

HILAL: Hessian-Informed Layer Allocation for Heterogeneous Analog–Digital Inference

Aniss Bessalah

Ecole Nationale Supérieure d'Informatique

Algiers, Algeria

ka_bessalah@esi.dz

Karima Benatchba

Ecole Nationale Supérieure d'Informatique

Algiers, Algeria

k_benatchba@esi.dz

Hatem Mohamed Abdelmoumen

Ecole Nationale Supérieure d'Informatique

Algiers, Algeria

kh_abdelmoumen@esi.dz

Hadjer Benmeziane

IBM Research Europe

Rüschlikon, Switzerland

Hadjer.Benmeziane@ibm.com

Abstract—Heterogeneous AI accelerators that combine high-precision digital cores with energy-efficient analog in-memory computing (AIMC) units offer a promising path to overcome the energy and scalability limits of deep learning. A key challenge, however, is to determine which neural network layers can be executed on noisy analog units without compromising accuracy. Existing mapping strategies rely on ad-hoc heuristics and lack principled noise-sensitivity estimation. We propose HILAL (Hessian-Informed Layer Allocation), a framework that systematically quantifies layer robustness to analog noise using two complementary metrics: Hessian-based Noise Impact and Spectral Concentration Ratio. Layers are partitioned into robust and sensitive groups via clustering, enabling threshold-free mapping to analog or digital units. To further mitigate accuracy loss, we gradually offload layers to AIMC while retraining with noise-injection. Experiments on convolutional networks and transformers across CIFAR-10/100, ImageNet and SQuAD show that HILAL is on average 3.09x faster in search and mapping runtime than SOTA methods while achieving less accuracy degradation and maximizing analog utilization.

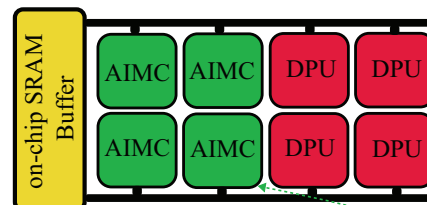
Index Terms—analog in-memory computing, neural network mapping, heterogeneous accelerators, noise robustness

I. INTRODUCTION

The growing demand for scaling Deep Neural Network (DNN) inference has pushed the limits of conventional digital accelerators. While digital processing units (DPUs) provide high precision and programmability, their energy efficiency is fundamentally constrained by frequent data movement between memory and compute [1]. Analog in-memory computing (AIMC) has recently emerged as a promising alternative that directly performs matrix–vector multiplications in memory arrays, offering orders-of-magnitude improvements in throughput and energy efficiency [2], [3]. However, AIMC units suffer from limited precision, device variability, and noise, which makes them unsuitable for executing neural networks without accuracy drop [4], [5].

Heterogeneous accelerators combine the strengths of both domains by tightly integrating digital cores with analog AIMC tiles [6]–[9]. A high-level depiction of such architectures is shown in Fig. 1a. The digital units provide accuracy and robustness for sensitive operations, while the analog units

a) Generic Heterogeneous Architecture



b) Heterogeneous Mapping of a Hybrid Conv-Attention Network



* Only mapping sensitive operations are kept
Memory and activations are omitted for the sake of clarity.

Fig. 1. (a) Generic heterogeneous accelerator architecture composed of digital processing units (DPUs) and analog in-memory computing (AIMC) tiles connected via an on-chip SRAM buffer. (b) Example mapping of a hybrid Conv–Attention network. Robust layers (e.g., convolution, feed-forward) are assigned to AIMC, while sensitive layers (e.g., attention projections, classifier) are executed on DPUs.

accelerate large linear computations with high efficiency. This heterogeneous design has emerged as a compelling path to break the energy bottleneck of AI inference systems, provided that layers can be mapped appropriately to the two compute domains.

A key challenge in this context is *layer allocation*: deciding which network layers are robust enough to be executed on noisy AIMC tiles and which layers require the precision of digital units. Existing methods lack a principled way to quantify robustness. Many adopt simplistic rules, such as mapping the first and last layers to digital and the rest to analog, assuming these layers are the most sensitive [6]. Others rely on heuristics that demand extensive retraining or empirical experimentation, which is time-consuming and does not generalize across archi-

tectures and datasets [10]–[12]. As illustrated in Fig. 1b, such assumptions are overly rigid: convolutional and feed-forward layers may often be robust enough for AIMC execution, whereas attention projections and classifiers typically require digital precision. A principled allocation strategy must therefore go beyond static heuristics and capture the true noise sensitivity of each layer.

Motivated by these limitations, we introduce **HILAL** (Hessian-Informed Layer Allocation), a framework that leverages second-order sensitivity analysis to guide layer mapping in heterogeneous accelerators. At its core, **HILAL** defines a new robustness signal called *Hessian-based Noise Impact (HNI)*, which estimates how much accuracy would degrade if a layer were executed on AIMC units. To complement this, **HILAL** also measures each layer’s *spectral concentration*, which reflects how strongly the layer amplifies small structured errors. Using these two signals, **HILAL** clusters layers into robust and sensitive groups, and derives hybrid mappings that maximize AIMC utilization without relying on hand-tuned thresholds or expensive retraining.

The main contributions of this work are as follows:

- We propose **HILAL**¹, the first Hessian-based framework for principled layer allocation in heterogeneous analog–digital accelerators, offering a hardware-agnostic way to quantify layer robustness to non-idealities.
- We introduce a clustering-based allocation strategy that separates robust and sensitive layers without ad hoc thresholds, enabling balanced mappings that preserve accuracy while maximizing analog utilization.
- We design a progressive hardware-aware retraining schedule that gradually offloads layers to AIMC, mitigating accuracy loss from noise.
- We evaluate **HILAL** on CNNs (AlexNet, VGG-16, ResNet) and Transformers (ViT, MobileBERT) across CIFAR-10/100, ImageNet and SQuAD, achieving up to 60% analog utilization with under 3% accuracy loss.

II. RELATED WORKS

Mapping strategies for heterogeneous analog–digital inference have been explored in only a few works, and existing approaches can be broadly categorized into *channel-wise* and *layer-wise* methods. Channel-wise schemes decide, for each convolutional channel (or group of channels), whether to execute it on analog or digital resources; layer-wise schemes assign entire layers at a time. The choice of granularity has significant implications for complexity, communication, and how well the mapping reflects true robustness.

Channel-wise methods (e.g., *Harmonica* [12] and *ODiMO* [11]) primarily target convolutions by selecting the most sensitive channels for digital computation and offloading the rest to analog cores. While this fine granularity can improve fidelity, it fragments the computation and introduces overheads that are rarely modeled in full: activations and partial sums must be routed across units, mixed-precision accumulation becomes

necessary, and additional ADC/DAC conversions arise when different channels use different precisions and compute domains. In practice, these communication and conversion costs complicate deployment and can erode the nominal efficiency benefits that motivate heterogeneous execution in the first place.

Layer-wise methods, in contrast, operate at the granularity of entire layers. A representative example is *LionHeart* [10], which explores the mapping space using heuristics (e.g., MAC-based ranking) and validates candidates via hardware-aware retraining against a user-defined accuracy target. Although effective, this strategy requires time-consuming training for a high number of candidate mappings and hinges on external thresholds, making it difficult to disentangle true noise robustness from exploration artifacts. Our work is also layer-wise but departs from these heuristics: rather than searching through many candidates, we aim to *predict* robust versus sensitive layers from the outset using principled sensitivity metrics. Importantly, we do not attempt to optimize efficiency in this paper; we assume a capable heterogeneous accelerator and focus on deriving mappings that fully exploit it while maintaining accuracy.

Finally, our approach is inspired by curvature-informed quantization. HAWQ [13], [14] demonstrated that the Hessian’s spectral properties are effective signals for per-layer precision selection, enabling robust mixed-precision assignments with minimal tuning. We adapt this insight to the analog–digital mapping problem: **HILAL** computes Hessian-informed descriptors tailored to analog noise (e.g., parameter-normalized expected loss increase and spectral concentration) and uses them to partition layers into robust and sensitive groups. This yields a threshold-free, theoretically grounded layer allocation that avoids channel-wise fragmentation and the heavy retraining loops characteristic of heuristic search.

We target architecture-agnostic mapping for heterogeneous analog–digital inference. Because we do not assume a specific AIMC substrate or micro-architecture, we evaluate **HILAL** on accuracy preservation vs. analog utilization under calibrated analog-noise models, and on mapping stability and overheads. This isolates the contribution of the allocation methodology from implementation-dependent performance numbers.

III. METHODOLOGY

We present **HILAL**, a framework that transforms a floating-point pretrained model into a mapped, analog-aware model with minimal accuracy degradation. A high-level overview is shown in Fig. 2. **HILAL** operates in three stages: (1) **Per-layer sensitivity metrics computation**, where we construct robustness descriptors for each candidate layer; (2) **Sensitivity-based clustering**, which partitions layers into analog-robust and digital-sensitive groups to derive a binary mapping; (3) **Layer-wise Analog Fine-Tuning (LAFT)**, a gradual offloading procedure in which analog-robust layers are sequentially mapped to AIMC units and fine-tuned with noise injection to preserve accuracy.

1) Per-layer sensitivity metrics computation

¹The full code of our methodology is available: <https://github.com/anissbslh/HILAL>

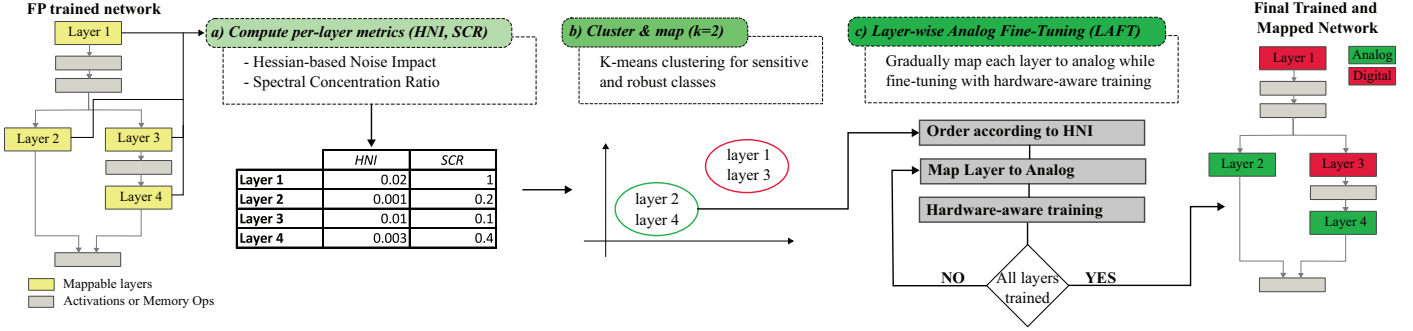


Fig. 2. Overview of the proposed Hessian-based mapping framework. Starting from a pretrained network, we compute per-layer descriptors from Hessian statistics and hardware noise parameters, cluster layers into robust and sensitive groups via k-means, and progressively map robust layers to analog units while performing hardware-aware retraining until convergence.

HILAL begins by assigning each mappable layer (i.e., convolution, fully connected, and projection layers) a two-dimensional robustness descriptor. The first axis, the **Hessian-based Noise Impact (HNI)**, quantifies the expected loss increase under analog noise by leveraging second-order curvature information of the loss landscape. This provides a principled estimate of how sensitive each layer is to noise injection. The second axis is the **spectral concentration ratio (SCR)**, defined as the ratio between the largest Hessian eigenvalue and the trace. A high ratio indicates that curvature is dominated by a single direction, making the layer more vulnerable to noise. Prior work such as HAWQ [13], [14] has demonstrated the importance of analyzing the Hessian Eigenspectrum for identifying fragile layers. Together, these two axes yield a compact yet informative representation of per-layer robustness to analog noise.

Definition 1 (Hessian-based Noise Impact). Let $\mathcal{L}(w)$ denote the training loss and $w_l \in \mathbb{R}^{d_l}$ the parameter vector of layer l . Suppose analog device imperfections induce a noise

$$w'_l = w_l + \Delta W_l, \quad \Delta W_l = \gamma_{\text{weight}} \cdot \max(|W_l|) \cdot \tau, \quad \tau \sim \mathcal{N}(0, I),$$

with scale γ_{weight} and dimension $d_l = |W_l|$. The *Hessian-based Noise Impact (HNI)* of layer l is defined as

$$\text{HNI}_l := \frac{\sigma_l^2}{2d_l} \text{Tr}(H_l), \quad \sigma_l^2 = (\gamma_{\text{weight}} \cdot \max(|W_l|))^2,$$

where $H_l = \nabla_{w_l}^2 \mathcal{L}$ is the layer Hessian.

Proposition 1. HNI_l equals the expected per-parameter loss increase under Gaussian weight noise, up to second-order approximation.

Proof. Applying a second-order Taylor expansion around w_l yields

$$\mathcal{L}(w_l + \Delta W_l) \approx \mathcal{L}(w_l) + \nabla \mathcal{L}(w_l)^\top \Delta W_l + \frac{1}{2} \Delta W_l^\top H_l \Delta W_l.$$

At a local minimizer, $\nabla \mathcal{L}(w_l) \approx 0$, so the expected increase in loss is

$$\mathbb{E}[\Delta \mathcal{L}] \approx \frac{1}{2} \mathbb{E}[\Delta W_l^\top H_l \Delta W_l].$$

Since $\Delta W_l \sim \mathcal{N}(0, \sigma_l^2 I)$, $\mathbb{E}[\Delta W_l^\top H_l \Delta W_l] = \sigma_l^2 \text{Tr}(H_l)$. Hence

$$\mathbb{E}[\Delta \mathcal{L}] \approx \frac{\sigma_l^2}{2} \text{Tr}(H_l).$$

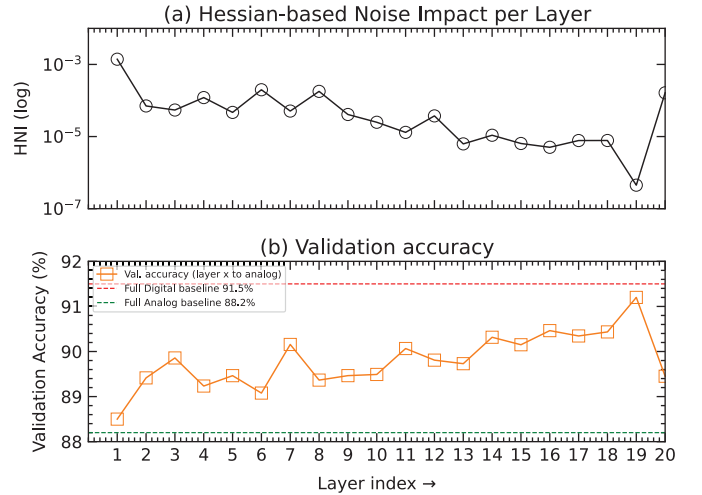


Fig. 3. **HNI correlates with robustness.** (a) HNI across ResNet-20 layers spans orders of magnitude. (b) When only one layer runs on analog, high-HNI layers drop accuracy by up to 6 points, while low-HNI layers stay near the 91.5% baseline.

Normalizing by d_l prevents bias toward larger layers, giving

$$\text{HNI}_l = \frac{1}{d_l} \mathbb{E}[\Delta \mathcal{L}],$$

as claimed.

Fig. 3 demonstrates the practical relevance of HNI on ResNet-20 (CIFAR-10). As shown in Fig. 3a, HNI varies by several orders of magnitude across layers, indicating large heterogeneity in noise sensitivity. Fig. 3b shows validation accuracy when only one layer is mapped to analog and fine-tuned. High-HNI layers lead to significant accuracy loss (up to 5–6 percentage points), while low-HNI layers remain within 1 point of the baseline (91.5%). This directly validates HNI: it reliably predicts which layers are fragile to analog noise and should remain digital.

Definition 2 (Spectral Concentration Ratio). For a layer l with Hessian $H_l \in \mathbb{R}^{d_l \times d_l}$, let $\lambda_{\max}(H_l)$ denote its largest

eigenvalue and $\text{Tr}(H_l)$ the trace. The SCR is defined as

$$\text{SCR}_l := \frac{\lambda_{\max}(H_l)}{\text{Tr}(H_l)}.$$

Intuition. SCR_l quantifies how concentrated the curvature is along the most sensitive direction of the loss landscape. A high ratio indicates that a single eigen-direction dominates. HAWQ [13], [14] demonstrated the utility of the Hessian’s Eigenvalues for identifying fragile layers in quantization; we extend its use to the analog-noise robustness setting. We form the per-layer descriptor

$$v_l = [\text{HNI}_l, \text{SCR}_l],$$

To ensure comparability across layers within the same network, each feature dimension is standardized using z-score normalization prior to clustering.

Efficient and accurate estimation of $\text{Tr}(H_l)$ and $\lambda_{\max}(H_l)$ is critical. For the trace, we adopt Hutchinson’s stochastic estimator with Rademacher probe vectors [15], which yields an unbiased estimate with variance controlled by the number of probes; in practice, $m = 10$ strikes a favorable balance between accuracy and cost. For the dominant eigenvalue, we employ the Power Iteration method initialized with a random vector [16]. A modest number of iterations (e.g., 20 steps) suffices to approximate λ_{\max} reliably. Both estimators rely only on Hessian–vector products [17], which are obtained via automatic differentiation in PyTorch [18], avoiding explicit construction of the Hessian.

2) Sensitivity-based clustering

Given the two-dimensional descriptors, we partition the set of mappable layers using k -means [19] with $k = 2$. The cluster whose centroid has the larger HNI value—and typically also a larger SCR—is labeled as *sensitive* and assigned to digital execution, while the complementary cluster is labeled *robust* and mapped to analog units.

A common alternative in the literature is to apply user-defined thresholds on sensitivity scores. However, such thresholds are problematic in our setting: (i) the optimal cutoff can vary significantly across layers due to differences in scale and function, and (ii) there is no established ground truth for thresholding metrics such as HNI or SCR. Clustering instead adapts automatically to the score distribution of each model, eliminating the need for manual calibration while maintaining interpretability.

To address the sensitivity of k -means to random initialization, we run the algorithm with multiple restarts and retain the solution with the lowest *inertia*, defined as the within-cluster sum of squared distances from each point to its assigned centroid. This criterion selects the partition that best captures the natural grouping of the descriptors. In addition, we validate the stability of the resulting analog/digital mapping across different runs in our ablation studies.

3) Layer-wise Analog Fine-Tuning (LAFT)

Mapping all analog-assigned layers at once often causes a sharp accuracy drop. A single fine-tuning stage struggles to recover from this. To mitigate this, we propose *Layer-wise Analog Fine-Tuning (LAFT)*, a staged schedule where analog-robust

Algorithm 1: Layer-wise Analog Fine-Tuning (LAFT)

Input: Pretrained FP model \mathcal{M} ; analog-assigned layer set \mathcal{A} from clustering; ordering π over \mathcal{A} by increasing HNI; noise model $\mathcal{N}(\cdot)$; stage epochs E ; batch size B ; optimizer \mathcal{O} ; early-stop patience P

Output: Analog-aware model \mathcal{M}^*

Initialize $\mathcal{M}_0 \leftarrow \mathcal{M}$ (all layers FP, no noise)

Initialize mapped set $\mathcal{S} \leftarrow \emptyset$

for $t = 1$ to $|\mathcal{A}|$ **do**

$l \leftarrow \pi(t)$ // Next most-robust layer

$\mathcal{S} \leftarrow \mathcal{S} \cup \{l\}$ // Cumulative mapping

Enable noise injection for layers in \mathcal{S}

Reset stage-best validation and patience counter

for $e = 1$ to E **do**

for each mini-batch (x, y) of size B **do**

Forward: compute $\hat{y} = \text{Forward}_{\mathcal{M}_t}(x)$

where

for $j \in \mathcal{S}$, apply noise injection via

$w'_j = w_j + \Delta W_j, \Delta W_j \sim \mathcal{N}(\cdot)$

Compute loss $\mathcal{L}(\hat{y}, y)$

Backward: gradients via autodiff

(reparameterizing through noise)

Update parameters with optimizer \mathcal{O}

end

Evaluate on validation set; update stage-best

metric

if no improvement for P epochs **then**

| break

end

Optionally decay learning rate

end

end

Return $\mathcal{M}^* \leftarrow \mathcal{M}_{|\mathcal{A}|}$

layers are introduced gradually. Specifically, we sort the analog-assigned layers by increasing HNI (most robust first) and then, at each stage, offload the next layer to analog, enable noise injection for that layer during forward passes, and fine-tune the model for a small number of epochs. Each stage stops early if validation accuracy does not improve for P epochs. This progressive exposure limits the noise magnitude per step, giving gradients the opportunity to compensate locally for the newly introduced noise. As shown in Algorithm 1, the set of analog layers grows cumulatively until all robust layers have been mapped. Empirically, LAFT consistently yields higher final accuracy at comparable analog utilization than one-shot mapping with a monolithic retrain. This observation aligns with prior findings that staged adaptation improves resilience in AIMC training pipelines [10], [20].

LAFT always begins from a high-accuracy FP checkpoint rather than an analog-trained model. This design ensures (i) *isolation of allocation effects*, since performance changes can be attributed directly to mapping decisions rather than noisy optimization dynamics; (ii) *stability and efficiency*, as pretrained

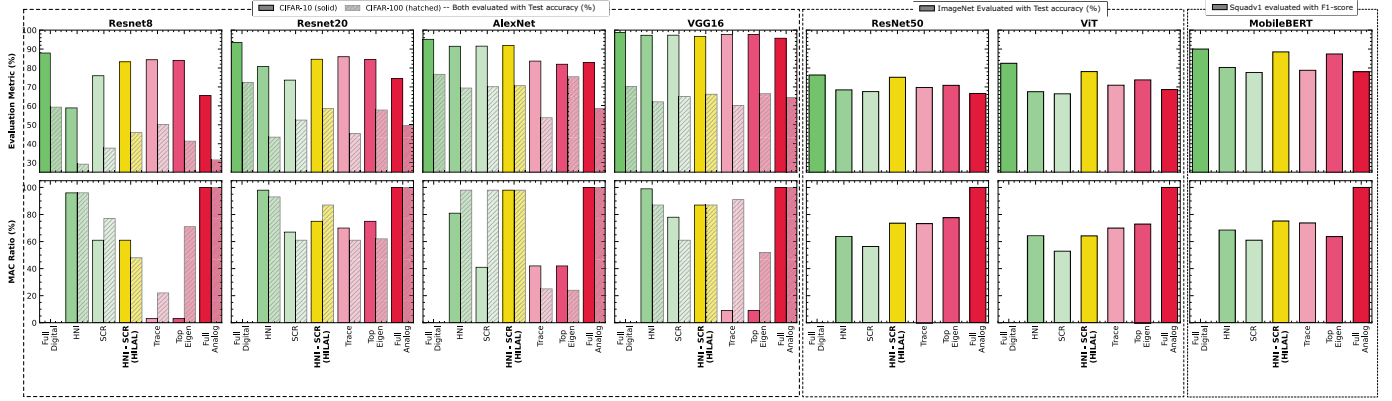


Fig. 4. Evaluation of HILAL (highlighted in yellow) across CIFAR-10/100 CNNs, ImageNet (ResNet-50/ViT), and SQuAD v1.1 (MobileBERT). **Top row:** task performance (test accuracy for CIFAR-10/100 and ImageNet; F1 score for SQuAD). **Bottom row:** analog utilization, reported as the MAC ratio. Solid bars correspond to CIFAR-10; hatched bars to CIFAR-100. Baselines include Full Digital, Full Analog, and variants using only HNI, SCR, or single Hessian statistics. HILAL consistently balances high performance with high analog utilization.

FP models converge rapidly with small fine-tuning adjustments, while analog-from-scratch training is slower and less stable; (iii) *fairness to baselines*, which also assume FP starts; and (iv) *architecture-agnostic generality*, since analog-from-scratch would risk overfitting to a specific noise instantiation.

IV. RESULTS AND COMPARISON

A. Experimental Setup

We evaluate HILAL on vision and NLP tasks: CIFAR-10/100 with AlexNet [21], VGG-16 [22], ResNet-8/20 [23]; ImageNet with ResNet-50 (and ViT-B/16 for completeness) [24], [25]; and SQuAD v1.1 with MobileBERT [26], [27]. For CIFAR/ImageNet, we split the original training set into 90% train / 10% val; for SQuAD, we use the official dev set; test sets are used only for final reporting.

All runs use a V100-based cluster. CNNs train for 200 epochs; MobileBERT for 3. Optimization is SGD with momentum and cross-entropy. On CIFAR/ImageNet we use cosine LR ($T_{\max}=200$, batch 256); on SQuAD we use a linear LR ($T_{\max}=3$, batch 8). Digital training uses LR= 0.01 (CIFAR/ImageNet) and 0.05 (SQuAD), momentum 0.85/0.99, weight decay 0/0.0005. For analog HWA retraining [20], [28], we initialize from digital weights and simulate non-idealities with AIHWKIT [29] (PCMLikeNoiseModel): digital biases, 256×256 tiles with even layer splitting, and additive Gaussian noise ($\sigma=0.08$) during training. Analog LR is 0.001 (CIFAR) and 0.02 (SQuAD), momentum 0.85/0.9, weight decay 0/0.003. Results are averaged over 20 independent runs to account for device stochasticity; we report 1-day accuracy to capture PCM drift effects [30].

B. General Results

Fig. 4 presents the performance of CNN architectures under HILAL and its variants. We report both the 1-day test accuracy and the achieved MAC ratio on CIFAR-10 and CIFAR-100.

As expected, the Full Digital mapping provides the highest accuracy across all models, but at the cost of efficiency, with 0% MAC ratios. Conversely, the Full Analog mapping achieves

the maximum MAC ratios, but accuracy collapses (dropping below 70% in most cases, and down to $\sim 60\%$ on VGG-16 with CIFAR-100) due to noise accumulation.

HILAL achieves a favorable trade-off in most cases. For instance, on CIFAR-10 with VGG-16, HILAL maintains accuracy above 95%, reducing the degradation relative to Full Digital to less than 2 percentage points, while still reaching a MAC ratio of about 85%. On ResNet-20 with CIFAR-100, HILAL recovers nearly 10 percentage points in accuracy compared to Full Analog, while sustaining a MAC ratio above 80%.

To further evaluate generality, we extended our experiments to large-scale models as shown in Fig. 4. With Full Digital mapping, both models achieve their expected baseline accuracy ($\sim 76\%$ for ResNet-50 and $\sim 82\%$ for ViT-B/16), but with negligible MAC ratio. Full Analog execution again delivers very high efficiency, but accuracy drops by more than 10 percentage points. In contrast, HILAL limits accuracy degradation to under 2–3 points (e.g., 74% for ResNet-50 and 79% for ViT-B/16) while still securing MAC ratios in the range of 65–70%. This confirms that our method scales effectively to large-scale vision models and outperforms variants that plateaued around 65–72% accuracy or used less than 70% analog compute.

On MobileBERT with SQuAD, Full Digital obtained an F1 score of 90.3 with negligible MAC ratio, while Full Analog dropped to 78.0. HILAL preserved most of the accuracy while enabling a 75% MAC ratio, achieving the best balance between accuracy and efficiency.

Overall, the results demonstrate that HILAL is not limited to CNNs or vision tasks. It generalizes to modern architectures such as transformers and language models, consistently achieving high efficiency without substantial accuracy loss.

C. Comparison with mapping optimization methods

We compare HILAL to LionHEART [10], ODiMo [11], Harmonica [12], and DIANA [6] on ResNet-8/20, VGG-16, AlexNet (CIFAR-10), ResNet-50 (ImageNet), and MobileBERT (SQuAD). Fig. 5 summarizes the results. Across all networks *except* ResNet-50, HILAL delivers the highest

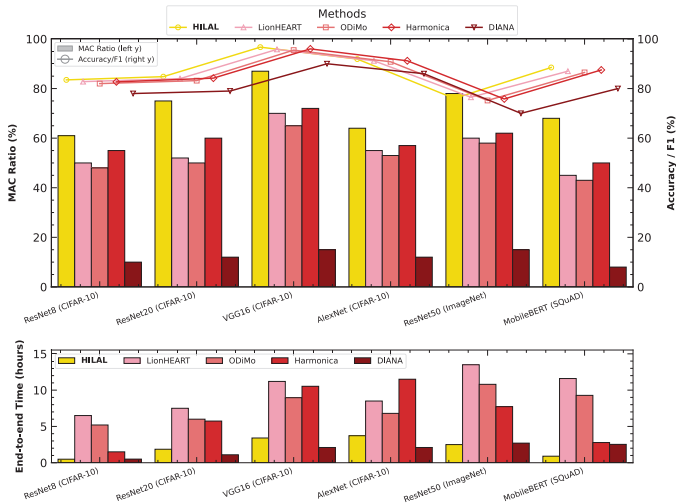


Fig. 5. Comparison with mapping-optimization methods. Top: Analog utilization (MAC Ratio, left y) and Accuracy/F1 (right y). Bottom: End-to-end execution time (hours).

accuracy while sustaining markedly higher analog utilization (MAC Ratio). On ResNet-50, LionHEART is slightly more accurate but at a substantially higher end-to-end time cost. DIANA attains the shortest times (seconds-scale) but consistently exhibits lower accuracy. Among accuracy-preserving methods, HILAL is consistently faster than Harmonica and ODiMo and significantly faster than LionHEART, while enabling more analog compute.

We note that our layer-wise analog fine-tuning schedule (LAFT) can *occasionally* outpace DIANA in wall-clock time: by retraining gradually, layer-by-layer, LAFT avoids repeated full-model restarts and reduces unnecessary backpropagation on stabilized layers, shortening convergence on some networks. A per-model breakdown of MAC Ratio, Accuracy/F1, and end-to-end time is provided in Fig. 5.

V. ABLATION STUDY

Our ablation study validates the design choices underlying our per-layer descriptor. We examine whether both components are needed to capture sensitivity to analog noise. We compare mapping outcomes from the two-dimensional descriptor [HNI, SCR] against two baselines relying on a single statistic: HNI-only and SCR-only.

Fig. 6 shows that relying on a single feature degrades analog/digital partitioning. In ResNet-8 and ResNet-20, mappings from HNI-only or SCR-only fail to identify noise-sensitive layers. As a result, these metrics assign such layers to the analog cluster, leading to an accuracy drop after retraining. On ResNet-8, HNI-only reaches 58.92% 1-day test accuracy on CIFAR-10 and 29.22% on CIFAR-100, compared to 83.27% on CIFAR-10 and 45.85% on CIFAR-100 with our descriptor. For ResNet-20, SCR-only achieves 73.60% on CIFAR-10 and 52.49% on CIFAR-100, while [HNI, SCR] improves to 84.62% and 58.53%, with a higher MAC ratio.

These results reflect complementary roles. HNI captures the expected effect of stochastic weight perturbations but is agnos-

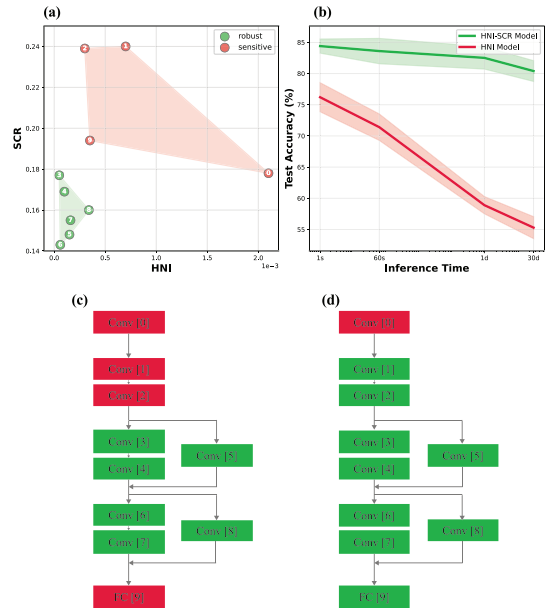


Fig. 6. Ablation study on the components of the layer descriptor. (a) HNI and SCR values for each layer of ResNet-8 on CIFAR-10, showing a natural separation into robust and sensitive layers. (b) Test accuracy over time for models using the full descriptor [HNI, SCR] vs. [HNI] only. Resulting layer-to-AIMC allocations using (c) [HNI, SCR] and (d) [HNI] only

tic to curvature directionality. SCR captures concentration in the Hessian spectrum but lacks a measure of overall noise-induced loss increase. Each alone misses key information, yielding poor mappings. Together, they provide a faithful representation of layer sensitivity, allowing k -means to separate robust and fragile layers across architectures.

The superiority of the joint descriptor is not confined to these ResNet variants. Across most models in our evaluation suite, [HNI, SCR] yields higher post-retraining accuracy than either ablation, improving by 2–20% depending on architecture.

VI. CONCLUSION

In this paper, we introduced HILAL, a Hessian-guided framework for layer allocation in heterogeneous analog–digital deployment of neural networks. Our method computes an interpretable two-dimensional descriptor per layer from Hessian trace and spectral concentration, clusters layers automatically, and performs progressive hardware-aware retraining while mapping robust layers to analog. The resulting mappings are threshold-free, and produce favorable accuracy vs analogization trade-offs compared to recent baselines. Extensive ablation studies show that the normalized HNI and SCR are effective sensitivity predictors, and LAFT substantially improves recovery compared to one-shot mapping. Future work includes extending the method to joint latency/energy-aware optimization objectives, exploring channel/tile granularity mappings with scalable search, and validating mappings on AIMC hardware.

REFERENCES

- [1] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, "A modern primer on processing in memory," in *Emerging computing: from devices to systems: looking beyond Moore and Von Neumann*. Springer, 2022, pp. 171–243.
- [2] I. Boybat, M. Le Gallo, S. Nandakumar, T. Moraitis, T. Parnell, T. Tuma, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, "Neuromorphic computing with multi-memristive synapses," *Nature communications*, vol. 9, no. 1, p. 2514, 2018.
- [3] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature nanotechnology*, vol. 15, no. 7, pp. 529–544, 2020.
- [4] I. Boybat, B. Kersting, S. G. Sarwat, X. Timoneda, R. L. Bruce, M. BrightSky, M. Le Gallo, and A. Sebastian, "Temperature sensitivity of analog in-memory computing using phase-change memory," in *2021 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2021, pp. 28–3.
- [5] X. Yang, B. Taylor, A. Wu, Y. Chen, and L. O. Chua, "Research progress on memristor: From synapses to computing systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 5, pp. 1845–1857, 2022.
- [6] K. Ueyoshi, I. A. Papistas, P. Houshmand, G. M. Sarda, V. Jain, M. Shi, Q. Zheng, S. Giraldo, P. Vranckx, J. Doevenspeck *et al.*, "Diana: An end-to-end energy-efficient digital and analog hybrid neural network soc," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65. IEEE, 2022, pp. 1–3.
- [7] S. Jain, H. Tsai, C.-T. Chen, R. Muralidhar, I. Boybat, M. M. Frank, S. Woźniak, M. Stanisavljevic, P. Adusumilli, P. Narayanan *et al.*, "A heterogeneous and programmable compute-in-memory accelerator architecture for analog-ai using dense 2-d mesh," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 1, pp. 114–127, 2022.
- [8] G. Raveendran Nair, F. Jiang, J. Zhang, and Y. Cao, "A 16nm heterogeneous accelerator for energy-efficient sparse and dense ai computing," in *Proceedings of the 29th ACM/IEEE International Symposium on Low Power Electronics and Design*, 2024, pp. 1–6.
- [9] I. Boybat, T. Boesch, M. Allegra, M. Baldo, J. Bertolini-Agnoletto, G. Burr, A. Buschini, A. Cabrini, E. Calvetti, C. Cappetta *et al.*, "Heterogeneous embedded neural processing units utilizing pcm-based analog in-memory computing," in *2024 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2024, pp. 1–4.
- [10] C. Lammie, Y. Wang, F. Ponzina, J. Klein, H. Benmeziane, M. Zapater, I. Boybat, A. Sebastian, G. Ansaloni, and D. Atienza, "Lionheart: A layer-based mapping framework for heterogeneous systems with analog in-memory computing tiles," *IEEE Transactions on Emerging Topics in Computing*, 2025.
- [11] M. Risso, A. Burrello, G. M. Sarda, L. Benini, E. Macii, M. Poncino, M. Verhelst, and D. J. Pagliari, "Precision-aware latency and energy balancing on multi-accelerator platforms for dnn inference," in *2023 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2023, pp. 1–6.
- [12] P. Behnam, U. Kamal, A. Shafiee, A. Tumanov, and S. Mukhopadhyay, "Harmonica: Hybrid accelerator to overcome imperfections of mixed-signal dnn accelerators," in *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2024, pp. 619–630.
- [13] Z. Dong, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer, "Hawq: Hessian aware quantization of neural networks with mixed-precision," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 293–302.
- [14] Z. Dong, Z. Yao, D. Arfeen, A. Gholami, M. W. Mahoney, and K. Keutzer, "Hawq-v2: Hessian aware trace-weighted quantization of neural networks," *Advances in neural information processing systems*, vol. 33, pp. 18 518–18 529, 2020.
- [15] M. F. Hutchinson, "A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines," *Communications in Statistics-Simulation and Computation*, vol. 18, no. 3, pp. 1059–1076, 1989.
- [16] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.
- [17] B. A. Pearlmutter, "Fast exact multiplication by the hessian," *Neural computation*, vol. 6, no. 1, pp. 147–160, 1994.
- [18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.
- [19] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation," *Electronics*, vol. 9, no. 8, p. 1295, 2020.
- [20] M. J. Rasch, C. Mackin, M. Le Gallo, A. Chen, A. Fasoli, F. Odermatt, N. Li, S. Nandakumar, P. Narayanan, H. Tsai *et al.*, "Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators," *Nature communications*, vol. 14, no. 1, p. 5282, 2023.
- [21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [25] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.
- [26] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "Mobilebert: a compact task-agnostic bert for resource-limited devices," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2158–2170.
- [27] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016.
- [28] C. Lammie, A. Vasilopoulos, J. Büchel, G. Camposampiero, M. Le Gallo, M. Rasch, and A. Sebastian, "Improving the accuracy of analog-based in-memory computing accelerators post-training," in *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2024, pp. 1–5.
- [29] M. Le Gallo, C. Lammie, J. Büchel, F. Carta, O. Fagbohunge, C. Mackin, H. Tsai, V. Narayanan, A. Sebastian, K. El Maghraoui *et al.*, "Using the ibm analog in-memory hardware acceleration kit for neural network training and inference," *APL Machine Learning*, vol. 1, no. 4, 2023.
- [30] I. Boybat, S. Nandakumar, M. Le Gallo, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, "Impact of conductance drift on multi-pcm synaptic architectures," in *2018 Non-Volatile Memory Technology Symposium (NVMTS)*. IEEE, 2018, pp. 1–4.