EXPERT-LEVEL ANALOG–DIGITAL MAPPING OF MIXTURE-OF-EXPERTS WITH THEORETICAL GENERALIZATION GUARANTEES

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-Experts (MoE) models enable scalable learning by activating only a subset of parameters per input, but remain memory-bound due to frequent movement of expert weights. Analog in-memory computing (AIMC) offers a promising solution by performing matrix–vector multiplications (MVMs) directly within memory, reducing data movement and improving energy efficiency. However, existing AIMC deployments map all MVMs uniformly to the analog units, leading to significant accuracy degradation for large MoE models due to hardware non-idealities. In this paper, we show that noise sensitivity varies significantly across MVM operations, with dense modules (such as attention layers) and certain experts being particularly vulnerable to analog noise. We then introduce a theoretically grounded metric, the maximum neuron norm score (MaxNNScore), to identify noise-sensitive experts. Our theoretical results prove that selectively mapping high-MaxNNScore experts to the digital units significantly improves the tolerable noise level while maintaining generalization compared to uniform analog execution. Empirical evaluations on large-scale MoE models, including DeepSeekMoE (16B), Qwen-1.5 MoE (14B) and OLMoE (7B), demonstrate that our approach recovers near full-precision performance (within 3-8% drop) over several benchmark LLM tasks while mapping only 11-29% of parameters to digital.

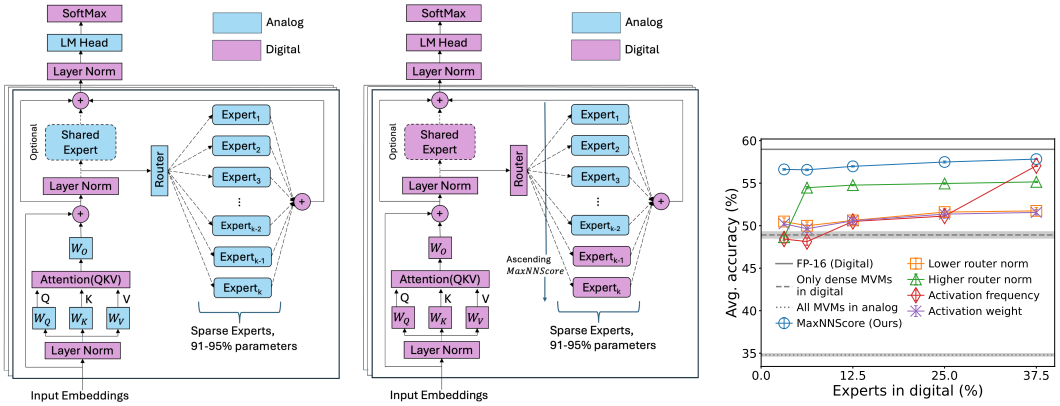
1 INTRODUCTION

Mixture-of-Experts (MoE) architectures enable the scaling of large neural networks by activating only a small subset of parameters per input (Shazeer et al., 2017; Riquelme et al., 2021; Fedus et al., 2022; Chowdhury et al., 2023). This sparsity has made MoE a key component in modern large language models (Jiang et al., 2024; Guo et al., 2025; Yang et al., 2025). However, despite reduced per-token computation, these models remain memory-bound: each forward pass requires fetching expert weights from memory, resulting in substantial memory traffic and energy consumption. As model sizes grow, data movement, rather than computation, becomes the dominant bottleneck (Frantar & Alistarh, 2024; Chowdhury et al., 2024; Jegham et al., 2025; Fernandez et al., 2025).

Analog in-memory computing (AIMC) offers a promising solution to this bottleneck by performing matrix-vector multiplications (MVMs) directly within non-volatile memory (NVM) arrays (Burr et al., 2017; Joshi et al., 2020; Sebastian et al., 2020; Fujiwara et al., 2024). By co-locating storage and computation, AIMC significantly reduces data movement and improves energy efficiency for large-scale models. Its effectiveness has been primarily on moderate-scale dense neural networks, where model sizes and layer widths remain relatively limited (Rasch et al., 2023; Büchel et al., 2025a; Hou et al., 2025b;a). AIMC has recently been extended to large-scale MoE models in Büchel et al. (2025b), which adopts a similar design principle as in smaller models: MVM operations of linear layers are mapped to analog NVM tiles, while other components, such as nonlinear activations, normalization layers, and attention, are executed on digital processing units (DPUs), as illustrated in Figure 1(a). Although this architecture is heterogeneous at the system level (analog vs. digital), it treats all expert MVMs uniformly by mapping them entirely to the analog NVM arrays.

Due to hardware non-idealities, including DAC/ADC quantization, device noise, and weight programming errors, analog MVMs are inherently approximate (Hou et al., 2025b). Mapping all expert

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) existing (b) proposed (c)

Figure 1: **(a)** Existing AIMC mapping, where MVMs of all linear layers in analog, while other operations (e.g., activation, attention, layer norm, softmax) are in digital (Büchel et al., 2025b). **(b)** Proposed mapping of MoE, where noise-sensitive experts identified via MaxNNScore and the dense MVMs are mapped to digital. **(c)** Average performance over seven benchmark tasks vs. the percentage of experts mapped to digital for DeepSeekMoE (simulated on AIHWKIT-Lightning (Büchel et al., 2024)). Mapping dense MVMs (only $\sim 9\%$ of total model parameters) in digital recovers more than half of the accuracy drop, while additionally mapping only $\sim$3% of the experts in digital limits the accuracy drop to within 2% of FP-16, significantly outperforming other expert selection baselines.

computations to analog can therefore lead to significant accuracy degradation, especially in large MoE models with many experts and MVM operations. Moreover, existing approaches that mitigate this degradation rely on noise-aware retraining (Rasch et al., 2023; Büchel et al., 2025a), which becomes prohibitively expensive at the scale of modern MoE models. Meanwhile, MoE models exhibit intrinsic heterogeneity across experts: different experts specialize in different tasks and contribute unequally to model predictions. This suggests a new opportunity: one can exploit expert-level differences to enable heterogeneous mapping of MVMs across analog and digital units to balance energy efficiency and generalization accuracy. This leads to a fundamental question that has not been addressed:

How can we identify noise-sensitive experts for heterogeneous analog–digital mapping at the level of expert MVMs in MoE, with generalization guarantees?

In this paper, we address this question both theoretically and empirically. Our analysis shows that experts composed of neurons with large ℓ_2-norms are more susceptible to noise and non-idealities in analog NVM devices. Accordingly, we map the MVMs of these noise-sensitive experts to digital units, while executing the remaining experts on analog hardware. We further assign the MVMs of dense modules, including the query, key, and value projections, the LM head, and shared experts (if any), to digital units, as they are highly sensitive to analog noise while accounting for only a small fraction of the total parameters and compute. We summarize our main contributions as follows:

1. **The first analog noise-sensitivity analysis across MVM operations in large MoE models.** To the best of our knowledge, this paper presents the first systematic analysis of analog noise sensitivity across different MVM operations in large MoE models under AIMC non-idealities. Our results show that noise sensitivity varies significantly across experts, with a subset exhibiting significant performance degradation under analog noise, motivating their execution in digital. In addition, we find that dense MVMs in MoE, although accounting for only a small fraction of total parameters and compute, are consistently sensitive to analog noise, making them better suited for digital execution.

2. **A theoretically grounded metric with noise-tolerance guarantees for expert selection**. We introduce the maximum neuron norm score (MaxNNScore), defined as the product of the maximum neuron ℓ_2 norms of all linear projection layers of an expert, as a principled metric to identify noise-sensitive experts for digital execution. We establish theoretical guarantees on the tolerable analog noise level for both uniform analog execution and heterogeneous analog–digital mapping. We prove that selective mapping of high-MaxNNScore experts to digital significantly increases the

noise tolerance compared to uniform analog execution, providing the first theoretical foundation for noise-aware heterogeneous mapping in MoE and justify $\mathrm{MaxNNScore}$ as a principled metric.

3. **Empirical validation of heterogeneous expert mapping on large-scale MoE models**. We evaluate our heterogeneous mapping strategy on large-scale MoE models, including DeepSeekMoE (Dai et al., 2024) (16B parameters), Qwen-1.5 MoE (Team, 2024) (14B parameters) and OLMoE (Muennighoff et al., 2025) (7B parameters), across a diverse set of LLM benchmarks. For example, our results show that the $\mathrm{MaxNNScore}$-based expert selection, combined with digital mapping of dense MVMs, limits the accuracy drop to within 3% of average full-precision performance over seven benchmark datasets while mapping < 9% of the total model parameters to digital (see Figure 1(c)). Furthermore, the metric $\mathrm{MaxNNScore}$ consistently outperforms other expert selection metrics, demonstrating strong robustness under analog noise.

1.1 Related works

Analog in-memory computing of large models. The evaluation of in-memory computing performance is largely limited to smaller-sized models of different architectures, such as CNNs (Joshi et al., 2020; Gokmen et al., 2019), RNNs Rasch et al. (2023), LSTMs(Tsai et al., 2019), and encoder-only Transformers (Spoon et al., 2021). Zhang et al. (2024) minimize DAC-ADC noise for LLMs through activation-shifting and weight-reshaping. Büchel et al. (2025a) conduct large-scale noise-aware retraining for foundation LLMs. Hou et al. (2025b) conduct sensitivity analysis of LLMs on various analog noise sources and propose a training-free rescaling method to mitigate the noise. In all of the cases, the model sizes (1-3B) are well below typical MoE parameter ranges.

Efficient inference of Mixture-of-Experts. Due to the large memory requirements of MoE, inference cost reduction has drawn much attention recently (Koishekenov et al., 2023; Li et al., 2024b; Chowdhury et al., 2024; Frantar & Alistarh, 2024; Huang et al., 2025). Most of the works focus on expert pruning and quantization to mitigate inference cost. Büchel et al. (2025b) demonstrate improved scalability of MoE in 3D AIMC architecture compared to dense models.

Theoretical generalization guarantees of neural networks. Due to high non-convexity, the theoretical generalization analysis of neural networks is mostly limited to two layers (Chowdhury et al., 2023; Bu et al., 2024; Li et al., 2025b). Feature learning dynamics is widely used to gain insights about different training and inference properties (Li et al., 2024a; 2025a). Most works require a simplified data model to facilitate analysis (Brutzkus et al., 2018; Karp et al., 2021; Shi et al., 2022).

2 Background

2.1 The Mixture-of-Experts architecture

An MoE block replaces the dense feed-forward network (FFN) of a Transformer layer with a set of sparsely activated FFN modules, referred to as *experts*. A routing network determines which experts are activated for each input token, enabling conditional computation. Common routing strategies include *token-choice routing* (Fedus et al., 2022), where each token selects its top experts, and *expert-choice routing* (Zhou et al., 2022), where each expert selects a subset of tokens.

For an MoE block with k experts, each expert $s \in [k]$ is implemented as a two-layer MLP or gated MLP with projection matrices $\mathbf{W}_{\mathrm{up}}^{(s)} \in \mathbb{R}^{d\times m}$ and $\mathbf{W}_{\mathrm{down}}^{(s)} \in \mathbb{R}^{m\times d}$. In the gated-MLP variant, an additional gate-projection matrix $\mathbf{W}_{\mathrm{gate}}^{(s)} \in \mathbb{R}^{d\times m}$ is used. The routing network is parameterized by a routing matrix $\mathbf{\Sigma} \in \mathbb{R}^{d\times k}$.

For an input token $\mathbf{x} \in \mathbb{R}^d$, the routing network computes routing scores $\mathbf{x}^\top\mathbf{\Sigma}$. A sparse subset of experts is activated according to the routing strategy. The output token is computed as $\mathbf{x}_{\mathrm{out}} = \sum_{s\in\mathcal{A}} \mathbf{f}_s(\mathbf{x})$, where $\mathcal{A} \subseteq [k]$ denotes the set of activated experts and $\mathbf{f}_s(\cdot)$ is the expert function.

For a standard MLP expert in an MoE block,

$$\mathbf{f}_s(\mathbf{x}) = G^{(s)}\big[\phi(\mathbf{x}^\top\mathbf{W}_{\mathrm{up}}^{(s)})\mathbf{W}_{\mathrm{down}}^{(s)}\big]^\top, \tag{1}$$

and for a gated-MLP expert,

$$\mathbf{f}_s(\mathbf{x}) = G^{(s)}\big[\big(\phi(\mathbf{x}^\top\mathbf{W}_{\mathrm{up}}^{(s)}) \odot (\mathbf{x}^\top\mathbf{W}_{\mathrm{gate}}^{(s)})\big)\mathbf{W}_{\mathrm{down}}^{(s)}\big]^\top. \tag{2}$$

Here, $\phi(\cdot)$ denotes an element-wise activation function, $\odot$ denotes the Hadamard product, and $G^{(s)}$ is the routing weight for expert s, obtained by normalizing routing scores.

2.2 Noise models of analog in-memory computing of MVMs

In analog in-memory computing (AIMC) paradigm, the matrix-vector multiplication (MVM) of a linear layer is performed in non-volatile memory (NVM) devices (e.g., PCM (Joshi et al., 2020), ReRAM (Wan et al., 2022)), where weights are programmed as NVM cell conductances. As shown in Figure 2, in an analog crossbar array of NVM, the digital inputs are converted to analog signals via a digital-to-analog converter (DAC), and the resulting column currents, representing the analog output, are converted back to digital by an analog-to-digital converter (ADC) for subsequent digital operation.

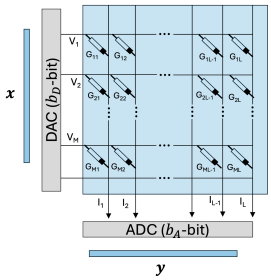


Figure 2: Schematic of an analog crossbar array

The MVM operation in AIMC accelerators is not exact. The nonidealities arise from weight programming noise and other system-level nonidealities, as well as quantization noise in DACs and ADCs. Following the standard AIMC crossbar model given in Rasch et al. (2023), the output of the i-th column of the tile, $\mathbf{y}_i$ can be expressed as,

$$\mathbf{y}_i := \mathrm{ADC}_{\beta_{\text{out}}}^{b_A}\left(\mathrm{DAC}_{\beta_{\text{in}}}^{b_D}(\mathbf{x})^\top \hat{\mathbf{W}}_{:,i} + \mathcal{N}(0,\sigma_s^2)\right) \tag{3}$$

Here, $\mathrm{DAC}_{\beta_{\text{in}}}^{b_D}(\cdot)$ clips the input to $[-\beta_{\text{in}},\beta_{\text{in}}]$ and quantize to b_D bits. $\mathrm{ADC}_{\beta_{\text{out}}}^{b_A}(\cdot)$ applies the same function with $\beta_{\text{out}} = \lambda\beta_{\text{in}}\max(|\mathbf{W}_{:,i}|)$. $\hat{\mathbf{W}}_{:,i}$ is the noisy version of the original weights $\mathbf{W}_{:,i}$, programmed at the column i of the crossbar array. The noise is referred to as the *weight programming noise*, arising from the imprecise weight programming in NVM devices. Following Rasch et al. (2023); Le Gallo et al. (2023); Büchel et al. (2025a), the noise can be mathematically modeled by,

$$\hat{W}_{j,i} = W_{j,i} + \mathcal{N}(0,\sigma_{j,i}^2), \sigma_{j,i} = c_0 W_{max} + \sum_{u=1}^{3} c_u \frac{|W_{j,i}|^u}{W_{max}^{u-1}} \tag{4}$$

Here, $W_{j,i}$ is the j-th element of $\mathbf{W}_{:,i}$ and $W_{max} := \max(|\mathbf{W}_{:,i}|)$. c_0, c_1, c_2, and c_3 are the weight programming noise magnitudes. The model shows that the noisy weight $\hat{W}_{j,i}$ is the sum of the original weight $W_{j,i}$ and a zero-mean Gaussian noise that depends on $W_{j,i}$ and W_{max}.

σ_s in (3) represents the *system noise*, which arises from the imprecision of the peripheral circuitry, and can be expressed as $\sigma_s = c_s W_{max}$. Here, the system noise magnitude, c_s, is amplified by W_{max} so that the analog output is de-normalized to offset the weight normalization while mapping weights within the programmable conductance range of the NVM tiles (Rasch et al., 2023; Hou et al., 2025b).

The bit-widths b_D and b_A control the DAC and ADC noises, respectively. Higher values of b_D and b_A imply lower DAC and ADC noise. c_0, c_1, c_2, c_3, and c_s collectively control the overall analog device noise level. Higher values of these coefficients imply higher analog device noise.

3 Analog–Digital heterogeneous mapping of MVMs in MoE models

Before describing our overall method, we first define the metric for selecting digital experts.

Maximum neuron norm score – a metric to select experts. For a projection matrix $\mathbf{W} \in \mathbb{R}^{d\times m}$ with m neurons, we define the maximum neuron norm of this matrix $\mathbf{W}$ as

$$\mathrm{MaxNNorm}(\mathbf{W}) := \max_{i\in[m]} \|\mathbf{W}_{:,i}\|_2, \tag{5}$$

where $\mathbf{W}_{:,i} \in \mathbb{R}^d$ denotes the weight vector of neuron i.

For expert s, we define the *maximum neuron norm score* of a gated-MLP expert s as the product of the maximum neuron norms of three matrices, $\mathbf{W}_{\text{up}}^{(s)}$, $\mathbf{W}_{\text{down}}^{(s)}$, and $\mathbf{W}_{\text{gate}}^{(s)}$,

$$\mathrm{MaxNNScore}^{(s)} := \prod_{*\in\{\text{up,down,gate}\}} \mathrm{MaxNNorm}(\mathbf{W}_*^{(s)}). \tag{6}$$

Heterogeneous mapping of MoE

Step 1. Map all dense MVMs in digital.
Step 2. Rank experts in each MoE block by $\mathrm{MaxNNScore}$ of (6) from high to low.
Step 3. Map MVMs of the top Γ fraction of the experts to digital, rest in analog.

Figure 3: The proposed heterogeneous mapping method of MVMs in MoE models.

where the gate term in (6) is omitted when defining the $\mathrm{MaxNNScore}$ for a standard MLP expert s.

Our **heterogeneous mapping of MVMs of MoEs** is summarized in Figure 3. We map the MVMs of all densely activated modules (e.g., query, key, value, and output projection matrices, the shared experts (if any), and the LM head) and the top Γ fraction of experts (ranked by the $\mathrm{MaxNNScore}$) in each MoE block to digital, and the rest of the experts' MVMs to the analog NVM tiles.

Rationale behind mapping dense MVMs in digital. The number of parameters in the dense MVMs (e.g., query, key, value, and output projection matrices, any shared expert, and the LM head in MoE-LLM) is insignificant. Typically, 5-9% parameters reside in these modules in total. On the other hand, the processing of all input tokens by these modules indicates high sensitivity to analog noise. Mapping these modules in analog suggests a large drop in model performance for a minimal gain in efficiency. We therefore propose to map them in digital.

Rationale for using the $\mathrm{MaxNNScore}$ as the selection metric. As discussed in Rasch et al. (2023); Hou et al. (2025b); Büchel et al. (2025a), weight programming noise and system noise dominate other sources of analog noise (e.g., ADC/DAC quantization) and constitute the primary sources of non-idealities in analog devices. Moreover, as shown in Section 2.2, both types of noise scale with the maximum weight magnitude ($W_{\max}$). Thus, intuitively, the $\mathrm{MaxNNScore}$ can capture the worst-case amplification of such noise within an expert in an MoE block.

Rationale for theoretical generalization analysis. As discussed above, $\mathrm{MaxNNScore}$ intuitively captures the experts' relative noise level. However, the complete AIMC model in (3) infers that the analog noise sensitivity is not solely determined by the absolute noise magnitude. Rather, it depends on the experts' signal-to-noise ratio (SNR), and their overall effect on model accuracy. In Section 4, we present a theoretical generalization analysis to confirm the variability of $\mathrm{MaxNNScore}$ across experts, to quantify their SNRs, and their overall effect on model accuracy for a rigorous justification.

4 Theoretical generalization guarantee of expert-level heterogeneous mapping

4.1 Key informal theoretical insights

Before formally presenting our theoretical setup and results, we first summarize our key theoretical findings. We consider an MoE block for a binary sequence classification task, a simplified yet representative setup, commonly used for theoretical generalization analysis (Chowdhury et al., 2024; 2026). The sequence labels are determined by a set of task-relevant tokens, among which one appears more frequently than the other in the input sequences. Specifically, the less-frequent task-relevant token appears with frequency α, while the more-frequent token appears with frequency $(1-\alpha)$, where $\alpha < \frac{1}{4}$. We theoretically show the following:

1. **Experts specialized in more-frequent tokens suffer larger analog noise**. We prove that experts specializing in more-frequent task-relevant tokens tend to develop neurons with larger ℓ_2 norms. Since analog noise scales with weight magnitude, these experts experience larger effective analog perturbations and are therefore more sensitive to hardware non-idealities.

2. **Selective digital mapping significantly increases tolerable analog noise**. We characterize the tolerable analog noise level required to maintain desirable generalization performance. In particular, we show that selectively mapping a fraction of experts with the largest $\mathrm{MaxNNScore}$ to digital units allows the remaining experts executed in analog to tolerate $\Omega\left(\frac{1-\alpha}{\alpha}\right)$ times larger noise magnitude compared to uniform analog execution.

4.2 ANALYTICAL SETUP

We adopted a similar setup to Chowdhury et al. (2026), as described below.

Analytical model and the learning task. We analyze a simplified MoE model of a single MoE block of standard MLP, trained on a binary sequence classification task. Given a sequence of tokens $\mathbf{X} \in \mathbb{R}^{d\times n}$, where each column $j \in [n]$ represents a token $\mathbf{x}^{(j)} \in \mathbb{R}^d$. The sequence label is denoted by $y \in \{+1, -1\}$. The model output is given by,

$$f(\mathbf{X}) := \frac{1}{d} \sum_{j\in[n]} \mathbf{1}^\top \mathbf{x}_{\text{out}}^{(j)} \tag{7}$$

where $\mathbf{x}_{\text{out}}^{(j)}$ is the j-th output token of the MoE block and $\mathbf{1} \in \mathbb{R}^d$ is the all-ones vector. Each expert's down-projection matrix is fixed throughout the training and given by $\mathbf{W}_{\text{down}}^{(s)} = a^{(s)}\mathbf{1}^{m\times d}$, where $\mathbf{1}^{m\times d}$ is an all-ones matrix and $a^{(s)} \in \{+1, -1\}$. We consider expert-choice routing, where for each expert s, the tokens corresponding to the top-l entries of $[\mathbf{X}^\top\mathbf{\Sigma}]_{:,s}$ are routed to that expert. The routing weight of token j, routed to expert s, is evaluated as,

$$G_j^{(s)} = \exp([\mathbf{X}^\top\mathbf{\Sigma}]_{j,s}) / \sum_{i\in J_s(\mathbf{X})} \exp([\mathbf{X}^\top\mathbf{\Sigma}]_{i,s}) \tag{8}$$

where, $J_s(\mathbf{X})$ is the index set of tokens routed to expert s.

Sequence sampling model. Let $(\mathbf{X}, y) \sim \mathcal{D}$, where tokens of $\mathbf{X}$ drawn from an orthonormal set $\mathcal{P} \subset \mathbb{R}^d$. Two vectors $\mathbf{o}_1, \mathbf{o}_2 \in \mathcal{P}$ define the task-relevant set $\mathcal{P}_r = \{\pm\mathbf{o}_1, \pm\mathbf{o}_2\}$; all others are task-irrelevant. Each sequence contains exactly one task-relevant token, determining the label: sequences containing $\pm\mathbf{o}_1$ are labeled $+1$, and those containing $\pm\mathbf{o}_2$ are labeled -1. Remaining tokens are drawn independently from $\mathcal{P}\backslash\{\mathbf{o}_1, \mathbf{o}_2\}$. With probability $\alpha \in (0, 1/4)$, the task-relevant token is $\mathbf{o}_1$ or $\mathbf{o}_2$.

Training Setup. We analyze the case, where the network given in (7) is trained for T steps via SGD with batch size B to minimize the Hinge loss, i.e., $\max(1 - yf^{(t)}(\mathbf{X}), 0)$.

Expert specialization. To quantify expert specialization, we define, for each expert s and task-relevant token $\mathbf{v} \in \mathcal{P}_r$, the probability

$$p_{\mathbf{v}}^{(s)} := \mathbb{P}\Big[G_j^{(s,t)} \geq 1/l \;\Big|\; x^{(j)} = \mathbf{v} \text{ for some } j \in [n]\Big] \tag{9}$$

over the data distribution $\mathcal{D}$. A value of $p_{\mathbf{v}}^{(s,t)} = 1$ indicates that the task-relevant token $\mathbf{v}$ is routed to expert s in every sequence containing $\mathbf{v}$, i.e., expert s is fully specialized on $\mathbf{v}$.

4.3 THEORETICAL GENERALIZATION GUARANTEES

Lemma 4.1. *Suppose, the model in (7) is trained for $T = \Theta(l^2\sqrt{\log l}/\alpha)$ steps. For any $\mathbf{v} \in \{\mathbf{o}_1, \mathbf{o}_2\}$ and any expert s, s', such that $p_{\mathbf{v}}^{(s,T)} = 1$ and $p_{-\mathbf{v}}^{(s',T)} = 1$, we have*

$$\text{MaxNNScore}^{(s)} < \text{MaxNNScore}^{(s')}$$

Lemma 4.1 shows that the expert specialized on more-frequent task-relevant token (e.g., the expert s' with $p_{-\mathbf{o}_1}^{(s')} = 1$) maintains larger *maximum neuron norm score* compared to the expert specialized on less-frequent task-relevant token (e.g., the expert s with $p_{\mathbf{o}_1}^{(s')} = 1$). At the same time, because both weight-programming noise in (4) and system noise increase with neuron magnitude, as shown in Section 2.2, experts specialized in more-frequent task-relevant tokens, while being more important for task performance, are also more susceptible to analog noise. Thus, selectively mapping these experts to digital units can significantly improve the robustness of the resulting analog–digital implementation. To formalize this intuition, we compare the generalization of two execution settings: (i) Uniform analog execution, where the MVMs of all experts are mapped to analog hardware, with the resulting model denoted by $f_A(\mathbf{X})$; (ii) Heterogeneous analog–digital mapping, where experts are selectively mapped according to our method in Figure 3, with the resulting model denoted by $f_H(\mathbf{X})$.

Theorem 4.2. *Suppose the model given in (7) is trained for $T = \Theta(l^2\sqrt{\log l}/\alpha)$ steps. Let γ denote the fraction of the experts $s \in [k]$ in the trained model that satisfies $p_{-\mathbf{v}}^{(s,T)} = 1$ for $v \in \{\mathbf{o}_1, \mathbf{o}_2\}$.*

Then, if the DAC and ADC bit-widths satisfy

$$b_D \geq \log_2(1 + \Omega(\sqrt{d})), b_A \geq \log_2(1 + \Omega(l)) \tag{10}$$

and the maximum analog noise magnitude, $\max(c_0, c_1, c_2, c_3, c_s)$ satisfies

$$\max(c_0, c_1, c_2, c_3, c_s) \leq c_A := O(\frac{\alpha}{1-\alpha} \times \frac{1}{l\sqrt{\log(kmld^2)}}), \tag{11}$$

then with high probability, the all-experts-in-analog mapped MoE model, denoted by f_A, has guaranteed generalization i.e., $\mathbb{P}[\forall(\mathbf{X}, y) \sim \mathcal{D} : y f_A^{(T)}(\mathbf{X}) > 0] = 1$.

Moreover, if $\Gamma \geq \gamma$ fraction of the experts with top MaxNNScore *are mapped in digital while others in analog, b_D, b_A satisfy (10), and the maximum analog noise magnitude, $\max(c_0, c_1, c_2, c_3, c_s)$ satisfies*

$$\max(c_0, c_1, c_2, c_3, c_s) \leq c_H := \frac{1-\alpha}{\alpha} c_A, \tag{12}$$

then with high probability, the heterogeneous-expert mapped MoE model, denoted by f_H, has guaranteed generalization. i.e., $\mathbb{P}[\forall(\mathbf{X}, y) \sim \mathcal{D} : y f_H^{(T)}(\mathbf{X}) > 0] = 1$.

Theorem 4.2 characterizes the maximum tolerable analog noise for guaranteed generalization under the all-experts-in-analog and the proposed heterogeneous mapping, given in (11) and (12), respectively. Mapping γ fraction of experts to digital, i.e., those specialized on more frequent task-relevant tokens, allows the remaining experts to tolerate a larger noise magnitude, by a factor of $\Omega\big(\frac{1-\alpha}{\alpha}\big)$.

5 Experiments

5.1 Experimental setup

Evaluated models. We evaluate the proposed heterogeneous mapping on three pre-trained MoE LLMs: DeepSeekMoE (Dai et al., 2024), Qwen-1.5 MoE (Team, 2024), and OLMoE (Muennighoff et al., 2025), with 16B, 14B, and 7B parameters, respectively. DeepSeekMoE and Qwen-1.5 MoE have 28 and 24 Transformer layers, respectively; DeepSeekMoE uses a dense FFN in the first layer and MoE FFNs thereafter. Each MoE block in both models includes a dense shared expert that processes all tokens. OLMoE has 16 Transformer layers, all with MoE FFNs. DeepSeekMoE, Qwen-1.5 MoE, and OLMoE have 64, 60, and 64 experts per MoE layer, respectively.

Evaluated task. We test the performance of these models on 7 benchmark LLM tasks: PIQA (Bisk et al., 2020), ARC-Challenge and ARC-Easy (Clark et al., 2018), BoolQ (Clark et al., 2019), WinoGrande (Sakaguchi et al., 2021), MathQA (Amini et al., 2019), and MMLU (Hendrycks et al., 2021). These tasks assess diverse LLM capabilities, including professional knowledge, commonsense and mathematical reasoning, trivia, and natural language inference.

Analog simulation. We simulate the analog inference using IBM's AIHWKIT-Lightning[1] (Büchel et al., 2024), with NVM tile size of 512. The DAC and ADC bit-widths (i.e., b_D and b_A) are set to 8, balancing the accuracy-latency trade-off (Büchel et al., 2025a; Rasch et al., 2023; Hou et al., 2025b).

Calibrating DAC and ADC. We calibrate DAC and ADC hyperparameters using Wikitext2 (Merity et al., 2016) and C4 (Raffel et al., 2020). Specifically, we set $\beta_{\text{in}} = \kappa \cdot \text{std}(\mathbf{x})$, where $\text{std}(\mathbf{x})$ is the exponential moving average of the input standard deviation over C4 test samples. We calibrate λ and κ using Wikitext2 test samples.

The weight programming and system noise. The weight programming noise model (i.e., the values of c_0, c_1, c_2, and c_3) is obtained by fitting data collected from IBM's Hermes project chip Le Gallo et al. (2023), which is a state-of-the-art 64-core PCM-based AIMC chip. Following (Rasch et al., 2023; Hou et al., 2025b), the nominal value of the system noise magnitude (i.e., c_s) is set to 0.04.

[1] Simulating the analog inference for deep neural network evaluations is a standard practice in AIMC literature (Rasch et al., 2023; Hou et al., 2025b; Büchel et al., 2025a; Fagbohungbe et al., 2025; Hou et al., 2025a; Wu et al., 2026), allowing rapid evaluation of AIMC-based neural network training and inference algorithms.

Table 1: Task-wise accuracy (%) of different methods on the 7 benchmark LLM tasks.

Model	Method	Param. in dig. (%)	Exp. in dig. (%)	PIQA	BoolQ	ARC-c	ARC-e	Wino.	MMLU	MathQA	Avg.
Deep Seek MoE (16B)	FP-16	-	-	80.36	72.78	47.61	72.85	70.17	37.67	31.42	58.98
	All analog	0	0	50.25 ± 0.34	47.57 ± 0.60	25.27 ± 0.42	25.85 ± 0.33	50.81 ± 0.69	24.54 ± 0.21	19.28 ± 0.30	34.80 ± 0.14
	Dense in dig.	8.72	0	68.77 ± 0.48	62.97 ± 0.55	37.07 ± 0.56	62.70 ± 0.51	59.60 ± 0.58	25.35 ± 0.20	25.84 ± 0.29	48.91 ± 0.37
	MaxNN Score (**Ours**)	11.57	3.125	77.66 ± 0.15	**70.73 ± 0.25**	**44.68 ± 0.25**	70.45 ± 0.17	**67.76 ± 0.25**	35.38 ± 0.10	29.67 ± 0.19	**56.62 ± 0.08**
		14.43	6.25	**77.86 ± 0.16**	**70.87 ± 0.24**	44.83 ± 0.44	70.30 ± 0.17	**67.35 ± 0.20**	35.41 ± 0.19	29.35 ± 0.18	56.57 ± 0.09
		20.13	12.50	77.81 ± 0.20	**71.84 ± 0.10**	**45.12 ± 0.31**	**70.97 ± 0.08**	67.80 ± 0.12	35.66 ± 0.20	29.61 ± 0.23	56.98 ± 0.07
	ENS	11.57	3.125	**77.75 ± 0.18**	70.44 ± 0.40	44.50 ± 0.32	**70.49 ± 0.15**	67.18 ± 0.45	**35.66 ± 0.12**	**29.68 ± 0.29**	56.53 ± 0.10
		14.43	6.25	77.39 ± 0.13	70.53 ± 0.38	**45.17 ± 0.34**	**70.87 ± 0.25**	66.94 ± 0.19	**35.99 ± 0.10**	**29.74 ± 0.15**	**56.67 ± 0.14**
		20.13	12.50	**78.21 ± 0.24**	71.61 ± 0.19	44.98 ± 0.22	70.84 ± 0.23	**68.24 ± 0.21**	**35.92 ± 0.20**	**30.22 ± 0.22**	**57.15 ± 0.08**
	Higher router norm	11.57	3.125	68.89 ± 0.42	62.55 ± 0.69	36.63 ± 0.44	61.66 ± 0.49	59.76 ± 0.44	25.33 ± 0.30	25.63 ± 0.35	48.64 ± 0.34
		14.43	6.25	76.24 ± 0.27	67.74 ± 0.29	42.55 ± 0.30	68.14 ± 0.25	64.11 ± 0.47	33.71 ± 0.16	28.80 ± 0.21	54.47 ± 0.13
		20.13	12.50	76.32 ± 0.13	67.81 ± 0.31	43.03 ± 0.23	68.23 ± 0.19	65.34 ± 0.45	33.80 ± 0.13	28.86 ± 0.13	54.77 ± 0.07
Qwen -1.5 MoE (14B)	FP-16	-	-	80.52	79.79	44.37	69.19	68.9	61.24	35.34	62.77
	All analog	0	0	51.28 ± 0.77	46.37 ± 0.57	26.23 ± 0.51	26.46 ± 0.25	50.11 ± 0.53	25.18 ± 0.14	19.69 ± 0.24	35.05 ± 0.25
	Dense in dig.	12.98	0	58.54 ± 0.64	55.74 ± 0.72	27.48 ± 0.29	42.54 ± 0.76	52.73 ± 0.33	30.21 ± 0.76	22.75 ± 0.14	41.43 ± 0.40
	MaxNN Score (**Ours**)	15.88	3.33	**78.28 ± 0.25**	**77.00 ± 0.25**	41.66 ± 0.42	**65.22 ± 0.24**	**66.63 ± 0.37**	**58.21 ± 0.11**	30.81 ± 0.28	**59.69 ± 0.08**
		18.78	6.67	**78.20 ± 0.14**	76.78 ± 0.23	41.97 ± 0.28	**65.58 ± 0.29**	**66.24 ± 0.36**	**58.69 ± 0.10**	31.41 ± 0.24	**59.84 ± 0.08**
		25.58	13.33	**79.11 ± 0.20**	77.63 ± 0.07	**42.40 ± 0.29**	**66.27 ± 0.13**	**66.98 ± 0.20**	**59.04 ± 0.09**	**33.06 ± 0.27**	**60.64 ± 0.07**
	ENS	15.88	3.33	77.78 ± 0.20	76.32 ± 0.19	**41.89 ± 0.22**	64.20 ± 0.25	65.20 ± 0.36	58.07 ± 0.14	**31.14 ± 0.19**	59.23 ± 0.09
		18.78	6.67	78.09 ± 0.11	**76.89 ± 0.24**	**42.46 ± 0.49**	64.75 ± 0.27	65.53 ± 0.48	58.45 ± 0.10	**31.75 ± 0.39**	59.71 ± 0.12
		25.58	13.33	78.32 ± 0.22	**77.68 ± 0.09**	42.04 ± 0.33	64.96 ± 0.27	66.67 ± 0.44	58.84 ± 0.09	32.65 ± 0.16	60.17 ± 0.14
	Act. weight	15.98	3.33	62.06 ± 0.60	59.28 ± 0.55	30.60 ± 0.57	48.32 ± 0.88	54.14 ± 0.48	37.29 ± 0.97	23.94 ± 0.19	45.10 ± 0.47
		18.78	6.67	72.48 ± 0.33	69.27 ± 0.29	36.55 ± 0.46	55.86 ± 0.32	58.35 ± 0.27	51.50 ± 0.25	26.27 ± 0.38	52.90 ± 0.13
		25.58	13.33	73.49 ± 0.39	70.35 ± 0.30	37.75 ± 0.49	58.19 ± 0.27	59.94 ± 0.45	52.04 ± 0.14	26.65 ± 0.37	54.06 ± 0.24
OL MoE (7B)	FP-16	-	-	79.71	69.85	49.66	76.94	68.90	53.65	28.44	61.03
	All analog	0	0	51.03 ± 0.28	45.36 ± 0.50	24.95 ± 0.34	29.02 ± 0.20	50.18 ± 0.64	24.52 ± 0.21	19.77 ± 0.15	34.98 ± 0.07
	Dense in dig.	5.37	0	51.72 ± 0.27	47.11 ± 0.29	25.82 ± 0.32	32.00 ± 0.17	50.99 ± 0.59	24.28 ± 0.12	20.28 ± 0.25	36.03 ± 0.10
	MaxNN Score (**Ours**)	11.28	6.25	**65.65 ± 0.30**	**58.06 ± 0.36**	**35.38 ± 0.36**	**53.69 ± 0.33**	**58.29 ± 0.33**	**35.45 ± 0.27**	**23.19 ± 0.41**	**47.10 ± 0.17**
		17.20	12.50	**66.97 ± 0.34**	**59.75 ± 0.31**	**36.76 ± 0.20**	**57.54 ± 0.39**	58.76 ± 0.24	**39.07 ± 0.29**	**23.56 ± 0.45**	**48.92 ± 0.13**
		29.03	25.00	**70.39 ± 0.30**	**63.66 ± 0.23**	**39.40 ± 0.38**	**63.36 ± 0.30**	60.38 ± 0.59	**45.40 ± 0.18**	**24.97 ± 0.22**	**52.51 ± 0.16**
	ENS	11.28	6.25	64.06 ± 0.36	56.10 ± 0.27	33.86 ± 0.38	48.42 ± 0.43	57.89 ± 0.63	29.49 ± 0.37	22.09 ± 0.31	44.56 ± 0.18
		17.20	12.50	66.97 ± 0.52	57.83 ± 0.24	35.87 ± 0.46	54.67 ± 0.43	**59.39 ± 0.48**	32.73 ± 0.16	22.58 ± 0.29	47.15 ± 0.13
		29.03	25.00	68.82 ± 0.32	58.26 ± 0.23	37.37 ± 0.51	56.89 ± 0.37	**60.89 ± 0.54**	34.58 ± 0.33	23.00 ± 0.21	48.55 ± 0.12
	Lower router norm	11.28	6.25	60.48 ± 0.24	52.42 ± 0.24	31.55 ± 0.32	44.20 ± 0.27	55.51 ± 0.51	27.61 ± 0.18	21.81 ± 0.19	41.94 ± 0.12
		17.20	12.50	60.71 ± 0.31	53.10 ± 0.42	31.03 ± 0.32	45.61 ± 0.38	55.49 ± 0.34	28.86 ± 0.25	22.41 ± 0.19	42.46 ± 0.07
		29.03	25.00	62.83 ± 0.35	57.60 ± 0.39	33.48 ± 0.53	51.24 ± 0.32	55.85 ± 0.30	35.53 ± 0.21	23.25 ± 0.21	45.69 ± 0.12

5.2 BASELINES

Analog-aware baseline. To the best of our knowledge, this paper is the first to study expert-level analog-digital mapping for AIMC accelerators. Consequently, no prior analog-noise-aware expert selection baseline exists in the literature. For a meaningful comparison, we design an analog-noise-aware baseline, the *expert noise sensitivity* (ENS), inspired by analog hardware-aware retraining. In analog hardware-aware retraining, additive Gaussian noise is injected into the weights during the forward pass of training (Büchel et al., 2025a). Specifically, the noisy weight corresponding to the j-th weight of the i-th neuron is given by, $\hat{W}_{j,i} = W_{j,i} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \gamma^2_{\text{weight}} W^2_{max})$. Motivated by this formulation, we define the ENS of expert s as,

$$\text{ENS}(s) := \frac{1}{\mathcal{T}_s} \sum_{\mathbf{x} \in \mathcal{T}_s} \mathbb{E}_\epsilon \left[\left|\left| \mathbf{f_s}(\mathbf{x}; \mathbf{W}) - \mathbf{f_s}(\mathbf{x}; \hat{\mathbf{W}}) \right|\right|_2 \right] \tag{13}$$

where $\mathbf{f_s}(\cdot)$ denotes the output of expert s, $\hat{\mathbf{W}}$ is obtained by injecting the analog-aware noise described above, and $\mathcal{T}_s$ denotes the set of tokens routed to expert s over a calibration dataset. We evaluate ENS on the Wikitext2 test set using $\gamma_{\text{weight}} = 0.05$ (as selected in Büchel et al. (2025a) for analog-noise-aware training). Experts with higher ENS values are mapped to digital.

MoE compression-based baselines. We further compare our method with expert importance metrics used for MoE pruning (Koishekenov et al., 2023; Chowdhury et al., 2024) and mixed-precision quantization (Li et al., 2024b; Huang et al., 2025): *Activation Frequency and Weight.* Experts are ranked in decreasing order of activation frequency or average routing weight over a calibration dataset,

Table 2: Accuracy, energy efficiency, and latency comparison on OLMoE for batch size of 32.

Method	Configuration	Avg. Acc. (%)	Energy Eff. (Tokens/J)	Latency (ms)
Full-precision*	GPU (INT8)	61.03	20	0.122
Analog-digital Mapping (Ours)	25% digital	**52.51 ± 0.16**	**72**	**0.048**
	12.5% digital	**48.92 ± 0.13**	**126**	**0.032**
	6.25% digital	**47.10 ± 0.10**	**207**	**0.023**
	0% digital	36.03 ± 0.10	564	0.018
GPTQ*	4-bit (INT4)	48.07	40	0.061
	2-bit (INT2)	34.97	80	0.031
Expert-pruning*	25% pruned	49.55	27	0.092
	37.5% pruned	40.95	32	0.076
	50% pruned	37.25	40	0.061

* Energy efficiency and latency for full-precision, GPTQ, and expert pruning are upper-bound estimates.

with higher-ranked experts mapped to digital units. *Lower and Higher Router Norm.* Experts are ranked in decreasing order of router norm. Prior work favors lower router norm (i.e., lower-rank) experts for higher quantization bit-width (Chowdhury et al., 2026) and higher router norm (i.e., higher rank) experts to be unpruned (Chowdhury et al., 2024). We evaluate both by mapping either lower- or higher-router-norm experts to digital units.

5.3 Results on large-scale MoE models on benchmark LLM tasks

Heterogeneous expert mapping with different expert selection metrics. Table 1 reports task-wise accuracy for our method along with the top two of the other digital expert selection baselines considered in this paper; complete results are provided in Table 3 of Appendix A. Mapping all MVMs to analog ("All Analog") causes over 24% average accuracy loss across all models, while mapping dense MVMs to digital partially recovers performance (recovers more than half of the performance drop for DeepSeekMoE). With our method, mapping less than 3.5% of sparse experts to digital recovers accuracy to within 3% of FP-16 performance for shared-expert-based models such as DeepSeekMoE and Qwen-1.5 MoE. OLMoE does not contain shared experts, and mapping only the dense MVMs to digital yields limited improvement. However, accuracy improves substantially for our method, consistently outperforming all competing baselines by large margins. Note that our method significantly outperforms MoE compression-based metrics and performs within the margin of error of the analog-aware ENS baseline on DeepSeekMoE, while consistently outperforming it on Qwen-1.5 MoE and OLMoE. Since ENS directly measures expert sensitivity to analog noise, these results demonstrate the effectiveness of our metric in identifying experts critical for digital mapping.

Gains in energy efficiency and latency. We estimate the energy (Tokens/J) and latency (ms) to generate a batch of tokens using the proposed mapping for OLMoE in Table 2. Evaluation details are provided in Appendix C. As shown in the table, compared to mapping all experts in analog, the energy efficiency decreases with the percentage of experts mapped in digital. However, the loss in energy efficiency is compensated for by the large gain in model performance. More importantly, the energy efficiency of the analog-digital mapping is still significantly larger (more than 2 orders of magnitude) than that of the digital inference in GPU (INT8). We further compare the energy efficiency and latency of the proposed mapping to popular digital MoE compression methods including low-bit quantization (GPTQ) and expert-pruning. To achieve similar energy efficiency and latency ratings as ours, the digital compression methods require an incomparably large drop in accuracy.

6 Conclusion

We develop a heterogeneous mapping framework for analog MoE models, where noise-sensitive experts are executed on digital units while the remaining experts are executed on AIMC. We show that mapping dense modules and experts with large neuron norms to digital units substantially improves robustness against AIMC non-idealities. We provide discussions on the independence of our analysis of underlying analog hardware, and the broader practical impact of our proposed method in Appendix H and Appendix I, respectively. Future work includes system-level designs for dynamically scheduling expert computations across analog and digital accelerators under compute, latency, and energy constraints.

AI USE STATEMENT

In this work, we **have NOT used** generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, and implement methods. Assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, and interpret results **are NOT applicable** to this work. We occasionally used generative AI tools to aid or polish writing. We have reviewed all AI-assisted polishing of writings before including them in the paper. We take responsibility for the final content of this work, including text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the complete setup for our theoretical analysis in Section 4.2. We include additional details related to our analysis in Appendix J. We provide the proof of Lemma 4.1 in Appendix K, and the proof of Theorem 4.2 in Appendix L. We provide the details of our experimental setup, including model architecture, parameter size, evaluated tasks, analog simulation and calibration methods, and noise scaling magnitude in Section 5.1. We further provide the calibrated hyperparameters in Appendix M. We include the code of our experiments as supplementary materials for reproducibility.

REFERENCES

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1245. URL https://aclanthology.org/N19-1245.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns over-parameterized networks that provably generalize on linearly separable data. In *International Conference on Learning Representations*, 2018.

Dake Bu, Wei Huang, Taiji Suzuki, Ji Cheng, Qingfu Zhang, Zhiqiang Xu, and Hau-San Wong. Provably neural active learning succeeds via prioritizing perplexing samples. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 4642–4695, 2024.

Julian Büchel, Iason Chalas, Giovanni Acampa, An Chen, Omobayode Fagbohungbe, Hsinyu Tsai, Kaoutar El Maghraoui, Manuel Le Gallo, Abbas Rahimi, and Abu Sebastian. Analog foundation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=zo4zYTR8vn.

Julian Büchel, Athanasios Vasilopoulos, William Andrew Simon, Irem Boybat, HsinYu Tsai, Geoffrey W Burr, Hernan Castro, Bill Filipiak, Manuel Le Gallo, Abbas Rahimi, et al. Efficient scaling of large language models with mixture of experts and 3d analog in-memory computing. *Nature Computational Science*, 5(1):13–26, 2025b.

Geoffrey W Burr, Robert M Shelby, Abu Sebastian, Sangbum Kim, Seyoung Kim, Severin Sidler, Kumar Virwani, Masatoshi Ishii, Pritish Narayanan, Alessandro Fumarola, et al. Neuromorphic computing using non-volatile memory. *Advances in Physics: X*, 2(1):89–124, 2017.

Julian Büchel, William Andrew Simon, Corey Lammie, Giovanni Acampa, Kaoutar El Maghraoui, Manuel Le Gallo, and Abu Sebastian. Aihwkit-lightning: A scalable hw-aware training toolkit for analog in-memory computing. In *NeurIPS 2024 Workshop Machine Learning with new Compute Paradigms*, 2024. URL https://openreview.net/forum?id=QNdxOgGmhR.

Mohammed Nowaz Rabbani Chowdhury, Shuai Zhang, Meng Wang, Sijia Liu, and Pin-Yu Chen. Patch-level routing in mixture-of-experts is provably sample-efficient for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6074–6114. PMLR, 2023.

Mohammed Nowaz Rabbani Chowdhury, Meng Wang, Kaoutar El Maghraoui, Naigang Wang, Pin-Yu Chen, and Christopher Carothers. A provably effective method for pruning experts in fine-tuned sparse mixture-of-experts. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=1oU4FKpVx5.

Mohammed Nowaz Rabbani Chowdhury, Kaoutar El Maghraoui, Hsinyu Tsai, Naigang Wang, Geoffrey W. Burr, Liu Liu, and Meng Wang. Efficient quantization of mixture-of-experts with theoretical generalization guarantees. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yiMlVBAoQi.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Damai Dai, Chengqi Deng, Chenggang Zhao, Rx Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297, 2024.

Omobayode Fagbohungbe, Corey Lammie, Malte J Rasch, Takashi Ando, Tayfun Gokmen, and Vijay Narayanan. Assessing the performance of analog training for transfer learning. In *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5. IEEE, 2025.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Jared Fernandez, Clara Na, Vashisth Tiwari, Yonatan Bisk, Sasha Luccioni, and Emma Strubell. Energy considerations of large language model inference and efficiency optimizations. In *LTI Student Research Symposium 2025*, 2025. URL https://openreview.net/forum?id=aXNty1YGe0.

Elias Frantar and Dan Alistarh. Qmoe: Sub-1-bit compression of trillion parameter models. In P. Gibbons, G. Pekhimenko, and C. De Sa (eds.), *Proceedings of Machine Learning and Systems*, volume 6, pp. 439–451, 2024.

Hidehiro Fujiwara, Haruki Mori, Wei-Chang Zhao, Kinshuk Khare, Cheng-En Lee, Xiaochen Peng, Vineet Joshi, Chao-Kai Chuang, Shu-Huan Hsu, Takeshi Hashizume, et al. 34.4 a 3nm, 32.5 tops/w, 55.0 tops/mm 2 and 3.78 mb/mm 2 fully-digital compute-in-memory macro supporting int12$\times$ int12 with a parallel-mac architecture and foundry 6t-sram bit cell. In *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, volume 67, pp. 572–574. IEEE, 2024.

Tayfun Gokmen, Malte J Rasch, and Wilfried Haensch. The marriage of training and inference for scaled deep learning analog hardware. In *2019 IEEE International Electron Devices Meeting (IEDM)*, pp. 22–3. IEEE, 2019.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Yayue Hou, Zhenyu Liu, Garrett Gagnon, Hsinyu Tsai, Kaoutar El Maghraoui, Geoffrey W Burr, and Liu Liu. Sage: Saliency-aware grouping for efficient mapping of llms on analog compute-in-memory. In *2025 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9. IEEE, 2025a.

Yayue Hou, Hsinyu Tsai, Kaoutar El Maghraoui, Tayfun Gokmen, Geoffrey W Burr, and Liu Liu. Nora: Noise-optimized rescaling of llms on analog compute-in-memory accelerators. In *2025 Design, Automation & Test in Europe Conference (DATE)*, pp. 1–7. IEEE, 2025b.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and XIAOJUAN QI. Mixture compressor for mixture-of-experts LLMs gains more. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=hheFYjOsWO.

Shubham Jain, Hsinyu Tsai, Ching-Tzu Chen, Ramachandran Muralidhar, Irem Boybat, Martin M Frank, Stanisław Woźniak, Milos Stanisavljevic, Praneet Adusumilli, Pritish 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*, 31(1): 114–127, 2022.

Nidhal Jegham, Marwan Abdelatti, Chan Young Koh, Lassad Elmoubarki, and Abdeltawab Hendawi. How hungry is ai? benchmarking energy, water, and carbon footprint of llm inference. *arXiv preprint arXiv:2505.09598*, 2025.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Vinay Joshi, Manuel Le Gallo, Simon Haefeli, Irem Boybat, Christophe Piveteau, Martino Dazzi, Bipin Rajendran, Abu Sebastian, and Evangelos Eleftheriou. Accurate deep neural network inference using computational phase-change memory. *Nature communications*, 11(1):2473, 2020.

Stefani Karp, Ezra Winston, Yuanzhi Li, and Aarti Singh. Local signal adaptivity: Provable feature learning in neural networks beyond kernels. In *Advances in Neural Information Processing Systems*, pp. 24883–24897, 2021.

Yeskendir Koishekenov, Alexandre Berard, and Vassilina Nikoulina. Memory-efficient nllb-200: Language-specific expert pruning of a massively multilingual machine translation model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3567–3585, 2023.

Manuel Le Gallo, Riduan Khaddam-Aljameh, Milos Stanisavljevic, Athanasios Vasilopoulos, Benedikt Kersting, Martino Dazzi, Geethan Karunaratne, Matthias Brändli, Abhairaj Singh, Silvia M Mueller, et al. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. *Nature Electronics*, 6(9):680–693, 2023.

Hongkang Li, Meng Wang, Songtao Lu, Xiaodong Cui, and Pin-Yu Chen. How do nonlinear transformers learn and generalize in in-context learning? In *International Conference on Machine Learning*, pp. 28734–28783. PMLR, 2024a.

Hongkang Li, Songtao Lu, Pin-Yu Chen, Xiaodong Cui, and Meng Wang. Training nonlinear transformers for chain-of-thought inference: A theoretical generalization analysis. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=n7n8McETXw.

Hongkang Li, Yihua Zhang, Shuai Zhang, Pin-Yu Chen, Sijia Liu, and Meng Wang. When is task vector provably effective for model editing? a generalization analysis of nonlinear transformers. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Pingzhi Li, Xiaolong Jin, Yu Cheng, and Tianlong Chen. Examining post-training quantization for mixture-of-experts: A benchmark. *arXiv preprint arXiv:2406.08155*, 2024b.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Evan Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoe: Open mixture-of-experts language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xXTkbTBmqq.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Malte J Rasch, Diego Moreda, Tayfun Gokmen, Manuel Le Gallo, Fabio Carta, Cindy Goldberg, Kaoutar El Maghraoui, Abu Sebastian, and Vijay Narayanan. A flexible and fast pytorch toolkit for simulating training and inference on analog crossbar arrays. In *2021 IEEE 3rd international conference on artificial intelligence circuits and systems (AICAS)*, pp. 1–4. IEEE, 2021.

Malte J Rasch, Charles Mackin, Manuel Le Gallo, An Chen, Andrea Fasoli, Frédéric Odermatt, Ning Li, SR Nandakumar, Pritish Narayanan, Hsinyu Tsai, et al. Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators. *Nature communications*, 14(1):5282, 2023.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Abu Sebastian, Manuel Le Gallo, Riduan Khaddam-Aljameh, and Evangelos Eleftheriou. Memory devices and applications for in-memory computing. *Nature nanotechnology*, 15(7):529–544, 2020.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Zhenmei Shi, Junyi Wei, and Yingyu Liang. A theoretical analysis on feature learning in neural networks: Emergence from inputs and advantage over fixed features. In *International Conference on Learning Representations*, 2022.

Katie Spoon, Hsinyu Tsai, An Chen, Malte J Rasch, Stefano Ambrogio, Charles Mackin, Andrea Fasoli, Alexander M Friz, Pritish Narayanan, Milos Stanisavljevic, et al. Toward software-equivalent accuracy on transformer-based deep neural networks with analog memory devices. *Frontiers in Computational Neuroscience*, 15:675741, 2021.

Qwen Team. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters", February 2024. URL https://qwenlm.github.io/blog/qwen-moe/.

Hsinyu Tsai, Stefano Ambrogio, Charles Mackin, Pritish Narayanan, Robert M Shelby, Kamil Rocki, An Chen, and Geoffrey W Burr. Inference of long-short term memory networks at software-equivalent accuracy using 2.5 m analog phase change memory devices. In *2019 Symposium on VLSI Technology*, pp. T82–T83. IEEE, 2019.

Weier Wan, Rajkumar Kubendran, Clemens Schaefer, Sukru Burc Eryilmaz, Wenqiang Zhang, Dabin Wu, Stephen Deiss, Priyanka Raina, He Qian, Bin Gao, et al. A compute-in-memory chip based on resistive random-access memory. *Nature*, 608(7923):504–512, 2022.

Zhaoxian Wu, Quan Xiao, Tayfun Gokmen, Omobayode Fagbohungbe, and Tianyi Chen. Analog in-memory training on general non-ideal resistive elements: The impact of response functions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=WhEPg4mUs6.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Bonan Zhang, Chia-Yu Chen, and Naveen Verma. Reshape and adapt for output quantization (RAOQ): Quantization-aware training for in-memory computing systems. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58739–58762. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhang24i.html.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Y Zhao, Andrew M. Dai, Zhifeng Chen, Quoc V Le, and James Laudon. Mixture-of-experts with expert choice routing. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022.

A COMPLETE RESULTS ON LARGE-SCALE MOE MODELS ON BENCHMARK LLM TASKS

Table 3: Task-wise accuracy (%) of different methods on the 7 benchmark LLM tasks.

Model	Method	Param. in dig. (%)	Exp. in dig. (%)	PIQA	BoolQ	ARC-c	ARC-e	Wino.	MMLU	MathQA	Avg.
Deep Seek MoE (16B)	FP-16	-	-	80.36	72.78	47.61	72.85	70.17	37.67	31.42	58.98
	All analog	0	0	50.25 ± 0.34	47.57 ± 0.60	25.27 ± 0.42	25.85 ± 0.33	50.81 ± 0.69	24.54 ± 0.21	19.28 ± 0.30	34.80 ± 0.14
	Dense in dig.	8.72	0	68.77 ± 0.48	62.97 ± 0.55	37.07 ± 0.56	62.70 ± 0.51	59.60 ± 0.58	25.35 ± 0.20	25.84 ± 0.29	48.91 ± 0.37
	MaxNN Score (**Ours**)	11.57	3.125	77.66 ± 0.15	**70.73 ± 0.25**	**44.68 ± 0.25**	70.45 ± 0.17	**67.76 ± 0.25**	35.38 ± 0.10	29.67 ± 0.19	**56.62 ± 0.08**
		14.43	6.25	**77.86 ± 0.16**	**70.87 ± 0.24**	44.83 ± 0.44	70.30 ± 0.17	**67.35 ± 0.20**	35.41 ± 0.19	29.35 ± 0.18	56.57 ± 0.09
		20.13	12.50	77.81 ± 0.20	**71.84 ± 0.10**	**45.12 ± 0.31**	**70.97 ± 0.08**	67.80 ± 0.12	35.66 ± 0.20	29.61 ± 0.23	56.98 ± 0.07
	ENS	11.57	3.125	**77.75 ± 0.18**	70.44 ± 0.40	44.50 ± 0.32	**70.49 ± 0.15**	67.18 ± 0.45	**35.66 ± 0.12**	**29.68 ± 0.29**	56.53 ± 0.10
		14.43	6.25	77.39 ± 0.13	70.53 ± 0.38	**45.17 ± 0.34**	**70.87 ± 0.25**	66.94 ± 0.19	**35.99 ± 0.10**	**29.74 ± 0.15**	**56.67 ± 0.14**
		20.13	12.50	**78.21 ± 0.24**	71.61 ± 0.19	44.98 ± 0.22	70.84 ± 0.23	**68.24 ± 0.21**	**35.92 ± 0.20**	**30.22 ± 0.22**	**57.15 ± 0.08**
	Lower router norm	11.57	3.125	70.79 ± 0.56	64.27 ± 0.80	39.02 ± 0.42	64.68 ± 0.58	62.05 ± 0.44	26.29 ± 0.22	26.33 ± 0.30	50.49 ± 0.43
		14.43	6.25	69.87 ± 0.82	63.28 ± 0.64	38.68 ± 0.51	64.58 ± 0.82	62.29 ± 0.42	25.71 ± 0.27	25.51 ± 0.19	49.99 ± 0.45
		20.13	12.50	70.40 ± 0.68	65.63 ± 0.77	38.86 ± 0.56	65.20 ± 0.57	62.12 ± 0.23	25.93 ± 0.30	26.16 ± 0.45	50.62 ± 0.47
	Higher router norm	11.57	3.125	68.89 ± 0.42	62.55 ± 0.69	36.63 ± 0.44	61.66 ± 0.49	59.76 ± 0.44	25.33 ± 0.30	25.63 ± 0.35	48.64 ± 0.34
		14.43	6.25	76.24 ± 0.27	67.74 ± 0.29	42.55 ± 0.30	68.14 ± 0.25	64.11 ± 0.47	33.71 ± 0.16	28.80 ± 0.21	54.47 ± 0.13
		20.13	12.50	76.32 ± 0.13	67.81 ± 0.31	43.03 ± 0.23	68.23 ± 0.19	65.34 ± 0.45	33.80 ± 0.13	28.86 ± 0.13	54.77 ± 0.07
	Act. freq.	11.57	3.125	68.66 ± 0.94	62.14 ± 0.73	36.85 ± 0.58	61.09 ± 0.41	59.50 ± 0.39	25.47 ± 0.34	25.29 ± 0.28	48.43 ± 0.47
		14.43	6.25	67.81 ± 0.72	62.36 ± 0.64	36.09 ± 0.37	61.55 ± 0.59	58.32 ± 0.65	25.27 ± 0.31	25.51 ± 0.27	48.13 ± 0.43
		20.13	12.50	71.03 ± 0.55	66.26 ± 0.67	38.70 ± 0.43	63.75 ± 0.39	59.88 ± 0.63	26.80 ± 0.33	27.04 ± 0.34	50.50 ± 0.40
	Act. weight	11.57	3.125	70.88 ± 0.48	64.98 ± 0.50	38.38 ± 0.38	63.49 ± 0.30	60.66 ± 0.43	26.79 ± 0.28	26.32 ± 0.27	50.22 ± 0.23
		14.43	6.25	69.85 ± 0.74	64.41 ± 0.70	37.85 ± 0.50	63.41 ± 0.54	59.63 ± 0.46	26.19 ± 0.41	26.08 ± 0.27	49.64 ± 0.44
		20.13	12.50	71.25 ± 0.78	66.51 ± 0.60	38.05 ± 0.36	64.61 ± 0.33	60.88 ± 0.54	26.49 ± 0.37	26.26 ± 0.36	50.58 ± 0.41
Qwen -1.5 MoE (14B)	FP-16	-	-	80.52	79.79	44.37	69.19	68.9	61.24	35.34	62.77
	All analog	0	0	51.28 ± 0.77	46.37 ± 0.57	26.23 ± 0.51	26.46 ± 0.25	50.11 ± 0.53	25.18 ± 0.14	19.69 ± 0.24	35.05 ± 0.25
	Dense in dig.	12.98	0	58.54 ± 0.64	55.74 ± 0.72	27.48 ± 0.29	42.54 ± 0.76	52.73 ± 0.33	30.21 ± 0.76	22.75 ± 0.14	41.43 ± 0.40
	MaxNN Score (**Ours**)	15.88	3.33	**78.28 ± 0.25**	**77.00 ± 0.25**	41.66 ± 0.42	**65.22 ± 0.24**	**66.63 ± 0.37**	**58.21 ± 0.11**	30.81 ± 0.28	**59.69 ± 0.08**
		18.78	6.67	**78.20 ± 0.14**	76.78 ± 0.23	41.97 ± 0.28	**65.58 ± 0.29**	**66.24 ± 0.36**	**58.69 ± 0.10**	31.41 ± 0.24	**59.84 ± 0.08**
		25.58	13.33	**79.11 ± 0.20**	77.63 ± 0.07	**42.40 ± 0.29**	**66.27 ± 0.13**	**66.98 ± 0.20**	**59.04 ± 0.09**	**33.06 ± 0.27**	**60.64 ± 0.07**
	ENS	15.88	3.33	77.78 ± 0.20	76.32 ± 0.19	**41.89 ± 0.22**	64.20 ± 0.25	65.20 ± 0.36	58.07 ± 0.14	**31.14 ± 0.19**	59.23 ± 0.09
		18.78	6.67	78.09 ± 0.11	**76.89 ± 0.24**	**42.46 ± 0.49**	64.75 ± 0.27	65.53 ± 0.48	58.45 ± 0.10	**31.75 ± 0.39**	59.71 ± 0.12
		25.58	13.33	78.32 ± 0.22	**77.68 ± 0.09**	42.04 ± 0.33	64.96 ± 0.27	66.67 ± 0.44	58.84 ± 0.09	32.65 ± 0.16	60.17 ± 0.14
	Lower router norm	15.88	3.33	60.08 ± 1.74	59.70 ± 1.22	30.44 ± 0.94	46.12 ± 3.06	54.19 ± 0.49	35.51 ± 2.19	24.30 ± 0.76	44.34 ± 1.39
		18.78	6.67	61.38 ± 0.70	60.18 ± 0.50	31.61 ± 0.66	49.04 ± 0.60	54.56 ± 0.37	34.76 ± 0.59	25.65 ± 0.47	45.32 ± 0.48
		25.58	13.33	73.79 ± 0.36	72.86 ± 0.38	39.00 ± 0.48	61.61 ± 0.37	60.17 ± 0.52	52.14 ± 0.54	29.82 ± 0.28	55.63 ± 0.28
	Higher router norm	15.98	3.33	61.40 ± 0.86	58.57 ± 0.49	30.30 ± 0.33	48.22 ± 0.95	54.06 ± 0.52	35.77 ± 1.31	23.38 ± 0.26	44.53 ± 0.58
		18.78	6.67	63.12 ± 0.58	59.47 ± 0.49	30.52 ± 0.39	49.92 ± 0.89	54.71 ± 1.09	36.70 ± 1.02	24.32 ± 0.63	45.54 ± 0.63
		25.58	13.33	60.49 ± 0.78	57.08 ± 0.64	29.78 ± 0.48	48.11 ± 0.89	52.48 ± 0.75	32.42 ± 1.50	22.53 ± 0.51	43.27 ± 0.62
	Act. freq.	15.88	3.33	57.67 ± 0.54	54.44 ± 0.29	27.40 ± 0.56	41.09 ± 0.37	51.71 ± 0.51	29.75 ± 0.48	22.42 ± 0.27	40.65 ± 0.30
		18.78	6.67	57.19 ± 0.42	54.82 ± 0.49	27.95 ± 0.33	41.62 ± 0.63	52.82 ± 0.31	30.22 ± 0.81	22.70 ± 0.29	41.05 ± 0.35
		25.58	13.33	57.61 ± 0.45	54.54 ± 0.08	27.45 ± 0.49	42.24 ± 0.54	52.21 ± 0.38	29.41 ± 0.49	22.71 ± 0.18	40.89 ± 0.22
	Act. weight	15.98	3.33	62.06 ± 0.60	59.28 ± 0.55	30.60 ± 0.57	48.32 ± 0.88	54.14 ± 0.48	37.29 ± 0.97	23.94 ± 0.19	45.10 ± 0.47
		18.78	6.67	72.48 ± 0.33	69.27 ± 0.29	36.55 ± 0.46	55.86 ± 0.32	58.35 ± 0.27	51.50 ± 0.25	26.27 ± 0.38	52.90 ± 0.13
		25.58	13.33	73.49 ± 0.39	70.35 ± 0.30	37.75 ± 0.49	58.19 ± 0.27	59.94 ± 0.45	52.04 ± 0.14	26.65 ± 0.37	54.06 ± 0.24
OL MoE (7B)	FP-16	-	-	79.71	69.85	49.66	76.94	68.90	53.65	28.44	61.03
	All analog	0	0	51.03 ± 0.28	45.36 ± 0.50	24.95 ± 0.34	29.02 ± 0.20	50.18 ± 0.64	24.52 ± 0.21	19.77 ± 0.15	34.98 ± 0.07
	Dense in dig.	5.37	0	51.72 ± 0.27	47.11 ± 0.29	25.82 ± 0.32	32.00 ± 0.17	50.99 ± 0.59	24.28 ± 0.12	20.28 ± 0.25	36.03 ± 0.10
	MaxNN Score (**Ours**)	11.28	6.25	**65.65 ± 0.30**	**58.06 ± 0.36**	**35.38 ± 0.36**	**53.69 ± 0.33**	**58.29 ± 0.33**	**35.45 ± 0.27**	**23.19 ± 0.41**	**47.10 ± 0.17**
		17.20	12.50	**66.97 ± 0.34**	**59.75 ± 0.31**	**36.76 ± 0.20**	**57.54 ± 0.39**	58.76 ± 0.24	**39.07 ± 0.29**	**23.56 ± 0.45**	**48.92 ± 0.13**
		29.03	25.00	**70.39 ± 0.30**	**63.66 ± 0.23**	**39.40 ± 0.38**	**63.36 ± 0.30**	60.38 ± 0.59	**45.40 ± 0.18**	**24.97 ± 0.22**	**52.51 ± 0.16**
	ENS	11.28	6.25	64.06 ± 0.36	56.10 ± 0.27	33.86 ± 0.38	48.42 ± 0.43	57.89 ± 0.63	29.49 ± 0.37	22.09 ± 0.31	44.56 ± 0.18
		17.20	12.50	66.97 ± 0.52	57.83 ± 0.24	35.87 ± 0.46	54.67 ± 0.43	**59.39 ± 0.48**	32.73 ± 0.16	22.58 ± 0.29	47.15 ± 0.13
		29.03	25.00	68.82 ± 0.32	58.26 ± 0.23	37.37 ± 0.51	56.89 ± 0.37	**60.89 ± 0.54**	34.58 ± 0.33	23.00 ± 0.21	48.55 ± 0.12
	Lower router norm	11.28	6.25	60.48 ± 0.24	52.42 ± 0.24	31.55 ± 0.32	44.20 ± 0.27	55.51 ± 0.51	27.61 ± 0.18	21.81 ± 0.19	41.94 ± 0.12
		17.20	12.50	60.71 ± 0.31	53.10 ± 0.42	31.03 ± 0.32	45.61 ± 0.38	55.49 ± 0.34	28.86 ± 0.25	22.41 ± 0.19	42.46 ± 0.07
		29.03	25.00	62.83 ± 0.35	57.60 ± 0.39	33.48 ± 0.53	51.24 ± 0.32	55.85 ± 0.30	35.53 ± 0.21	23.25 ± 0.21	45.69 ± 0.12
	Higher router norm	11.28	6.25	51.89 ± 0.34	46.31 ± 0.20	26.51 ± 0.38	32.39 ± 0.39	50.73 ± 0.56	24.30 ± 0.08	20.85 ± 0.26	36.14 ± 0.10
		17.20	12.50	52.06 ± 0.35	48.67 ± 0.38	26.66 ± 0.45	34.41 ± 0.30	52.00 ± 0.55	24.37 ± 0.09	20.43 ± 0.17	36.95 ± 0.19
		29.03	25.00	53.65 ± 0.42	48.74 ± 0.30	27.22 ± 0.62	35.33 ± 0.27	52.83 ± 0.38	24.24 ± 0.13	20.65 ± 0.22	37.53 ± 0.17
	Act. freq.	11.28	6.25	55.21 ± 0.37	47.77 ± 0.28	28.05 ± 0.41	36.87 ± 0.49	52.10 ± 0.54	24.39 ± 0.10	21.20 ± 0.23	37.95 ± 0.17
		17.20	12.50	55.04 ± 0.64	48.63 ± 0.35	27.67 ± 0.59	37.64 ± 0.25	52.47 ± 0.37	24.35 ± 0.10	21.25 ± 0.26	38.15 ± 0.12
		29.03	25.00	56.78 ± 0.30	48.72 ± 0.48	28.62 ± 0.33	39.22 ± 0.45	52.72 ± 0.39	24.18 ± 0.10	21.38 ± 0.16	38.81 ± 0.09
	Act. weight	11.28	6.25	54.95 ± 0.68	47.97 ± 0.62	27.59 ± 0.50	36.87 ± 0.33	51.57 ± 0.35	24.31 ± 0.09	21.03 ± 0.22	37.76 ± 0.25
		17.20	12.50	56.04 ± 0.15	49.14 ± 0.29	28.13 ± 0.37	37.09 ± 0.33	53.08 ± 0.72	24.50 ± 0.14	20.98 ± 0.24	38.43 ± 0.14
		29.03	25.00	59.89 ± 0.30	50.05 ± 0.33	30.29 ± 0.36	41.52 ± 0.36	55.25 ± 0.42	24.38 ± 0.10	21.94 ± 0.36	40.48 ± 0.16

B ROBUSTNESS UNDER INCREASING ANALOG NOISE LEVELS

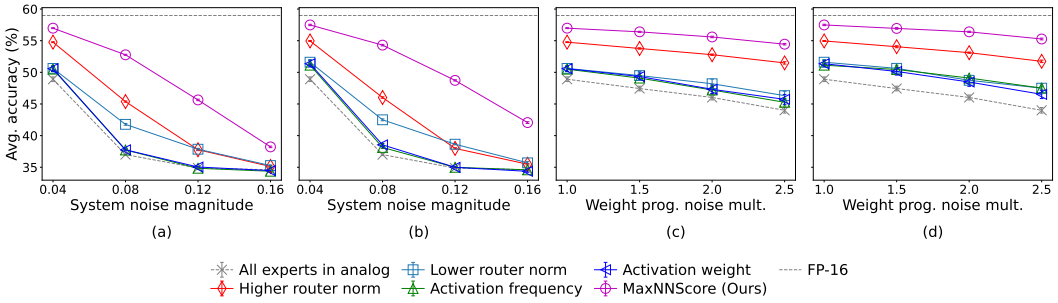


Figure 4: (a), (b): Varying system noise with 12.5% and 25% experts in digital, respectively. (c), (d): Varying programming noise with 12.5% and 25% experts in digital, respectively.

We vary the magnitudes of weight-programming noise and system noise on DeepSeekMoE to evaluate the robustness of our expert-level analog–digital mapping strategy under increasing analog noise levels. For weight-programming noise, we scale the coefficients c_0, c_1, c_2, and c_3 by a common multiplier and vary this multiplier to control the noise magnitude. For system noise, we directly vary c_s. As shown in Figure 4 , our method achieves substantially higher robustness to both types of noise compared to all baseline expert selection metrics.

C THE DETAILS FOR EVALUATING ENERGY EFFICIENCY AND LATENCY

To evaluate the energy efficiency and latency, we consider the AIMC accelerator described in Büchel et al. (2025b). The accelerator contains 256 NVM tiles. The analog MVM latency and energy consumption are provided as 100 ns and 10,000 pJ, respectively. We consider that the dense MVMs and the digital experts are mapped to the heavy compute cores of the accelerator. For a fair comparison with GPU inference on an NVIDIA A100 GPU, we estimate that the digital cores in AIMC operate at 624 TOP/s and 1.56 TOPs/Watt. We further consider that the accelerator contains an HBM of 1,555GB/s effective bandwidth, 400 watts, and 30 watts of active and passive power, respectively. We first evaluate the experts' token distribution in OLMoE for the Wikitext2 dataset. We then use the token distribution along with the latency and energy data provided above to evaluate the total energy required to generate 64 tokens for each of 32 input samples (i.e., batch size is equal to 32), which provides us with the energy efficiency in Tokens/J.

We evaluate an upper bound on the energy efficiency of the NVIDIA A100 GPU as provided in Table 2. The quantities are evaluated under 100% model FLOPs utilization (MFU), with 624 tera operations per second (624 TOP/s) at 400Watt, and with data transfer bandwidth of 1,555 GB/s for the accelerator. In that case, the upper bound of the throughput is evaluated as,

$$\text{Throughput (tokens/s)} \quad = \frac{\text{No. of tokens generated}}{\max\left(\frac{\text{Total TOPs}}{\text{MFU624TOPs/s}}, \frac{\text{Total weight transfer}}{1555\text{GB/s}}\right)} \tag{14}$$

The corresponding energy efficiency can be directly calculated using the throughput and power rating of the device.

D ENERGY EFFICIENCY AND LATENCY FOR SHARED-EXPERT-BASED MODEL

A natural question is whether mapping the shared experts in the digital offsets the latency and energy-efficiency advantages of the proposed analog-digital mapping, since the shared experts process every token. Our results in Table 4 for DeepSeekMoE show that the energy efficiency and latency gains are still preserved. Specifically, mapping only 6.25% of the sparse experts to digital in addition to the shared experts increases the average accuracy to 56.57%, while still maintaining 4× higher energy efficiency (18 Tokens/J vs. 4 Tokens/J) and nearly 4.8× lower latency (0.149 ms vs. 0.72 ms) than full-precision GPU inference. These results demonstrate that the digital overhead introduced by the shared experts is relatively small compared to the overall efficiency gains obtained from analog execution.

Table 4: Accuracy, energy efficiency, and latency comparison on DeepSeekMoE.

Method	**Configuration**	**Avg. Accuracy (%)**	**Energy Efficiency (Tokens/J)**	**Latency (ms)**
Full-precision*	GPU (INT8)	58.98	4	0.72
Shared experts in **digital** (Ours)	12.5% experts in digital	**56.98 ± 0.07**	**11**	**0.229**
	6.25% experts in digital	**56.57 ± 0.09**	**18**	**0.149**
	0% experts in digital	**48.91 ± 0.37**	**47**	**0.066**
Shared experts in **analog**	12.5% experts in digital	45.90 ± 0.20	15	0.177
	6.25% experts in digital	45.71 ± 0.12	29	0.099
	0% experts in digital	38.52 ± 0.23	564	0.025

E JUSTIFICATION FOR USING l_2-NORM OVER OTHER NORMS FOR MaxNNScore EVALUATION

The derivation of Theorem 4.2 characterizes the impact of analog noise through the magnitude of the expert weights. In particular, for experts with comparable l_1-norm magnitudes, the analog noise model in Section 2.2 implies that weight vectors with larger l_2-norms are more sensitive to analog noise, as their weight mass is concentrated on fewer large-magnitude weights. While the l_∞-norm captures only the largest absolute weight and may therefore be overly sensitive to a single extreme value, the l_2-norm accounts for the contributions of all weights while placing greater emphasis on large-magnitude weights. Consequently, the l_2-norm provides a more discriminative measure of analog-noise sensitivity and naturally serves as the ranking criterion for expert selection.

To empirically validate this theoretical insight, we compare l_1-norm-based and l_∞-norm-based analog-digital mappings with the proposed l_2-norm-based mapping on OLMoE.

As shown in Table 5, the proposed l_2-norm-based mapping consistently achieves higher accuracy. These results demonstrate that the l_2-norm more effectively captures expert sensitivity to analog noise.

Table 5: Comparison of different neuron norm-based expert selection methods.

Method	**Experts in Digital (%)**	**Avg. Accuracy (%)**
ℓ_2-norm **(Ours)**	25.00	$\mathbf{52.51 \pm 0.16}$
	37.50	$\mathbf{54.13 \pm 0.12}$
	50.00	$\mathbf{56.17 \pm 0.19}$
	62.50	$\mathbf{57.68 \pm 0.13}$
	75.00	$\mathbf{58.70 \pm 0.09}$
ℓ_1-norm	25.00	50.10 ± 0.19
	37.50	51.95 ± 0.25
	50.00	55.65 ± 0.11
	62.50	57.28 ± 0.07
	75.00	58.63 ± 0.05
ℓ_∞-norm	25.00	52.28 ± 0.08
	37.50	54.05 ± 0.14
	50.00	55.81 ± 0.07
	62.50	57.24 ± 0.09
	75.00	58.40 ± 0.13

F SENSITIVITY ANALYSIS OF ADC/DAC BIT-WIDTH

We conduct a sensitivity analysis by varying the ADC/DAC precision from 6 to 10 bits, and the results are summarized Table 6.

Table 6: Average accuracy (%) under different ADC/DAC precisions and expert mapping strategies on OLMoE

ADC/DAC Bits	Method	Experts in Digital (%)	Avg. Accuracy (%)
10	**Ours**	6.25	**47.50 ± 0.18**
		12.50	**49.28 ± 0.12**
		25.00	**53.01 ± 0.10**
	Higher router norm	6.25	36.07 ± 0.18
		12.50	37.16 ± 0.16
		25.00	37.51 ± 0.17
	Act. freq.	6.25	38.13 ± 0.09
		12.50	38.38 ± 0.13
		25.00	38.92 ± 0.15
	Act. weight	6.25	37.84 ± 0.18
		12.50	38.53 ± 0.21
		25.00	40.36 ± 0.18
8	**Ours**	6.25	**47.10 ± 0.17**
		12.50	**48.92 ± 0.13**
		25.00	**52.51 ± 0.16**
	Higher router norm	6.25	36.14 ± 0.10
		12.50	36.95 ± 0.19
		25.00	37.53 ± 0.17
	Act. freq.	6.25	37.95 ± 0.17
		12.50	38.15 ± 0.12
		25.00	38.81 ± 0.09
	Act. weight	6.25	37.76 ± 0.25
		12.50	38.43 ± 0.14
		25.00	40.48 ± 0.16
6	**Ours**	6.25	**42.79 ± 0.16**
		12.50	**44.35 ± 0.13**
		25.00	**48.43 ± 0.08**
	Higher router norm	6.25	35.26 ± 0.13
		12.50	35.91 ± 0.14
		25.00	36.45 ± 0.15
	Act. freq.	6.25	36.37 ± 0.16
		12.50	36.54 ± 0.15
		25.00	37.27 ± 0.15
	Act. weight	6.25	36.35 ± 0.18
		12.50	36.90 ± 0.18
		25.00	38.30 ± 0.13

As expected, increasing the ADC/DAC precision improves the overall inference accuracy for all methods by reducing quantization noise introduced during analog computation. Nevertheless, the proposed MaxNNScore consistently achieves the highest accuracy across all bit-width configurations and digital expert ratios.

G TOKEN VISUALIZATION OF NEURONS CORRESPONDING TO MaxNNorm



Figure 5: Token visualization of the first MoE block of OLMoE. Expert 30, 50, and 14 correspond to lowest three MaxNNorm neurons over the up-projection matrices. Similarly, Expert 38, 57, and 6 correspond to the highest MaxNNorm neurons over the up-projection matrices.

We visualize the top activating tokens of the neurons corresponding to maximum neuron norm for OLMoE. Figure 5 presents the results for the first layer of the model. Expert 30, 50, and 14 are the lowest MaxNNorm experts for the up-projection matrices. Expert 38, 57, and 6 are highest MaxNNorm experts for the up-projection layers. As we can see, the lowest neuron norm experts are activating by semantically less-frequent tokens (e.g., 'ach', 'ell', 'Ireland', 'ite', 'erry' etc.). On the other hand highest neuron norm experts are activating to semantically more-frequent tokens (e.g., 'the', 'a', 'this', 'them', 'each', 'and', etc.). This justify our intuition that higher MaxNNorm experts specialize to more-frequent tokens.

H Discussion on the independence of the analysis of the underlying analog hardware

Our analysis is agnostic of underlying analog hardware. The programming noise of both PCM- and ReRAM-based NVM devices is commonly modeled using third- or fourth-degree polynomials in the analog in-memory computing (AIMC) literature (Rasch et al., 2023; Büchel et al., 2025a; Wan et al., 2022) and in widely used simulators such as AIHWKIT (Rasch et al., 2021). The polynomial coefficients are obtained by fitting experimental measurements from the target device. Consequently, the noise model adopted in Section 2.2 is a general empirical analog-noise model rather than one specific to PCM devices, making our theoretical analysis applicable to a broad range of NVM devices.

The effect of device-specific properties, including cell resolution, is captured through the noise coefficients. Device-specific characteristics, such as the NVM cell resolution, influence the fitted coefficients of the polynomial noise model (Rasch et al., 2021). Therefore, rather than assuming a particular cell resolution, in Figure 4 of our paper, we systematically sweep the noise-model coefficients to emulate a broad range of analog noise levels arising from different device technologies and internal device properties. As shown in the figure, the proposed analog-digital mapping consistently outperforms all competing baselines across this range of noise conditions, demonstrating that our conclusions are not tied to a specific analog hardware implementation.

I Discussion on practical and broader impact of the proposed analog-digital mapping

The proposed method can be directly extended to larger and more complex models. Unlike ENS, which requires manual identification of noise sensitivity of each expert separately and hence leads to substantially large compute and time for digital expert identification for models with a high number of experts and layers, the proposed analog–digital mapping is independent of the model size and can be applied to larger MoE architectures without modification. Specifically, the same expert-selection procedure is applied independently within each MoE layer: $\mathrm{MaxNNScore}$ is computed for every expert, the experts are ranked according to their scores, and the selected experts are assigned to digital while the remaining experts are mapped to the analog arrays. Therefore, extending the methodology to larger models simply requires repeating the same procedure for the additional experts and layers.

The proposed method is directly compatible with existing heterogeneous AIMC accelerators. Existing analog in-memory computing (AIMC) accelerators already employ a heterogeneous execution paradigm (Büchel et al., 2025b;a; Jain et al., 2022). In particular, a substantial portion of transformer inference, including $\mathrm{Attention}(Q, K, V)$, SwiGLU activation, residual connections, layer normalization, and $\mathrm{SoftMax}$, is executed in the digital core of the accelerator, while matrix-vector multiplications are performed on the analog arrays. Our proposed analog–digital mapping naturally follows this execution model: the selected digital experts can be executed in the existing digital core alongside these operations, requiring no fundamental architectural modifications to the accelerator.

The proposed method provides a practical direction for future AIMC accelerators. Current AIMC accelerators already rely on heterogeneous execution, but they treat all matrix-vector multiplications uniformly. Our work introduces a theory-guided analog–digital mapping strategy that selectively assigns only the most analog-noise-sensitive experts to digital hardware while keeping the remaining experts in analog. This substantially improves inference accuracy with only a modest increase in digital computation, thereby preserving the memory, energy-efficiency, and latency advantages of analog in-memory computing. We believe this methodology provides a practical design paradigm for deploying increasingly large and complex MoE models on future heterogeneous AIMC accelerators.

J PRELIMINARIES

For theoretical analysis, we adopt exactly the same setting as in Chowdhury et al. (2026). We re-state the setting here.

As we theoretically analyze the expert-choice routing, for any training step t, equation (7) can be re-written as,

$$f^{(t)}(\mathbf{X}) = \sum_{s=1}^{k} f_s^{(t)}(\mathbf{X}), \text{where } f_s^{(t)}(\mathbf{X}) = a^{(s)} \sum_{j \in J_s^{(t)}(\mathbf{X})} G_j^{(s,t)} \sum_{r=1}^{m} \phi(\langle \mathbf{w}_r^{(s,t)}, \mathbf{x}^{(j)} \rangle) \tag{15}$$

where, $J_s^{(t)}(\mathbf{X})$ is the set of tokens routed to expert s at step t, $\mathbf{w}_r^{(s,t)}$ is the r-th neuron of $\mathbf{W}_{\text{up}}^{(s)}$ at step t, and $\mathbf{x}^{(j)}$ is the j-th token of the sequence $\mathbf{X}$.

Tokens with top l routing scores, i.e., the tokens of $\mathbf{X}$ with top l indices of $[\mathbf{X}^\top \mathbf{\Sigma}]_{:,s}$ are routed to expert s.

We consider $\phi = \text{ReLU}(\cdot)$, i.e., the rectified linear unit. The routing weight of j-th token at expert s can be calculated as,

$$G_j^{(s,t)} = \begin{cases} \dfrac{\exp(\langle \mathbf{w}_s^{(t)}, \mathbf{x}^{(j)} \rangle)}{\sum_{i \in J_s^{(t)}(\mathbf{X})} \exp(\langle \mathbf{w}_s^{(t)}, \mathbf{x}^{(i)} \rangle)}, & j \in J_s^{(t)}(\mathbf{X}), \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

Here, $\mathbf{w}_s^{(t)} = \mathbf{\Sigma}_{:,s}^{(t)}$ is the routing vector corresponding to the expert s at step t.

The model is trained to minimize the Hinge loss

$$\hat{l}^{(t)}(\mathbf{X}, y) = \max(1 - y f^{(t)}(\mathbf{X}), 0) \tag{17}$$

while the gradients are evaluated on

$$l^{(t)}(\mathbf{X}, y) = 1 - y f^{(t)}(\mathbf{X}) \tag{18}$$

For any input $(\mathbf{X}, y)$, the gradient for $\mathbf{w}_r^{(s,t)}$ is evaluated as,

$$\frac{\partial l^{(t)}(\mathbf{X}, y)}{\partial \mathbf{w}_r^{(s,t)}} = -y a^{(s)} \sum_{j \in J_s^{(t)}(\mathbf{X})} G_j^{(s,t)} \mathbf{x}^{(j)} 1_{\langle \mathbf{w}_r^{(s,t)}, \mathbf{x}^{(j)} \rangle \geq 0} \tag{19}$$

The model is trained via Stochastic Gradient Descent algorithm (SGD) with batch size B. The expert weights are updated with learning rate η_e, and the routing parameters are updated with a learning rate η_r. The batch gradient for $\mathbf{w}_r^{(s,t)}$ is evaluated as,

$$\frac{\partial l}{\partial \mathbf{w}_r^{(s,t)}} = \frac{1}{B} \sum_{\mathbf{X} \in \mathcal{B}_t} \frac{\partial l^{(t)}(\mathbf{X}, y)}{\partial \mathbf{w}_r^{(s,t)}} \tag{20}$$

Notations:

1. $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ hides the factor $\log(poly(d))$ with a sufficiently large polynomial $poly(\cdot)$
2. With high probability (abbreviated as $w.h.p.$) refers to the probability $1 - \dfrac{1}{poly(d)}$.

Definitions and assumptions:

Initial router alignment. We consider that at the beginning of training, i.e., at $t = 0$, router $s \in [k]$ is aligned to task-relevant token $\mathbf{v} \in \mathcal{P}_r$, if $p_{\mathbf{v}}^{(s,0)} = \Omega(1)$. In other words, router $s \in [k]$ is aligned with $\mathbf{v}$, if it selects $\mathbf{v}$ with non-trivial gating values at least for a constant fraction of sequences containing $\mathbf{v}$.

For any $\mathbf{v} \in \mathcal{P}_r$, we define a complementary expert proficiency measure for the expert s at time t as, $\bar{p}_{\mathbf{v}}^{(s,t)} := \mathbb{P}\Big[(\mathbf{X}, y) \sim \mathcal{D} : \exists j \in J_s^{(t)}(\mathbf{X}) \text{ such that } \mathbf{x}^{(j)} = \mathbf{v} \big| \exists j \in [n] \text{ such that } \mathbf{x}^{(j)} = \mathbf{v}\Big]$.

Let $S_{\mathbf{v}}$ ($\mathbf{v} \in \mathcal{P}_r$) denote the set of experts whose routers are aligned with $\mathbf{v}$ in the initial model.

Without the loss of generality, we assume that for any $s \in S_{\mathbf{v}}$, $\bar{p}_{-\mathbf{v}}^{(s,0)} = O(1/d)$.

We define $C_n := \max\Big\{\Big|\Big|\mathbf{w}_r^{(s,0)}\Big|\Big|\Big\}_{s\in[k],r\in[m]}$.

We define, $S_+ := \{s \in [k] : a^{(s)} = +1\}$, and $S_- := \{s \in [k] : a^{(s)} = -1\}$.

We define, $\gamma_{\mathbf{v}} := \dfrac{|S_{\mathbf{v}}|}{|S_+|}$ for $\mathbf{v} \in \{\pm\mathbf{o}_1\}$ and $\gamma_{\mathbf{v}} := \dfrac{|S_{\mathbf{v}}|}{|S_-|}$ for $\mathbf{v} \in \{\pm\mathbf{o}_2\}$.

Therefore, $\gamma = \max\{\gamma_{\mathbf{v}}\}_{\mathbf{v}\in\{\mathbf{o}_1,\mathbf{o}_2\}}$.

Without the loss of generality, we assume that $\forall \mathbf{v} \in \mathcal{P}_r, \gamma_{\mathbf{v}} = \Omega(1)$.

We define, $C_p := \min\{\langle \mathbf{w}_s^{(0)}, \mathbf{q} - \mathbf{q}'\rangle\}_{s\in[k],\mathbf{q}\in\mathcal{P}\cup\{-\mathbf{o}_1,-\mathbf{o}_2\},\mathbf{q}'\in\mathcal{P}\cup\{-\mathbf{o}_1,-\mathbf{o}_2\}\setminus\{\mathbf{q}\}}$.

We assume that $C_p > 0$.

We assume that for any $\mathbf{v} \in \mathcal{P}_r$ and any $s \in S_{\mathbf{v}}$, $\frac{\big|\{r\in[m]:\langle \mathbf{w}_r^{(s,0)}, \mathbf{v}\rangle \geq 0\}\big|}{m} = \Omega(1)$,
and $\big||S_+| - |S_-|\big| = O(\sqrt{k})$.

We define,

- $G_{\mathbf{v}}^{(s,t)}$: Routing weights of the token $x^{(j)} = \mathbf{v}$ for some $j \in [n]$ and $\mathbf{v} \in \mathcal{P}_r$ at expert s and step t
- $G_{\mathbf{q}}^{(s,t)}$: Routing weights of the token $x^{(j)} = \mathbf{q}$ for some $j \in [n]$ and $\mathbf{q} \in \mathcal{P}\setminus\{\mathbf{o}_1, \mathbf{o}_2\}$ at expert s and step t
- $l_{\mathbf{q}}^{(s,t)} := \Big|\{j \in J_s^{(t)}(\mathbf{X}) : \mathbf{x}^{(j)} = \mathbf{q}\}\Big|$, is the number of copies of the task-irrelevant vector $\mathbf{q} \in \mathcal{P}\setminus\{\mathbf{o}_1, \mathbf{o}_2\}$ in the set of top l tokens for the input sequence $\mathbf{X}$ at expert s and step t

Components of the neurons' gradients.

For any input $(\mathbf{X}, y) \sim \mathcal{D}$, the gradient component of $\mathbf{w}_r^{(s,t)}$ along any task-relevant vector $\mathbf{v} \in \mathcal{P}_r$ and along any task-irrelevant vector $\mathbf{q} \in \mathcal{P}\setminus\{\mathbf{o}_1, \mathbf{o}_2\}$ at iteration t are evaluated as follows:

$$\Big\langle \frac{\partial l(\mathbf{X}, y)}{\partial \mathbf{w}_r^{(s,t)}}, \mathbf{q} \Big\rangle = \begin{cases} 0 & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{q}\rangle < 0 \\ 0 & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{q}\rangle \geq 0 \text{ but, } \nexists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = \mathbf{q} \\ -ya^{(s)} l_{\mathbf{q}}^{(s,t)} G_{\mathbf{q}}^{(s,t)} & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{q}\rangle \geq 0 \text{ and, } \exists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = \mathbf{q} \end{cases} \tag{21}$$

$$\Big\langle \frac{\partial l(\mathbf{X}, y)}{\partial \mathbf{w}_r^{(s,t)}}, \mathbf{v} \Big\rangle = \begin{cases} 0 & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{v}\rangle < 0 \text{ but } \nexists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = -\mathbf{v} \\ ya^{(s)} G_{-\mathbf{v}}^{(s,t)}(\mathbf{X}) & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{v}\rangle < 0 \text{ and } \exists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = -\mathbf{v} \\ 0 & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{v}\rangle \geq 0 \text{ but } \nexists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = \mathbf{v} \\ -ya^{(s)} G_{\mathbf{v}}^{(s,t)}(\mathbf{X}) & \text{if } \langle \mathbf{w}_r^{(s,t)}, \mathbf{v}\rangle \geq 0 \text{ and } \exists j \in J_s^{(t)}(\mathbf{X}) \text{ s.t. } \mathbf{x}^{(j)} = \mathbf{v} \end{cases} \tag{22}$$

Lemma J.1 (Lemma J.5(ii) of Chowdhury et al. (2026)). *For any expert $s \in S_{\mathbf{v}}$ such that $\mathbf{v} \in \{\mathbf{o}_1, \mathbf{o}_2\}$, and $\forall \mathbf{q} \in \mathcal{P} \backslash \{\mathbf{o}_1, \mathbf{o}_2\}$, by selecting $\eta_r = O\left(\frac{\eta_e C_p}{ml^2 C_n^2}\right)$ and $B = \tilde{\Omega}\left(d^2\right)$, we can ensure that after $T' = O\left(\frac{lC_n}{\alpha \eta_e}\right)$ steps, $p_{\mathbf{v}}^{(s,T')} \geq p_{\mathbf{v}}^{(s,0)}$ and, $\bar{p}_{-\mathbf{v}}^{(s,T')} \leq \bar{p}_{-\mathbf{v}}^{(s,0)}$.*

Lemma J.2 (Lemma J.6(ii) of Chowdhury et al. (2026)). *For any expert $s \in S_{\mathbf{v}}$ such that $\mathbf{v} \in \{-\mathbf{o}_1, -\mathbf{o}_2\}$, and $\forall \mathbf{q} \in \mathcal{P} \backslash \{\mathbf{o}_1, \mathbf{o}_2\}$, by selecting $\eta_r = O\left(\frac{\eta_e C_p}{ml^2 C_n^2}\right)$ and $B = \tilde{\Omega}\left(d^2\right)$, we can ensure that after $T' = O\left(\frac{lC_n}{(1-\alpha)\eta_e}\right)$ steps, $p_{\mathbf{v}}^{(s,T')} \geq p_{\mathbf{v}}^{(s,0)}$ and, $\bar{p}_{-\mathbf{v}}^{(s,T')} \leq \bar{p}_{-\mathbf{v}}^{(s,0)}$.*

Lemma J.3 (Lemma I.1(i) of Chowdhury et al. (2026)). *After $T = \Theta(l^2 C_n \sqrt{\log l}/\alpha\eta_e\sqrt{C_p})$ steps of SGD with batch size $B = \tilde{\Omega}(d^2)$ and $\eta_r = O\left(\frac{\eta_e C_p}{ml^2 C_n^2}\right)$, for all $s \in S_{\mathbf{v}}$ and $\mathbf{v} \in \mathcal{P}_r = \{\pm\mathbf{o}_1, \pm\mathbf{o}_2\}$, we have $p_{\mathbf{v}}^{(s,T)} = 1$, $\bar{p}_{-\mathbf{v}}^{(s,T)} = 0$, and $\forall \mathbf{x}^{(j)} = \mathbf{v}$ for some $j \in [n]$, $G_j^{(s,T)} > \frac{1}{2}$.*

K Proof of Lemma 4.1

Lemma K.1 (Full version of Lemma 4.1). *Suppose, the network given in (7) is trained for $T = \Theta(l^2 C_n \sqrt{\log l}/\alpha\eta_e\sqrt{C_p})$ steps with batch size $B = \tilde{\Omega}(d^2)$ and $\eta_r = O(\frac{\eta_e C_p}{ml^2 C_n^2})$. Then, for any $\mathbf{v} \in \{\mathbf{o}_1, \mathbf{o}_2\}$ and any expert s, s', such that $p_{\mathbf{v}}^{(s,T)} = 1$ and $p_{-\mathbf{v}}^{(s',T)} = 1$, we have*

$$\mathrm{MaxNNScore}^{(s)} < \mathrm{MaxNNScore}^{(s')} \tag{23}$$

Proof. From Lemma J.3, for any $s \in S_+$, and for any $\mathbf{v} \in \{\mathbf{o}_1, -\mathbf{o}_1\}$, if $p_{\mathbf{v}}^{(s,T)} = 1$, then $s \in S_{\mathbf{v}}$.

Similarly, for any $s \in S_-$, and for any $\mathbf{v} \in \{\mathbf{o}_2, -\mathbf{o}_2\}$, if $p_{\mathbf{v}}^{(s,T)} = 1$, then $s \in S_{\mathbf{v}}$.

Now, WLOG let us assume $s \in S_{\mathbf{o}_1}$ and $s' \in S_{-\mathbf{o}_1}$.

Then, using Lemma J.1, $\forall r \in [m]$, $\forall t \leq T$, $\left|\langle \frac{\partial l}{\partial \mathbf{w}_r^{(s,t)}}, \mathbf{o}_1 \rangle\right| \leq \frac{\alpha}{2} + \tilde{O}(\frac{1}{\sqrt{B}})$, which implies, for $B = \tilde{\Omega}(d^2)$, $\forall r \in [m]$, $\left|\langle \mathbf{w}_r^{(s,T)}, \mathbf{o}_1 \rangle\right| = O(l^2 C_n \sqrt{\log l}/\eta_e \sqrt{C_p})$.

On the other hand, $\forall r \in [m]$, $\forall t \leq T$, if $\langle \mathbf{w}_r^{(s,t)}, \mathbf{o}_2 \rangle \geq 0$, then $\langle \frac{\partial l}{\partial \mathbf{w}_r^{(s,t)}}, \mathbf{o}_2 \rangle \geq 0$, and if $\langle \mathbf{w}_r^{(s,t)}, \mathbf{o}_2 \rangle \leq 0$, then $\langle \frac{\partial l}{\partial \mathbf{w}_r^{(s,t)}}, -\mathbf{o}_2 \rangle \geq 0$. Therefore, $\left|\langle \mathbf{w}_r^{(s,T)}, \mathbf{o}_2 \rangle\right| = O(C_n)$.

Again, for any $\mathbf{q} \in \mathcal{P} \backslash \{\mathbf{o}_1, \mathbf{o}_2\}$, $\forall r \in [m]$, $\forall t \leq T$, $\left|\langle \frac{\partial l}{\partial \mathbf{w}_r^{(s,t)}}, \mathbf{q} \rangle\right| \leq O(\frac{1}{d}) + \tilde{O}(\frac{1}{\sqrt{B}})$, which implies, $\left|\langle \mathbf{w}_r^{(s,T)}, \mathbf{q} \rangle\right| = O(C_n)$.

Therefore, $\mathrm{MaxNNScore}^{(s)} = O(l^2 C_n \sqrt{\log l}/\sqrt{C_p})$.

Now, as Lemma J.2 holds, $\forall r \in [m]$ such that $\langle \mathbf{w}_r^{(s',t)}, -\mathbf{o}_1 \rangle \geq 0$,

$\forall t \geq \frac{\alpha T}{1-\alpha}$, $\langle \frac{\partial l}{\partial \mathbf{w}_r^{(s',0)}}, -\mathbf{o}_1 \rangle \leq -\Omega(1-\alpha) + \tilde{O}\left(\frac{1}{\sqrt{B}}\right)$.

Therefore, $\forall r \in [m]$ such that $\langle \mathbf{w}_r^{(s',0)}, -\mathbf{o}_1 \rangle \geq 0$, $\langle \mathbf{w}_r^{(s',T)}, -\mathbf{o}_1 \rangle = \Omega((1-2\alpha) l^2 C_n \sqrt{\log l}/\alpha\sqrt{C_p})$.

Therefore, $\mathrm{MaxNNScore}^{(s')} = \Omega((1-2\alpha) l^2 C_n \sqrt{\log l}/\alpha\sqrt{C_p})$ which completes the proof. □

L PROOF OF THEOREM 4.2

Theorem L.1 (Full version of Theorem 4.2)**.** *Suppose, the network given in (7) is trained for* $T = \Theta(l^2 C_n \sqrt{\log l}/\alpha\eta_e\sqrt{C_p})$ *steps with batch size* $B = \tilde{\Omega}(d^2)$ *and* $\eta_r = O(\frac{\eta_e C_p}{ml^2 C_n^2})$*. Suppose,* γ *fraction of the experts* $s \in [k]$ *in the trained model satisfies* $p_{-\mathbf{v}}^{(s,T)} = 1$ *for* $v \in \{\mathbf{o}_1, \mathbf{o}_2\}$*.*

Then, if the DAC and ADC bit-widths satisfy

$$b_D \geq \log_2(1 + \Omega(\sqrt{d})), b_A \geq \log_2(1 + \Omega(l)) \tag{24}$$

and the maximum analog noise magnitude, $\max(c_0, c_1, c_2, c_3, c_s)$ *satisfies*

$$\max(c_0, c_1, c_2, c_3, c_s) \leq c_A := O(\frac{\alpha}{1-\alpha} \times \frac{1}{l\sqrt{\log(kmld^2)}}), \tag{25}$$

then with high probability, the all-experts-in-analog mapped MoE model, denoted by f_A*, has guaranteed generalization i.e.,*

$$\mathbb{P}[\forall(\mathbf{X}, y) \sim \mathcal{D} : y f_A^{(T)}(\mathbf{X}) > 0] = 1. \tag{26}$$

Moreover, if $\Gamma \geq \gamma$ *fraction of the experts with top* MaxNNScore *are mapped in digital while others in analog,* b_D, b_A *satisfy (24), and the maximum analog noise magnitude,* $\max(c_0, c_1, c_2, c_3, c_s)$ *satisfies*

$$\max(c_0, c_1, c_2, c_3, c_s) \leq c_H := \frac{1-\alpha}{\alpha} c_A, \tag{27}$$

then with high probability, the heterogeneous-expert mapped MoE model, denoted by f_H*, has guaranteed generalization. i.e.,*

$$\mathbb{P}[\forall(\mathbf{X}, y) \sim \mathcal{D} : y f_H^{(T)}(\mathbf{X}) > 0] = 1. \tag{28}$$

Proof. From Lemma J.3, for any $s_1 \in S_{\mathbf{o}_1}$, $p_{\mathbf{o}_1}^{(s_1,T)} = 1$ and $\forall \mathbf{x}^{(j)} = \mathbf{o}_1$ for some $j \in [n]$, $G_j^{(s_1,T)} \geq \frac{1}{2}$.

Furthermore, $\sum_{r\in[m]} \langle \mathbf{w}_r^{(s_1,T)}, \mathbf{o}_1 \rangle = \Omega(mlC_n\sqrt{\frac{\log l}{C_p}})$.

Therefore, for any $(\mathbf{X}, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $\mathbf{x}^{(j)} = \mathbf{o}_1$,

$\sum_{s_1 \in S_{\mathbf{o}_1}} f_{s_1}^{(T)}(\mathbf{X}) = \Omega(\gamma_{o_1} mlC_n\sqrt{\frac{\log l}{C_p}})$.

On the other hand, from statement (i) of Lemma J.3, for any $s_3 \in S_{-o_1}$, $p_{o_1}^{(s_3,T)} = 0$.

Therefore, for any $(x, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $x^{(j)} = o_1$,

$\sum_{s\in S_+} f_s^{(T)}(x) = \Omega(\gamma_{o_1} kmlC_2\sqrt{\frac{\log l}{C_p}})$.

Again, for any $q \in \mathcal{P}\backslash\{\mathbf{o}_1, \mathbf{o}_2\}$, $\forall s \in [k], \sum_{r\in[m]} \langle \mathbf{w}_r^{(s,T)}, \mathbf{q} \rangle = O(mC_n)$, and $\forall s \in S_-, \sum_{r\in[m]} \langle \mathbf{w}_r^{(s,T)}, \mathbf{o}_1 \rangle = O(mC_n)$.

Therefore, for any $(\mathbf{X}, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $\mathbf{x}^{(j)} = \mathbf{o}_1$, $f^{(T)}(\mathbf{X}) = \Omega(\gamma_{o_1} kmlC_n\sqrt{\frac{\log l}{C_p}}) - O(klmC_n)$, which implies for any $(\mathbf{X}, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $\mathbf{x}^{(j)} = \mathbf{o}_1$, $yf^{(T)}(\mathbf{X}) > 0$.

(**Condition 1**) Therefore, to ensure that for any $(\mathbf{X}, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $\mathbf{x}^{(j)} = \mathbf{o}_1$, $yf_A^{(T)}(\mathbf{X}) > 0$, we need $f^{(T)}(\mathbf{X}) - f_A^{(T)}(\mathbf{X}) = O(kmlC_n\sqrt{\frac{\log l}{C_p}})$.

(**Condition 2**) Similarly, to ensure that for any $(\mathbf{X}, y) \sim \mathcal{D}$ such that $\exists j \in [n]$ with $\mathbf{x}^{(j)} = -\mathbf{o}_1$, $yf_A^{(T)}(\mathbf{X}) > 0$, we need $f^{(T)}(\mathbf{X}) - f_A^{(T)}(\mathbf{X}) = O(\frac{1-\alpha}{\alpha}kmlC_n\sqrt{\frac{\log l}{C_p}})$.

Now, for any $s_1 \in S_{\mathbf{o}_1}$, $\text{MaxNNScore}^{(s_1)} = O(l^2C_n\sqrt{\frac{\log l}{C_p}})$, and $s_3 \in S_{-\mathbf{o}_1}$ $\text{MaxNNScore}^{(s_3)} = O(\frac{1-\alpha}{\alpha}l^2C_n\sqrt{\frac{\log l}{C_p}})$, which implies for any $r \in [m]$, $\max(\mathbf{w}_r^{(s_1,T)}) \leq O(l^2C_n\sqrt{\frac{\log l}{C_p}})$, and $\max(\mathbf{w}_r^{(s_3,T)}) \leq O(\frac{1-\alpha}{\alpha}l^2C_n\sqrt{\frac{\log l}{C_p}})$.

Similarly, for any $s_2 \in S_{\mathbf{o}_2}$, $\text{MaxNNScore}^{(s_2)} = O(l^2C_n\sqrt{\frac{\log l}{C_p}})$, and $s_4 \in S_{-\mathbf{o}_2}$ $\text{MaxNNScore}^{(s_4)} = O(\frac{1-\alpha}{\alpha}l^2C_n\sqrt{\frac{\log l}{C_p}})$, which implies for any $r \in [m]$, $\max(\mathbf{w}_r^{(s_2,T)}) \leq O(l^2C_n\sqrt{\frac{\log l}{C_p}})$, and $\max(\mathbf{w}_r^{(s_4,T)}) \leq O(\frac{1-\alpha}{\alpha}l^2C_n\sqrt{\frac{\log l}{C_p}})$.

Now, for any input token $\mathbf{x}^{(j)}$, we express the output of DAC by $\hat{\mathbf{x}}^{(j)} = \mathbf{x}^{(j)} + \Delta\mathbf{x}^{(j)}$ where $\Delta\mathbf{x}^{(j)}$ is the quantization noise component introduced by DAC. Similarly, for any neuron $\mathbf{w}_r^{(s,T)}$, we express the programmed weight vector of the neuron by $\hat{\mathbf{w}}_r^{(s,T)} = \mathbf{w}_r^{(s,T)} + \Delta\mathbf{w}_r^{(s,T)}$, where $\Delta\mathbf{w}_r^{(s,T)}$ is the programming noise component introduced during weight programming.

We now express the analog-computed output of $\langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle$ by

$$\begin{aligned}\widehat{\langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle} &= \langle \hat{\mathbf{w}}_r^{(s,T)}, \hat{\mathbf{x}}^{(j)} \rangle + N_{sys} + N_{adc} \\ &= \langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle + \langle \mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)} \rangle + \langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle + \langle \Delta\mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)} \rangle \\ &\quad + N_{sys} + N_{adc}\end{aligned}$$

Here, N_{sys} and N_{adc} denote the system and ADC noise, respectively.

Now, to ensure that conditions 1 and 2 hold, we need $\forall s \in [k], r \in [m]$

$$\left| \widehat{\langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle} - \langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle \right| = O(lC_n\sqrt{\frac{\log l}{C_p}})$$

Now, any element $i \in [d]$ of $\Delta\mathbf{x}^{(j)}$, $\Delta\mathbf{x}_i^{(j)} \sim \text{Unif}(-\frac{\Delta_D}{2}, \frac{\Delta_D}{2})$ where Δ_D is the quantization bin size of DAC for input $\mathbf{x}^{(j)}$.

Therefore,

$$\mathbb{P}\left[\left| \langle \mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)} \rangle \right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}}) \right] \leq \exp\left(-\frac{O(l^2C_n^2\frac{\log l}{C_p})}{\sum_{i=1}^d \mathbf{w}_{r_i}^{(s,T)^2}\Delta_D^2}\right) = \exp\left(-\frac{O(l^2C_n^2\frac{\log l}{C_p})}{\left|\left|\mathbf{w}_r^{(s,T)}\right|\right|^2\Delta_D^2}\right)$$

Now, as $\left|\left|\mathbf{w}_r^{(s,T)}\right|\right| = O(\frac{1-\alpha}{\alpha} l^2 C_n \sqrt{\frac{\log l}{C_p}})$, we have

$$\mathbb{P}\left[\left|\langle \mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-\frac{O(C_n^2\frac{\log l}{C_p})}{\sum_{i=1}^d \mathbf{w}_{r_i}^{(s,T)^2}\Delta_D^2}\right)$$
$$= \exp\left(-O(\frac{\alpha^2}{(1-\alpha)^2 l^2 \Delta_D^2})\right)$$

Taking the union bound over $s \in [k]$ and $r \in [m]$,

by selecting $\Delta_D = O(\frac{\alpha}{(1-\alpha)l\sqrt{\log(kmld^2)}})$ we can ensure that

$$\mathbb{P}\left[\left|\langle \mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \frac{1}{4d^2}, \forall s \in [k], r \in [m] \tag{29}$$

Now, for any element $i \in [d]$ of $\Delta\mathbf{w}_r^{(s,T)}$, $\Delta\mathbf{w}_{r_i}^{(s,T)} \sim \mathcal{N}(0, \sigma_p^2)$ where

$\sigma_p = c_0 \max(\mathbf{w}_r^{(s,T)}) + \sum_{i=1}^3 c_u \frac{|\mathbf{w}_{r_i}|^u}{\left(\max(\mathbf{w}_r^{(s,T)})\right)^{u-1}}$.

Therefore, $\langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)}\rangle \sim \mathcal{N}(0, \sigma_p^2 ||\mathbf{x}^{(j)}||^2)$.

Now, as $||\mathbf{x}^{(j)}|| = 1$, we have $\langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)}\rangle \sim \mathcal{N}(0, \sigma_p^2)$.

Therefore,

$$\mathbb{P}\left[\left|\langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-\frac{O(l^2C_n^2\frac{\log l}{C_p})}{\sigma_p^2}\right)$$

Now, as $\max(\mathbf{w}_r^{(s,T)}) = O(\frac{1-\alpha}{\alpha} l^2 C_n \sqrt{\frac{\log l}{C_p}})$,

we have $\sigma_p = O(\max(c_0, c_1, c_2, c_3)\frac{1-\alpha}{\alpha} l^2 C_n \sqrt{\frac{\log l}{C_p}})$.

Therefore,

$$\mathbb{P}\left[\left|\langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-O(\frac{\alpha^2}{(1-\alpha)^2 l^2 \max(c_0, c_1, c_2, c_3)^2})\right)$$

Taking the union bound over $s \in [k]$ and $r \in [m]$,

by selecting $\max(c_0, c_1, c_2, c_3) = O(\frac{\alpha}{(1-\alpha)l\sqrt{\log(kmld^2)}})$ we can ensure that

$$\mathbb{P}\left[\left|\langle \Delta\mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \frac{1}{4d^2}, \forall s \in [k], r \in [m] \tag{30}$$

Now, any pair of elements $\Delta\mathbf{x}^{(j)}$ and $\Delta\mathbf{w}_{r_i}^{(s,T)}$, $(\Delta\mathbf{x}_i^{(j)} \times \Delta\mathbf{w}_{r_i}^{(s,T)}) \sim \text{subgaussian}(0, \sigma_p^2\Delta_D^2/12)$.

Therefore,

$$\mathbb{P}\left[\left|\langle \Delta\mathbf{w}_r^{(s,T)}, \Delta\mathbf{x}^{(j)}\rangle\right| = \Omega(lC_n\sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-\frac{O(l^2C_n^2\frac{\log l}{C_p})}{d\sigma_p^2\Delta_D^2}\right)$$

Now, for our selection of $\max(c_0, c_1, c_2, c_3)$, $\sigma_p = O(\frac{1}{\sqrt{\log(kmld^2)}} l C_n \sqrt{\frac{\log l}{C_p}})$.

Therefore,

$$\mathbb{P}\left[\left|\langle \Delta \mathbf{w}_r^{(s,T)}, \Delta \mathbf{x}^{(j)} \rangle\right| = \Omega(l C_n \sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-O(\frac{\log(kmld^2)}{d\Delta_D^2})\right)$$

Taking the union bound over $s \in [k]$ and $r \in [m]$,

by selecting $\Delta_D = O(\frac{1}{\sqrt{d}})$ we can ensure that

$$\mathbb{P}\left[\left|\langle \Delta \mathbf{w}_r^{(s,T)}, \Delta \mathbf{x}^{(j)} \rangle\right| = \Omega(C_n \sqrt{\frac{\log l}{C_p}})\right] \leq \frac{1}{4d^2}, \forall s \in [k], r \in [m] \tag{31}$$

Now, $N_{sys} \sim \mathcal{N}(0, c_s^2 \max(\mathbf{w}_r^{(s,T)})^2)$.

Therefore,

$$\mathbb{P}\left[N_{sys} = \Omega(l C_n \sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-\frac{O(l^2 C_n^2 \frac{\log l}{C_p})}{c_s^2 \max(\mathbf{w}_r^{(s,T)})^2}\right)$$

Now, as $\max(\mathbf{w}_r^{(s,T)}) = O(\frac{1-\alpha}{\alpha} l^2 C_n \sqrt{\frac{\log l}{C_p}})$, we have

$$\mathbb{P}\left[N_{sys} = \Omega(C_n \sqrt{\frac{\log l}{C_p}})\right] \leq \exp\left(-\frac{O(\frac{\alpha^2}{(1-\alpha)^2 l^2})}{c_s^2}\right)$$

Taking the union bound over $s \in [k]$ and $r \in [m]$,

by selecting $c_s = O(\frac{\alpha}{(1-\alpha) l \sqrt{\log(kmld^2)}})$ we can ensure that

$$\mathbb{P}\left[N_{sys} = \Omega(C_n \sqrt{\frac{\log l}{C_p}})\right] \leq \frac{1}{4d^2}, \forall s \in [k], r \in [m] \tag{32}$$

Now, to ensure that $N_{adc} = O(l C_n \sqrt{\frac{\log l}{C_p}})$ we need the quantization bin-size of ADC, $\Delta_A = O(l C_n \sqrt{\frac{\log l}{C_p}})$.

Now, as $\forall s \in [k], r \in [m], \langle \mathbf{w}_r^{(s,T)}, \mathbf{x}^{(j)} \rangle = O(l^2 C_n \sqrt{\frac{\log l}{C_p}})$, we need

$$b_A \geq \log_2(1 + \Omega(l))$$

Now, taking union bound of (29), (30), (31), and (32), we need

$$\max(c_0, c_1, c_2, c_3, c_s) = O(\frac{\alpha}{(1-\alpha) l \sqrt{\log(kmld^2)}})$$

and

$$b_D \geq \log_2(1 + \Omega(\sqrt{d}))$$

to ensure that conditions 1 and 2 hold with probability at least $1 - \frac{1}{d^2}$

Now, mapping γ fraction of the experts with $\mathrm{MaxNNScore}$ in digital implies for the remaining analog experts, $\max(\mathbf{w}_r^{(s,T)}) = O(l^2 C_n \sqrt{\frac{\log l}{C_p}})$ and $\left|\left|\mathbf{w}_r^{(s,T)}\right|\right| = O(l^2 C_n \sqrt{\frac{\log l}{C_p}})$.

Therefore, following the same procedure, we can show that now we need

$$\max(c_0, c_1, c_2, c_3, c_s) = O(\frac{1}{l\sqrt{\log(kmld^2)}})$$

to ensure that conditions 1 and 2 hold with probability at least $1 - \frac{1}{d^2}$

□

M HYPERPARAMETER CALIBRATION RESULTS FOR DAC AND ADC

The results of calibrating hyperparameters κ and λ are given in Table 7 to Table 14.

Table 7: κ vs. Perplexity (PPL) for OLMoE. Noise added only at the experts.

κ	10	18	25	30	**35**	40	45	50
PPL	10.98	7.59	7.10	7.00	**6.97**	6.99	7.02	7.07

Table 8: λ vs. Perplexity (PPL) for OLMoE ($\kappa = 35$). Noise added only at the experts.

λ	0.75	0.9	**1.0**	1.125	1.25	1.5	1.75	2.0	2.25	2.5	2.75
PPL	7.15	7.10	**7.10**	7.11	7.13	7.19	7.26	7.32	7.42	7.48	7.57

Table 9: κ vs. Perplexity (PPL) for OLMoE. Noise added to both experts and dense modules.

κ	10	15	20	25	**30**	35	40	45
PPL	22.26	8.66	7.50	7.16	**7.07**	7.07	7.12	7.18

Table 10: λ vs. Perplexity (PPL) for OLMoE ($\kappa = 30$). Noise added to experts and dense modules.

λ	0.75	**1.0**	1.25	1.5
PPL	7.42	**7.26**	7.28	7.38

Table 11: κ vs. Perplexity (PPL) for DeepSeekMoE. Noise added to experts.

κ	30	35	**40**	45	50	60
PPL	6.64	6.62	6.62	6.63	6.64	6.68

Table 12: λ vs. Perplexity (PPL) for DeepSeekMoE ($\kappa = 40$). Noise added to experts only.

λ	0.9	1.0	1.25	1.5	2.0	**2.25**	2.5	3.0	4.0
PPL	6.97	6.86	6.73	6.69	6.68	6.67	6.67	6.69	6.76

Table 13: κ vs. Perplexity (PPL) for DeepSeekMoE. Noise added to experts and dense modules.

κ	30	**35**	40	45	50
PPL	7.06	**7.05**	7.09	7.18	7.30

Table 14: λ vs. Perplexity (PPL) for DeepSeekMoE ($\kappa = 35$). Noise added to experts and dense modules.

λ	1.0	1.25	**1.5**	1.75	2.0	2.25
PPL	7.75	7.52	7.51	7.58	7.70	7.85