MF-QAT: Multi-Format Quantization-Aware Training for Elastic Inference

Anonymous ACL submission

Abstract

Quantization-aware training (QAT) is typically performed for a single target numeric format, while practical deployments often need to choose numerical precision at inference time based on hardware support or runtime constraints. We study *multi-format QAT*, where a single model is trained to be robust across multiple quantization formats. We find that multi-format QAT can match single-format QAT at each target precision, yielding one model that performs well overall across different formats, even formats that were not seen during training. To enable practical deployment, we propose the *Slice-and-Scale* conversion procedure for both MXINT and MXFP that converts a high-precision representation into lower-precision formats without re-training. Building on this, we introduce a pipeline that (i) trains a model with multi-format QAT, (ii) stores a single anchor format checkpoint (MXINT8/MXFP8), and (iii) allows on-the-fly conversion to lower MXINT or MXFP formats at runtime with negligible-or no-additional accuracy degradation. Together, these components provide a practical path to elastic precision scaling and allow selecting the runtime format at inference time across diverse deployment targets.

1 Introduction

Deep neural networks increasingly rely on quantization to meet the latency, throughput, and memory constraints of modern deployment (Jacob et al., 2017; Esser et al., 2020; Dettmers et al., 2022; Xiao et al., 2024; Lin et al., 2024). In many production settings, however, the target numeric format is not fixed: different accelerators support different formats, and the same device might want to serve at different precisions for different batches based on the current load of the system (Cai et al., 2020; Lee et al., 2025a; Jin et al., 2020; Yi et al., 2025; Chen et al., 2025b). Standard quantization-aware training (QAT) commonly optimizes a model for a *single* format (Chen et al., 2025a; Ma et al., 2024), which creates a brittle dependency between training and deployment.

This paper asks a simple question: can we train one model that is robust across multiple quantization formats, and deploy it in a way that supports further precision reduction at inference time with minimal loss?

We answer both affirmatively. First, we discover that models trained with multi-format QAT perform on par with single-format QAT at the target precisions, so a single model can serve multiple deployment formats effectively.

Second, we design Slice-and-Scale format transformation methods for both MXINT and MXFP that produce lower-precision representations from higher-precision ones. These transformations enable a practical workflow in which a single stored checkpoint can be converted to different precisions efficiently without re-expanding to FP32 model weights.

Third, we propose an inference scheme MF-QAT that combines these ideas: we train a model with multi-format QAT, store the resulting model in a single anchor microscaling format (MXINT8/MXFP8), and allow it to be further quantized directly to an even lower precision at inference time. Empirically, this final step incurs only minimal additional accuracy degradation, enabling elastic precision scaling depending on runtime constraints.

2 Background and related work

Quantization and QAT. Post-training quantization and quantization-aware training (QAT) are standard approaches to reduce memory footprint and improve throughput (Jacob et al., 2017; Esser et al., 2020; Dettmers et al., 2022; Xiao et al., 2024; Lin et al., 2024; Xu et al., 2025; Saxena et al., 2025). QAT simulates the effects of quantization

during training, typically via straight-through estimators (Yin et al., 2019), allowing the model to adapt to the induced noise. While effective, most QAT pipelines optimize for a *single* target format (e.g., a fixed bitwidth and quantization scheme) (Chen et al., 2025a; Ma et al., 2024), which can lead to accuracy drops when the deployment format differs.

Microscaling formats. Many modern accelerators support integer microscaling (MXINT) and floating-point microscaling (MXFP), which employ shared scaling factors over blocks of values to achieve efficient computation while retaining sufficient dynamic range. In this framework, a tensor is represented by low-precision elements together with a shared scale for each block. Accordingly, a microscaling format is defined by (i) the scale-factor data type, (ii) the element data type and precision, and (iii) the scaling block size (Rouhani et al., 2023a). From this perspective, MXINT and MXFP share the same block-wise scaling abstraction and differ only in whether the elements are encoded as integers or floating-point numbers. Due to their ability to reduce memory usage and inference latency with negligible accuracy loss, microscaling formats are becoming an increasingly important quantization primitive in modern hardware (Sharify et al., 2024; Rouhani et al., 2023b; Yang et al., 2025b; Lee et al., 2025b).

Elastic inference serving. Recent work on elastic inference goes beyond single-format quantization by enabling a higher-precision checkpoint to be converted into lower-precision variants, thereby reducing or eliminating the need to store multiple full model copies. Representative approaches include overlay-style, nested, and Matryoshka-style parameterizations that expose multiple precision levels from a largely shared memory footprint (Park et al., 2024; Lee et al., 2025a; Chen et al., 2025b; Nair et al., 2025). For example, MatQuant exploits the nested structure of integer data types, allowing lower-bitwidth models to be extracted from the most significant bits of a higher-bitwidth integer representation. This is closely related to our goal of supporting multiple deployment precisions from a single checkpoint, but it focuses on nested integer formats such as INT8, INT4, and INT2. Other elastic-serving methods consider clustering-based quantization formats (Park et al., 2024), FP16/FP8 formats (Lee et al., 2025a), or runtime-adaptive precision selection using techniques such as precision-specific batch normalization, token- or layer-level precision switching, and lightweight LoRA adapters (Yi et al., 2025; Chen et al., 2025b; Liu et al., 2025; Jin et al., 2020; Bulat and Tzimiropoulos, 2021).

In contrast, our work focuses on microscaling formats, including both MXINT and MXFP. We do not modify the network architecture with additional normalization layers or precision-specific adapters. Instead, we directly quantize the weights to obtain a single high-precision MX checkpoint that can be efficiently converted to many lower-precision MX deployment formats through Slice-and-Scale, which explicitly transforms both element values and shared block scales. Our multi-format QAT scheme achieves minimal or no accuracy degradation across many deployment formats, including formats not seen during QAT.

3 Methods

3.1 Models, tasks, and datasets

We conduct experiments on the following pretrained LLMs: `Llama-2-7b-hf`, `Llama-3.2-1B`, and `Llama-3.2-3B` from the Llama2 (Touvron et al., 2023) and the Llama-3.2 (Grattafiori et al., 2024) families and `Qwen3-0.6B-Base`, `Qwen3-1.7B-Base`, and `Qwen3-4B-Base`, from the recently released Qwen3 family (Yang et al., 2025a). We also include multimodal language models belonging to Qwen3-VL family: `Qwen3-VL-2B-Instruct` and `Qwen3-VL-4B-Instruct` (Bai et al., 2025) for our evaluations.

For quantization-aware finetuning, we use 128 examples from the WikiText-2 train split (Merity et al., 2016), and we evaluate the multiformat QAT approach on a range of tasks which measure the *language modeling ability*: perplexity on WikiText-2 validation split (Merity et al., 2016), *common sense reasoning ability*: normalized 0-shot accuracy on HellaSwag (Zellers et al., 2019), *language understanding*: 0-shot accuracy on MMLU (Hendrycks et al., 2021), and *mathematical understanding*: normalized 0-shot accuracy on MathQA (Amini et al., 2019). For multimodal vision language models, we use ChartQA (Masry et al., 2022) for *visual question answering*.

3.2 Multi-format training scheme

Weight-only quantization. We perform weight-only quantization in the text decoder stack (exclud-

ing *lm_head*). During QAT, only the quantized weight parameters are updated, all other parameters are frozen. Similarly, for the full precision finetuning baseline, only these weight parameters are updated.

Training formats and schedule. We train sequentially on MXINT and MXFP weight formats with bitwidths $b \in \{2, 4, 6, 8\}$ and $\{4(E2M1), 6(E3M2), 8(E4M3)\}$ respectively. We train in *increasing* bit order (2→4→6→8), because lower-precision weights typically require larger updates to jump out of quantization bin, training in the opposite direction can destabilize the higher-precision quantization settings learned earlier. Each bitwidth is trained for 1 epoch, resulting in 4 epochs total for MXINT multi-format QAT and 3 epochs total for MXFP multi-format QAT. Each epoch contains the same 128 data examples.

Baselines. For comparison, we (i) finetune the model in full precision, and (ii) run single-format QAT for each target format. We use the same number of epochs as the multi-format QAT runs for fair comparison. For models larger than 2B, we observed severe overfitting for single-format QAT models, hence, the number of epochs are reduced to 1 for all runs for these models (each format in multi-format QAT is assigned equal number of steps within the epoch).

Evaluation. We evaluate on MXINT formats with bitwidths $b \in \{2, 3, 4, 5, 6, 7, 8\}$ and MXFP formats with bitwidths $b \in \{4(E2M1), 5(E2M2), 6(E3M2), 7(E3M3), 8(E4M3)\}$. For each trained variant (full-precision finetune, single-format QAT, and multi-format QAT), we apply post-training quantization (PTQ) to convert the resulting checkpoint to the target evaluation format before measuring performance on WikiText-2 validation. This isolates the effect of the training procedure on robustness to format changes and ensures that all compared variants are evaluated in the same target format.

Optimization details. For each model and training variant, we sweep learning rates in $\{10^{-4}, 10^{-5}, 10^{-6}\}$ for MXINT and $\{10^{-5}, 10^{-6}, 10^{-7}\}$ for MXFP, and report results for the best-performing learning rate. We use `torch.optim.AdamW` with default hyperparameters.

3.3 Slice-and-Scale MXINT (SSMXINT) format conversion

To enable fast conversion between higher-precision and lower-precision microscaling integer (MXINT) formats, we propose *Slice-and-Scale MXINT* (SSMXINT). SSMXINT converts a tensor represented in a higher-precision MXINT format (e.g., MXINT8) into a lower-precision MXINT format by (i) *slicing* the least-significant bits of the elements with rounding and (ii) *scaling* the shared block scale so that the represented real range is preserved.

Notation (used in Sec. 3.3–3.4). For a block of scalar floats $V = \{V_i\}_{i=1}^{k}$ with block size of k, MX conversion returns a shared scale X and elements $\{P_i\}_{i=1}^{k}$. The reconstruction is $\hat{V}_i = XP_i$. Let $e_{\max}(f)$ denote the exponent of the largest normal number in element format f, and let $\mathrm{quantize}_f(\cdot)$ denote conversion to f (rounding plus clamping). We use $\mathrm{Round}(\cdot)$ for element-wise round-to-nearest. Following the MX format conversion algorithm in (Rouhani et al., 2023b),

$$\texttt{shared_exp} = \left\lfloor \log_2\left(\max_i |V_i|\right)\right\rfloor - e_{\max}(f), \quad (1)$$

$$X = 2^{\texttt{shared_exp}}, \quad (2)$$

$$P_i = \mathrm{quantize}_f\left(\frac{V_i}{X}\right). \quad (3)$$

MXINT representation. For MXINT, the shared scale follows Eq.1 with $f = \mathrm{MXINT}(b)$, where b is the number of bits for the element format:

$$\texttt{shared_exp} = \left\lfloor \log_2\left(\max_i |V_i|\right)\right\rfloor - e_{\max}(b), \quad (4)$$

$$X = 2^{\texttt{shared_exp}}. \quad (5)$$

Elements are then computed by $P_i = \mathrm{clip}_b(\mathrm{Round}(V_i/X))$.

High-to-low conversion via slice-and-scale. Let (X_h, P_h, b_h) be a high-precision MXINT block and (X_ℓ, P_ℓ, b_ℓ) the target one, with $b_h > b_\ell$. Since V is fixed, the computation of scale for different MXINT precisions only differs in the value of $e_{\max}(b)$. Hence, it is theoretically equivalent to obtain the low-precision MXINT scale from the high-precision MXINT scale accounting for the difference in $e_{\max}(b)$ versus obtaining the low-precision MXINT scale from the original full precision tensor.

Define $\Delta e = e_{\max}(b_h) - e_{\max}(b_\ell)$. For signed MXINT, this is equivalent to $\Delta e = b_h - b_\ell$. SSMXINT converts by

$$P_{\ell,i} = \mathrm{clip}_{b_\ell}\left(\mathrm{Round}\left(\frac{P_{h,i}}{2^{\Delta e}}\right)\right), \qquad X_\ell = X_h\, 2^{\Delta e}. \quad (6)$$

Hence $\hat{V}_{\ell,i} = X_\ell P_{\ell,i} \approx X_h P_{h,i} = \hat{V}_{h,i}$, where the residual error is due to rounding in the low-precision cast. Since the element value is in integer format and is divided by powers of two, this conversion is equivalent to a right shift and round on integer elements: for each $P_{h,i}$, divide by $2^{\Delta e}$ (right shift by Δe), drop the Δe least-significant bits, round using the most-significant dropped bit, and keep the rounded value in the target element representation. The scale update $X_\ell = X_h 2^{\Delta e}$ compensates for the element down-scaling so the reconstructed block remains numerically close to the high-precision one.

3.4 Slice-and-Scale MXFP (SSMXFP) format conversion

The same idea can be applied to MXFP.

MXFP representation. Let the MXFP element format be parameterized by exponent bits η and mantissa bits μ (total element bits $b = 1 + \eta + \mu$), with $\eta >= \mu$. $\text{quantize}_{\eta,\mu}(\cdot)$ denotes quantizing (including rounding and clamping) to the target MXFP element format. Let $e_{\max}(\eta)$ denote the exponent of the largest normal number in this MXFP format. Following the same Eq. 1,

$$\texttt{shared_exp} = \left\lfloor \log_2\left(\max_i |V_i|\right)\right\rfloor - e_{\max}(\eta), \quad (7)$$

$$X = 2^{\texttt{shared_exp}}, \quad (8)$$

followed by $P_i = \text{quantize}_{\eta,\mu}(V_i/X)$.

High-to-low conversion via slice-and-scale. Let high precision use (η_h, μ_h) and low precision use (η_ℓ, μ_ℓ), with $\eta_h > \eta_\ell$ and $\Delta e = e_{\max}(\eta_h) - e_{\max}(\eta_\ell)$. Following the same slice-and-scale rule,

$$P_{\ell,i} = \text{quantize}_{\eta_\ell,\mu_\ell}\left(\frac{P_{h,i}}{2^{\Delta e}}\right), \qquad X_\ell = X_h\, 2^{\Delta e}. \quad (9)$$

Compared with SSMXINT, SSMXFP incurs additional elementwise conversion cost, since MXFP elements cannot be reduced to lower precision through integer bit shifts alone. Depending on hardware support, this conversion may be implemented either as a direct cast from the high-precision anchor format to the target format, or through an FP32 intermediate. In either case, SSMXFP is still substantially simpler than standard MX requantization: the shared block scale is adjusted by a fixed format-dependent exponent offset, without requiring block-wise maximum search or scale recomputation.

3.5 Training and inference with anchor format storage

Let A denote the anchor format (MXINT8 or MXFP8), and t a runtime target format derived from A.

Training. We maintain full-precision master weights W_{fp}. For each forward pass,

$$W_A = Q_A(W_{\text{fp}}), \qquad W_t = Q_{A\to t}(W_A), \quad (10)$$

where Q_A quantizes to the anchor format and $Q_{A\to t}$ denotes SSMXINT or SSMXFP conversion. The model runs with W_t, and gradients are propagated through both operators using straight-through estimators (Yin et al., 2019) to update W_{fp}.

Inference. We store only the anchor checkpoint W_A. At runtime, we generate $W_t = Q_{A\to t}(W_A)$ on demand for any target format t. This enables elastic precision selection without retraining.

4 Results

4.1 Multi-format QAT across MXINT and MXFP

We compare multi-format QAT against precision-specific single-format QAT and full-precision fine-tuning followed by post-training quantization across both MXINT and MXFP formats. Figure 1 reports zero-shot WikiText-2 perplexity for `Qwen3-4B-Base` as a function of evaluation bit-width. Results for all models are provided in Appendix A.1.

Brittleness of single-format QAT. As expected, single-format QAT performs best at its target bit-width. However, these models are highly brittle under format mismatch. In the MXINT setting (Figure 1, left), models trained at 4, 6, or 8 bits degrade sharply when evaluated at 2 or 3 bits. Conversely, the 2-bit model performs poorly at higher precisions. This shows that precision-specific QAT tends to over-specialize to its training format.

Robustness of multi-format QAT. In contrast, multi-format QAT (purple) closely tracks the best achievable performance across the full precision range. At the explicitly trained bit-widths ($\{2, 4, 6, 8\}$), it remains consistently close to the corresponding single-format optimum, while also generalizing well to unseen intermediate bit-widths ($\{3, 5, 7\}$). Overall, multi-format QAT removes the need to commit to a single deployment precision, enabling robust performance under dynamic inference-time format selection.

Model	QAT/FT Precision	PTQ Precision						
		MXINT2	MXINT3*	MXINT4	MXINT5*	MXINT6	MXINT7*	MXINT8
Llama-2-7B	Full Precision FT	22.7	37.5	44.4	44.7	**45.8**	46.0	46.0
	MXINT2 QAT	**29.3**	**39.6**	44.4	44.6	45.1	45.6	45.1
	MXINT4 QAT	22.9	38.7	**44.6**	44.8	45.6	45.9	46.0
	MXINT6 QAT	22.1	38.4	43.7	44.6	45.2	46.1	46.0
	MXINT8 QAT	21.9	37.7	44.2	44.8	45.2	46.0	46.0
	Multi-format QAT	23.1	38.7	43.9	**44.9**	45.5	**46.2**	**46.1**
Llama-3.2-1B	Full Precision FT	21.9	25.6	34.9	40.9	41.3	**41.3**	41.2
	MXINT2 QAT	**23.6**	26.2	34.7	40.2	41.2	40.6	40.8
	MXINT4 QAT	22.8	**28.3**	**36.1**	41.1	**41.4**	40.9	41.5
	MXINT6 QAT	22.4	25.8	35.1	41.3	**41.4**	40.7	41.5
	MXINT8 QAT	22.1	25.6	35.2	**41.4**	**41.4**	**41.3**	**41.6**
	Multi-format QAT	23.3	28.2	35.2	40.8	41.3	40.8	41.4
Llama-3.2-3B	Full Precision FT	25.2	35.6	49.5	51.1	52.1	51.7	51.8
	MXINT2 QAT	**25.5**	36.6	49.2	51.4	52.3	**52.3**	52.0
	MXINT4 QAT	24.5	36.6	49.3	51.4	52.5	51.7	51.6
	MXINT6 QAT	24.1	35.4	49.4	**51.5**	**52.6**	51.8	51.9
	MXINT8 QAT	24.9	35.5	49.4	51.4	**52.6**	51.9	51.8
	Multi-format QAT	23.9	**36.7**	**49.7**	51.2	52.4	52.0	**52.1**
Qwen3-0.6B	Full Precision FT	24.0	25.0	38.4	43.5	45.9	46.7	**46.9**
	MXINT2 QAT	**24.4**	24.9	38.1	42.3	45.6	45.8	45.9
	MXINT4 QAT	23.3	25.3	**39.6**	**44.1**	**47.2**	46.1	46.5
	MXINT6 QAT	23.6	24.7	38.3	43.8	46.4	**47.2**	46.8
	MXINT8 QAT	22.6	**25.4**	38.0	43.7	46.5	47.0	46.8
	Multi-format QAT	22.0	25.2	**39.6**	43.7	46.1	46.9	46.2
Qwen3-1.7B	Full Precision FT	**24.1**	29.2	48.6	53.1	54.2	54.6	55.2
	MXINT2 QAT	22.7	28.8	48.7	53.4	54.3	54.1	**55.5**
	MXINT4 QAT	22.7	**29.5**	48.6	52.9	**55.1**	**54.7**	55.4
	MXINT6 QAT	22.5	29.1	48.9	**53.7**	54.5	54.6	55.1
	MXINT8 QAT	23.9	28.6	48.8	53.2	54.7	54.3	55.1
	Multi-format QAT	22.7	29.3	**49.4**	**53.7**	54.7	**54.7**	**55.5**
Qwen3-4B	Full Precision FT	**24.6**	31.1	60.1	60.5	60.9	61.7	61.7
	MXINT2 QAT	22.7	31.6	59.4	60.2	60.9	61.9	62.0
	MXINT4 QAT	23.3	**32.8**	**60.8**	60.5	61.6	62.2	62.0
	MXINT6 QAT	22.2	31.9	60.1	**60.6**	**61.8**	**62.7**	**62.9**
	MXINT8 QAT	23.0	31.4	60.1	60.5	61.6	62.4	62.8
	Multi-format QAT	23.9	31.8	60.0	60.4	61.5	61.9	62.4

Table 1: Average 0-shot accuracy (higher is better) on MMLU (Hendrycks et al., 2021), MathQA (Amini et al., 2019), and HellaSwag (Zellers et al., 2019) for *the MXINT formats*. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively. Detailed per-individual task results in Appendix B, Tables 4-6.

4.2 Multi-format QAT with downstream task

Tables 1 and 2 present the average 0-shot accuracy across MMLU (Hendrycks et al., 2021), MathQA (Amini et al., 2019), and HellaSwag (Zellers et al., 2019) for six models from the Llama and Qwen families when quantized to MXINT and MXFP formats, respectively. Across all six models and both formats, multi-format QAT either outperforms the corresponding single-format QAT baselines or remains within 1% accuracy of the best-performing model. The only exception is the MXINT2 format, where accuracy stays within 3% for all models except `Llama-2-7B`. This is achieved despite training a single model to handle all precisions simultaneously.

For instance, on `Qwen3-1.7B` under MXINT (Table 1), multi-format QAT achieves the best or second-best accuracy at six out of seven evaluation precisions, including at unseen precisions such as MXINT5 (53.7%) and MXINT7 (54.7%). A similar trend holds under the MXFP formats (Table 2), where multi-format QAT on `Qwen3-1.7B` reaches 54.3% and 54.8% at MXFP6 and MXFP7-matching or exceeding all single-format baselines. These results confirm that the downstream task performance mirrors the perplexity trends observed in Section 4.1: a single multi-format QAT model can robustly serve diverse deployment precisions with negligible-or no-additional accuracy degradation at individual formats, eliminating the need to train and store separate models for each target precision.

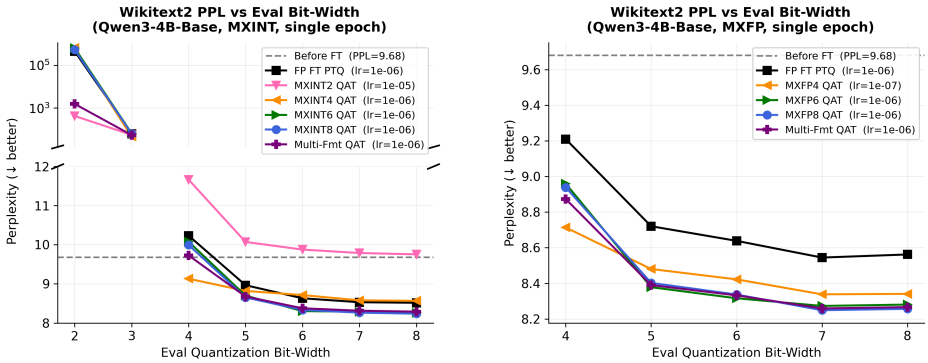

Figure 1: Multi-format QAT on `Qwen3-4B-Base`, trained for one epoch on 128 WikiText-2 examples. **Left**: MXINT formats. **Right**: MXFP formats. For each training variant, we show the result at the best learning rate, chosen as the one with the lowest evaluation loss among the three learning rates in Section 3.2. The x-axis denotes evaluation quantization bit-width, and the y-axis denotes WikiText-2 validation perplexity after post-training quantization to the corresponding bit-width. The horizontal dashed line marks the original high-precision model (FP16/BF16/FP32, depending on pretraining). The black solid line corresponds to full-precision fine-tuning, the purple solid line to multi-format QAT, and the other colored solid lines to single-format QAT.

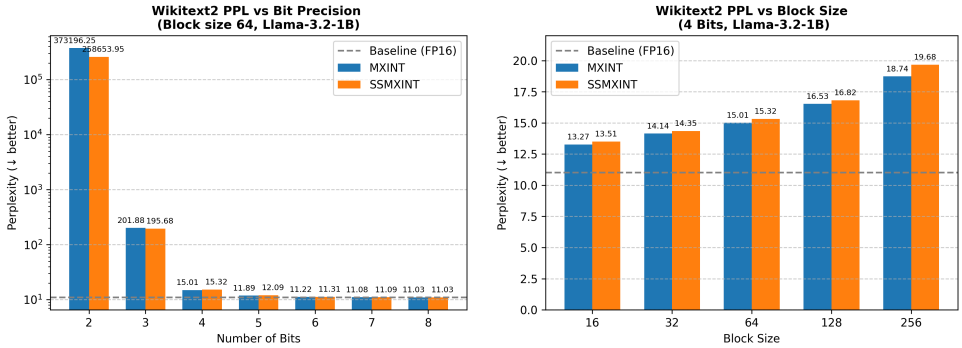

Figure 2: WikiText-2 perplexity for `Llama-3.2-1B`, comparing direct MXINT quantization and SSMXINT. **Left:** Varying bit precision at block size 64. **Right:** Varying block size at 4-bit precision. Horizontal dashed line denotes the baseline model.

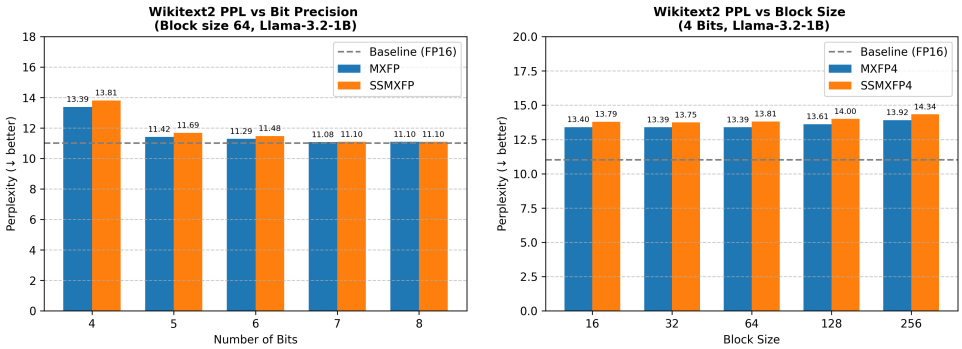

Figure 3: WikiText-2 perplexity for `Llama-3.2-1B`, comparing direct MXFP quantization and SSMXFP. **Left:** Varying bit precision at block size 64. **Right:** Varying block size at 4-bit precision. Horizontal dashed line denotes the baseline model.

We further evaluate multi-format QAT on the multimodal Qwen3-VL models using ChartQA (Table 3). Despite higher variance inherent to visual reasoning tasks, multi-format QAT remains competitive across precisions-for example, achieving the best accuracy at MXINT6 (30.0%) on `Qwen3-VL-4B` and matching the top single-format result at MXFP7 (39.0%) on `Qwen3-VL-2B`. These

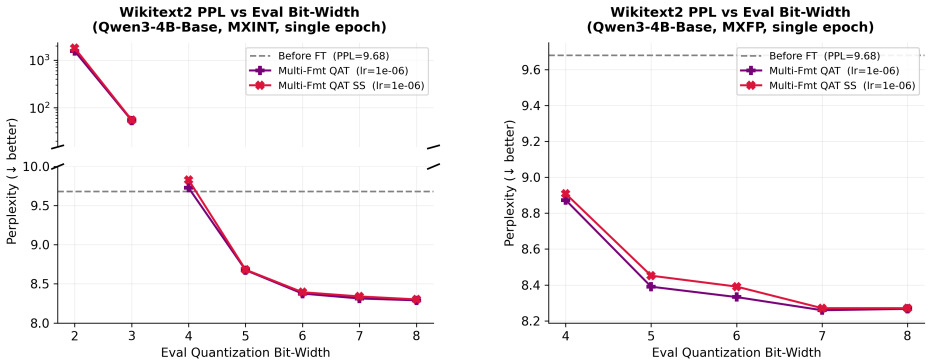

Figure 4: Multi-format QAT with Slice-and-Scale on `Qwen3-4B-Base`. **Left:** MXINT formats. **Right:** MXFP formats. Plotting conventions match Figure 1.

results confirm that the benefits of multi-format QAT extend to multimodal workloads.

4.3 Effectiveness of Slice-and-Scale conversion

The previous subsection shows that multi-format QAT is robust across target precisions while requiring only a single full-precision model during training. We now show that storage can be reduced further by replacing this with an 8-bit anchor checkpoint, which is converted on demand to lower-precision MX formats using Slice-and-Scale (SS). The details of the SS algorithm are described in Sections 3.3 and 3.4.

We evaluate SS conversion using two metrics: average layer-wise MSE on 100 random tensors of shape $(1, 1024)$, and zero-shot WikiText-2 perplexity for `Llama-3.2-1B`. Here, MXINT/MXFP denote direct quantization into target OCP MX formats (Rouhani et al., 2023a), while SSMXINT/SSMXFP denote our conversion from an 8-bit anchor (MXINT8/MXFP8) to lower precisions without access to full-precision weights.

SS closely matches direct quantization at the tensor level across bit precisions and block sizes for both MXINT and MXFP. Full MSE results are provided in Appendix C.

End-to-end language modeling performance. Figures 2 and 3 show that SS achieves WikiText-2 perplexity nearly identical to direct target-format quantization across all evaluated settings. These results show that a single higher-precision anchor checkpoint can be down-converted at runtime to lower-precision MX formats with minimal loss.

4.4 Multi-format QAT with Slice and Scale

We next evaluate the full pipeline combining multi-format QAT with Slice-and-Scale (SS) conversion. Figure 4 shows results for `Qwen3-4B-Base`. Results for all models are deferred to Appendix A.2. We train this variant using the anchor-storage procedure in Section 3.5, cycling through target formats uniformly during training.

Figure 4 shows that the Slice-and-Scale variant (red) closely matches standard multi-format QAT (purple) across the full precision range. For MXINT, the two curves are nearly indistinguishable; for MXFP, only a small gap appears at intermediate bit-widths, consistent with Section 4.3.

These results show that a single low-precision anchor checkpoint, together with on-the-fly Slice-and-Scale conversion, recovers nearly the same performance as standard multi-format QAT while reducing checkpoint storage overhead.

5 Conclusion

We presented a study of multi-format quantization-aware training (MF-QAT) and showed that training a single model across multiple quantization formats can match the performance of single-format QAT at each target precision, providing a single model that remains effective across formats. To make these benefits deployable, we introduced a Microscaling format transformation method that derives a lower-precision MXINT/MXFP representation from a higher-precision one without retraining. Finally, we proposed an elastic inference scheme that stores a single low-precision checkpoint and enables further on-the-fly quantization to even lower precision at runtime with minimal loss degradation. These results suggest a practical path to decoupling training from rigid deployment-format choices, improving robustness across heterogeneous hardware, and enabling dynamic precision scaling under varying runtime constraints.

Model	QAT/FT Precision	PTQ Precision				
		MXFP4	MXFP5*	MXFP6	MXFP7*	MXFP8
Llama-2-7B	Full Precision FT	43.3	45.5	45.7	**46.0**	**46.2**
	MXFP4 QAT	**43.7**	**46.0**	**46.1**	45.9	45.6
	MXFP6 QAT	43.3	45.5	45.8	45.7	45.8
	MXFP8 QAT	43.6	45.4	45.7	**46.0**	45.8
	Multi-format QAT	43.5	45.5	45.8	45.8	45.8
Llama-3.2-1B	Full Precision FT	37.4	41.1	**41.5**	41.5	41.7
	MXFP4 QAT	**38.2**	41.4	41.2	**42.0**	**41.8**
	MXFP6 QAT	37.3	41.1	41.1	41.8	41.4
	MXFP8 QAT	37.8	41.0	41.1	41.8	**41.8**
	Multi-format QAT	37.6	**41.5**	41.2	41.4	41.5
Llama-3.2-3B	Full Precision FT	**50.6**	**52.5**	51.6	51.8	51.6
	MXFP4 QAT	50.2	51.9	51.8	51.9	**51.8**
	MXFP6 QAT	50.0	52.3	51.9	**52.0**	51.7
	MXFP8 QAT	50.2	**52.5**	**52.3**	51.8	51.7
	Multi-format QAT	49.9	**52.5**	52.2	51.8	51.7
Qwen3-0.6B	Full Precision FT	**44.0**	46.1	46.3	45.7	45.5
	MXFP4 QAT	43.7	45.9	46.7	**46.1**	**46.2**
	MXFP6 QAT	43.7	**46.5**	46.7	45.6	45.4
	MXFP8 QAT	43.6	**46.5**	**46.8**	45.6	45.2
	Multi-format QAT	43.5	46.1	46.4	45.6	45.4
Qwen3-1.7B	Full Precision FT	51.8	53.7	53.7	54.3	54.7
	MXFP4 QAT	**52.8**	53.7	53.9	**54.8**	**55.2**
	MXFP6 QAT	51.7	54.0	**54.3**	54.3	54.7
	MXFP8 QAT	52.3	**54.3**	54.0	54.4	54.4
	Multi-format QAT	52.2	54.1	**54.3**	**54.8**	54.7
Qwen3-4B	Full Precision FT	60.1	61.2	61.1	61.4	61.4
	MXFP4 QAT	**60.3**	**62.2**	61.9	62.3	**62.4**
	MXFP6 QAT	**60.3**	61.6	61.5	**62.4**	62.0
	MXFP8 QAT	60.2	61.9	62.1	62.3	62.2
	Multi-format QAT	**60.3**	61.9	**62.2**	**62.4**	62.3

Table 2: Average 0-shot accuracy (higher is better) on MMLU (Hendrycks et al., 2021), MathQA (Amini et al., 2019), and HellaSwag (Zellers et al., 2019) for *the MXFP formats*. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively. Detailed per-individual task results in Appendix B, Table 7.

Model	QAT/FT Precision	PTQ Precision				
		MXINT4	MXINT5*	MXINT6	MXINT7*	MXINT8
Qwen3-VL-2B	Full Precision FT	15.0	52.0	36.0	**44.0**	35.0
	MXINT4 QAT	**23.0**	**55.0**	32.0	39.0	38.0
	MXINT6 QAT	17.0	50.0	37.0	41.0	**40.0**
	MXINT8 QAT	20.0	47.0	**40.0**	40.0	32.0
	Multi-format QAT	15.0	48.0	31.0	34.0	34.0
Qwen3-VL-4B	Full Precision FT	23.0	**25.0**	28.0	**23.0**	25.0
	MXINT4 QAT	24.0	24.0	**30.0**	**23.0**	**28.0**
	MXINT6 QAT	**28.0**	21.0	27.0	22.0	24.0
	MXINT8 QAT	27.0	22.0	28.0	22.0	26.0
	Multi-format QAT	25.0	23.0	**30.0**	22.0	23.0
Model	**QAT/FT Precision**	**PTQ Precision**				
		MXFP4	**MXFP5***	**MXFP6**	**MXFP7***	**MXFP8**
Qwen3-VL-2B	Full Precision FT	30.0	33.0	29.0	30.0	27.0
	MXFP4 QAT	**42.0**	**37.0**	**34.0**	**39.0**	**43.0**
	MXFP6 QAT	33.0	34.0	31.0	38.0	37.0
	MXFP8 QAT	35.0	35.0	**34.0**	32.0	28.0
	Multi-format QAT	32.0	34.0	33.0	**39.0**	36.0
Qwen3-VL-4B	Full Precision FT	**18.0**	**29.0**	**27.0**	23.0	**24.0**
	MXFP4 QAT	16.0	27.0	24.0	24.0	**24.0**
	MXFP6 QAT	**18.0**	27.0	25.0	**27.0**	20.0
	MXFP8 QAT	**18.0**	**29.0**	25.0	24.0	23.0
	Multi-format QAT	16.0	26.0	24.0	24.0	23.0

Table 3: ChartQA (Masry et al., 2022) accuracy for Qwen3-VL (Bai et al., 2025) multi-modal models when quantized to the MXINT and MXFP formats with different precisions. Rows indicate the QAT/FT training precision; columns indicate the PTQ inference precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively.

Limitations

MF-QAT provides a practical framework for elastic inference over microscaling formats, and our experiments demonstrate strong quantization quality and cross-format robustness across a broad range of MXINT and MXFP deployment formats. Nevertheless, several aspects of the framework merit further study. First, our evaluation primarily focuses on model-level accuracy, while the potential system-level benefits are analyzed conceptually. An important next step is to integrate Slice-and-Scale with real accelerator backends and measure conversion overhead, latency, throughput, and memory footprint in realistic serving settings. Such measurements would be particularly valuable for MXFP formats, where conversion is more general than integer bit shifting and may involve intermediate floating-point operations.

Second, our experiments adopt a fixed multi-format QAT schedule to study whether a single checkpoint can generalize across many deployment precisions. This provides a controlled setting for evaluating cross-format generalization, but leaves open a broader training design space. Future work could explore alternative fine-tuning datasets, matched optimization budgets, different precision orders, randomized precision sampling, and joint training over all target formats. These studies would provide a more complete understanding of how data, training horizon, and precision ordering affect the robustness of multi-format QAT.

Finally, while our evaluation spans multiple language and vision-language model families, further studies across larger model scales, additional quantization pipelines, and more generation-oriented benchmarks would strengthen the empirical picture. Combining Slice-and-Scale with stronger QAT or PTQ methods, and evaluating on instruction following, mathematical reasoning, and code generation tasks, would further establish the generality of elastic microscaling inference.

6 Potential risks.

This work focuses on efficient quantized inference and does not introduce new datasets, user-facing applications, or model capabilities beyond those of the underlying pretrained models. The main potential risk is that improved inference efficiency may lower the cost of deploying both beneficial and harmful applications of existing language models. At the same time, the method can reduce memory and compute requirements for serving, which may improve accessibility and energy efficiency. Responsible deployment should therefore follow the safety, privacy, and usage constraints of the underlying models and applications.

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards Interpretable Math Word Problem Solving with Operation-Based Formalisms. *arXiv preprint*. ArXiv:1905.13319 [cs].

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-VL Technical Report.

Adrian Bulat and Georgios Tzimiropoulos. 2021. BitMixer: Mixed-precision networks with runtime bit-width selection. *arXiv preprint*. ArXiv:2103.17267 [cs].

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 2020. Once-for-All: Train One Network and Specialize it for Efficient Deployment. *arXiv preprint*. ArXiv:1908.09791 [cs].

Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. 2025a. EfficientQAT: Efficient Quantization-Aware Training for Large Language Models. *arXiv preprint*. ArXiv:2407.11062 [cs].

Shaoyuan Chen, Zhixuan Chen, Dawei Yang, Zhihang Yuan, and Qiang Wu. 2025b. OTARo: Once Tuning for All Precisions toward Robust On-Device LLMs. *arXiv preprint*. ArXiv:2511.13147 [cs].

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. *arXiv preprint*. ArXiv:2208.07339 [cs].

Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. 2020. Learned Step Size Quantization. *arXiv preprint*. ArXiv:1902.08153 [cs].

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *arXiv preprint*. ArXiv:2009.03300 [cs].

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *arXiv preprint*. ArXiv:1712.05877 [cs].

Qing Jin, Linjie Yang, and Zhenyu Liao. 2020. AdaBits: Neural Network Quantization with Adaptive Bit-Widths. *arXiv preprint*. ArXiv:1912.09666 [cs].

Haeun Lee, Omin Kwon, Yeonhong Park, and Jae W. Lee. 2025a. NestedFP: High-Performance, Memory-Efficient Dual-Precision Floating Point Support for LLMs. *arXiv preprint*. ArXiv:2506.02024 [cs].

Jungi Lee, Junyong Park, Soohyun Cha, Jaehoon Cho, and Jaewoong Sim. 2025b. Mx+: Pushing the limits of microscaling formats for efficient large language model serving. *Preprint*, arXiv:2510.14557.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. *arXiv preprint*. ArXiv:2306.00978 [cs].

Fangxin Liu, Zongwu Wang, Jinhong Xia, Junping Zhao, Shouren Zhao, Jinjin Li, Jian Liu, Li Jiang, and Haibing Guan. 2025. FlexQuant: A Flexible and Efficient Dynamic Precision Switching Framework for LLM Quantization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 4152–4161, Suzhou, China. Association for Computational Linguistics.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. *arXiv preprint*. ArXiv:2402.17764 [cs].

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning. *arXiv preprint*. ArXiv:2203.10244 [cs].

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. *arXiv preprint*. ArXiv:1609.07843 [cs].

Pranav Nair, Puranjay Datta, Jeff Dean, Prateek Jain, and Aditya Kusupati. 2025. Matryoshka quantization. *Preprint*, arXiv:2502.06786.

Yeonhong Park, Jake Hyun, SangLyul Cho, Bonggeun Sim, and Jae W. Lee. 2024. Any-Precision LLM: Low-Cost Deployment of Multiple, Different-Sized LLMs. *arXiv preprint*. ArXiv:2402.10517 [cs].

Bita Darvish Rouhani, Nitin Garegrat, Tom Savell, Ankit More, Kyung-Nam Han, Ritchie Zhao, Mathew Hall, Jasmine Klar, Eric Chung, Yuan Yu, Michael Schulte, Ralph Wittig, Ian Bratt, Nigel Stephens, Jelena Milanovic, John Brothers, Pradeep Dubey, Marius Cornea, Alexander Heinecke, and 7 others. 2023a. OCP Microscaling Formats (MX) Specification. *Open Compute Project*.

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. 2023b. Microscaling Data Formats for Deep Learning. *arXiv preprint*. ArXiv:2310.10537 [cs].

Utkarsh Saxena, Sayeh Sharify, Kaushik Roy, and Xin Wang. 2025. ResQ: Mixed-Precision Quantization of Large Language Models with Low-Rank Residuals. *arXiv preprint*. ArXiv:2412.14363 [cs].

Sayeh Sharify, Utkarsh Saxena, Zifei Xu, Wanzin Yazar, Ilya Soloveychik, and Xin Wang. 2024. Post Training Quantization of Large Language Models with Microscaling Formats. *arXiv preprint*. ArXiv:2405.07135 [cs].

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint*. ArXiv:2307.09288 [cs].

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2024. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. *arXiv preprint*. ArXiv:2211.10438 [cs].

Zifei Xu, Sayeh Sharify, Wanzin Yazar, Tristan Webb, and Xin Wang. 2025. Understanding the Difficulty of Low-Precision Post-Training Quantization for LLMs. *arXiv preprint*. ArXiv:2410.14570 [cs].

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Hanmei Yang, Summer Deng, Amit Nagpal, Maxim Naumov, Mohammad Janani, Tongping Liu, and Hui Guan. 2025b. An empirical study of microscaling formats for low-precision llm training. In *2025 IEEE 32nd Symposium on Computer Arithmetic (ARITH)*, pages 1–8.

Ke Yi, Yuhui Xu, Heng Chang, Yuan Meng, Tong Zhang, and Jia Li. 2025. One QuantLLM for ALL: Fine-tuning Quantized LLMs Once for Efficient Deployments. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23057–23066, Vienna, Austria. Association for Computational Linguistics.

Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. 2019. Understanding Straight-Through Estimator in Training Activation Quantized Neural Nets. *arXiv preprint*. ArXiv:1903.05662 [cs].

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? *arXiv preprint*. ArXiv:1905.07830 [cs].

A Wikitext2 PPL Eval results for all models

A.1 Additional plots for Multi-format QAT across MXINT and MXFP

This appendix contains plots for Section 4.1 on all models. Each figure below contains results for one model with left figure for MXINT and right figure for MXFP. Plotting conventions follow Figure 1.

A.2 Additional plots for Multi-format QAT with Slice and Scale

This appendix contains plots for Section 4.4 on all models. Each figure below contains results for one model with left figure for MXINT and right figure for MXFP. Plotting conventions follow Figure 4.

B Complete results of main result tables

Detailed results of Table 1 in the main paper, including task-by-task performance for the MXINT formats, are shown in Table 4 (MMLU accuracy (Hendrycks et al., 2021)), Table 5 (MathQA accuracy (Amini et al., 2019)), and Table 6 (HellaSwag accuracy (Zellers et al., 2019)). Similarly, per-task accuracy for the MXFP formats is indicated in Table 7. Our results show that, across diverse downstream tasks, a single multi-format QAT model can effectively generalize to multiple deployment precisions while maintaining accuracy for each format, avoiding the need to train and store separate models for every target precision.

C Tensor-level reconstruction error for Slice-and-Scale

Figures 19 and 20 report average layer-wise MSE on 100 random tensors of shape $(1, 1024)$ under two sweeps: varying bit precision at fixed block size 64, and varying block size at fixed 4-bit precision. As expected, reconstruction error decreases with higher precision and smaller block size. Across all settings, SS closely matches direct quantization for both MXINT and MXFP. Although SSMXFP exhibits a modestly larger relative gap at intermediate bit-widths, the absolute difference remains small, confirming that SS preserves the numerical behavior of the 8-bit anchor with high fidelity.

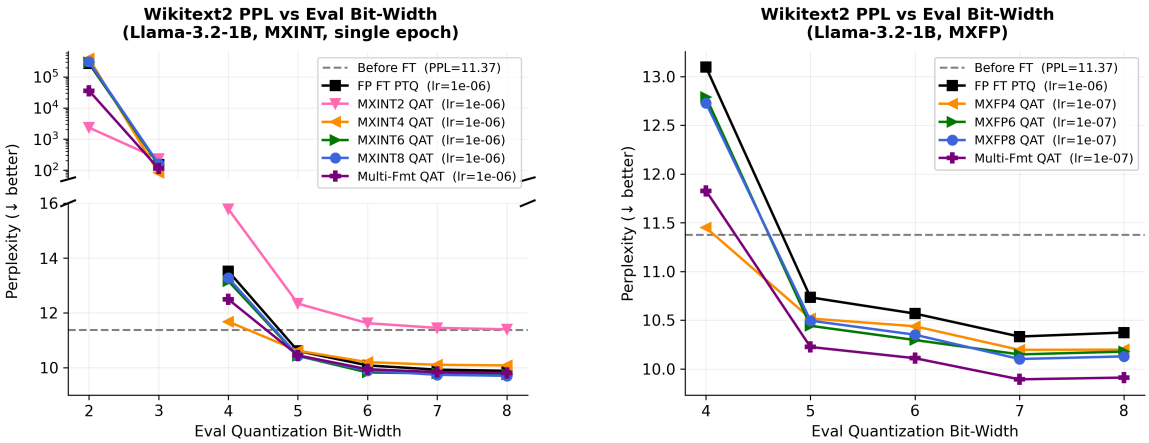

Figure 5: Multi-format QAT results for `Llama-3.2-1B`.

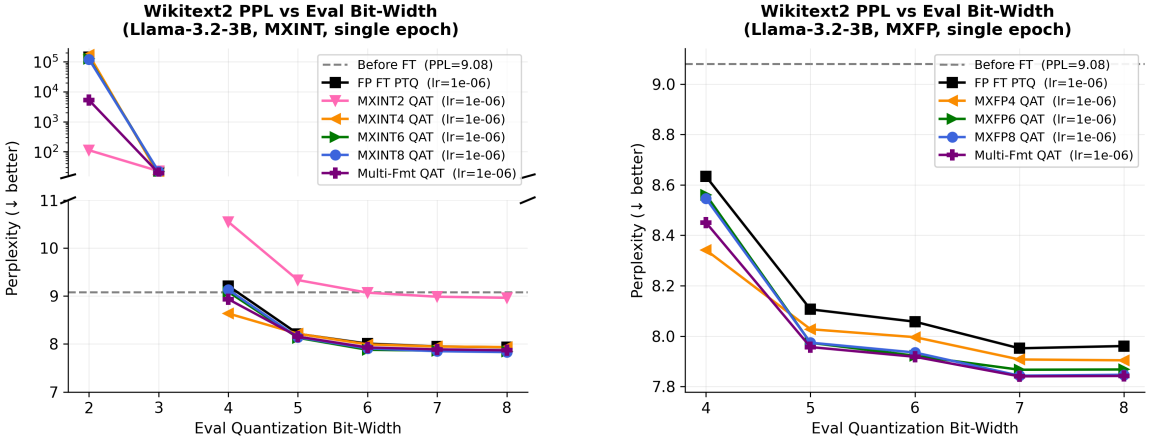

Figure 6: Multi-format QAT results for `Llama-3.2-3B`.

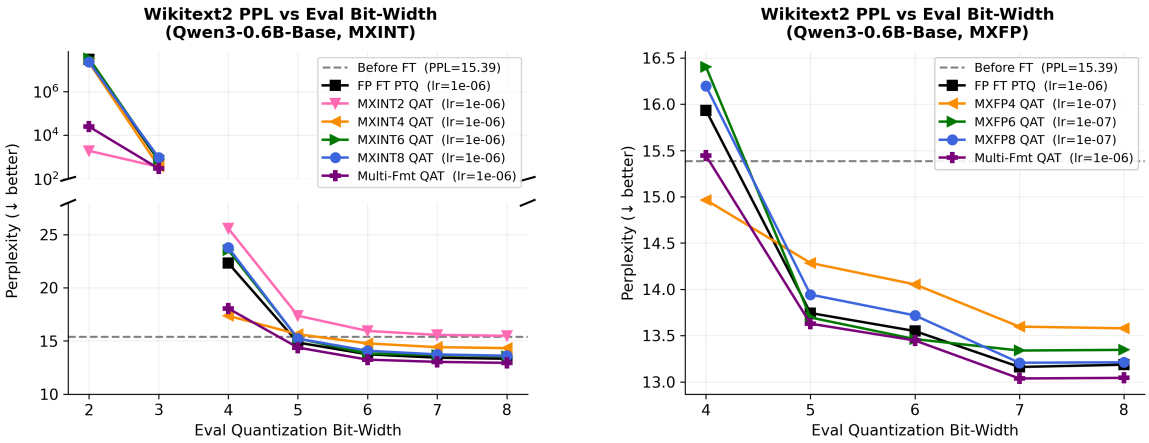

Figure 7: Multi-format QAT results for `Qwen3-0.6B-Base`.

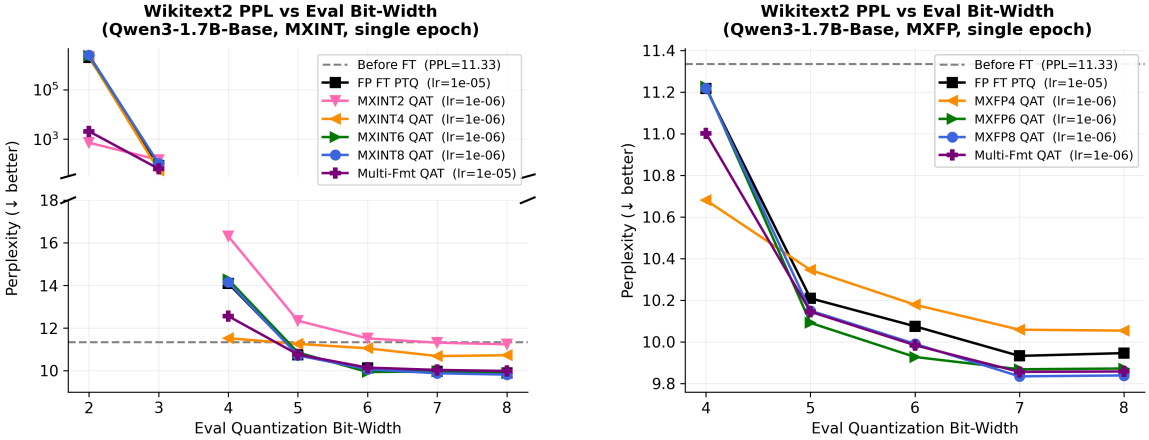

Figure 8: Multi-format QAT results for `Qwen3-1.7B-Base`.

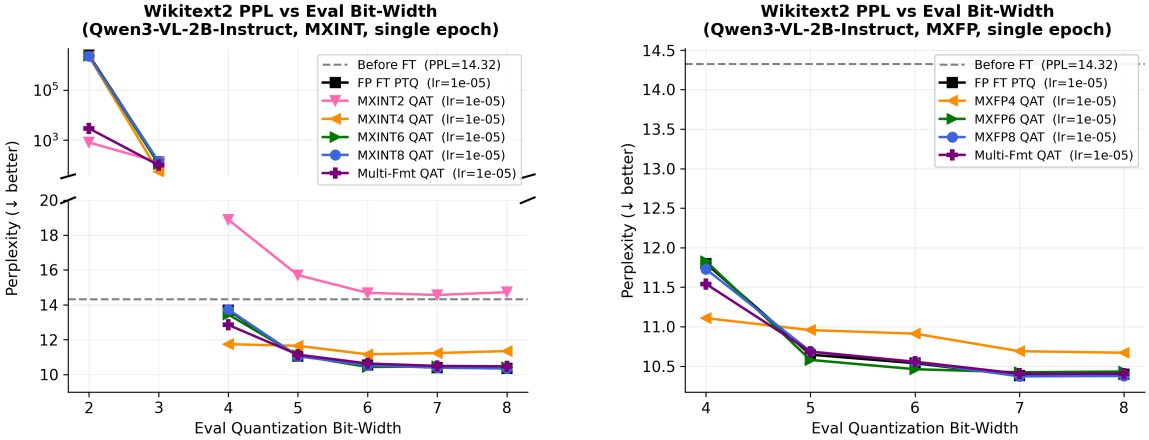

Figure 9: Multi-format QAT results for `Qwen3-VL-2B-Instruct`.

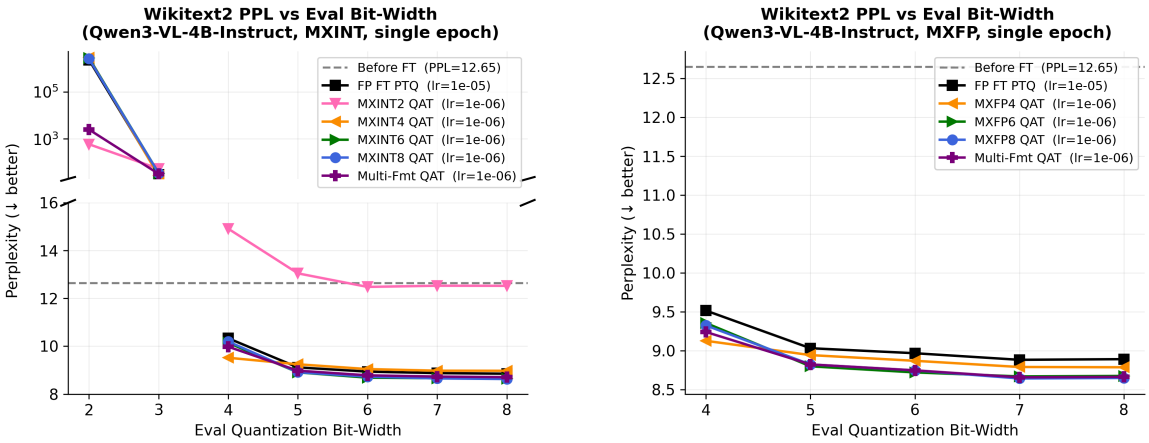

Figure 10: Multi-format QAT results for `Qwen3-VL-4B-Instruct`.

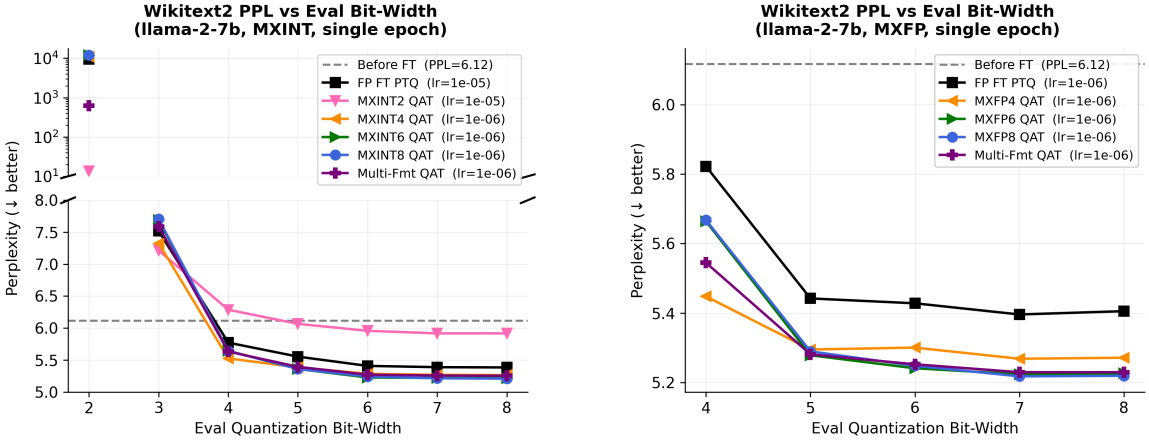

Figure 11: Multi-format QAT results for `llama-2-7b`.

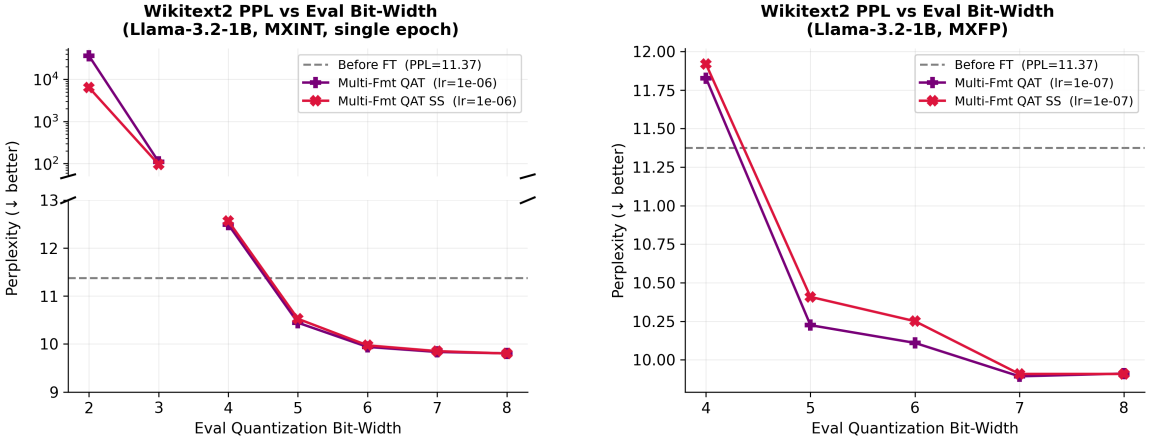

Figure 12: Multi-format QAT with Slice-and-Scale results for `Llama-3.2-1B`.

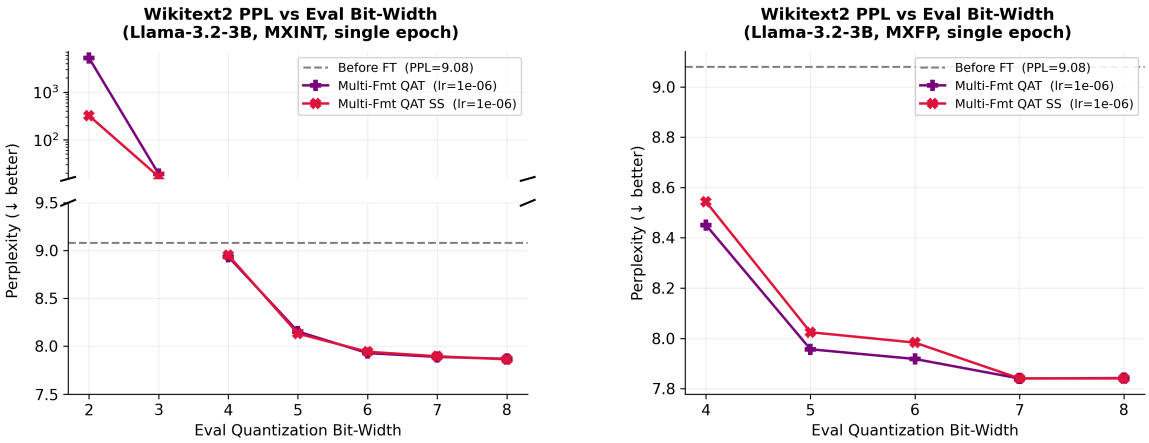

Figure 13: Multi-format QAT with Slice-and-Scale results for `Llama-3.2-3B`.

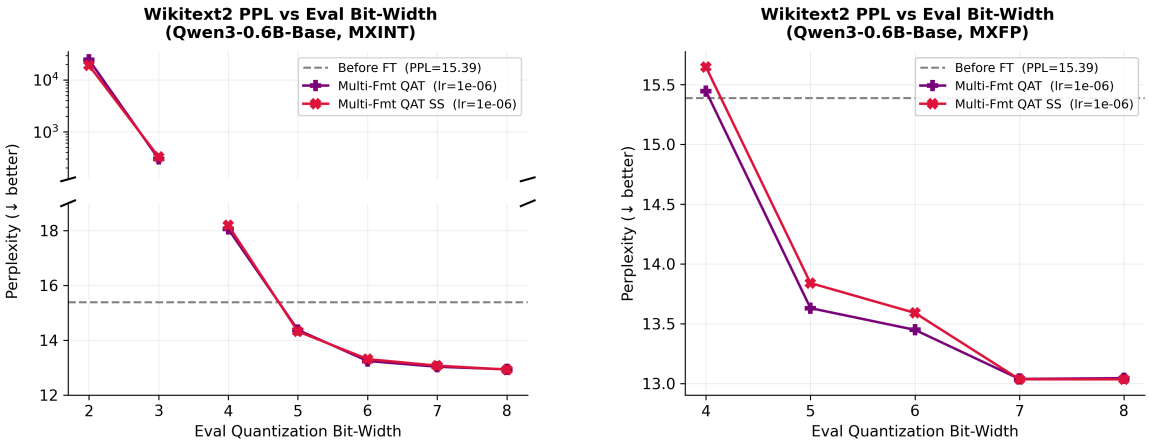

Figure 14: Multi-format QAT with Slice-and-Scale results for `Qwen3-0.6B-Base`.

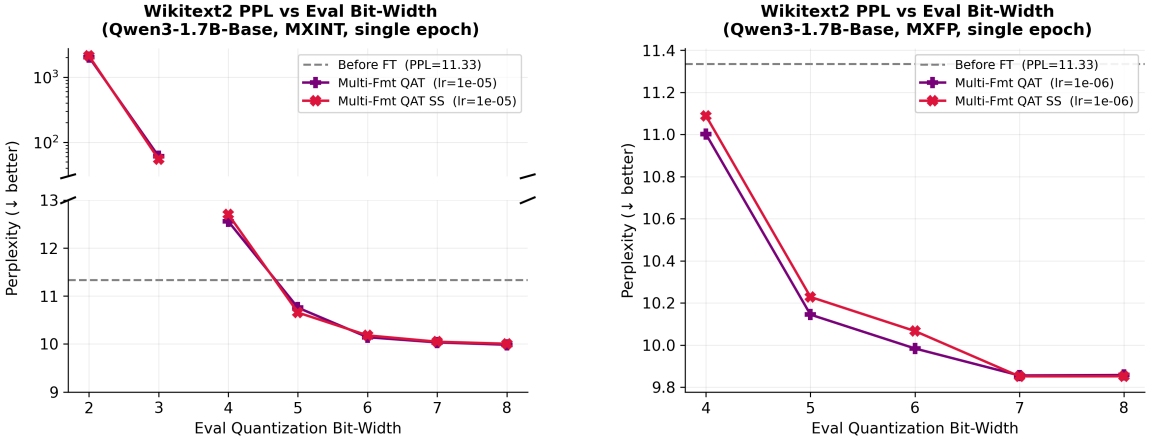

Figure 15: Multi-format QAT with Slice-and-Scale results for `Qwen3-1.7B-Base`.

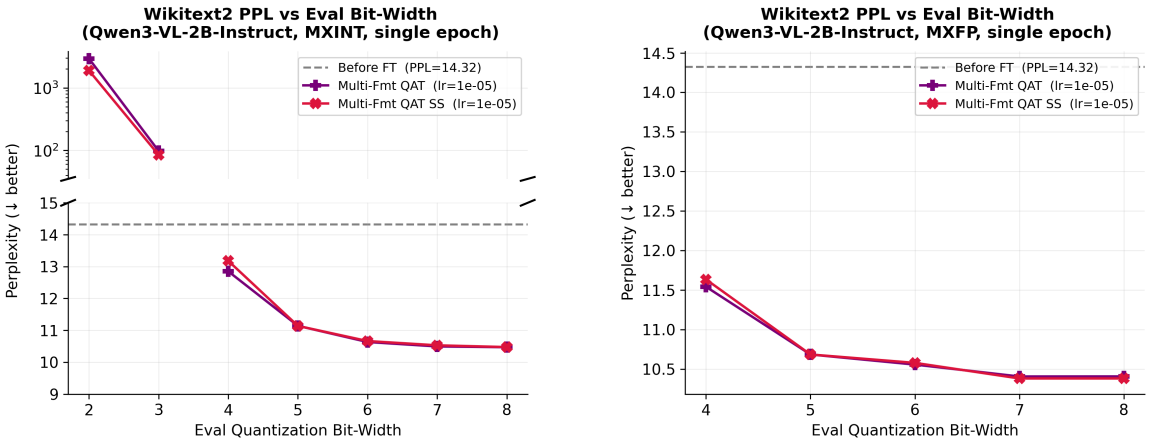

Figure 16: Multi-format QAT with Slice-and-Scale results for `Qwen3-VL-2B-Instruct`.

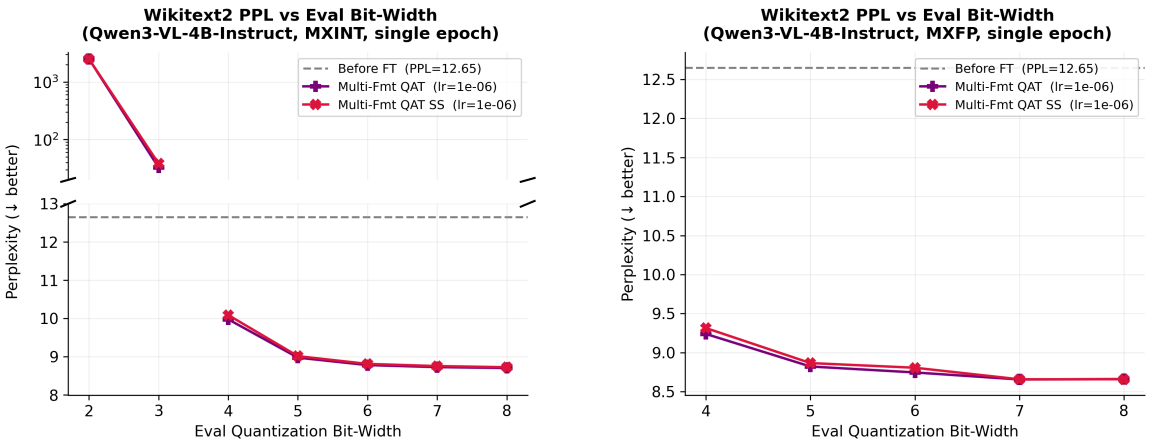

Figure 17: Multi-format QAT with Slice-and-Scale results for `Qwen3-VL-4B-Instruct`.

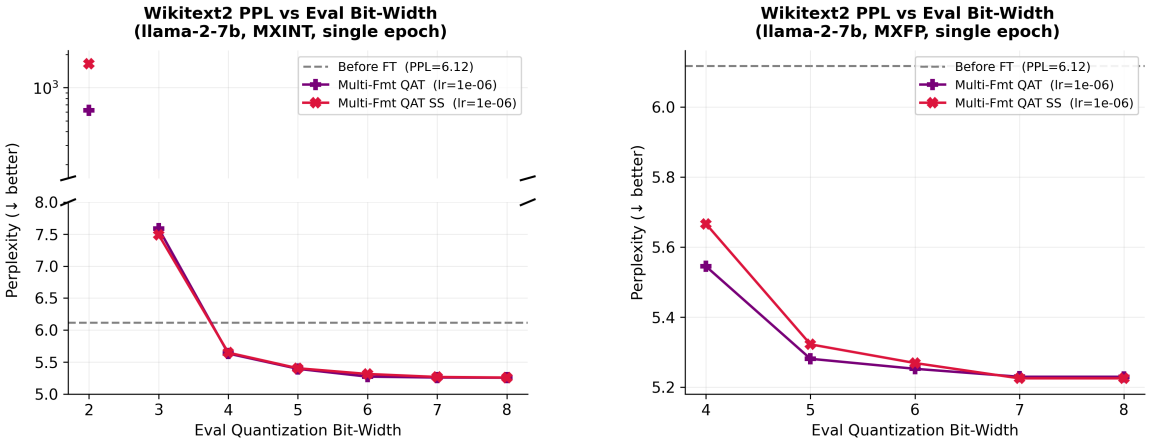

Figure 18: Multi-format QAT with Slice-and-Scale results for `llama-2-7b`.

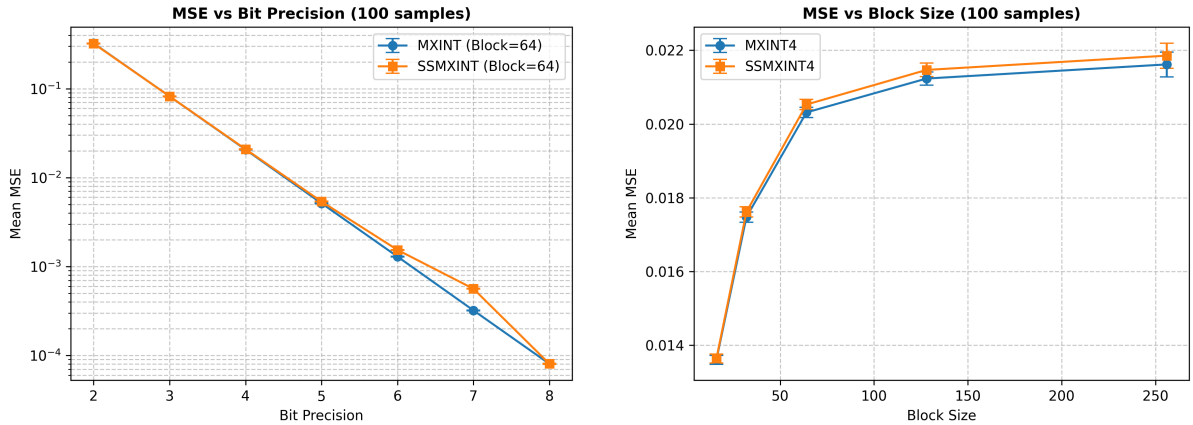

Figure 19: Tensor reconstruction MSE for direct MXINT quantization and Slice-and-Scale conversion (SSMXINT) on 100 random tensors. **Left:** Varying bit precision at block size 64. **Right:** Varying block size at 4-bit precision.

Model	QAT/FT Precision	PTQ Precision						
		MXINT2	MXINT3*	MXINT4	MXINT5*	MXINT6	MXINT7*	MXINT8
Llama-2-7B	Full Precision FT	25.0	28.1	39.3	41.1	42.3	42.5	42.7
	MXINT2 QAT	24.5	**29.4**	39.2	40.1	41.2	41.7	41.0
	MXINT4 QAT	**25.3**	28.6	**39.7**	**41.8**	42.7	42.6	**42.9**
	MXINT6 QAT	24.5	28.0	38.5	41.1	41.9	42.6	42.5
	MXINT8 QAT	24.3	27.1	39.1	40.7	42.1	42.4	42.7
	Multi-format QAT	23.8	28.3	39.0	41.4	**42.9**	**43.2**	**42.9**
Llama-3.2-1B	Full Precision FT	23.8	23.3	28.8	37.4	39.2	39.6	39.6
	MXINT2 QAT	**25.7**	23.9	28.2	37.4	39.3	39.5	39.8
	MXINT4 QAT	24.0	**24.6**	28.4	**38.6**	39.2	39.0	39.4
	MXINT6 QAT	23.5	23.8	28.2	37.8	39.3	39.0	39.7
	MXINT8 QAT	22.2	23.7	**29.1**	37.3	39.7	**39.8**	**40.4**
	Multi-format QAT	24.4	23.9	27.8	37.5	**40.1**	39.1	39.8
Llama-3.2-3B	Full Precision FT	26.5	31.1	50.2	55.8	56.1	55.7	55.7
	MXINT2 QAT	25.3	30.3	50.5	55.9	56.0	55.9	56.0
	MXINT4 QAT	26.7	**31.7**	**51.4**	**56.0**	56.4	55.7	**56.1**
	MXINT6 QAT	**26.8**	29.5	50.6	55.2	56.0	55.6	55.8
	MXINT8 QAT	**26.8**	31.0	50.3	55.6	56.0	**56.0**	55.8
	Multi-format QAT	23.8	31.4	50.6	55.3	**56.5**	55.7	55.8
Qwen3-0.6B	Full Precision FT	25.4	**25.5**	37.7	48.8	50.6	51.4	52.0
	MXINT2 QAT	23.9	**25.5**	38.1	48.2	51.4	51.3	52.0
	MXINT4 QAT	25.2	24.1	**42.0**	49.2	**52.3**	51.4	52.0
	MXINT6 QAT	25.1	24.7	37.6	49.5	50.5	**51.8**	**52.1**
	MXINT8 QAT	**25.5**	25.0	37.4	**49.7**	50.7	51.7	52.0
	Multi-format QAT	24.0	23.4	40.5	49.2	50.7	51.5	51.9
Qwen3-1.7B	Full Precision FT	24.3	25.7	54.5	61.5	63.3	63.4	63.5
	MXINT2 QAT	24.5	25.8	54.6	61.6	63.0	63.6	64.0
	MXINT4 QAT	**24.8**	**27.4**	55.1	61.4	**63.8**	64.0	**64.2**
	MXINT6 QAT	24.1	27.0	**55.7**	62.0	62.9	**64.1**	64.0
	MXINT8 QAT	24.5	26.1	54.7	61.8	63.4	64.0	64.0
	Multi-format QAT	24.4	26.4	55.5	**62.7**	63.2	63.0	63.0
Qwen3-4B	Full Precision FT	24.9	26.6	69.9	73.0	73.8	74.4	74.6
	MXINT2 QAT	23.9	26.5	**70.1**	72.7	74.1	74.0	74.0
	MXINT4 QAT	24.2	**28.5**	**70.1**	73.2	74.0	74.5	74.7
	MXINT6 QAT	24.5	28.0	69.8	**73.3**	73.9	**74.6**	**74.8**
	MXINT8 QAT	23.3	26.9	69.9	73.2	74.1	74.5	**74.8**
	Multi-format QAT	**25.4**	28.3	69.8	72.8	**74.2**	**74.6**	74.3

Table 4: MMLU accuracy (Hendrycks et al., 2021) for various models when quantized to the MXINT formats. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively.

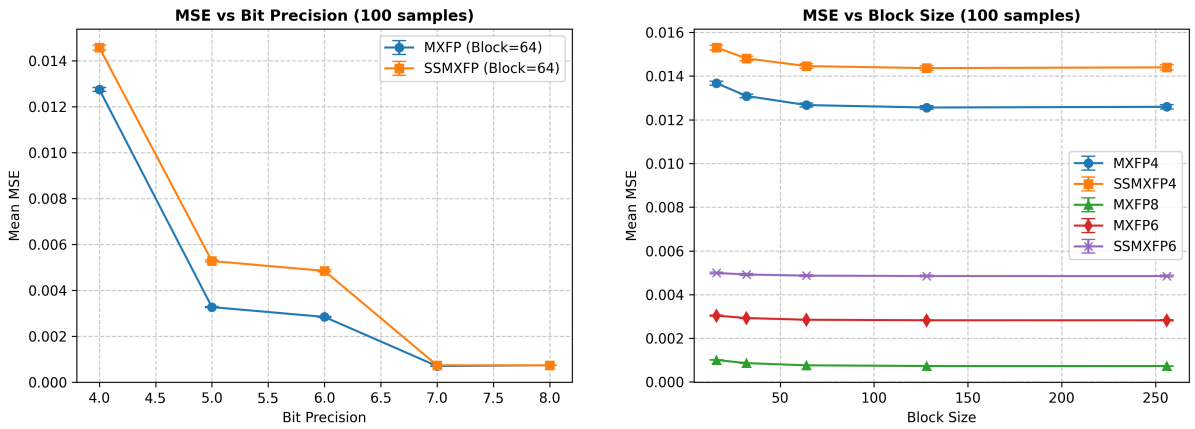

Figure 20: Tensor reconstruction MSE for direct MXFP quantization and Slice-and-Scale conversion (SSMXFP) on 100 random tensors. **Left:** Varying bit precision at block size 64. **Right:** Varying block size at 4-bit precision.

Model	QAT/FT Precision	PTQ Precision						
		MXINT2	MXINT3*	MXINT4	MXINT5*	MXINT6	MXINT7*	MXINT8
Llama-2-7B	Full Precision FT	18.4	25.0	27.8	28.0	**28.8**	28.4	**28.4**
	MXINT2 QAT	**20.6**	26.4	**28.0**	**29.0**	28.2	28.4	27.8
	MXINT4 QAT	17.8	26.4	27.6	27.6	27.8	**28.6**	28.2
	MXINT6 QAT	15.0	**26.8**	26.8	27.4	27.2	28.4	28.0
	MXINT8 QAT	16.6	25.6	27.4	28.0	27.4	**28.6**	28.0
	Multi-format QAT	18.4	24.8	26.8	27.4	27.6	28.4	**28.4**
Llama-3.2-1B	Full Precision FT	17.8	17.8	24.4	28.0	28.2	**28.0**	27.6
	MXINT2 QAT	18.2	18.0	24.0	26.8	28.2	26.2	26.6
	MXINT4 QAT	18.0	19.6	**25.2**	28.0	**28.6**	27.6	**28.4**
	MXINT6 QAT	17.6	17.6	23.8	**28.8**	28.4	27.6	27.8
	MXINT8 QAT	18.6	18.2	24.0	**28.8**	28.2	**28.0**	28.0
	Multi-format QAT	**20.0**	**21.2**	**25.2**	28.4	**28.6**	27.4	**28.4**
Llama-3.2-3B	Full Precision FT	**23.2**	25.4	**35.4**	33.4	36.4	35.0	34.8
	MXINT2 QAT	21.0	27.4	33.8	33.4	36.2	**35.2**	**35.0**
	MXINT4 QAT	19.6	27.4	33.4	33.0	36.0	34.6	33.8
	MXINT6 QAT	18.2	26.4	34.2	**34.0**	**36.8**	34.2	34.2
	MXINT8 QAT	19.4	25.6	34.8	33.4	36.2	34.4	34.4
	Multi-format QAT	20.6	**28.0**	35.0	33.4	35.8	34.8	34.6
Qwen3-0.6B	Full Precision FT	18.8	19.6	28.6	**33.4**	36.0	36.0	**35.8**
	MXINT2 QAT	19.8	18.6	27.6	29.6	34.8	35.0	35.0
	MXINT4 QAT	**20.6**	18.2	28.6	33.2	**38.2**	34.8	35.6
	MXINT6 QAT	19.6	19.0	28.6	31.8	37.2	**37.0**	**35.8**
	MXINT8 QAT	17.0	20.2	28.6	33.2	37.4	**37.0**	**35.8**
	Multi-format QAT	17.6	**20.6**	**28.8**	32.0	37.0	36.6	35.6
Qwen3-1.7B	Full Precision FT	**19.8**	**20.4**	36.8	40.6	40.8	**41.2**	42.6
	MXINT2 QAT	18.6	19.6	37.2	**41.6**	**42.8**	41.0	**43.4**
	MXINT4 QAT	15.4	19.8	35.8	40.4	42.4	41.0	42.8
	MXINT6 QAT	19.0	19.2	36.2	40.6	41.8	41.0	42.4
	MXINT8 QAT	16.8	19.4	36.8	40.2	41.4	40.6	42.4
	Multi-format QAT	18.4	**20.4**	**37.6**	40.8	40.8	40.8	43.0
Qwen3-4B	Full Precision FT	**21.0**	22.6	47.2	47.4	46.4	48.4	48.2
	MXINT2 QAT	19.6	24.2	47.8	**49.6**	47.8	50.4	**51.0**
	MXINT4 QAT	19.2	**24.4**	**49.6**	48.0	47.8	49.4	49.6
	MXINT6 QAT	16.2	23.8	47.6	48.0	**48.4**	**50.6**	**51.0**
	MXINT8 QAT	20.8	23.2	47.8	47.8	48.0	50.0	**51.0**
	Multi-format QAT	19.4	22.8	47.2	47.8	46.8	48.6	50.4

Table 5: MathQA normalized accuracy (Amini et al., 2019) for various models when quantized to the MXINT formats. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively.

Model	QAT/FT Precision	PTQ Precision						
		MXINT2	MXINT3*	MXINT4	MXINT5*	MXINT6	MXINT7*	MXINT8
Llama-2-7B	Full Precision FT	24.8	59.4	66.2	65.0	66.2	67.2	66.8
	MXINT2 QAT	**42.8**	**63.0**	66.0	64.8	65.8	66.6	66.6
	MXINT4 QAT	25.6	61.2	**66.4**	65.0	66.2	66.6	67.0
	MXINT6 QAT	26.8	60.4	65.8	65.2	**66.4**	**67.4**	**67.4**
	MXINT8 QAT	24.8	60.4	66.2	65.6	66.0	67.0	67.2
	Multi-format QAT	27.0	60.0	65.8	**65.8**	66.0	67.0	67.0
Llama-3.2-1B	Full Precision FT	24.0	35.6	51.6	57.2	56.4	**56.4**	56.4
	MXINT2 QAT	**26.8**	36.8	52.0	56.4	56.2	56.2	56.0
	MXINT4 QAT	26.4	**40.8**	**54.6**	56.6	56.4	56.0	56.6
	MXINT6 QAT	26.2	36.0	53.4	57.4	**56.6**	55.6	**57.0**
	MXINT8 QAT	25.4	35.0	52.6	**58.0**	56.2	56.0	56.4
	Multi-format QAT	25.4	39.6	52.6	56.4	55.2	55.8	56.0
Llama-3.2-3B	Full Precision FT	25.8	50.2	63.0	64.2	63.8	64.4	64.8
	MXINT2 QAT	**30.2**	**52.0**	63.4	64.8	64.8	**65.8**	65.0
	MXINT4 QAT	27.2	50.8	63.0	**65.2**	65.0	64.8	65.0
	MXINT6 QAT	27.4	50.2	63.4	**65.2**	65.0	65.6	65.6
	MXINT8 QAT	28.4	50.0	63.0	**65.2**	**65.6**	65.2	65.2
	Multi-format QAT	27.2	50.8	**63.6**	64.8	65.0	65.4	**65.8**
Qwen3-0.6B	Full Precision FT	27.8	30.0	49.0	48.4	51.0	52.6	**53.0**
	MXINT2 QAT	**29.4**	30.6	48.6	49.2	50.6	51.0	50.6
	MXINT4 QAT	24.0	**33.6**	48.2	**50.0**	51.0	52.0	51.8
	MXINT6 QAT	26.2	30.4	48.6	**50.0**	**51.6**	**52.8**	52.6
	MXINT8 QAT	25.4	31.0	48.0	48.2	51.4	52.4	52.6
	Multi-format QAT	24.4	31.6	**49.6**	49.8	50.6	52.6	51.2
Qwen3-1.7B	Full Precision FT	28.2	**41.4**	54.6	57.2	58.4	59.2	59.4
	MXINT2 QAT	25.0	41.0	54.4	57.0	57.2	57.6	59.0
	MXINT4 QAT	27.8	**41.4**	54.8	56.8	59.0	59.0	59.2
	MXINT6 QAT	24.4	41.2	54.8	**58.4**	58.8	58.6	58.8
	MXINT8 QAT	**30.4**	40.4	54.8	57.6	59.2	58.4	58.8
	Multi-format QAT	25.4	41.2	**55.0**	57.6	**60.2**	**60.2**	**60.4**
Qwen3-4B	Full Precision FT	**27.8**	44.2	**63.2**	**61.2**	62.6	62.4	62.2
	MXINT2 QAT	24.6	44.0	60.2	58.4	60.8	61.2	61.0
	MXINT4 QAT	26.4	**45.6**	62.6	60.2	63.0	62.6	61.6
	MXINT6 QAT	26.0	43.8	62.8	60.4	63.0	**62.8**	**62.8**
	MXINT8 QAT	24.8	44.2	62.6	60.6	62.8	62.6	62.6
	Multi-format QAT	26.8	44.4	63.0	60.6	**63.4**	62.6	62.6

Table 6: HellaSwag normalized accuracy (Zellers et al., 2019) for various models when quantized to the MXINT formats. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively.

QAT/FT Precision	PTQ Precision														
	MXFP4			MXFP5*			MXFP6			MXFP7*			MXFP8		
	MMLU	**Math**	**HellaS**	**MMLU**	**Math**	**HellaS**	**MMLU**	**Math**	**HellaS**	**MMLU**	**Math**	**HellaS**	**MMLU**	**Math**	**HellaS**
Llama-2-7B															
Full Prec. FT	37.4	28.8	63.6	42.0	**28.6**	65.8	42.4	**28.4**	66.4	42.3	28.4	**67.4**	**42.3**	**28.8**	**67.4**
MXFP4 QAT	**37.8**	**29.6**	63.8	**42.5**	28.4	**67.2**	42.5	28.2	**67.6**	**42.6**	28.2	67.0	42.1	27.6	67.0
MXFP6 QAT	36.7	29.0	64.2	41.9	**28.6**	66.0	42.5	**28.4**	66.6	41.8	28.4	67.0	41.9	28.4	67.2
MXFP8 QAT	36.7	**29.6**	**64.4**	42.1	27.8	66.2	42.4	28.2	66.4	42.4	**28.8**	66.8	42.1	28.4	66.8
Multi-format QAT	**37.8**	28.8	64.0	41.7	28.4	66.4	**42.6**	**28.4**	66.4	42.3	28.4	66.8	42.1	**28.8**	66.6
Llama-3.2-1B															
Full Prec. FT	30.6	**26.6**	55.0	39.2	25.8	**58.4**	**40.2**	26.0	**58.4**	39.4	27.6	**57.6**	39.7	27.6	**57.8**
MXFP4 QAT	**31.9**	**26.6**	56.2	40.2	**27.0**	57.0	38.9	**26.8**	57.8	40.1	**28.8**	57.0	39.9	**29.0**	56.4
MXFP6 QAT	31.1	24.8	56.0	40.2	25.0	58.2	39.5	26.4	57.4	40.2	28.0	57.2	39.8	27.4	57.0
MXFP8 QAT	31.0	25.6	**56.8**	40.4	24.8	57.8	39.4	26.0	57.8	**40.3**	28.2	56.8	**40.1**	28.0	57.2
Multi-format QAT	31.3	25.4	56.2	**40.8**	26.0	57.8	39.7	26.4	57.6	39.6	28.0	56.6	39.8	28.4	56.4
Llama-3.2-3B															
Full Prec. FT	53.7	**33.2**	**65.0**	56.2	**36.4**	64.8	56.1	35.2	63.6	55.8	**35.6**	64.0	55.7	**35.0**	64.0
MXFP4 QAT	**54.2**	32.4	64.0	56.2	35.2	64.4	55.9	35.2	64.4	**55.9**	35.2	64.6	55.9	34.4	65.0
MXFP6 QAT	53.7	32.0	64.2	**56.5**	35.2	65.2	55.7	35.6	64.4	**55.9**	34.6	**65.6**	56.0	33.8	**65.2**
MXFP8 QAT	54.0	32.0	64.6	56.3	**36.4**	64.8	55.9	**36.2**	**64.8**	55.7	34.8	65.0	**56.3**	33.8	65.0
Multi-format QAT	54.0	31.6	64.2	56.3	35.4	**65.8**	**56.3**	35.6	**64.8**	55.7	34.4	65.2	55.9	34.0	**65.2**
Qwen3-0.6B															
Full Prec. FT	**48.1**	**33.2**	50.6	49.8	37.6	50.8	50.6	36.2	52.0	51.1	34.0	**52.0**	50.8	34.2	51.6
MXFP4 QAT	47.6	32.6	51.0	50.1	36.6	51.0	51.2	**36.6**	52.2	51.4	**35.0**	51.8	51.2	**35.2**	**52.2**
MXFP6 QAT	48.0	31.6	**51.4**	**50.5**	37.4	**51.6**	51.1	36.4	**52.6**	**51.5**	34.4	51.0	51.0	34.0	51.2
MXFP8 QAT	47.9	32.2	50.8	**50.5**	**38.0**	51.0	**51.3**	**36.6**	52.4	51.3	33.6	51.8	51.1	33.0	51.4
Multi-format QAT	47.6	32.2	50.6	50.2	36.8	51.2	**51.3**	36.2	51.8	51.4	34.8	50.6	**51.3**	34.0	50.8
Qwen3-1.7B															
Full Prec. FT	**60.3**	38.8	56.4	61.8	40.4	**59.0**	62.5	39.2	**59.4**	62.5	40.6	**59.8**	62.6	41.4	**60.0**
MXFP4 QAT	59.9	**41.0**	57.4	63.2	40.8	57.2	63.4	39.8	58.6	64.1	**42.4**	58.0	**63.9**	**42.8**	58.8
MXFP6 QAT	59.8	38.0	57.2	63.4	40.4	58.2	63.6	**40.2**	59.0	64.0	40.6	58.2	63.7	41.6	58.8
MXFP8 QAT	59.9	39.4	**57.6**	63.1	**41.0**	58.8	63.3	40.0	58.8	63.7	41.2	58.2	63.7	41.2	58.4
Multi-format QAT	59.9	40.0	56.6	**63.7**	40.2	58.4	**63.8**	**40.2**	58.8	**64.2**	41.2	59.0	63.7	41.6	58.8
Qwen3-4B															
Full Prec. FT	70.8	47.8	**61.6**	72.8	47.4	63.4	73.2	47.6	62.4	74.3	48.0	62.0	74.3	47.8	62.2
MXFP4 QAT	**72.2**	47.4	61.2	**73.8**	**48.8**	**64.0**	**74.1**	48.6	**63.0**	74.4	49.6	**63.0**	**74.5**	49.8	**63.0**
MXFP6 QAT	71.4	**48.4**	61.0	73.2	48.2	63.4	73.2	48.4	62.8	**74.5**	49.8	62.8	**74.5**	48.6	62.8
MXFP8 QAT	71.3	47.8	**61.6**	73.3	48.6	63.8	73.7	49.6	**63.0**	74.0	49.8	**63.0**	74.3	49.4	62.8
Multi-format QAT	72.0	47.6	61.4	73.7	**48.8**	63.2	74.0	**49.8**	62.8	74.4	**50.2**	62.6	74.4	**50.0**	62.4

Table 7: Comparison of MMLU (Hendrycks et al., 2021), MathQA (Amini et al., 2019), and HellaSwag (Zellers et al., 2019) accuracies for various models when quantized to the MXFP formats. Each row corresponds to a different QAT/FT training precision, and each column shows the PTQ evaluation precision. Columns marked with * denote precisions not seen during multi-format/QAT training. QAT, PTQ, and FT denote Quantization Aware Training, Post Training Quantization, and Finetuning, respectively.