Quantization Error Propagation: Revisiting Layer-Wise Post-Training Quantization

Yamato Arai[1,2] Yuma Ichikawa[1,3]

[1]Fujitsu Limited [2]The University of Tokyo [3]RIKEN Center for AIP

Introduction

Motivation: Efficient LLM Deployment

Large Language Models (LLMs) deliver strong performance but are highly *memory-* and *compute-*intensive, which makes deployment in edge or latency-sensitive environments challenging.

Layer-wise Post-Training Quantization (PTQ)

- Quantizes weights on a *layer-by-layer* basis using a small calibration set.
- No backpropagation or retraining, low additional compute and memory.
- Widely used in practice (QuIP [1], GPTQ [2], AWQ [3], RTN, etc.).

However, recent progress in layer-wise PTQ is saturating:

- Accuracy degrades significantly in low-bit regimes (e.g., 2–3 bits).
- Empirical observations suggest that **quantization errors accumulate and grow across layers.**

Goal and Contribution

Goal

Revisiting the core design of layer-wise PTQ by making the *quantization errors propagation across layers explicit*, while preserving full compatibility with existing PTQ pipelines and keeping computational overhead low.

Our Contribution

Diagnose the problem: We identify and quantify how quantization errors *accumulate* and *amplify* as depth increases in standard layer-wise PTQ.

Introduce Quantization Error Propagation (QEP), a general framework that:

- Reformulates per-layer objectives to explicitly *propagate* and *compensate* upstream errors.
- Adds a tunable propagation strength that balances error reduction, robustness, and runtime.
- **Remains fully orthogonal and *plug-and-play* with GPTQ, AWQ, QuIP, RTN, and other layer-wise methods.**

Demonstrate consistent gains: QEP provides robust improvements across models and benchmarks, with particularly strong benefits at low-bit (INT2/INT3) quantization.

Background: Layer-wise PTQ

The layers are quantized sequentially, $\{\boldsymbol{W}_l\}_{l=1}^{L} \to \{\widehat{\boldsymbol{W}}_l\}_{l=1}^{L}$, by solving the following:

$$\min_{\widehat{\boldsymbol{W}}_l \in \mathbb{Q}^{n_l \times d_l}} \left\| \boldsymbol{W}_l \mathrm{X}_l - \widehat{\boldsymbol{W}}_l \mathrm{X}_l \right\|_F^2, \tag{1}$$

where the set $\mathbb{Q} \subset \mathbb{R}$ denotes the discrete quantization set, consisting of a finite set of 2^b distinct quantization levels. Two common choices for activation quantization are:

- Full Precision Activations: $\mathrm{X}_l = \boldsymbol{X}_l$, computed with original weights.
- Quantized Activations: $\mathrm{X}_l = \widehat{\boldsymbol{X}}_l$, computed with weights.

This yields efficient Hessian-based implementations, $\mathrm{H}_l = \mathrm{X}_l \mathrm{X}_l^\top$, used in GPTQ, AWQ, QuIP, and related methods.

Bottleneck: Quantization Error Accumulation & Growth

We quantize the first $n = 10$ Transformer blocks and keep the remainder in full precision. Let $f_m(\boldsymbol{X})$ and $\widehat{f}_m(\boldsymbol{X})$ denote the original and partially quantized outputs at Transformer block m.

$$\Delta_m = \left\| f_m(\boldsymbol{X}) - \widehat{f}_m(\boldsymbol{X}) \right\|_F^2. \tag{2}$$

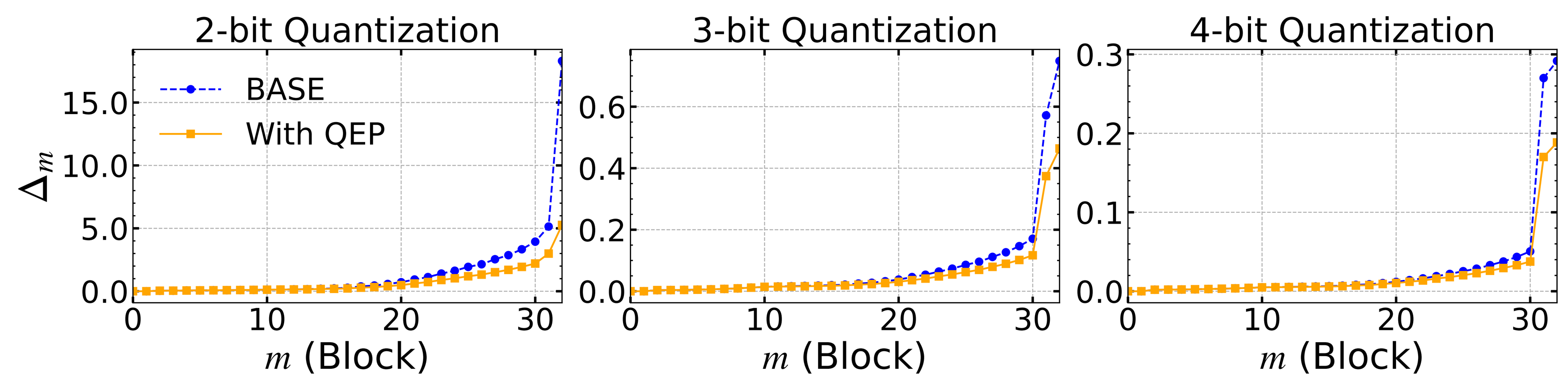


Figure 1. Error accumulation and growth in Llama-2-7B. **RTN shows near-exponential growth of Δ_m within quantized blocks and continued growth in later full-precision blocks, while QEP suppresses both.**

Within the quantized region, **errors grow nearly exponentially**. They continue to increase in later *unquantized* layers. The core issue is *independent* layer-wise optimization: each layer is optimized separately and ignores accumulated errors.

QEP: Quantization Error Propagation

To address this error accumulation, QEP minimizes the following objective rather than matching outputs under a shared input X_l:

$$\min_{\widehat{\boldsymbol{W}}_l \in \mathbb{Q}^{n_l \times d_l}} \left\| \boldsymbol{W}_l \boldsymbol{X}_l - \widehat{\boldsymbol{W}}_l \widehat{\boldsymbol{X}}_l \right\|_F^2. \tag{3}$$

$\boldsymbol{X}_l$ is computed with full-precision upstream weights, while $\widehat{\boldsymbol{X}}_l$ uses quantized upstream weights. This forces $\widehat{\boldsymbol{W}}_l$ to approximate the original layer and **compensate** for the accumulated error $\boldsymbol{\delta}_l = \boldsymbol{X}_l - \widehat{\boldsymbol{X}}_l$. However, this objective is no longer governed solely by the standard Hessian.

Weight Correction: Keeping Hessian Efficiency

The objective in Eq. (3) is equivalent to (when $\alpha_l = 1$):

$$\min_{\widehat{\boldsymbol{W}}_l \in \mathbb{Q}^{n_l \times d_l}} \left\| \boldsymbol{W}_l^*(\alpha_l) \widehat{\boldsymbol{X}}_l - \widehat{\boldsymbol{W}}_l \widehat{\boldsymbol{X}}_l \right\|_F^2, \quad \boldsymbol{W}_l^*(\alpha_l) = \boldsymbol{W}_l + \alpha_l \boldsymbol{W}_l \boldsymbol{\delta}_l \widehat{\boldsymbol{X}}_l^\top \widehat{\boldsymbol{H}}_l^{-1}. \tag{4}$$

$\alpha_l \in [0, 1]$ controls the correction strength: $\alpha_l = 0$ recovers **baseline** layer-wise PTQ; $\alpha_l = 1$ gives full QEP correction; intermediate values trade off **error reduction** and **robustness**.

This formulation preserves the GPTQ-style quadratic structure, enabling reuse of Hessian-based acceleration and existing implementations.

Theorem (Quantization Error Guarantee)

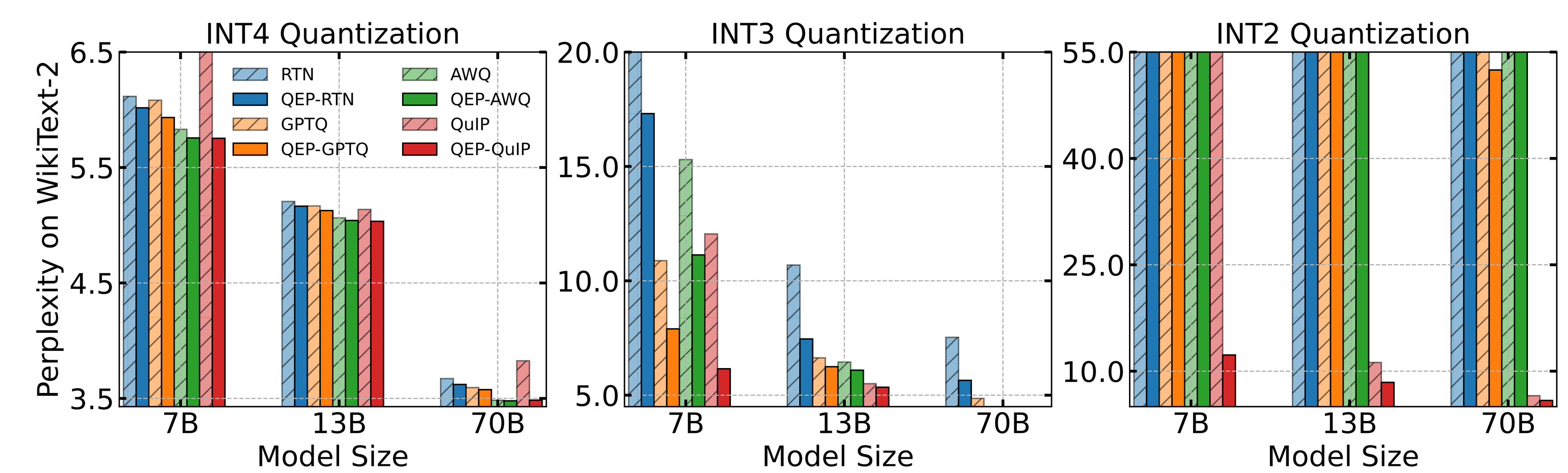
Consider an L-layer network $f_{\boldsymbol{\theta}}(\boldsymbol{X}) = \sigma_L(\boldsymbol{W}_L \cdots \sigma_1(\boldsymbol{W}_1 \boldsymbol{X}))$, where each activation σ_l is Lipschitz. Let $\widehat{\boldsymbol{\theta}}_{\mathrm{QEP}}$ and $\widehat{\boldsymbol{\theta}}_{\mathrm{BASE}}$ denote the parameters obtained with the QEP objective with propagation and the standard independent layer-wise objective, respectively. Then

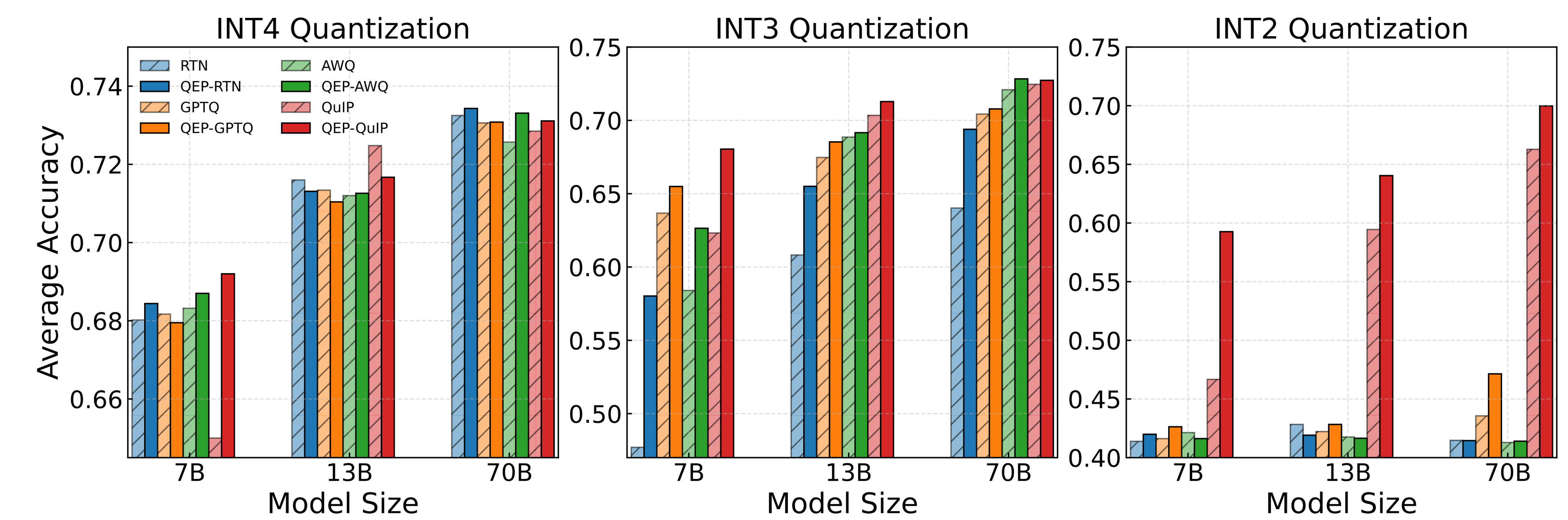
$$\left\| f_{\boldsymbol{\theta}}(\boldsymbol{X}) - f_{\widehat{\boldsymbol{\theta}}_{\mathrm{QEP}}}(\boldsymbol{X}) \right\|_F \le \left\| f_{\boldsymbol{\theta}}(\boldsymbol{X}) - f_{\widehat{\boldsymbol{\theta}}_{\mathrm{BASE}}}(\boldsymbol{X}) \right\|_F. \tag{5}$$

Experiments: Setup

- **Quantization**: RTN, GPTQ, AWQ, and QuIP with W4A16, W3A16, and W2A16.
- **Model**: Llama-2 (7B, 13B, 70B), Llama-3-8B, and Mistral-7B.
- **QEP**: Default $\alpha_l = 1/2$, except for MLP layers in the 70B model, where some layers use $\alpha_l = 0$.
- **Eval**: Perplexity (PPL) on WikiText and zero-shot accuracy on ARC-Easy, PIQA, and StoryCloze.

Result: QEP Improves Layer-wise PTQ (Llama2; other models in paper)





QEP consistently strengthens the performance of layer-wise PTQ methods.

- **INT4/INT3**: QEP further improves strong baselines such as AWQ.
- **INT2**: Baseline methods often diverge (high PPL, low ACC), whereas QEP restores usable performance. QEP-enabled QuIP achieves SOTA PPL among layer-wise PTQ methods at INT2.

Runtime & Robustness

- QEP introduces only modest overhead; computing its correction terms is far cheaper than full GPTQ or AWQ optimization.
- QEP shows strong robustness to calibration data, substantially mitigating the overfitting issues observed in GPTQ and AWQ.

References

[1] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M. De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36:4396–4429, 2023.

[2] Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. Gptaq: Efficient finetuning-free quantization for asymmetric calibration. In *Forty-second International Conference on Machine Learning*, 2025.

[3] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

arai.yamato@fujitsu.com, ichikawa.yuma@fujitsu.com