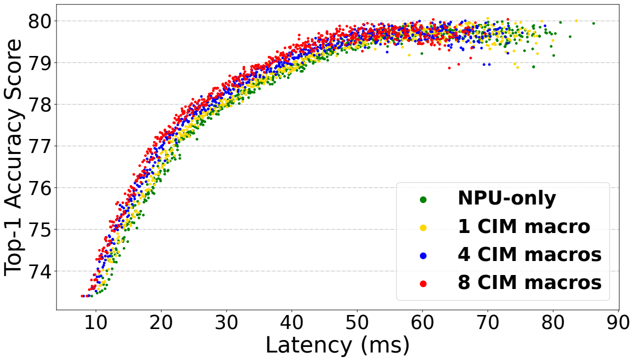


Figure 7: H4H-NAS results showing hybrid CNN/ViT model top-1 accuracy under varying search latency constraints, when using NPU with 0, 1, 4 or 8 CIM macros.

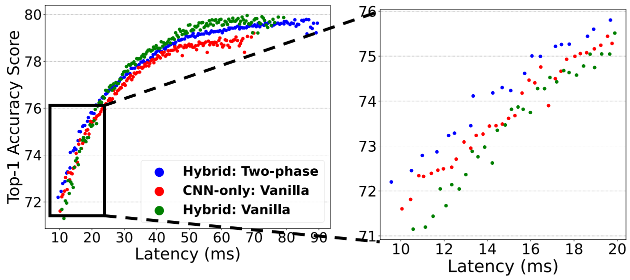


Figure 8: Top-1 accuracy searched by latency of two-phase incremental training of hybrid models vs. vanilla training of hybrid models vs. vanilla CNN-only models on NPU-only systems. The right side is a zoom-in on small-sized models.

Heterogeneous Systems Save Energy. We also conduct energy-constrained search for hybrid models in systems with different numbers of CIM macros. Heterogeneous systems improve energy efficiency. Given the same energy requirement, a system with 8 CIM macros can support hybrid models with 0.614% higher accuracy. Additionally, it achieves an average energy consumption reduction of 11.80% and up to 33.13%.

Effects of Multiple CIM Macros. Compute can be parallelized on CIMs when more than one CIM is available in the system. In Figure 7, the overall inference latency decreases with more CIMs. However, the improvement in energy efficiency does not scale proportionally with the introduction of more CIMs. Intuitively, adding more macros without altering their internal structure does not significantly change the energy efficiency of operations.

6 ABLATION STUDY

6.1 ResNet vs. MobileNet-v2 vs. Hybrid Model

To evaluate the efficacy of hybrid model architectures on heterogeneous edge systems, we conduct searches for optimal models based on ResNet, MobileNet-v2 (IRB), and hybrid CNN/ViT structures. As depicted in Figure 8, the accuracy of subnets increases with more latency being afforded. Hybrid models achieve significantly better performance than IRB-based models given the same latency budget (similar results hold for energy—not shown). It is also worth noting that IRB-based models strictly outperform ResNet counterparts given all constraints. (No ResNet-based subnet exceeds a 77% top-1 accuracy.) Similar trends also exist in systems with CIM components, also not presented due to space constraints.

Efficient CNN/ViT Basic Block. We also study the ratio of the number of ViT blocks over IRB in different subnets. Interestingly, H4H-NAS tends to incorporate both CNN and ViT while maintaining a balance between them for efficient inference. Almost all searched subnets exhibit a similar

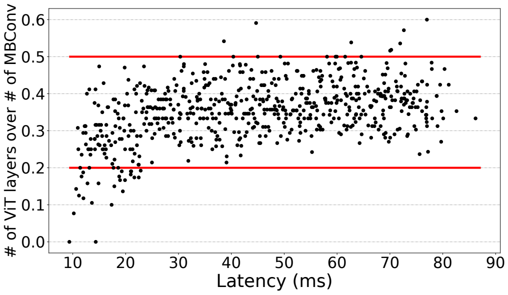


Figure 9: Ratio between number of ViT layers and number of MBConv layers, in each subnet.

Model & Recipe	Min_net	Max_net
CNN-only	71.691	78.802
Hybrid: Vanilla	71.346 (−0.345)	79.914 (+1.112)
Hybrid: TPI-unfreeze	72.140 (+0.449)	79.248 (+0.446)
Hybrid TPI-freeze	72.201 (**+0.510**)	79.782 (**+0.980**)

Table 2: Top-1 accuracy of minimum and maximum subnets after different training recipes. TPI refers to two-phase incremental training. Freeze/unfreeze refers to whether to prevent/proceed the weight updates on CNNs in the second TPI phase. Parentheses numbers are accuracy improvements.

proportion of 2–5 ViT combined with 10 IRB, as indicated by the region between the red lines in Figure 9. This phenomenon indicates that maybe repeated blocks with a fixed proportion of both IRB and ViT are preferred.

This finding favors an alternating structure over single-type models or LeViT structures, aligning with recent hand-crafted hybrid architectures [46]. IRB abstracts *neighboring* information in a feature map into tokens, while ViT translates the token embeddings in a *global* environment using attention layers. Therefore, hybrid architectures often offer better accuracy/performance trade-offs.

6.2 Two-Phase Incremental Supernet Training

Two-phase vs. Vanilla. To assess our two-phase incremental (TPI) training, we compare the training quality of the CNN-only supernet, vanilla training on the hybrid models, and our TPI training on the hybrid models. The results are presented in Table 2 and Figure 8.

Vanilla training suffers from gradient conflicts. Our TPI training, on the other hand, resolves such gradient conflicts and produces small-sized subnets that perform no worse than those in the CNN-only baseline. This can be observed in Figure 8 and the "min_net" column in Table 2, where TPI training results in better-performing subnets than the CNN-only baseline. Furthermore, TPI training preserves the efficacy of hybrid space in medium/large-sized subnets. See the saturation curve in Figure 8 and the "max_net" column in Table 2.

CNN Freezing. We also compare whether to update the weights of already-trained CNN parts in the second phase of supernet training, corresponding to the freeze/unfreeze strategy in Table 2 (rows 3-4). The results show that freezing the CNN parts and only training the ViT parts can bring up to 0.53% accuracy improvements. One potential explanation for this observation is that freezing the CNN parts and only updating the ViT parts prevents the training of global information comprehenders (transformers) from interfering with the already-trained local information extractors (CNN). Moreover, the two-phase training with CNN-freezing only adds 42% GPU work overhead than single-phase training of a hybrid supernet, which is acceptable since such training is only required once.

6.3 Increased Parallelism inside a CIM Macro

In Section 5, we demonstrated that using multiple CIM macros improves inference latency over a state-of-the-art single-macro system. Here, we further explore the benefits of introducing multiple compute units within a single CIM macro. Figure 10 illustrates an example of a single CIM macro with four compute units inside.

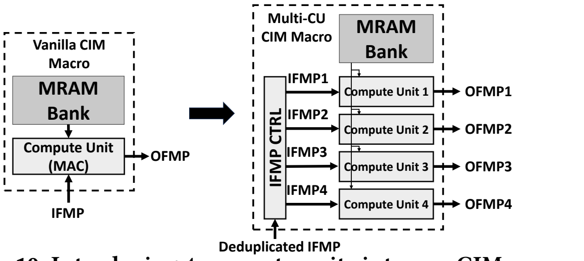


Figure 10: Introducing 4 compute units into one CIM macro.

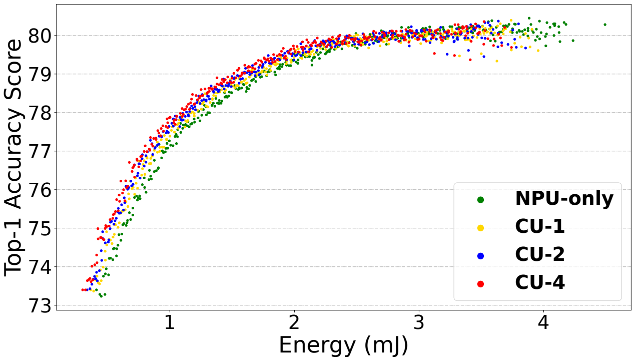


Figure 11: Energy-constrained H4H-NAS for a single-macro system with different numbers of compute units.

This design is promising for two reasons. Firstly, it provides another level of parallelism in computation. Secondly, it allows for the merging and transfer of repeated input data into the CIM macro. The input feature map (IFMP) controller reorganizes the dataflow required for computation. For example, depthwise convolutions can benefit from input deduplication if adjacent output elements are computed simultaneously, where the theoretical read reduction can reach 2/3 for (3,3)-kernels. Additionally, each compute unit only costs 14% area overhead, and IFMP controller is even < 0.1%.

We integrate our multi-CU design into H4H-NAS. Figure 11 shows energy-constrained searches on single-macro systems with different numbers of compute units. Under same accuracy, a single-macro system with 4 compute units reduces energy consumption by an average of 19.11% and up to 41.72% compared to NPU-only systems. Additionally, it achieves an average reduction of 9.34% compared to an NPU+CIM system with one compute unit per macro.

7 CONCLUSION

This paper presents H4H-NAS, a NAS-oriented framework that automates the design of efficient hybrid CNN+ViT models for heterogeneous edge systems equipped with both NPU and CIM. The framework achieves up to a 1.34% improvement in top-1 accuracy, along with up to 56.08% latency reduction and 41.72% energy savings. Key techniques include a highly flexible hybrid model search space, a two-phase incremental supernet training for hybrid models, a reliable performance profiler for heterogeneous systems, and system enhancements through increased CIM parallelism. Our framework is adaptable to future edge devices and provides insights into both ML model design and edge system optimization.

ACKNOWLEDGMENTS

We thank Win-San (Vince) Khwa, Xiaoyu Sun, Yi-Lun Lu and all the other folks at TSMC Corporate Research, for hardware modeling and all the help whenever needed. This research was supported by Meta Reality Labs Research, NSF grants CCF-1919223 and CNS-2211882, Parallel Data Lab (PDL) Consortium, and the Lee-Stanziale Ohana Endowed Fellowship.

REFERENCES

[1] Michael Abrash. 2021. Creating the Future: Augmented Reality, the next Human-Machine Interface. In *2021 IEEE International Electron Devices Meeting (IEDM)*. 1–11. https://doi.org/10.1109/IEDM19574.2021.9720526

[2] Arm®. Accessed 2024-10. Arm Ethos-U55 microNPU Description. https://www.arm.com/products/silicon-ip-cpu/ethos/ethos-u55.

[3] Arm®. Accessed 2024-10. Arm Ethos-U65 microNPU Description. https://www.arm.com/products/silicon-ip-cpu/ethos/ethos-u65.

[4] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 2019. Once-for-All: Train One Network and Specialize it for Efficient Deployment. In *International Conference on Learning Representations*.

[5] Yu-Der Chih, Po-Hao Lee, Hidehiro Fujiwara, Yi-Chun Shih, Chia-Fu Lee, Rawan Naous, Yu-Lin Chen, Chieh-Pu Lo, Cheng-Han Lu, Haruki Mori, Wei-Chang Zhao, Dar Sun, Mahmut E. Sinangil, Yen-Huei Chen, Tan-Li Chou, Kerem Akarvardar, Hung-Jen Liao, Yih Wang, Meng-Fan Chang, and Tsung-Yung Jonathan Chang. 2021. 16.4 An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications. In *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 64. 252–254. https://doi.org/10.1109/ISSCC42613.2021.9365766

[6] Krishna Teja Chitty-Venkata, Murali Emani, Venkatram Vishwanath, and Arun K. Somani. 2022. Neural Architecture Search for Transformers: A Survey. *IEEE Access* 10 (2022), 108374–108412. https://doi.org/10.1109/ACCESS.2022.3212767

[7] Yen-Cheng Chiu, Win-San Khwa, Chung-Yuan Li, Fang-Ling Hsieh, Yu-An Chien, Guan-Yi Lin, Po-Jung Chen, Tsen-Hsiang Pan, De-Qi You, Fang-Yi Chen, Andrew Lee, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Yu-Der Chih, Tsung-Yung Chang, and Meng-Fan Chang. 2023. A 22nm 8Mb STT-MRAM Near-Memory-Computing Macro with 8b-Precision and 46.4-160.1TOPS/W for Edge-AI Devices. In *2023 IEEE International Solid-State Circuits Conference (ISSCC)*. 496–498. https://doi.org/10.1109/ISSCC42615.2023.10067563

[8] Yen-Cheng Chiu, Win-San Khwa, Chia-Sheng Yang, Shih-Hsin Teng, Hsiao-Yu Huang, Fu-Chun Chang, Yuan Wu, Yu-An Chien, Fang-Ling Hsieh, Chung-Yuan Li, et al. 2023. A CMOS-integrated spintronic compute-in-memory macro for secure AI edge devices. *Nature Electronics* 6, 7 (2023), 534–543.

[9] Yen-Cheng Chiu, Chia-Sheng Yang, Shih-Hsin Teng, Hsiao-Yu Huang, Fu-Chun Chang, Yuan Wu, Yu-An Chien, Fang-Ling Hsieh, Chung-Yuan Li, Guan-Yi Lin, Po-Jung Chen, Tsen-Hsiang Pan, Chung-Chuan Lo, Win-San Khwa, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Chieh-Pu Lo, Yu-Der Chih, Tsung-Yung, Jonathan Chang, and Meng-Fan Chang. 2022. A 22nm 4Mb STT-MRAM Data-Encrypted Near-Memory Computation Macro with a 192GB/s Read-and-Decryption Bandwidth and 25.1-55.1TOPS/W 8b MAC for AI Operations. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. 178–180. https://doi.org/10.1109/ISSCC42614.2022.9731621

[10] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. 2021. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *Proceedings of the IEEE/CVF International Conference on computer vision*. 12239–12248.

[11] Zhaori Cong, Jinshan Yue, Shengzhe Yan, Zhuoyu Dai, Zeyu Guo, Zhihang Qiao, Yifan He, Wenyu Sun, Chunming Dou, Feng Zhang, and Yongpan Liu. 2024. Understanding the Upper Bounds of Energy Efficiency in a Computing-in-Memory Processor and How to Approach the Limit. In *61th Design Automation Conference (DAC '24), Work-In-Progress Posters*.

[12] Xin Dong, Barbara De Salvo, Meng Li, Chiao Liu, Zhongnan Qu, H.T. Kung, and Ziyun Li. 2022. SplitNets: Designing Neural Architectures for Efficient Distributed Computing on Head-Mounted Systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12559–12569.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

[14] Hidehiro Fujiwara, Haruki Mori, Wei-Chang Zhao, Mei-Chen Chuang, Rawan Naous, Chao-Kai Chuang, Takeshi Hashizume, Dar Sun, Chia-Fu Lee, Kerem Akarvardar, Saman Adham, Tan-Li Chou, Mahmut Ersin Sinangil, Yih Wang, Yu-Der Chih, Yen-Huei Chen, Hung-Jen Liao, and Tsung-Yung Jonathan Chang. 2022. A 5-nm 254-TOPS/W 221-TOPS/mm2 Fully-Digital Computing-in-Memory Macro Supporting Wide-Range Dynamic-Voltage-Frequency Scaling and Simultaneous MAC and Write Operations. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. 1–3. https://doi.org/10.1109/ISSCC42614.2022.9731754

[15] Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, and Yue Hu. 2021. Graph neural architecture search. In *International joint conference on artificial intelligence*. International Joint Conference on Artificial Intelligence.

[16] Yalda Ghasemi, Heejin Jeong, Sung Ho Choi, Kyeong-Beom Park, and Jae Yeol Lee. 2022. Deep learning-based object detection in augmented reality: A systematic review. *Computers in Industry* 139 (2022), 103661. https://doi.org/10.1016/j.compind.2022.103661

[17] Jorge Gomez, Saavan Patel, Syed Shakib Sarwar, Ziyun Li, Raffaele Capoccia, Zhao Wang, Reid Pinkham, Andrew Berkovich, Tsung-Hsun Tsai, Barbara De Salvo, and Chiao Liu. 2022. Distributed On-Sensor Compute System for AR/VR Devices: A Semi-Analytical Simulation Framework for Power Estimation. In *Proceedings of tinyML Research Symposium*.

[18] Chengyue Gong, Dilin Wang, Meng Li, Xinlei Chen, Zhicheng Yan, Yuandong Tian, Qiang Liu, and Vikas Chandra. 2022. NASViT: Neural Architecture Search for Efficient Vision Transformers with Gradient Conflict aware Supernet Training. In *International Conference on Learning Representations*.

[19] Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. 2021. Levit: a vision transformer in convnet's clothing for faster inference. In *IEEE/CVF international conference on computer vision*. 12259–12269.

[20] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D. Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, Asaf Nitzan, Gang Dong, Yuting Ye, Lingling Tao, Chengde Wan, and Robert Wang. 2020. MEgATrack: Monochrome Egocentric Articulated Hand-Tracking for Virtual Reality. *ACM Trans. Graph.* 39, 4, Article 87 (aug 2020).

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[22] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. 2019. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1314–1324.

[23] Wei-Hsing Huang, Tai-Hao Wen, Je-Min Hung, Win-San Khwa, Yun-Chen Lo, Chuan-Jia Jhang, Huna-Hsi Hsu, Yu-Hsiana Chin, Yu-Chiao Chen, Chuna-Chuan Lo, Ren-Shuo Liu, Kea-Tiong Tang, Chih-Cheng Hsieh, Yu-Der Chih, Tsung-Yung Chang, and Meng-Fan Chang. 2023. A Nonvolatile Al-Edge Processor with 4MB SLC-MLC Hybrid-Mode ReRAM Compute-in-Memory Macro and 51.4-251TOPS/W. In *2023 IEEE International Solid-State Circuits Conference (ISSCC)*. 15–17. https://doi.org/10.1109/ISSCC42615.2023.10067610

[24] Je-Min Hung, Yen-Hsiang Huang, Sheng-Po Huang, Fu-Chun Chang, Tai-Hao Wen, Chin-I Su, Win-San Khwa, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Yu-Der Chih, Tsung-Yung Jonathan Chang, and Meng-Fan Chang. 2022. An 8-Mb DC-Current-Free Binary-to-8b Precision ReRAM Nonvolatile Computing-in-Memory Macro using Time-Space-Readout with 1286.4-21.6TOPS/W for Edge-AI Devices. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. 1–3. https://doi.org/10.1109/ISSCC42614.2022.9731715

[25] Je-Min Hung, Cheng-Xin Xue, Hui-Yao Kao, Yen-Hsiang Huang, Fu-Chun Chang, Sheng-Po Huang, Ta-Wei Liu, Chuan-Jia Jhang, Chin-I Su, Win-San Khwa, et al. 2021. A four-megabit compute-in-memory macro with eight-bit precision based on CMOS and resistive random-access memory for AI edge devices. *Nature Electronics* 4, 12 (2021), 921–930.

[26] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. 2022. A Review of Yolo algorithm developments. *Procedia computer science* 199 (2022), 1066–1073.

[27] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. 2017. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*. 1–12.

[28] Hongbo Kang, Yiwei Zhao, Guy E. Blelloch, Laxman Dhulipala, Yan Gu, Charles McGuffey, and Phillip B. Gibbons. 2022. PIM-tree: A Skew-resistant Index for Processing-in-Memory. *PVLDB* 16, 4 (2022), 946–958. https://doi.org/10.14778/3574245.3574275

[29] Hongbo Kang, Yiwei Zhao, Guy E. Blelloch, Laxman Dhulipala, Yan Gu, Charles McGuffey, and Phillip B. Gibbons. 2023. PIM-trie: A Skew-resistant Trie for Processing-in-Memory. In *Proceedings of the 35th ACM Symposium on Parallelism in Algorithms and Architectures* (Orlando, FL, USA) *(SPAA '23)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3558481.3591070

[30] R. Khaddam-Aljameh, M. Stanisavljevic, J. Fornt Mas, G. Karunaratne, M. Braendli, F. Liu, A. Singh, S. M. Müller, U. Egger, A. Petropoulos, T. Antonakopoulos, K. Brew, S. Choi, I. Ok, F. L. Lie, N. Saulnier, V. Chan, I. Ahsan, V. Narayanan, S. R. Nandakumar, M. Le Gallo, P. A. Francese, A. Sebastian, and E. Eleftheriou. 2021. HERMES Core – A 14nm CMOS and PCM-based In-Memory Compute Core using an array of 300ps/LSB Linearized CCO-based ADCs and local digital processing. In *2021 Symposium on VLSI Circuits*. 1–2. https://doi.org/10.23919/VLSICircuits52068.2021.9492362

[31] W. S. Khwa, K. Akarvardar, Y. S. Chen, Y. C. Chiu, J. C. Liu, J. J. Wu, H. Y. Lee, S. M. Yu, C. H. Lee, T. C. Chen, Y. C. Lin, C. F. Hsu, T. Y. Lee, T. K. Ku, C. H. Kuo, J. Y. Wu, X. Y. Bao, C. S. Chang, Y. D. Chih, H.-S. P. Wong, and M. F. Chang. 2021. MLC PCM Techniques to Improve Nerual Network Inference Retention Time by 105X and Reduce Accuracy Degradation by 10.8X. In *2021 Symposium on VLSI Technology*. 1–2.

[32] Win-San Khwa, Yen-Cheng Chiu, Chuan-Jia Jhang, Sheng-Po Huang, Chun-Ying Lee, Tai-Hao Wen, Fu-Chun Chang, Shao-Ming Yu, Tung-Yin Lee, and Meng-Fan Chang. 2022. A 40-nm, 2M-Cell, 8b-Precision, Hybrid SLC-MLC PCM Computing-in-Memory Macro with 20.5 - 65.0TOPS/W for Tiny-Al Edge Devices. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. 1–3. https://doi.org/10.1109/ISSCC42614.2022.9731670

[33] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 4015–4026.

[34] Hsiang Tsung Kung and Charles E Leiserson. 1979. Systolic arrays (for VLSI). In *Sparse Matrix Proceedings 1978*, Vol. 1. Society for industrial and applied mathematics Philadelphia, PA, USA, 256–282.

[35] Po-Hao Lee, Chia-Fu Lee, Yi-Chun Shih, Hon-Jarn Lin, Yen-An Chang, Cheng-Han Lu, Yu-Lin Chen, Chieh-Pu Lo, Chung-Chieh Chen, Cheng-Hsiung Kuo, Tan-Li Chou, Chia-Yu Wang, J. J. Wu, Roger Wang, Harry Chuang, Yih Wang, Yu-Der Chih, and Tsung-Yung Jonathan Chang. 2023. 33.1 A 16nm 32Mb Embedded STT-MRAM with a 6ns Read-Access Time, a 1M-Cycle Write Endurance, 20-Year Retention at 150°C and MTJ-OTP Solutions for Magnetic Immunity. In *2023 IEEE International Solid-State Circuits Conference (ISSCC).* 494–496. https://doi.org/10.1109/ISSCC42615.2023.10067837

[36] Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. 2022. Efficientformer: Vision transformers at mobilenet speed. *Advances in Neural Information Processing Systems* 35 (2022), 12934–12949.

[37] Chiao Liu, Andrew Berkovich, Qing Chao, Song Chen, Ziyun Li, Hans Reyserhove, Syed Shakib Sarwar, and Tsung-Hsun Tsai. 2020. Intelligent vision sensors for AR/VR. In *Imaging Systems and Applications.* Optica Publishing Group, ITu5G–1.

[38] Shuying Liu and Weihong Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR).* 730–734. https://doi.org/10.1109/ACPR.2015.7486599

[39] Shiwei Liu, Peizhe Li, Jinshan Zhang, Yunzhengmao Wang, Haozhe Zhu, Wenning Jiang, Shan Tang, Chixiao Chen, Qi Liu, and Ming Liu. 2023. 16.2 A 28nm 53.8TOPS/W 8b Sparse Transformer Accelerator with In-Memory Butterfly Zero Skipper for Unstructured-Pruned NN and CIM-Based Local-Attention-Reusable Engine. In *2023 IEEE International Solid-State Circuits Conference (ISSCC).* 250–252. https://doi.org/10.1109/ISSCC42615.2023.10067360

[40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *IEEE/CVF international conference on computer vision.* 10012–10022.

[41] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 11976–11986.

[42] Lotfi Abdelkrim Mecharbat, Hadjer Benmeziane, Hamza Ouarnoughi, and Smail Niar. 2023. HyT-NAS: Hybrid Transformers Neural Architecture Search for Edge Devices. arXiv:2303.04440

[43] Sachin Mehta and Mohammad Rastegari. 2022. MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. In *International Conference on Learning Representations.*

[44] Haruki Mori, Wei-Chang Zhao, Cheng-En Lee, Chia-Fu Lee, Yu-Hao Hsu, Chao-Kai Chuang, Takeshi Hashizume, Hao-Chun Tung, Yao-Yi Liu, Shin-Rung Wu, Kerem Akarvardar, Tan-Li Chou, Hidehiro Fujiwara, Yih Wang, Yu-Der Chih, Yen-Huei Chen, Hung-Jen Liao, and Tsung-Yung Jonathan Chang. 2023. A 4nm 6163-TOPS/W/b $\mathbf{4790-TOPS/mm^2/b}$ SRAM Based Digital-Computing-in-Memory Macro Supporting Bit-Width Flexibility and Simultaneous MAC and Weight Update. In *2023 IEEE International Solid-State Circuits Conference (ISSCC).* 132–134. https://doi.org/10.1109/ISSCC42615.2023.10067555

[45] Onur Mutlu, Saugata Ghose, Juan Gómez-Luna, and Rachata Ausavarungnirun. 2023. *A Modern Primer on Processing in Memory.* Springer Nature Singapore, Singapore, 171–243. https://doi.org/10.1007/978-981-16-7487-7_7

[46] Namuk Park and Songkuk Kim. 2021. How Do Vision Transformers Work?. In *International Conference on Learning Representations.*

[47] Wonhoon Park, Junha Ryu, Sangjin Kim, Soyeon Um, Wooyoung Jo, Sangyoeb Kim, and Hoi-Jun Yoo. 2023. A 5.99 TFLOPS/W Heterogeneous CIM-NPU Architecture for an Energy Efficient Floating-Point DNN Acceleration. In *2023 IEEE International Symposium on Circuits and Systems (ISCAS).* 1–4. https://doi.org/10.1109/ISCAS46773.2023.10181869

[48] Alexander Plopski, Teresa Hirzle, Nahal Norouzi, Long Qian, Gerd Bruder, and Tobias Langlotz. 2022. The Eye in Extended Reality: A Survey on Gaze Interaction and Eye Tracking in Head-Worn Extended Reality. *ACM Comput. Surv.* 55, 3, Article 53 (mar 2022), 39 pages.

[49] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 33. 4780–4789.

[50] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. 2017. Large-scale evolution of image classifiers. In *International conference on machine learning.* PMLR, 2902–2911.

[51] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition.*

[52] Samsung Semiconductor. Accessed 2024-10. Samsung Neural SDK. https://developer.samsung.com/neural/overview.html.

[53] Sony Semiconductor. Accessed 2024-10. Sony AI camera. https://www.sony-semicon.com/en/technology/ivs/index.html.

[54] Yi-Chun Shih, Chia-Fu Lee, Yen-An Chang, Po-Hao Lee, Hon-Jarn Lin, Yu-Lin Chen, Chieh-Pu Lo, Ku-Feng Lin, Tien-Wei Chiang, Yuan-Jen Lee, Kuei-Hung Shen, Roger Wang, Wayne Wang, Harry Chuang, Eric Wang, Yu-Der Chih, and Jonathan Chang. 2020. A Reflow-capable, Embedded 8Mb STT-MRAM Macro with 9nS Read Access Time in 16nm FinFET Logic CMOS Process. In *2020 IEEE International Electron Devices Meeting (IEDM).* 11.4.1–11.4.4. https://doi.org/10.1109/IEDM13553.2020.9372115

[55] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning.* PMLR, 6105–6114.

[56] Mingxing Tan and Quoc Le. 2021. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning.* PMLR, 10096–10106.

[57] Fengbin Tu, Zihan Wu, Yiqi Wang, Weiwei Wu, Leibo Liu, Yang Hu, Shaojun Wei, and Shouyi Yin. 2023. 16.1 MuITCIM: A 28nm 2.24μJ/Token Attention-Token-Bit Hybrid Sparse Digital CIM-Based Accelerator for Multimodal Transformers. In *2023 IEEE International Solid-State Circuits Conference (ISSCC).* 248–250.

[58] Bo Wang, Chen Xue, Zhongyuan Feng, Zhaoyang Zhang, Han Liu, Lizheng Ren, Xiang Li, Anran Yin, Tianzhu Xiong, Yeyang Xue, Shengnan He, Yuyao Kong, Yongliang Zhou, An Guo, Xin Si, and Jun Yang. 2023. A 28nm Horizontal-Weight-Shift and Vertical-feature-Shift-Based Separate-WL 6T-SRAM Computation-in-Memory Macro for Depthwise Neural-Networks. In *IEEE International Solid-State Circuits Conference.*

[59] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. 2023. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 7464–7475. https://doi.org/10.1109/CVPR52729.2023.00721

[60] Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. 2021. AttentiveNAS: Improving Neural Architecture Search via Attentive Sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 6418–6427.

[61] Tai-Hao Wen, Je-Min Hung, Hung-Hsi Hsu, Yuan Wu, Fu-Chun Chang, Chung-Yuan Li, Chih-Han Chien, Chin-I Su, Win-San Khwa, Jui-Jen Wu, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Mon-Shu Ho, Yu-Der Chih, Tsung-Yung Jonathan Chang, and Meng-Fan Chang. 2023. A 28nm Nonvolatile AI Edge Processor using 4Mb Analog-Based Near-Memory-Compute ReRAM with 27.2 TOPS/W for Tiny AI Edge Devices. In *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits).* 1–2. https://doi.org/10.23919/VLSITechnologyandCir57934.2023.10185326

[62] Arissa Wongpanich, Hieu Pham, James Demmel, Mingxing Tan, Quoc Le, Yang You, and Sameer Kumar. 2021. Training EfficientNets at Supercomputer Scale: 83 percent ImageNet Top-1 Accuracy in One Hour. In *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW).* 947–950. https://doi.org/10.1109/IPDPSW52791.2021.00146

[63] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. 2023. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 16133–16142.

[64] Donglai Xiang, Timur Bagautdinov, Tuur Stuyck, Fabian Prada, Javier Romero, Weipeng Xu, Shunsuke Saito, Jingfan Guo, Breannan Smith, Takaaki Shiratori, Yaser Sheikh, Jessica Hodgins, and Chenglei Wu. 2022. Dressing Avatars: Deep Photorealistic Appearance for Physically Simulated Clothing. *ACM Trans. Graph.* 41, 6, Article 222 (nov 2022), 15 pages. https://doi.org/10.1145/3550454.3555456

[65] Shanshan Xie, Can Ni, Aseem Sayal, Pulkit Jain, Fatih Hamzaoglu, and Jaydeep P Kulkarni. 2021. 16.2 eDRAM-CIM: Compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing. In *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 64. IEEE, 248–250.

[66] Changjae Yi, Donghyun Kang, and Soonhoi Ha. 2022. Hardware-Software Codesign of a CNN Accelerator. In *2022 25th Euromicro Conference on Digital System Design (DSD).* 348–356. https://doi.org/10.1109/DSD57027.2022.00054

[67] De-Qi You, Yen-Cheng Chiu, Win-San Khwa, Chung-Yuan Li, Fang-Ling Hsieh, Yu-An Chien, Chung-Chuan Lo, Ren-Shuo Liu, Chi-Cheng Hsieh, Kea-Tiong Tang, Yu-Der Chih, Tsung-Yung Jonathan Chang, and Meng-Fan Chang. 2023. An 8b-Precision 8-Mb STT-MRAM Near-Memory-Compute Macro Using Weight-Feature and Input-Sparsity Aware Schemes for Energy-Efficient Edge AI Devices. *IEEE Journal of Solid-State Circuits* (2023), 1–12. https://doi.org/10.1109/JSSC.2023.3324335

[68] Jiahui Yu and Thomas S Huang. 2019. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE/CVF international conference on computer vision.* 1803–1811.

[69] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. 2020. Bignas: Scaling up neural architecture search with big single-stage models. In *Computer Vision–ECCV 2020: 16th European Conference, Part VII 16.* Springer,

702–717.

[70] Yiwei Zhao, Ziyun Li, Win-San Khwa, Xiaoyu Sun, Sai Qian Zhang, Syed Shakib Sarwar, Kleber Hugo Stangherlin, Yi-Lun Lu, Jorge Tomas Gomez, Jae-Sun Seo, Phillip B. Gibbons, Barbara De Salvo, and Chiao Liu. 2024. Neural Architecture Search of Hybrid Models for NPU-CIM Heterogeneous AR/VR Devices. *arXiv preprint arXiv:2410.08326* (2024).

[71] Kaixiong Zhou, Xiao Huang, Qingquan Song, Rui Chen, and Xia Hu. 2022. Auto-gnn: Neural architecture search of graph neural networks. *Frontiers in big Data* 5 (2022), 1029307.

[72] Barret Zoph and Quoc Le. 2016. Neural Architecture Search with Reinforcement Learning. In *International Conference on Learning Representations*.