Unleashing Low-Bit Inference on Ascend NPUs: A Comprehensive Evaluation of HiFloat Formats

Pengxiang Zhao, Hui-Ling Zhen, Xing Li, Han Bao, Weizhe Lin, Zhiyuan Yang, Ziwei Yu, Xin Wang, Mingxuan Yuan, Xianzhi Yu, Zhenhua Dong

Huawei Technologies Co., Ltd.

Abstract

As LLMs scale, low-bit floating-point formats like MXFP and NVFP4 offer new opportunities for precision and efficiency. In this work, we evaluate HiFloat (HiF8 and HiF4), a family of formats tailored for Ascend NPUs. Through rigorous comparison across weight-activation and KV-cache tasks, we provide three key insights: (1) INT8 suits narrow-range data, while floating-point formats excel with high-variance data; (2) in 4-bit regimes, HiF4's hierarchical scaling prevents the accuracy collapse seen in integer formats; and (3) HiFloat is fully compatible with state-of-the-art post-training quantization frameworks. Overall, HiFloat provides a solution for high-efficiency LLM inference on NPUs.

1 Introduction

Exponential scaling in Large Language Models (LLMs) has placed immense pressure on computational throughput and memory bandwidth. Quantization has emerged as a critical paradigm to alleviate these bottlenecks by reducing memory overhead without significantly sacrificing performance. Current literature has largely focused on integer-based methods (Frantar et al., 2022; Lin et al., 2023; Zhao and Yuan, 2025; Xiao et al., 2023; Shao et al., 2024; Li et al., 2024; Liu et al., 2024; Hooper et al., 2025; Li et al., 2025; Kim et al., 2024; Ashkboos et al., 2024; Ma et al., 2024; Liu et al., 2025; Zhang et al., 2024) for weight, activation, and KV cache quantization to enhance inference efficiency.

Building on these foundations, the field is increasingly pivoting toward low-bit floating-point formats. Hardware-native standards like Microscaling (MX) (Rouhani et al., 2023) and NVFP4 (Alvarez et al., 2025) represent a significant step in this evolution. For example, GPT-OSS (OpenAI, 2025) leverages MXFP4 quantization-aware training (QAT) to compress weights, enabling single-GPU execution for large-scale models.

In this work, we evaluate the HiFloat format family (Luo et al., 2024, 2026) tailored for Ascend NPUs. The 8-bit HiFloat (HiF8) features a dynamic mantissa allocation for extended range, while the 4-bit HiFloat (HiF4) employs a three-level hierarchical scaling structure. While HiF8 has been explored for training (Luo et al., 2024), the efficacy of the HiFloat family for post-training quantization (PTQ) remains unexamined. We address this gap with a comprehensive evaluation of their distributional properties, conversion accuracy, and end-to-end performance.

Furthermore, we explore the synergy between HiFloat and established PTQ frameworks like SmoothQuant (Xiao et al., 2023) and SVDQuant (Li et al., 2024). Our analysis demonstrates that these outlier-mitigation strategies complement HiFloat's inherent representational capacity, yielding compounded benefits for model compression. Beyond weight-activation quantization, we evaluate HiFloat for KV cache quantization. The results show that HiF4 achieves state-of-the-art performance, outperforming both 4-bit integer baselines and existing floating-point standards.

Our contributions are summarized as follows:

- **Mathematical Formulation:** We provide formal quantization and dequantization logic for the HiFloat family.
- **Distributional Analysis:** We analyze the representational properties of HiFloat formats.
- **Inference Evaluation:** We present the first comprehensive study of HiFloat for LLM inference, evaluating performance across the entire pipeline, including weights, activations, and KV cache.
- **Algorithmic Synergy Analysis:** We characterize the effective interplay between HiFloat and established PTQ frameworks like SmoothQuant and SVDQuant.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 844–864
July 2-7, 2026

2 Background and Related Work

Floating-point Representation. Floating-point (FP) representation is fundamental to modern deep learning, offering the dynamic range necessary to capture high-variance data distributions. Following the IEEE 754 standard (IEEE Standards Association, 2019), an FP number comprises a sign bit (s), an exponent (e), and a mantissa (m). We adopt the ExMy notation to indicate that x bits are allocated to the exponent and y bits to the mantissa.

The Microscaling Formats. The Microscaling (MX) specification (Rouhani et al., 2023) introduces block-based quantization formats, which partitions tensors into contiguous micro-blocks that share a single, high-precision scale factor. An MX representation is formally defined by the triplet $(k, \mathcal{S}, \mathcal{E})$, where k is the block size (typically 32), $\mathcal{S}$ denotes the shared scaling data type (typically E8M0), and $\mathcal{E}$ is the element data type. We detail MX types and conversion logic in Appendix A.

The NVFP4 Format. The NVFP4 (Alvarez et al., 2025) extends the MX paradigm using a two-level hierarchical scaling mechanism. Each 16-element micro-block is scaled by a fine-grained E4M3 factor, which is then modulated by a per-tensor FP32 scalar to maintain numerical fidelity. We detail its configuration and conversion logic in Appendix B.

PTQ Methods. PTQ methods quantize pre-trained models without extensive retraining, targeting three primary operands: weights (Frantar et al., 2022; Lin et al., 2023), weight-activations (Xiao et al., 2023; Li et al., 2024; Zhang et al., 2024), and KV cache (Liu et al., 2024; Li et al., 2025). To mitigate accuracy degradation caused by heavy-tailed distributions in LLMs, techniques such as outlier splitting (Kim et al., 2024; Li et al., 2024), magnitude smoothing (Lin et al., 2023; Wei et al., 2023; Xiao et al., 2023), and coordinate rotation (Ashkboos et al., 2024; Ma et al., 2024; Liu et al., 2025; Shao et al., 2025) are leveraged to suppress outliers and normalize feature distributions and preserve accuracy. While PTQ methodologies have matured for integer quantization, their efficacy remains under-explored in floating-point domains.

3 The 8-bit HiFloat Format: HiF8

The 8-bit HiFloat format (HiF8) (Luo et al., 2024) extends the IEEE 754 standard by introducing a dynamic prefix code to adapt bit-field allocation to numerical distributions. This design provides the representational flexibility required for diverse workloads through four distinct components (Figure 1). The sign field determines polarity, while the dot field distinguishes between normal and subnormal forms and dictates the dynamic bit-width split between the exponent and mantissa. The exponent field is context-dependent: it represents either an implicit zero or an exponent sign bit paired with an implicit leading bit of 1. This implicit bit is restored during decoding to maximize precision. For example, the configuration $\{\text{SE}, [1]\}$ represents ± 1, while $\{\text{SE}, [1], \text{E}\}$ covers $\{\pm 2, \pm 3\}$. All remaining bits are allocated to the mantissa.

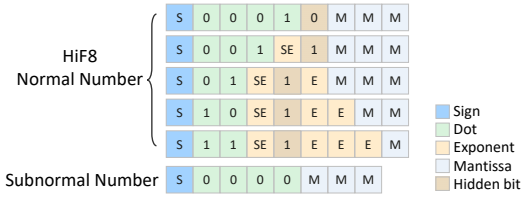


Figure 1: HiF8 Bit Layout.

Metric	E4M3	E5M2	HiF8
Max Precision (Bits)	3	2	3
Max Positive Normal	1.75×2^{8}	1.75×2^{15}	2^{15}
Min Positive Normal	2^{-6}	2^{-14}	2^{-15}
Max Positive Subnormal	1.75×2^{-7}	1.5×2^{-16}	2^{-16}
Min Positive Subnormal	2^{-9}	2^{-16}	2^{-22}

Table 1: Comparison across FP8 Formats.

Adhering to the IEEE 754 standard, the value of a HiF8 normal number is defined by:

$$(-1)^s \times (1 + m) \times 2^e, \tag{1}$$

whereas for subnormal numbers, the value is represented as:

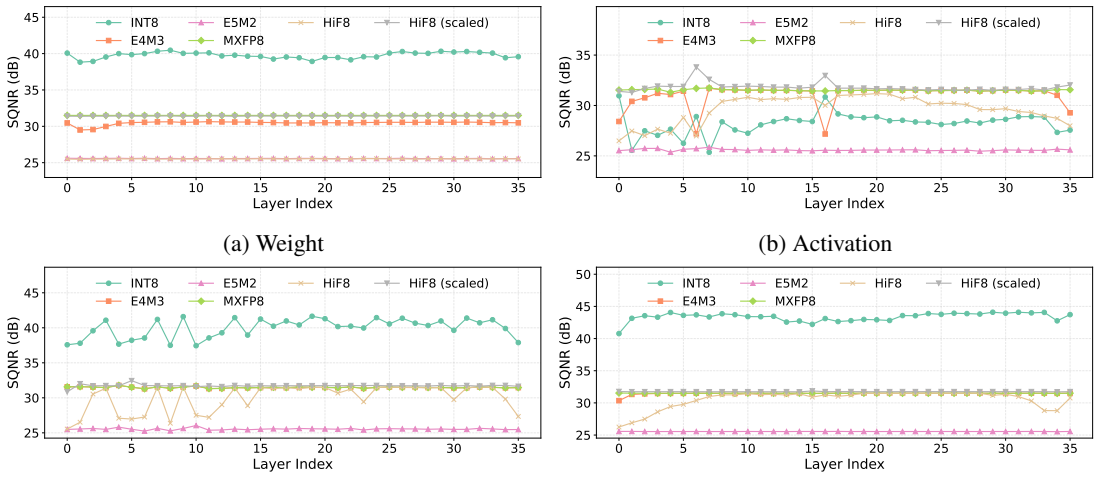
$$(-1)^s \times 2^{m-23}. \tag{2}$$

The dynamic range of HiF8 is summarized in Table 1, while the formal mapping procedure from high-precision inputs is detailed in Appendix C.

While HiF8 performance is established for training (Luo et al., 2024), its fidelity for static quantization remains unexamined. We address this by evaluating how various 8-bit formats preserve signal integrity using the Signal-to-Quantization-Noise Ratio (SQNR):

$$\text{SQNR}(\mathbf{X}, \hat{\mathbf{X}}) = 10 \log_{10} \left(\frac{\|\mathbf{X}\|_F^2}{\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2} \right),$$

where $\mathbf{X}$ and $\hat{\mathbf{X}}$ are the original and quantized



(a) Weight (b) Activation

(c) Key State (d) Value State

Figure 2: Layer-wise Quantization SQNR (↑) Comparison for Qwen3-8B across 8-bit Formats.

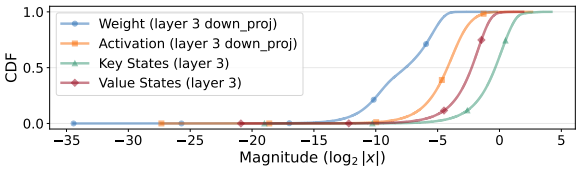


Figure 3: CDF of Qwen3-8B Representative Cases.

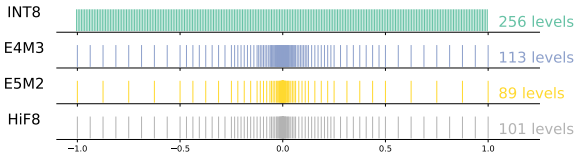


Figure 4: The Distribution of Representable Points within the interval $[-1, 1]$ of 8-bit Formats.

weight matrices, and $\|\cdot\|_F$ is the Frobenius norm. A higher SQNR signifies superior fidelity.

Weight Fidelity. Figure 2a illustrates a consistent 8-bit SQNR hierarchy across Qwen3-8B (Qwen Team, 2025) weights: INT8 (per-channel) > HiF8 (scaled) ≈ MXFP8 (E4M3) > E4M3 > E5M2 ≈ HiF8. Detailed per-weight metrics are in Appendix F.1. This trend reveals a fundamental shift in numerical requirements from training to inference. While HiF8's expansive dynamic range captures volatile gradients during training (Luo et al., 2024), it is suboptimal for the static, bounded distributions of pre-trained weights. As shown in Figure 3, weights concentrate heavily within $[2^{-9}, 2^{-5}]$, with negligible density in the extreme ranges where HiF8 allocates significant representational capacity. This mismatch results in bit waste. Within the primary range ($[-1, 1]$), INT8 utilizes its full 8-bit budget (256 levels), whereas E4M3, HiF8, and E5M2 provide only 113, 101, and 89 levels, respectively (Figure 4). Consequently, HiF8 exhibits increasing discretization coarseness as magnitudes deviate from zero (see Appendix G.1), sacrificing the fine-grained precision necessary for high-fidelity weight compression. To mitigate this,

Feature	INT8	MXFP8	HiF8	HiF8 (scaled)
Scaling Granularity	Per-axis	Block-based	None	Per-axis
Block Size	N	32	—	N
Scale Format	BF16	E8M0	—	FP32
Element Format	INT8	E4M3	HiF8	HiF8
Bits/Element	$\gtrsim 8.00$	8.25	8.00	$\gtrsim 8.00$

Table 2: Comparison of 8-bit Quantization Formats.

a per-channel scaling factor could be employed to realign the target tensor's range with HiF8's capacity; the specific algorithm for this approach is detailed in Appendix D. Denoted as HiF8 (scaled), such a configuration ($K = 16$) yield performance comparable to that of MXFP8. A comparison of scaling granularity and bit-width is provided in Table 2. MXFP8 typically requires a higher bit-budget per element compared to the HiF8 variants.

Activation Fidelity. Unlike weights, activations exhibit high dynamic volatility and emergent outliers. We evaluate 8-bit activation fidelity using Qwen3-8B on Wikitext-2 dataset (Merity et al., 2016). As shown in Figure 2b, there is HiF8 (scaled) ≈ MXFP8 (E4M3) > E4M3 > HiF8 > INT8 (per-token) > E5M2. Detailed per-weight's activation comparisons are provided in

Figure 5: HiF4 Hierarchical Quantization.

Feature	MXFP4	NVFP4	HiF4
Hierarchy Levels	1	2	3
Per-tensor Scaling	✗	✓	✗
Block Size	32	16	64
Sub-lock Size	✗	✗	8
Micro-block Size	✗	✗	4
Scaling Format	E8M0	FP32/E4M3	E6M2/E1/E1
Element Format	E2M1	E2M1	E1M2
Bits/Element	4.25	$\gtrsim$ 4.50	4.50

Table 3: Comparison of 4-bit Hierarchical Quantization.

Appendix F.2. Symmetric floating-point formats underutilize representable states due to activation asymmetry, wasting bit-budget on empty numerical regions; however, their wide dynamic range inherently accommodates extreme outliers. Conversely, while INT8 captures distributional shifts via zero-points, large-magnitude outliers force a coarse, uniform quantization step that significantly degrades resolution near zero, where activation density is highest. With $K = 4$, HiF8 (scaled) achieves performance comparable to that of MXFP8. A detailed distributional analysis is provided in Appendix G.1.

Ultimately, end-to-end performance depends on the complex interplay between weight and activation precision. For instance, the superior fidelity of INT8 for weights may be offset by its inability to capture activation dynamics. We evaluate the end-to-end quantization performance in Section 5.

KV Cache Fidelity. We evaluate KV cache fidelity by applying per-token quantization to Key and Value tensors across 64 Wikitext-2 sequences. As shown in Figures 2c and 2d, both tensors follow a consistent SQNR hierarchy: INT8 (per-token) $>$ HiF8 (scaled) $\approx$ MXFP8 (E4M3) $>$ E4M3 $>$ HiF8 $>$ E5M2.

This hierarchy suggests that the linear projections generating Key and Value states effectively mitigate the emergent outliers observed in raw activations (Figures 11c and 11d in Appendix G.1). Thus, the uniform resolution of INT8 is better suited for these narrower ranges than the expansive, but coarser, dynamic range of HiF8 or E5M2. With $K = 1$, HiF8 (scaled) achieves performance comparable to that of MXFP8.

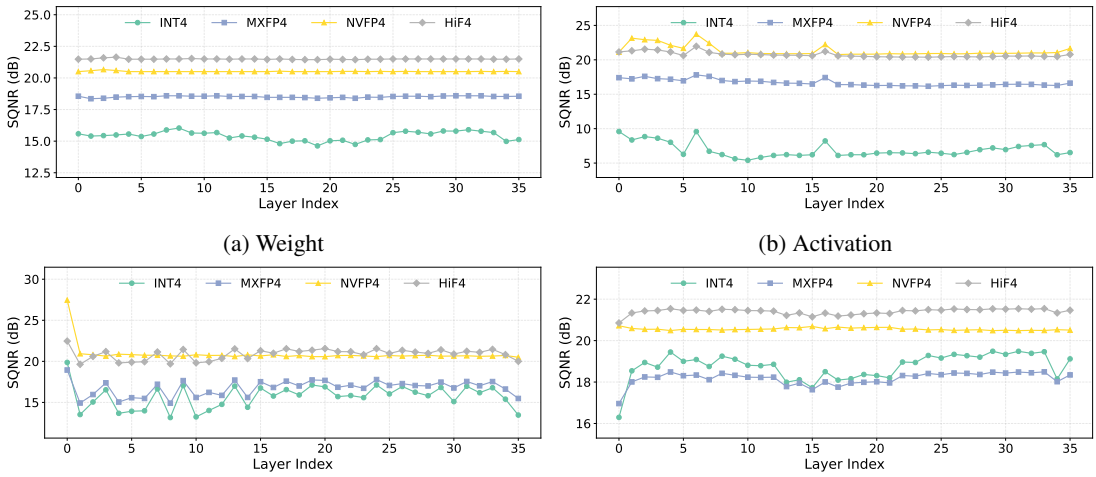
4 The 4-bit HiFloat Format: HiF4

As shown in Figure 5, The 4-bit HiFloat format (HiF4) (Luo et al., 2026) employs a three-level hierarchical encoding scheme. Each 64-element block shares an 8-bit unsigned E6M2 scale, which is refined by 1-bit E1M0 factors at both the 8-element sub-block and 4-element micro-block levels. The individual elements are then represented in a 4-bit E1M2 format. A summary of the 4-bit formats is provided in Table 3. The formal procedure for mapping high-precision inputs to the HiF4 representation is detailed in Appendix E.

Weight Fidelity. Halving the bit-budget to 4 bits creates a critical representational bottleneck. As shown in Figure 6a, a consistent performance hierarchy emerges across Qwen3-8B layers: HiF4 $>$ NVFP4 $>$ MXFP4 $>$ INT4 (per-channel). Detailed per-weight metrics are provided in Appendix F.1. This hierarchy demonstrates that at ultra-low precisions, hierarchical block-based scaling is essential to capture local variance and recover representational capacity. While INT4 suffers from extreme binning coarseness, the multi-level scaling of HiF4 and NVFP4 provides a significantly finer discretization grid than the single-level microscaling of MXFP4. A qualitative distributional analysis is provided in Appendix G.2.

Activation Fidelity. As shown in Figure 6b, the 4-bit activation SQNR hierarchy follows: NVFP4 $>$ HiF4 $>$ MXFP4 $>$ INT4 (per-token). Detailed metrics are provided in Appendix F.2. The superior performance of NVFP4 is primarily driven by its incorporation of an FP32 per-tensor global scaling factor. This mechanism provides the expansive dynamic range necessary to accommodate high-magnitude activation outliers. We provide a detailed analysis of these representational trade-offs in Appendix G.2

KV Cache Fidelity. Figures 6c and 6d illustrate 4-bit KV cache fidelity across Qwen3-8B. For Key states, NVFP4 excels in earlier layers, while HiF4 generally dominates as depth increases; in both regimes, hierarchical floating-point formats significantly outperform the uniform INT4 baseline. For Value states, the performance hierarchy shifts to: HiF4 $>$ NVFP4 $>$ INT4 (per-channel) $>$ MXFP4. This shift suggests that Value distribu-



(a) Weight (b) Activation

(c) Key State (d) Value State

Figure 6: Layer-wise Quantization SQNR (↑) Comparison for Qwen3-8B across 4-bit Formats.

tions are more stable and bounded than Key states; in such contexts, the uniform resolution of INT4 provides a slight advantage over the coarser, single-level scaling of MXFP4. These observations align with the representable distributions analyzed in Appendix G.2, where the finer granularity of HiF4 and NVFP4 consistently mitigates the coarse binning seen in INT4 and MXFP4.

5 Experimental Results

We evaluate the end-to-end performance of HiFloat formats across multiple models and assess their synergy with existing PTQ frameworks.

5.1 Settings

Following prior work, we evaluate model performance using perplexity on Wikitext-2 (Merity et al., 2016) and C4 (Raffel et al., 2023) with a 2,048 sequence length. Zero-shot accuracy is further assessed across adiverse suite, including ARC Challenge (Clark et al., 2018), ARC Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), Math-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), BoolQ (Clark et al., 2019), and LongBench (Bai et al., 2024), utilizing the LM Evaluation Harness library (Gao et al., 2024). We evaluate the vision-language models using VLMEvalKit (Duan et al., 2024) on several established benchmarks, including MME (Fu et al., 2025), SEEDBench (Li et al., 2023a), Hallusion-Bench (Guan et al., 2024), and POPE (Li et al., 2023b). For SmoothQuant, we optimize the hyper-parameter α via grid search across $[0.1, 0.9]$ with a 0.1 step size. For SVDQuant, we fix the low-rank component at rank 16.

5.2 Main Results

W8A8 Quantization. Tables 4 and 5 demonstrate high model resilience in the W8A8 regime. For Qwen3-8B (Qwen Team, 2025), all formats maintain over 99% of the BF16 baseline even with basic round-to-nearest (RTN), with INT8 achieving peak performance when paired with SVDQuant. While HiF8 slightly trails INT8 and MXFP8 under RTN, it achieves negligible accuracy loss when integrated with SmoothQuant or SVDQuant. Similarly, for openPangu-7B (Chen et al., 2025), INT8 leads under RTN quantization. SmoothQuant enables HiF8 to emerge as the top-performing format, effectively matching the BF16 baseline.

W4A4 Quantization. Tables 4 and 5 reveal a catastrophic failure for INT4 at W4A4, even when augmented with SmoothQuant and SVDQuant. This underscores that at ultra-low bit widths, uniform integer spacing becomes a fundamental bottleneck. While MXFP4 outperforms the integer baseline, it still incurs notable accuracy drops. In contrast, HiF4 and NVFP4 maintain model integrity. On Qwen3-8B, both formats remain highly competitive, with NVFP4 holding a marginal lead. For openPangu-7B, HiF4 achieves remarkable fidelity, limiting accuracy degradation to just 3.0% under RTN and narrowing to 2.7% with SmoothQuant or SVDQuant. This near-complete recovery of the BF16 baseline confirms that HiF4's hierarchi-

Models	Bit	Methods	Format	PPL (↓)		Acc (↑)				Overall	
				Wiki	C4	HellaSwag	MMLU	Arc-C	MATH-500	Avg.↑	Δ(%)↓
Qwen3-8B	-	-	BF16	9.72	15.43	0.571	0.730	0.555	0.784	1.000	-
	W8A8	RTN	INT	**9.63**	**15.40**	**0.572**	**0.727**	0.547	**0.800**	**1.002**	**-0.2**
			MXFP	9.76	15.48	0.570	**0.727**	**0.549**	0.796	1.000	0.0
			HiF	9.67	15.41	0.569	0.723	0.547	0.792	0.997	0.3
		SmoothQuant	INT	**9.59**	**15.29**	**0.572**	**0.728**	0.548	0.792	1.000	0.0
			MXFP	9.75	15.48	0.570	**0.728**	**0.561**	0.786	**1.001**	**-0.1**
			HiF	9.79	15.47	0.568	0.723	0.554	**0.798**	**1.001**	**-0.1**
		SVDQuant	INT	**9.58**	**15.28**	**0.573**	**0.729**	**0.552**	**0.796**	**1.003**	**-0.3**
			MXFP	9.70	15.40	0.571	**0.729**	0.551	0.792	1.001	-0.1
			HiF	9.64	15.36	0.570	0.728	0.550	0.794	1.001	-0.1
	W4A4	RTN	INT	3.6e4	3.7e4	0.260	0.248	0.196	0.026	0.277	72.3
			MXFP	11.21	18.13	0.519	0.652	0.480	0.720	0.898	10.2
			NVFP	**10.16**	**16.24**	0.551	**0.706**	0.521	**0.786**	**0.971**	**2.9**
			HiF	10.30	16.55	**0.555**	0.705	**0.529**	0.760	0.965	3.5
		SmoothQuant	INT	6.1e2	6.0e2	0.265	0.234	0.190	0.014	0.267	73.3
			MXFP	10.90	17.46	0.535	0.670	0.492	0.748	0.926	7.4
			NVFP	**10.12**	**16.21**	0.549	**0.708**	0.532	**0.806**	**0.983**	**1.7**
			HiF	10.27	16.31	**0.557**	0.705	**0.548**	0.772	0.978	2.2
		SVDQuant	INT	3.8e2	4.2e2	0.295	0.268	0.215	0.048	0.313	68.7
			MXFP	10.58	17.02	0.539	0.685	0.503	0.742	0.935	6.5
			NVFP	**9.92**	**15.88**	0.560	**0.719**	0.535	**0.774**	**0.980**	**2.0**
			HiF	9.98	16.05	**0.563**	0.717	**0.539**	0.768	**0.980**	**2.0**

Table 4: Performance Evaluation of Weight-Activation Quantization on Qwen3-8B. **Bold** indicates best performance per configuration; Δ shows degradation from BF16 baseline.

Models	Bit	Methods	Format	PPL (↓)		Acc (↑)				Overall	
				Wiki	C4	HellaSwag	MMLU	Arc-C	MATH-500	Avg.↑	Δ(%)↓
openPangu-7B	-	-	BF16	34.95	57.18	0.450	0.567	0.323	0.898	1.000	-
	W8A8	RTN	INT	38.76	63.02	**0.456**	0.549	**0.331**	0.890	**0.995**	**0.5**
			MXFP	**34.89**	**55.67**	0.446	0.535	0.321	**0.906**	0.986	1.4
			HiF	35.76	58.37	0.447	0.523	0.327	0.892	0.978	2.2
		SmoothQuant	INT	**35.79**	**56.90**	0.450	**0.588**	0.321	0.866	0.995	0.5
			MXFP	35.01	56.90	0.444	0.580	0.332	0.884	1.001	-0.1
			HiF	36.67	59.38	0.448	0.578	**0.340**	**0.902**	**1.014**	**-1.4**
		SVDQuant	INT	36.85	59.42	**0.454**	0.557	**0.328**	0.894	**0.998**	**0.2**
			MXFP	**34.92**	**56.25**	0.449	0.551	0.322	**0.900**	0.993	0.7
			HiF	35.28	57.45	0.449	0.543	0.324	0.896	0.989	1.1
	W4A4	RTN	INT	2.2e4	1.6e4	0.260	0.234	0.220	0.014	0.326	67.4
			MXFP	54.58	63.41	0.429	0.490	0.308	0.818	0.914	8.6
			NVFP	**38.96**	**57.51**	**0.435**	0.488	**0.322**	**0.866**	0.943	5.7
			HiF	41.32	59.69	**0.435**	**0.563**	0.310	0.862	**0.970**	**3.0**
		SmoothQuant	INT	6.4e2	7.3e2	0.312	0.275	0.220	0.072	0.393	60.7
			MXFP	45.55	62.67	0.430	0.524	**0.315**	0.830	0.938	6.2
			NVFP	**38.68**	**58.18**	0.437	0.486	0.313	**0.894**	0.952	4.8
			HiF	39.69	60.27	**0.443**	**0.543**	0.306	0.886	**0.973**	**2.7**
		SVDQuant	INT	856.2	724.5	0.313	0.325	0.249	0.256	0.510	49.0
			MXFP	46.25	60.85	0.433	0.513	0.313	0.824	0.930	7.0
			NVFP	**37.52**	**57.28**	0.439	0.525	**0.321**	**0.870**	0.962	3.8
			HiF	38.85	58.42	**0.440**	**0.559**	0.315	0.864	**0.973**	**2.7**

Table 5: Performance Evaluation of Weight-Activation Quantization on openPangu-7B. **Bold** indicates best performance per configuration; Δ shows degradation from BF16 baseline.

cal structure is uniquely suited for outlier-aware optimization under stringent 4-bit constraints. To demonstrate generalization across architectures and modalities, we further evaluate W4A4 RTN quantization across these data formats on the recent Qwen3.5-9B (Qwen Team, 2026) and Qwen3-VL-8B-Instruct (Qwen Team, 2025) models, with results detailed in Appendix G.3 Tables 9 and 10.

KV Cache Quantization. Building on the success of SmoothQuant in W8A8 and W4A4 regimes, we extend our evaluation to include KV cache quantization (Table 11, Appendix G.3). At 8-bit level, Qwen3-8B remains largely unaffected by these additional steps across all formats. However, for openPangu-7B, the performance gap widens, where HiF8 exhibits a more pronounced accuracy decline than MXFP8. The transition to W4A4 augmented by Q16KV4 and QKV4 serves as a stress test. MXFP4 incurs a 10% accuracy loss on Qwen3-8B and suffers a total collapse on openPangu-7B, with degradation surging beyond 60%. In contrast, NVFP4 and HiF4 demonstrate remarkable resilience. In particular, HiF4 consistently yields superior results for Qwen3-8B, limiting accuracy losses to 3.15% for Q16KV4 and 3.92% for QKV4.

Format	Prefill	Decode
BF16	5567.98	65.21
MXFP8	4993.81	45.98
HiF8	4764.07	44.45
MXFP4	4557.54	30.43

Table 6: Comparison of prefill and decode latencies (ms) for Qwen3-30B-A3B across evaluated data formats.

This trend persists for openPangu-7B, where HiF4 restricts losses to 11.03% and 13.84%, respectively. Table 12 in Appendix G.3 reflects similar findings on the LongBench benchmark. These results validate HiF4 as a uniquely robust format for end-to-end inference at ultra-low bit widths.

Real Wall-Clock Gains. We evaluate the end-to-end hardware latency of Qwen3-30B-A3B on Ascend NPUs, using a batch size of 64, a prefill sequence length of 2K, and 256 generated tokens. For HiF8, we profile the prefill and decode latencies using native kernels. As shown in Table 6, HiF8 achieves speedups of $1.17\times$ in prefill and $1.47\times$ in decoding over the BF16 baseline, while also outperforming MXFP8. For the 4-bit regime, although the HiF4 hardware architecture and quantization logic are fully specified for accuracy evaluation, highly optimized production kernels are currently under development. To provide empirical hardware validation, we benchmark MXFP4 on Ascend NPUs to establish a strict lower bound for HiF4 performance. Specifically, MXFP4 W4A4 yields $1.22\times$ and $2.14\times$ speedups for prefill and decode, respectively. Because HiF4 features a streamlined compute flow and native hardware co-design (Luo et al., 2026), we project its final wall-clock speedups will confidently exceed these MXFP4 baselines.

6 Conclusions

In this work, we present a systematic evaluation of the HiFloat formats for LLM inference, benchmarking them against industry standards such as MXFP and NVFP4. Our assessment across weights, activations, and KV cache reveals that numerical representation is the primary determinant of model resilience. We identify a clear performance bifurcation in the 8-bit regime: while floating-point formats like MXFP8 and HiF8 provide the dynamic range necessary for fluctuating activations, the uniform density of INT8 remains superior for weight quantization. This suggests that for weights, avoiding the exponent-waste of floating-point formats is more beneficial than having an expansive dynamic range. However, as precision shifts to the 4-bit regime, the limitations of uniform spacing become a fundamental bottleneck, leading to a catastrophic collapse in integer formats. In this constrained space, the hierarchical structures of NVFP4 and HiF4 prove essential for isolating outliers while maintaining local precision. Notably, HiF4 preserves over 96.5% of the BF16 baseline on Qwen3-8B and 97.0% on openPangu-7B using only RTN for W4A4; these results improve further when integrated with SmoothQuant or SVDQuant. When augmented with KV cache quantization, HiF4 consistently achieves superior results across all evaluated 4-bit formats, establishing it as a robust standard for low precision inference.

References

Eduardo Alvarez, Omri Almog, Eric Chung, Simon Layton, Dusan Stosic, Ronny Krashinsky, and Kyle Aubrey. 2025. Introducing NVFP4 for efficient and accurate low-precision inference. NVIDIA Technical Blog.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *Preprint*, arXiv:2404.00456.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, and 1 others. 2024. Longbench: A comprehensive benchmark for multilingual long-context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15509–15527.

Hanting Chen, Yasheng Wang, Kai Han, Dong Li, Lin Li, Zhenni Bi, Jinpeng Li, Haoyu Wang, Fei Mi, Mingjian Zhu, Bin Wang, Kaikai Song, Yifei Fu, Xu He, Yu Luo, Chong Zhu, Quan He, Xueyu Wu, Wei He, and 5 others. 2025. Pangu embedded: An efficient dual-system llm reasoner with metacognition. *Preprint*, arXiv:2505.22375.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind

Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, and 1 others. 2024. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11198–11201.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. 2025. Mme: A comprehensive evaluation benchmark for multimodal large language models. *Preprint*, arXiv:2306.13394.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. 2024. Hallusionbench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. *Preprint*, arXiv:2310.14566.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2025. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *Preprint*, arXiv:2401.18079.

IEEE Standards Association. 2019. *IEEE 754-2019: IEEE Standard for Floating-Point Arithmetic*. Institute of Electrical and Electronics Engineers (IEEE), New York, NY, USA.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. 2024. Squeezellm: Dense-and-sparse quantization. In *International Conference on Machine Learning*, pages 23901–23923. PMLR.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. 2023a. Seed-bench: Benchmarking multimodal llms with generative comprehension. *Preprint*, arXiv:2307.16125.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. 2024. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. *arXiv preprint arXiv:2411.05007*.

Xing Li, Zeyu Xing, Yiming Li, Linping Qu, Hui-Ling Zhen, Wulong Liu, Yiwu Yao, Sinno Jialin Pan, and Mingxuan Yuan. 2025. Kvtuner: Sensitivity-aware layer-wise mixed-precision kv cache quantization for efficient and nearly lossless llm inference. *arXiv preprint arXiv:2502.04420*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 292–305, Singapore. Association for Computational Linguistics.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2023. Awq: Activation-aware weight quantization for llm compression and acceleration. In *Proceedings of the 6th MLSys Conference*.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. 2025. Spinquant: Llm quantization with learned rotations. *Preprint*, arXiv:2405.16406.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen (Henry) Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024. Kivi: a tuning-free asymmetric 2bit quantization for kv cache. In *International Conference on Machine Learning*. PMLR.

Yuanyong Luo, Jing Huang, Yu Cheng, Ziwei Yu, Kaihua Tang, Xinda Ma, Xin Wang, Anping Tong, Guipeng Hu, Yun Xu, Mehran Taghian, Peng Wu, Guanglin Li, Yunke Peng, Tianchi Hu, Minqi Chen, Michael Bi Mi, Hu Liu, Xiping Zhou, and 3 others. 2026. Hifloat4 format for language model inference. *Preprint*, arXiv:2602.11287.

Yuanyong Luo, Zhongxing Zhang, Richard Wu, Hu Liu, Ying Jin, Kai Zheng, Minmin Wang, Zhanying He, Guipeng Hu, Luyao Chen, Tianchi Hu, Junsong Wang, Minqi Chen, Mikhaylov Dmitry, Korviakov Vladimir, Bobrin Maxim, Yuhao Hu, Guanfu Chen, and Zeyi Huang. 2024. Ascend hifloat8 format for deep learning. *Preprint*, arXiv:2409.16626.

Yuexiao Ma, Huixia Li, Xiawu Zheng, Feng Ling, Xuefeng Xiao, Rui Wang, Shilei Wen, Fei Chao, and

Rongrong Ji. 2024. Affinequant: Affine transformation quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Qwen Team. 2026. Qwen3.5: Towards native multimodal agents.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, and 14 others. 2023. Microscaling data formats for deep learning. *Preprint*, arXiv:2310.10537.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2024. Omniquant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations*.

Yuantian Shao, Peisong Wang, Yuanteng Chen, Chang Xu, Zhihui Wei, and Jian Cheng. 2025. Block rotation is all you need for mxfp4 quantization. *Preprint*, arXiv:2511.04214.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. 2023. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. In *International Conference on Learning Representations*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demeter, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38154–38168. PMLR.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Yu Zhang, Mingzi Wang, Lancheng Zou, Wulong Liu, Hui-Ling Zhen, Mingxuan Yuan, and Bei Yu. 2024. Mixpe: Quantization and hardware co-design for efficient llm inference. *arXiv preprint arXiv:2411.16158*.

Pengxiang Zhao and Xiaoming Yuan. 2025. GANQ: GPU-adaptive non-uniform quantization for large language models. In *International Conference on Machine Learning*. PMLR.

A MXFP Transformation Logic

The MX formats support a diverse family of configurations by varying the bit-width and numerical interpretation of the element data type $\mathcal{E}$, as summarized in Table 7.

Mapping a high-precision tensor to the MX domain involves a two-stage transformation comprising scale estimation and element-wise quantization. In standard configurations, for a given micro-block $\mathbf{x} \in \mathbb{R}^k$,the shared scale $s = 2^e \in \mathcal{S}$ is derived to align the block's dynamic range with the representable limits of the element data type $\mathcal{E}$. Let $q_{\max}$ denote the maximum representable magnitude of $\mathcal{E}$ (e.g., $q_{\max} = 6.0$ for the E2M1 configuration of MXFP4). The exponent e is determined by rounding the required scaling factor to the upward-adjacent power-of-two, constrained by the bit-width of the E8M0 format:

$$e = \text{clip}\left(\left\lceil \log_2 \frac{\max_i |x_i|}{q_{\max}} \right\rceil, -127, 127\right), \quad (3)$$

where $\text{clip}(v, a, b) = \max(a, \min(v, b))$. Subsequently, the individual elements x_i are normalized and mapped to the codewords $q_i \in \mathcal{E}$ via:

$$q_i = \text{round}_{\mathcal{E}}\left(\text{clip}\left(\frac{x_i}{s}, -q_{\max}, q_{\max}\right)\right). \quad (4)$$

Here, $\text{round}_{\mathcal{E}}(\cdot)$ denotes the rounding operator onto the set of representable values in $\mathcal{E}$, typically utilizing round-to-nearest (RTN) logic. The dequantized reconstruction is subsequently recovered as $\hat{x}_i = 2^e \cdot q_i$.

B NVFP4 Transformation Logic

The element data type for NVFP4 is defined as E2M1, as summarized in Table 8.

Mapping a tensor $\mathcal{X} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_n}$ in high-precision to the NVFP4 domain involves a hierarchical three-stage transformation: per-tensor scale estimation, per-block scale estimation, and element-wise quantization. First, a per-tensor scale s_2 is derived to normalize the overall dynamic range:

$$s_2 = \frac{\max(|\mathcal{X}|)}{v_{\max}}, \quad (5)$$

where $v_{\max} = \text{E4M3}_{\max} \times \text{E2M1}_{\max} = 448 \times 6$. The tensor is pre-scaled as $\tilde{\mathcal{X}} = \mathcal{X}/s_2$. Then, for each micro-block $\tilde{\mathbf{x}} \in [-v_{\max}, v_{\max}]^k$ within $\tilde{\mathcal{X}}$, a per-block scale s_1 is calculated and projected onto the E4M3 codeword set:

$$s_1 = \text{round}_{\text{E4M3}}\left(\frac{\max_i |\tilde{x}_i|}{6}\right). \quad (6)$$

Name	Block Size (k)	Scale ($\mathcal{S}$)	Element ($\mathcal{E}$)
MXFP8	32	E8M0	E5M2 / E4M3
MXFP6	32	E8M0	E3M2 / E2M3
MXFP4	32	E8M0	E2M1
MXINT8	32	E8M0	INT8

Table 7: Microscaling (MX) Configurations.

Configurations	Value
Block Size (k)	16
Per-block Scale ($\mathcal{S}_1$)	E4M3
Per-tensor Scale ($\mathcal{S}_2$)	FP32
Element ($\mathcal{E}$)	E2M1

Table 8: NVFP4 Configurations.

Finally, the normalized elements are mapped to the E2M1 codewords via:

$$q_i = \text{round}_{\text{E2M1}}\left(\frac{\tilde{x}_i}{s_1}\right). \quad (7)$$

The dequantized reconstruction is subsequently recovered as $\hat{x}_i = s_1 \cdot s_2 \cdot q_i$. By introducing a high-precision FP32 per-tensor scaling factor s_2 calibrated to $v_{\max}$, the NVFP4 format theoretically eliminates clipping errors during the quantization process. Specifically, s_2 ensures the pre-scaled elements $\tilde{x}_i$ are within the joint representable range of the E4M3 micro-scales and E2M1 codewords, while s_1 subsequently maps the local distribution to the dynamic range of the E2M1 elements. Consequently, the quantization noise in NVFP4 is dominated by rounding error, which arises from the projection of values onto the discrete grids of $\mathcal{S}_1$ and $\mathcal{E}$.

C HiF8 Transformation Logic

To transform a high-precision input $x \in \mathbb{R}$ into HiF8, the process executes a three-part pipeline. First, the binary exponent e of the input magnitude is determined by the floor of the base-2 logarithm:

$$e = \lfloor \log_2(|x| + \epsilon) \rfloor, \quad (8)$$

where ϵ is a small constant (e.g., 2^{-14} for FP16 or 2^{-45} for FP32) introduced for numerical stability and to define the underflow boundary. This e serves as the scaling factor that determines the dynamic range interval $[2^e, 2^{e+1})$ in which $|x|$ resides.

Unlike static floating-point formats, HiF8 utilizes a magnitude-dependent precision allocation. The quantity of mantissa bits n_m is governed by the magnitude of the exponent $|e|$ in accordance

with the bit layout design presented in Figure 1. This relationship is formally characterized by the following definition:

$$n_m = \begin{cases} 3, & \text{if } |e| \leq 3; \\ 2, & \text{if } 3 < |e| \leq 7; \\ 1, & \text{if } 7 < |e| \leq 15; \\ 0, & \text{if } |e| > 15. \end{cases} \tag{9}$$

The mantissa value m represents the fractional component of the value, such that $|x| = (1+m) \cdot 2^e$ for normal numbers.

To discretize m using n_m bits, we scale $|x|$ by the quantization step size $\Delta = 2^{e-n_m}$:

$$\tilde{x} = \frac{|x|}{2^{e-n_m}}. \tag{10}$$

We then apply the RTN logic to find the nearest discrete state $\hat{x}$:

$$\hat{x} = \lfloor \tilde{x} + 0.5 \rfloor. \tag{11}$$

The quantized mantissa value is reconstructed as $1+m_q = \hat{x} \cdot 2^{-n_m}$, where m_q represents the fractional mantissa stored in the bit-field.

Following Equation 1, the reconstructed value x_{hif8} is obtained by projecting the encoding back to the floating-point domain and applying the sign bit $(-1)^s = \text{sgn}(x)$:

$$x_{\text{hif8}} = (-1)^s \cdot \hat{x} \cdot 2^{e-n_m}. \tag{12}$$

The complete pseudocode for the HiF8 adaptive quantization and dequantization logic is summarized in Algorithm 1.

D Scaling HiF8 Quantization

The standard HiF8 format reserves significant representational capacity for extreme values that are rarely present in typical distributions (Figure 3). To address this, a scaling mechanism is proposed to better align the tensor's dynamic range with the high-density regions of the HiF8 encoding.

For a vector $\mathbf{x} \in \mathbb{R}^n$, a scaling factor s is calculated as:

$$s = \frac{K}{\max_i |\mathbf{x}_i| + \epsilon}, \tag{13}$$

where K is a predefined hyperparameter representing the target maximum magnitude, and $\epsilon > 0$ is a small constant to ensure numerical stability. The vector $\mathbf{x}$ is then scaled to produce the normalized representation $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = s \cdot \mathbf{x}. \tag{14}$$

Algorithm 1: HiF8 Adaptive Quantization

Input: Input x, small constant ϵ
Output: Quantized-dequantized x_{hif8}

$s \leftarrow \text{sign}(x)$;
$e \leftarrow \lfloor \log_2(x_{|x|} + \epsilon) \rfloor$;
`// Adaptive Mantissa Selection`
if $|e| \leq 3$ **then**
  $n_m \leftarrow 3$;
else if $|e| \leq 7$ **then**
  $n_m \leftarrow 2$;
else if $|e| \leq 15$ **then**
  $n_m \leftarrow 1$;
else
  $n_m \leftarrow 0$;
`// Quantization-Dequantization`
$\hat{x} \leftarrow \lfloor (|x|/2^{e-n_m}) + 0.5 \rfloor$;
$x_{\text{hif8}} \leftarrow s \cdot (\hat{x} \cdot 2^{e-n_m})$;
return x_{hif8};

The normalized components of $\tilde{\mathbf{x}}$ are subsequently mapped to the nearest HiF8 representational levels. During inference, it is recovered by dividing the scaling factor back. This adjustment seeks to reduce the bit waste identified in Section 3 and recover the precision necessary for high-fidelity compression.

E HiF4 Transformation Logic

Given a tensor $\mathcal{X} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_n}$, HiF4 quantization can be performed along any chosen dimension. Assume the k-th dimension is selected such that $d_k \equiv 0 \pmod{64}$, and let $b = d_k/64$. Without loss of generality, we consider the vector $\mathbf{x} \in \mathbb{R}^{d_k}$ by fixing all indices of $\mathcal{X}$ except for the k-th dimension, denoted as $\mathbf{x} = \mathcal{X}_{i_1,\ldots,:,\ldots,i_n}$. Subsequently, HiF4 quantization is applied to each vector in a block-based hierarchical scaling scheme.

First, the vector $\mathbf{x}$ is partitioned into blocks of size 64 by reshaping it into a tensor $\mathbf{X} \in \mathbb{R}^{b \times 8 \times 2 \times 4}$. For each block, we compute the maximum absolute value along each of the three sub-dimensions sequentially:

$$\mathbf{A}_3 = \max_k |\mathbf{X}_{h,i,j,k}| \in \mathbb{R}_+^{b \times 8 \times 2}, \tag{15a}$$

$$\mathbf{A}_2 = \max_j (\mathbf{A}_3)_{h,i,j} \in \mathbb{R}_+^{b \times 8}, \tag{15b}$$

$$\mathbf{A}_1 = \max_i (\mathbf{A}_2)_{h,i} \in \mathbb{R}_+^{b}, \tag{15c}$$

where $|\cdot|$ denotes the absolute value operator. Sub-

sequently, we derive a set of hierarchical scaling factors based on $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$, progressing from coarse to fine granularity.

For every 64 elements, we calculate a block-wise scaling factor $\mathbf{S}_1$ based on the normalized $\mathbf{A}_1$, which is then represented in the unsigned E6M2 format:

$$\tilde{\mathbf{A}}_1 = \text{clip}\left(\frac{\mathbf{A}_1}{7}, q_{\min}, q_{\max}\right), \tag{16a}$$

$$\mathbf{E}_1 = \left\lfloor \log_2 \tilde{\mathbf{A}}_1 \right\rfloor, \tag{16b}$$

$$\mathbf{M}_1 = \text{round}\left(\frac{\tilde{\mathbf{A}}_1}{2^{\mathbf{E}_1 - 2}}\right), \tag{16c}$$

$$\mathbf{S}_1 = \mathbf{M}_1 \times 2^{\mathbf{E}_1 - 2}, \tag{16d}$$

where $q_{\min} = 2^{-48}$ and $q_{\max} = 1.5 \times 2^{15}$ are the extreme values of the E6M2 format. Within each block, a sub-block scaling factor $\mathbf{S}_2$ is calculated every 8 elements based on $\mathbf{A}_2$ and $\mathbf{S}_1$. This factor is represented as a 1-bit exponent according to the following logic:

$$\tilde{\mathbf{A}}_2 = \text{clip}\left(\frac{\mathbf{A}_2}{\mathbf{S}_1}, 0, 4\right), \tag{17a}$$

$$\mathbf{E}_2 = \left\lfloor \frac{\tilde{\mathbf{A}}_2}{4} \right\rfloor \in \{0, 1\}, \tag{17b}$$

$$\mathbf{S}_2 = 2^{\mathbf{E}_2}. \tag{17c}$$

Within each sub-block, a scaling factor $\mathbf{S}_3$ is calculated every 4 elements based on $\mathbf{A}_3$, $\mathbf{S}_1$, and $\mathbf{S}_2$. This factor is also represented as a 1-bit exponent according to the following logic:

$$\tilde{\mathbf{A}}_3 = \text{clip}\left(\frac{\mathbf{A}_3}{\mathbf{S}_1 \times \mathbf{S}_2}, 0, 2\right), \tag{18a}$$

$$\mathbf{E}_3 = \left\lfloor \frac{\tilde{\mathbf{A}}_3}{2} \right\rfloor \in \{0, 1\}, \tag{18b}$$

$$\mathbf{S}_3 = 2^{\mathbf{E}_3}. \tag{18c}$$

Finally, the magnitude of each element is normalized by the hierarchical scaling factors $\mathbf{S}_1$, $\mathbf{S}_2$, and $\mathbf{S}_3$. The resulting values are then quantized into the E1M2 format as follows:

$$\tilde{\mathbf{X}} = \text{clip}\left(\frac{|\mathbf{X}|}{\mathbf{S}_1 \times \mathbf{S}_2 \times \mathbf{S}_3}, 0, 1.75\right), \tag{19a}$$

$$\hat{\mathbf{X}} = \left\lfloor \tilde{\mathbf{X}} \times 2^2 + 0.5 \right\rfloor \in \{0, 1, \ldots, 7\}. \tag{19b}$$

In summary, each element of $\mathbf{X}$ is represented by the tuple $(\text{sign}, \mathbf{E}_1, \mathbf{M}_1, \mathbf{E}_2, \mathbf{E}_3, \hat{\mathbf{X}})$. The corresponding dequantized value $\mathbf{X}_{\text{hif4}}$ is reconstructed as follows:

$$\mathbf{X}_{\text{hif4}} = \text{sign} \times \mathbf{M}_1 \times 2^{\mathbf{E}_1 + \mathbf{E}_2 + \mathbf{E}_3 - 4} \times \hat{\mathbf{X}}. \tag{20}$$

This formulation allows the hardware to reconstruct the value using simple integer addition of the exponent components $(\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3)$ followed by a bit-shift, avoiding the need for expensive floating-point multiplications for the scaling factors.

F Quantization SQNR Comparisons

F.1 Weight Quantization

We evaluate the impact of reduced precision on model fidelity by analyzing the layer-wise SQNR for Qwen3-8B across all 36 decoder layers. Figures 7 and 8 contrast the representational efficiency of investigated formats in 8-bit and 4-bit regimes.

As shown in Figure 7, in the 8-bit regime, all formats maintain high fidelity. INT8 consistently achieves the highest SQNR, outperforming floating-point formats by approximately 5–10 dB. This indicates that for 8-bit weights, the uniform density of an integer format is more effective than an expansive dynamic range. The bit-waste incurred by exponent bits in MXFP8 or HiF8 results in a coarser representation for the bulk of the weight distribution compared to the fine-grained, uniform steps of INT8. By employing the per-channel scaling mechanism detailed in Appendix D with $K = 16$, HiF8 (scaled) achieves performance comparable to that of MXFP8.

As illustrated in Figure 8, the transition to 4-bit precision marks a shift where uniform spacing becomes a bottleneck, causing INT4 SQNR to collapse, often falling below 15 dB. In this constrained regime, the hierarchical structure of HiF4 and NVFP4 proves essential. HiF4 consistently yields the highest SQNR across all components (W_q, W_k, W_v, W_o, and MLP layers). By utilizing multi-level scaling to isolate outliers while preserving local precision, HiF4 demonstrates superior structural resilience, validating its efficacy for high-fidelity inference at ultra-low bit-widths.

F.2 Activation Quantization

To assess the impact of reduced precision on model activations, we evaluate the layer-wise SQNR for Qwen3-8B with Wikitext-2 dataset. Unlike weights, activation distributions are dynamic and often characterized by prominent outliers, making them significantly more sensitive to quantization

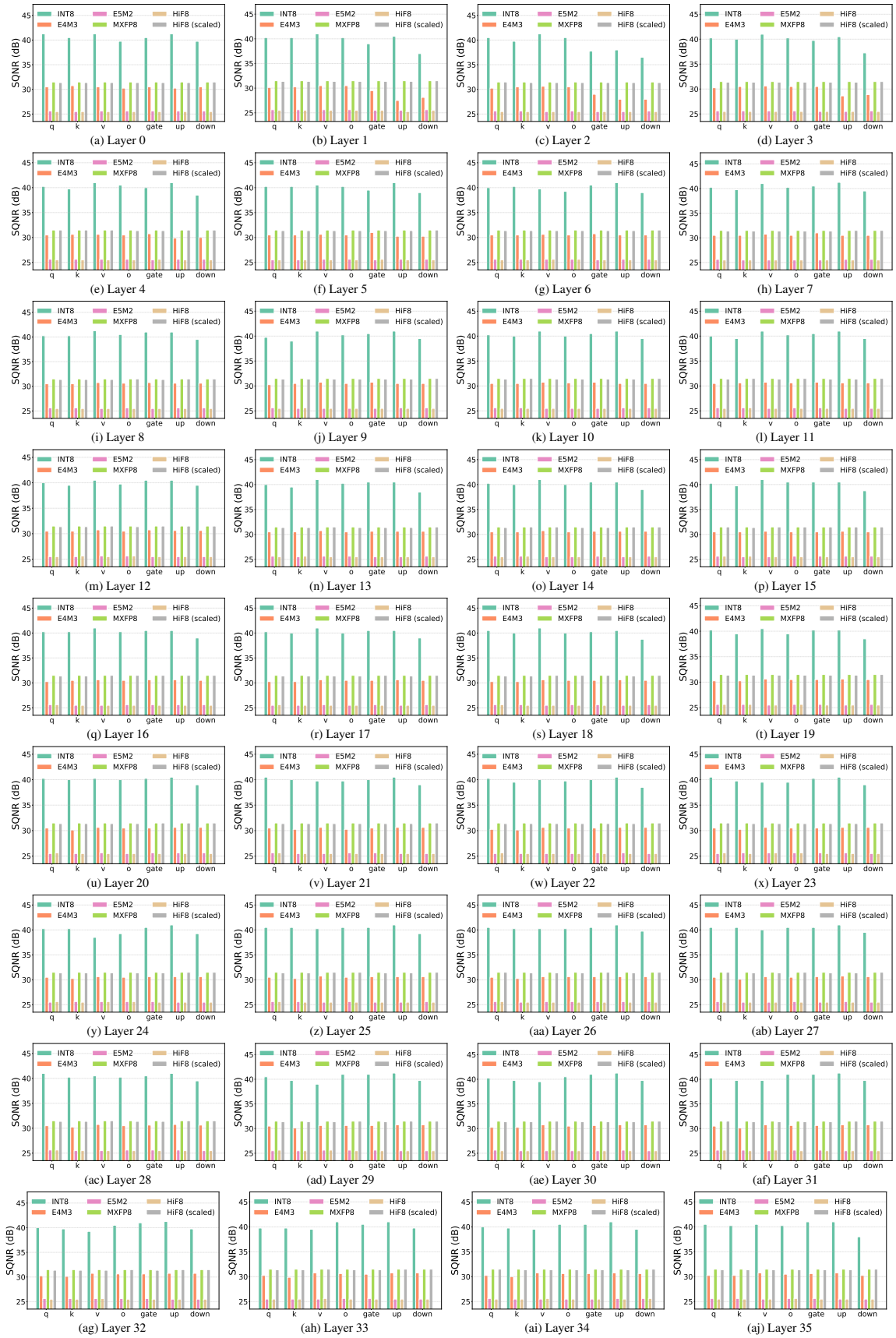


Figure 7: Weight Quantization SQNR Comparison for Qwen3-8B across 8-bit Formats.

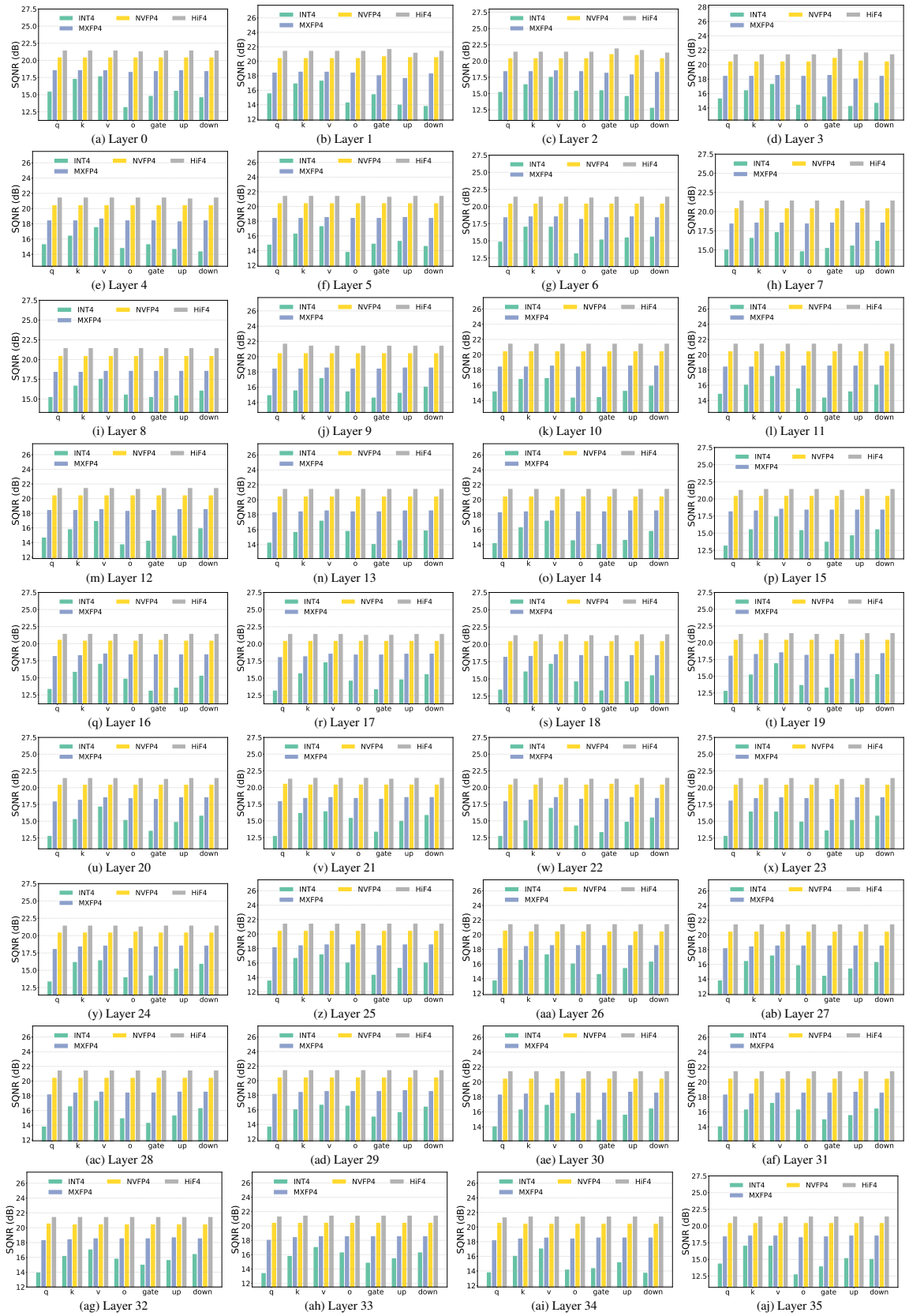


Figure 8: Weight Quantization SQNR Comparison for Qwen3-8B across 4-bit Formats.

Figure 9: Activation Quantization SQNR Comparison for Qwen3-8B across 8-bit Formats.

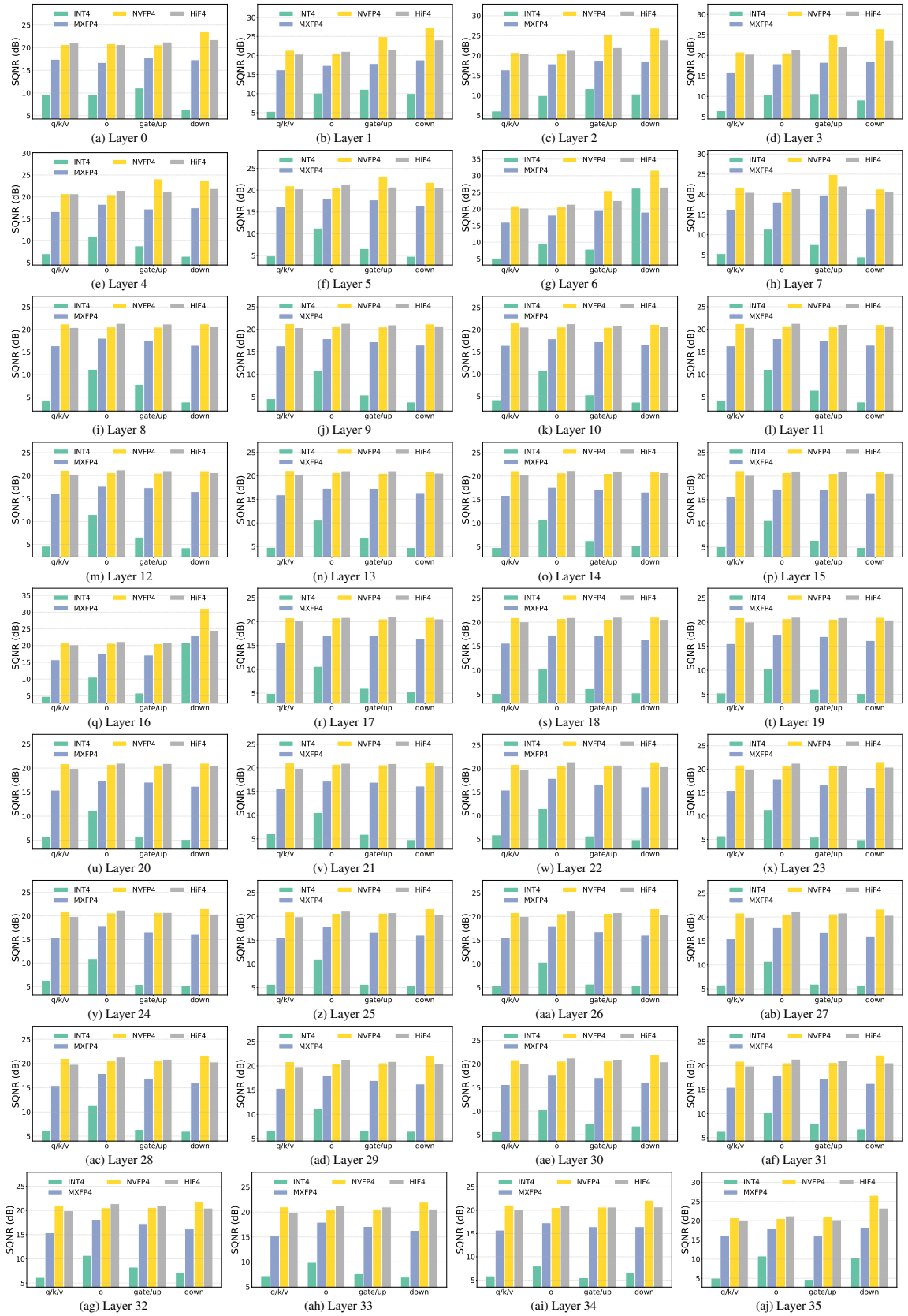
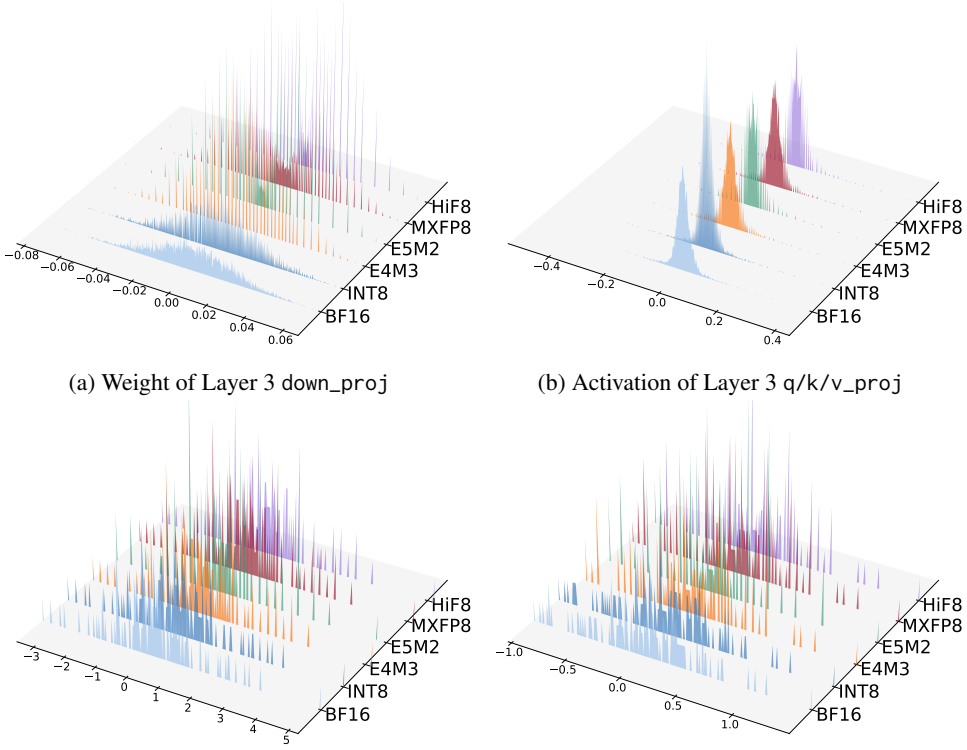


Figure 10: Activation Quantization SQNR Comparison for Qwen3-8B across 4-bit Formats.



(a) Weight of Layer 3 `down_proj`

(b) Activation of Layer 3 `q/k/v_proj`

(c) Key State of Layer 3

(d) Value State of Layer 3

Figure 11: Distributional Impact of 8-bit Quantization Formats on Qwen3-8B Layer 3 Components (z-axis: density).

bit-widths. Figures 9 and 10 illustrate the activation fidelity for the 8-bit and 4-bit regimes, respectively.

As shown in Figure 9, in the 8-bit regime, the performance hierarchy differs markedly from the weight analysis. While INT8 remains competitive in certain layers, floating-point formats, specifically HiF8 and MXFP8, demonstrate superior stability across the 36 decoder layers. Across most components (W_q, W_k, W_v, W_o, and MLP layers), HiF8 and MXFP8 maintain SQNR levels between 30 dB and 35 dB. The expanded dynamic range afforded by the exponent bits in these formats allows them to better accommodate the outlier spikes common in LLM activations. With $K = 4$, HiF8 (scaled) achieves performance comparable to that of MXFP8. In contrast, INT8 shows higher variability, particularly in deeper layers where activation magnitudes fluctuate more aggressively, leading to sporadic drops in fidelity compared to the more consistent floating-point baselines.

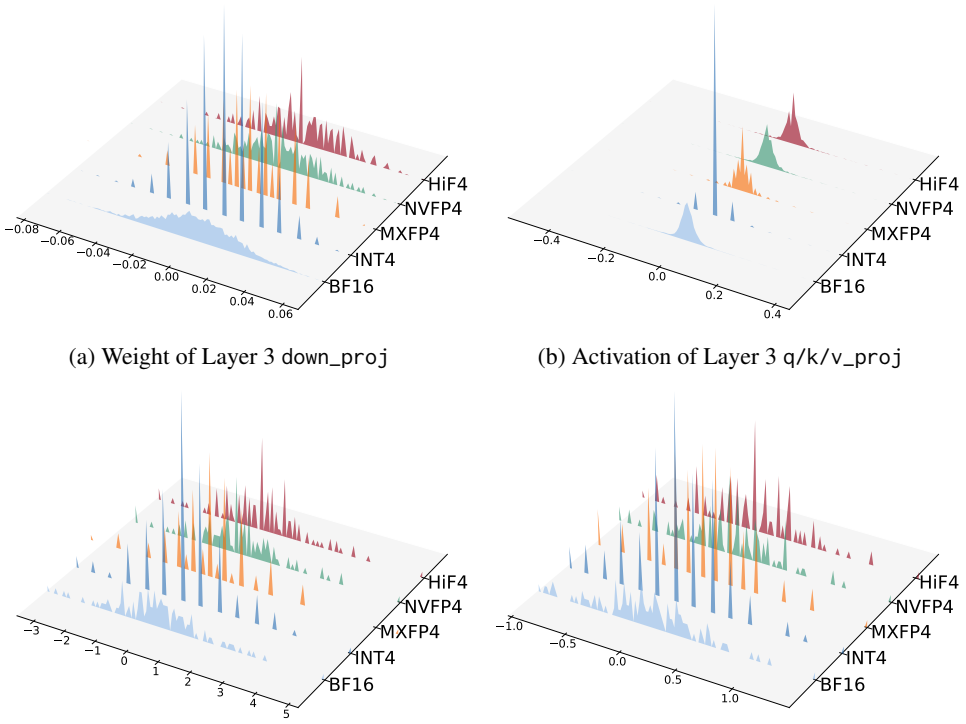
As illustrated in Figure 10, the transition to 4-bit precision reveals a near-total collapse for uniform integer quantization. INT4 SQNR values plummet below 10 dB in nearly all layers, and in several instances (such as the activations of W_q, W_k, and W_v in Layer 1-3), they drop to 5 dB. This confirms that the constraint bit budget and uniform spacing of INT4 are fundamentally incapable of representing the high-variance distributions of activations. In this low precision setting, hierarchical formats become mandatory for functional integrity. HiF4 and NVFP4 consistently achieve the highest SQNR.

G Distributional Analysis

G.1 8-bit Formats

Figure 11a illustrates the first channel of the Qwen3-8B Layer 3 `down_proj` weight distribution, which is characterized by a symmetric, bell-shaped density concentrated in a narrow range. Within this bounded space, INT8 provides the most faithful reconstruction because its uniform spacing maximizes representational density where the signal is most prevalent. In contrast, MXFP8 and HiF8 allocate a portion of their bit-budget to exponent ranges that remain largely unpopulated by weight values. This exponent-waste results in a coarser quantization grid near the center of the distribution, explaining why floating-point formats exhibit higher quantization noise for static weights.

The activation input to Layer 3 `q/k/v_proj` (Figure 11b) presents a sharp contrast, exhibiting significant asymmetry and heavy-tailed outliers. While INT8 utilizes a zero-point to address the global shift, its uniform nature forces a massive increase in step size to encompass extreme outliers, leading to a resolution collapse near zero. Con-



(a) Weight of Layer 3 `down_proj`

(b) Activation of Layer 3 `q/k/v_proj`

(c) Key State of Layer 3

(d) Value State of Layer 3

Figure 12: Distributional Impact of 4-bit Quantization Formats on Qwen3-8B Layer 3 Components (z-axis: density).

Figure 13: Visualization of Quantization Resolution within Blocks across Various 4-bit Formats. Left: Global overview of the 64-element activation vector, with the red box indicating the zoomed sub-region. Right: Element-wise magnification of indices 16–31.

versely, the logarithmic spacing of MXFP8 and HiF8 allows the grid to maintain high precision in the dense region near the origin while simultaneously providing the dynamic range necessary to capture high-magnitude activations. This adaptability confirms that for dynamic activations, the structural flexibility of floating-point formats outweighs the raw density of integer formats.

Figures 11c and 11d visualize the contrasting behaviors of Key and Value tensors. The Key states display a complex, multi-modal distribution where the hierarchical flexibility of HiF8 and MXFP8 is essential for capturing shifting peaks. The Value states appear more stable and bounded, similar to weight distributions but with slightly more variance. This stability explains why the performance gap between uniform and hierarchical formats is narrower for Keys than for Values; when a distribution is bounded, the expansive range of floating formats is less critical, allowing denser uniform formats to remain highly competitive.

G.2 4-bit Formats

The distribution in Figure 12a highlights the collapse of uniform quantization at ultra-low bit-widths. Because INT4 is restricted to only 16 representable levels, it must stretch its uniform grid to cover the entire weight range, resulting in extremely coarse binning and a total loss of local feature resolution. MXFP4 offers an improvement over the integer baseline but remains limited by its reliance on power-of-two scaling and a larger 32-element block size. This increased granularity prevents the format from adapting to local variations, causing the quantization intervals to widen signifi-

Models	Bit	Methods	Format	Acc (↑)						Overall	
				HellaSwag	MMLU	Arc-C	Arc-E	BoolQ	PIQA	Avg.↑	Δ(%)↓
Qwen3.5-9B	-	-	BF16	0.5836	0.7856	0.5410	0.8148	0.8933	0.7927	1.0000	-
	W4A4	RTN	MXFP	0.5534	0.7172	0.4889	0.7980	0.8590	**0.7878**	0.9531	4.69
			NVFP	**0.5714**	0.7619	0.5316	0.7992	0.8379	0.7851	0.9719	2.81
			HiF	0.5693	**0.7639**	**0.5384**	**0.8199**	**0.8722**	0.7856	**0.9860**	**1.40**

Table 9: Performance Evaluation of Weight-Activation Quantization on Qwen3.5-9B. **Bold** indicates best performance per configuration; Δ shows degradation from BF16 baseline.

Models	Bit	Methods	Format	MME (↑)	Acc (↑)			Overall	
					SEEDBench	HallusionBench	POPE	Avg.↑	Δ(%)↓
Qwen3-VL-8B-Instruct	-	-	BF16	1756.26	76.88	58.36	88.05	1.0000	-
	W4A4	RTN	MXFP	1667.30	73.70	55.94	87.88	0.9742	2.58
			NVFP	1652.73	**76.84**	56.15	87.74	0.9885	1.15
			HiF	**1706.73**	76.59	**58.15**	**88.53**	**0.9999**	**0.01**

Table 10: Performance Evaluation of Weight-Activation Quantization on Qwen3-VL-8B-Instruct. **Bold** indicates best performance per configuration; Δ shows degradation from BF16 baseline.

Models	SmoothQuant WA Precision	Data Format	QKV Quant. Precision	PPL (↓)		Acc (↑)				EM (↑)	Accuracy Loss Δ(%)↓
				Wikitext	C4	HellaSwag	MMLU	Arc-C	MATH500	GSM8K	
Qwen3-8B	-	BF16	-	9.72	15.43	0.5711	0.7302	0.5546	78.4	0.8787	-
	W8A8	INT8	-	9.59	15.29	0.5720	0.7276	0.5478	79.2	0.0334	13.48
			Q16KV8	9.58	15.29	0.5724	0.7292	0.5503	77.4	0.0326	13.70
			QKV8	9.59	15.30	0.5712	0.7287	0.5563	80.2	0.0326	13.10
		MXFP8	-	9.75	15.48	0.5695	0.7277	0.5606	78.6	0.8825	**-0.07**
			Q16KV8	9.77	15.48	0.5696	0.7264	0.5538	79.8	0.8832	**-0.08**
			QKV8	9.77	15.49	0.5707	0.7265	0.5580	79.0	0.8886	**-0.15**
		HiF8	-	9.79	15.47	0.5680	0.7227	0.5538	79.8	0.8787	*0.13*
			Q16KV8	9.81	15.50	0.5684	0.7258	0.5512	82.8	0.8741	*-0.29*
			QKV8	9.85	15.54	0.5658	0.7236	0.5435	81.0	0.8772	*0.39*
	W4A4	INT4	-	6.1e2	6.0e2	0.2651	0.2344	0.1903	1.4	0.0015	1478.49
			Q16KV4	6.3e2	6.3e2	0.2649	0.2333	0.2090	1.4	0.0008	1535.22
			QKV4	7.9e2	8.0e2	0.2591	0.2415	0.1937	0.8	0.0008	1928.26
		MXFP4	-	10.90	17.46	0.5352	0.6700	0.4923	74.8	0.8188	8.92
			Q16KV4	11.08	17.67	0.5267	0.6587	0.4821	74.2	0.7688	11.00
			QKV4	11.48	18.27	0.5120	0.6422	0.4761	68.0	0.7392	14.60
		NVFP4	-	10.12	16.21	0.5491	0.7079	0.5324	80.6	0.8711	**2.59**
			Q16KV4	10.20	16.28	0.5473	0.7041	0.5205	81.2	0.8484	*3.46*
			QKV4	10.28	16.41	0.5446	0.6988	0.5128	75.2	0.8522	*5.10*
		HiF4	-	10.27	16.31	0.5566	0.7050	0.5478	77.2	0.8552	*3.25*
			Q16KV4	10.23	16.31	0.5563	0.7051	0.5435	77.8	0.8582	**3.15**
			QKV4	10.32	16.47	0.5515	0.6939	0.5478	78.8	0.8309	**3.92**
openPangu-7B	-	BF16	-	34.95	57.18	0.4504	0.5667	0.3225	89.8	0.5049	-
	W8A8	INT8	-	35.79	56.93	0.4503	0.5882	0.3208	86.6	0.0250	13.90
			Q16KV8	35.86	57.11	0.4512	0.5894	0.3294	88.6	0.0281	13.13
			QKV8	35.91	57.04	0.4482	0.5866	0.3200	91.0	0.0227	13.49
		MXFP8	-	35.01	56.90	0.4439	0.5799	0.3319	88.4	0.5034	**-0.32**
			Q16KV8	36.03	58.37	0.4448	0.5454	0.3140	89.4	0.5095	**1.76**
			QKV8	36.31	58.89	0.4452	0.5467	0.3285	89.4	0.5140	**1.19**
		HiF8	-	36.67	59.38	0.4481	0.5783	0.3396	90.2	0.5178	*-0.15*
			Q16KV8	40.23	63.18	0.4433	0.5642	0.3200	87.0	0.4966	*4.74*
			QKV8	40.46	63.50	0.4433	0.5407	0.3140	85.8	0.5155	*5.42*
	W4A4	INT4	-	6.4e2	7.3e2	0.3124	0.2752	0.2201	7.2	0.0023	459.03
			Q16KV4	1.9e3	2.2e3	0.2796	0.2504	0.2048	2.2	0.0000	1375.22
			QKV4	2.3e3	2.6e3	0.2860	0.2406	0.2210	0.8	0.0008	1603.48
		MXFP4	-	45.55	62.67	0.4296	0.5236	0.3148	83.0	0.4488	10.46
			Q16KV4	81.51	88.81	0.4027	0.3296	0.3063	46.0	0.1698	51.59
			QKV4	92.62	97.77	0.3924	0.3448	0.2927	35.2	0.1077	62.39
		NVFP4	-	38.68	58.18	0.4372	0.4855	0.3131	89.4	0.5140	*4.46*
			Q16KV4	44.67	65.68	0.4332	0.4755	0.3311	84.8	0.3844	*12.77*
			QKV4	45.38	65.38	0.4299	0.4588	0.3012	79.8	0.3639	*16.21*
		HiF4	-	39.69	60.27	0.4434	0.5426	0.3055	88.6	0.5262	**3.88**
			Q16KV4	43.44	61.43	0.4361	0.4563	0.3063	84.2	0.4466	**11.03**
			QKV4	44.76	61.51	0.4368	0.4190	0.2944	83.0	0.4246	**13.84**

Table 11: Performance Evaluation of KV Cache and Attention Quantization with Various Formats based on the Weight and Activation Quantization using SmoothQuant. Q16KV8 and Q16KV4 denote KV cache quantization, while QKV8 and QKV4 simulate attention quantization. Accuracy Loss denotes the average accuracy loss compared with the BF16 baselines. **Bold red** and *italic blue* indicate the best and second best settings, respectively.

cantly as values deviate from zero. In contrast, the hierarchical structures of HiF4 and NVFP4 allow for a much denser concentration of representable states near the bulk of the distribution, preserving the original signal's bell-shaped contour even under stringent constraints.

The activation analysis in Figure 12b demonstrates that managing dynamic outliers is the primary challenge for 4-bit inference. NVFP4 achieves the most faithful reconstruction of the original signal, due to its FP32 global scaling factor and fine-grained 16-element block size. This lo-

Models	SmoothQuant WA Precision	Data Format	QKV Quant. Precision	Single-Document QA	Multi-Document QA	Summarization	Few-shot Learning	Synthetic	Code Completion	Average
Qwen3-8B	-	BF16	-	42.42	29.50	23.09	29.99	67.89	4.40	33.93
	W8A8	INT8	-	41.92	30.08	22.67	30.57	68.43	4.95	34.10
			Q16KV8	41.88	30.50	22.64	30.65	69.47	4.67	34.31
			QKV8	41.93	29.63	22.67	31.77	68.56	4.70	34.24
		MXFP8	-	42.77	29.95	22.91	32.20	68.97	5.29	**34.71**
			Q16KV8	42.64	29.48	23.07	33.31	68.78	5.36	**34.82**
			QKV8	42.56	29.57	22.64	31.30	68.72	4.96	34.30
		HiF8	-	42.23	29.16	22.80	32.67	69.14	3.84	34.41
			Q16KV8	42.05	28.97	22.91	33.83	68.89	3.94	34.55
			QKV8	42.15	28.49	22.86	33.18	68.83	3.55	34.30
	W4A4	INT4	-	1.17	1.11	2.13	0.59	0.91	7.81	1.83
			Q16KV4	1.14	1.42	2.45	0.54	1.24	7.31	1.93
			QKV4	1.03	1.12	2.54	0.51	1.43	7.75	1.93
		MXFP4	-	40.73	25.07	22.46	31.88	63.14	26.96	**34.47**
			Q16KV4	37.95	23.98	22.17	24.61	58.65	25.59	31.52
			QKV4	36.20	23.88	22.27	22.66	53.72	27.24	30.27
		NVFP4	-	39.34	28.61	23.00	25.42	65.10	8.44	32.27
			Q16KV4	39.25	26.39	22.93	26.01	61.97	7.87	31.43
			QKV4	38.74	25.88	22.72	25.95	61.39	9.83	*31.28*
		HiF4	-	39.70	27.92	22.63	26.77	67.50	5.85	*32.49*
			Q16KV4	41.08	27.91	22.82	28.37	66.14	7.34	**33.04**
			QKV4	39.65	27.70	22.60	26.02	64.56	5.47	**31.83**
openPangu-7B	-	BF16	-	36.25	24.78	20.50	41.84	60.70	23.96	34.45
	W8A8	INT8	-	37.22	26.20	20.29	40.94	60.41	26.85	*34.93*
			Q16KV8	36.83	26.71	20.44	41.30	59.72	25.70	*34.84*
			QKV8	36.86	26.94	20.49	40.05	61.81	26.60	**35.05**
		MXFP8	-	36.73	26.21	20.45	42.54	60.00	27.15	**35.14**
			Q16KV8	37.12	26.54	20.55	41.10	57.49	30.29	**34.97**
			QKV8	36.06	27.10	20.41	39.88	55.71	28.47	*34.18*
		HiF8	-	36.03	25.49	20.37	42.67	59.72	21.57	34.31
			Q16KV8	36.27	25.34	20.63	38.53	53.51	23.27	32.86
			QKV8	35.42	25.40	20.47	39.00	55.65	23.92	33.14
	W4A4	INT4	-	14.09	8.51	14.60	5.34	5.03	10.26	9.80
			Q16KV4	3.69	3.16	12.51	2.09	2.20	13.54	5.69
			QKV4	3.56	3.28	12.71	2.13	2.88	13.54	5.83
		MXFP4	-	33.14	20.18	20.51	29.81	58.86	27.63	30.78
			Q16KV4	26.00	14.91	19.64	12.21	23.75	18.59	19.02
			QKV4	21.89	13.35	19.23	11.33	17.85	15.79	16.59
		NVFP4	-	35.35	23.78	20.53	37.71	59.71	16.04	*32.41*
			Q16KV4	32.97	23.14	20.11	26.34	55.21	15.13	*28.86*
			QKV4	30.08	22.07	20.00	25.42	51.27	14.31	*27.27*
		HiF4	-	36.28	25.81	20.40	39.34	61.09	23.34	**34.16**
			Q16KV4	35.14	22.96	20.24	32.48	54.76	30.69	**31.85**
			QKV4	34.74	25.63	20.18	33.34	53.01	26.46	**31.79**

Table 12: LongBench Performance Evaluation of KV Cache and Attention Quantization with Various Formats based on the Weight and Activation Quantization using SmoothQuant. Q16KV8 and Q16KV4 denote KV cache quantization, while QKV8 and QKV4 simulate attention quantization. Accuracy Loss denotes the average accuracy loss compared with the BF16 baselines. **Bold red** and *italic blue* indicate the best and second best settings, respectively.

calized scaling effectively isolates high-magnitude outliers, preventing them from diluting the precision of the remaining values within the same block. HiF4 remains highly competitive but exhibits a slight loss of precision near zero; its coarser binning in the central region causes a high density of small-magnitude values to collapse into a limited number of states. This sensitivity is exacerbated by its larger block size relative to NVFP4, where a single extreme outlier can exert a disproportionate influence on the scaling factor for the entire group. To further investigate the impact of outliers on block-based quantization formats, we visualize a 64-element segment of the activation vector from the Qwen3-8B Layer 3 `down_proj` along with its corresponding quantization results in Figure 13. The magnified view of indices 16–31 demonstrates that NVFP4, bolstered by its FP32 global scale, successfully avoids the range inflation that typically plagues INT4. Although HiF4 lacks a global scaling factor, its three-level hierarchy effectively isolates local variance, thereby preserving a higher number of active quantization bins compared to MXFP4.

The distribution of Key states in Figure 12c is characterized by a broad range and a spiky, multimodal structure. This sparsity makes it the primary bottleneck for KV quantization. NVFP4 demonstrates the most precise alignment with this signal, as its 16-element block size allows the quantization grid to adapt rapidly to local magnitude shifts between different spikes. HiF4 remains highly resilient but exhibits a slight loss of resolution in the low-magnitude valleys between peaks. Because its hierarchical scaling must occasionally account for multiple peaks within a single block, the resulting grid can become slightly less dense in the regions between them. In contrast, MXFP4 and INT4 struggle significantly; their coarser scaling mechanisms are unable to track these rapid fluctuations, leading to a blurring of the Key distribution that likely contributes to the performance degradation seen in long-context benchmarks.

In contrast to the Key states, the Value distributions in Figure 12d appear more bounded and stable, mirroring the bell-shaped density often found

in weights. In this more predictable numerical space, the performance gap between formats begins to narrow, as the primary requirement is representational density rather than expansive dynamic range. HiF4 and NVFP4 both maintain a high concentration of representable states within the primary bulk of the Value distribution, ensuring that the majority of the signal is captured with high fidelity. However, INT4 still experiences a significant resolution loss, as 16 levels are insufficient to represent the nuanced variations within even a bounded Value tensor. MXFP4 also shows wider intervals than the hierarchical formats, confirming that even for well-behaved distributions like Values, a multi-level scaling approach is superior to a single-level power-of-two scale.

G.3 Additional Results

To demonstrate generalization across architectures and modalities, we further evaluate W4A4 RTN quantization across these data formats on the recent Qwen3.5-9B and Qwen3-VL-8B-Instruct models. As detailed in Table 9, evaluating Qwen3.5-9B under W4A4 RTN quantization further underscores the efficacy of HiF4. While MXFP4 and NVFP4 experience average performance degradations of 4.69% and 2.81% respectively compared to the BF16 baseline, HiF4 limits this drop to just 1.40%. The results for Qwen3-VL-8B-Instruct, presented in Table 10, further solidify the superiority of HiF4 on complex multimodal architectures. These results demonstrate that HiF4 generalizes seamlessly to novel architectural paradigms.

Table 11 provides a comprehensive performance evaluation of KV cache and attention quantization for the Qwen3-8B and openPangu-7B models. In the W8A8 regime, both models demonstrate high stability across all formats, with MXFP8 and HiF8 maintaining near-lossless performance. The transition to W4A4 precision coupled with KV cache and attention quantization reveals a critical bifurcation in format resilience. Traditional INT4 quantization suffers a catastrophic failure. While MXFP4 remains functional under weight and activation quantization alone, the addition of 4-bit KV cache and attention quantization triggers a severe performance decline. In contrast, HiF4 exhibits remarkable robustness under these extreme constraints. For Qwen3-8B, HiF4 consistently yields the superior result, restricting accuracy loss to 3.15% for Q16KV4 and 3.92% for QKV4. This trend is mirrored in the openPangu-7B results, where HiF4 maintains significantly higher model integrity than both MXFP4 and NVFP4 by limiting the loss to 11.03% and 13.84% respectively. These findings validate HiF4 as a uniquely effective format for end-to-end 4-bit inference, ensuring functional stability even when the entire model pipeline is subjected to ultra-low bit-width quantization. As shown in Table 12, the results from the LongBench dataset corroborate the findings from standard benchmarks, confirming that the hierarchical stability of HiF4 is critical for maintaining performance over long-context tasks, where the error accumulation may amplify the quantization errors caused by different low-bit data formats. HiF4 noticeably outperforms NVFP4 in term of the accuracy under the full 4-bit WA and KV cache quantization for both Qwen3-8B and openPanu-7B models.