UNDERSTANDING THE DIFFICULTY OF LOW-PRECISION POST-TRAINING QUANTIZATION FOR LLMS

Zifei Xu, Sayeh Sharify, Wazin Yazar, Tristan Webb & Xin Wang
d-Matrix Corporation
Santa Clara, CA 95054, USA
{xuzifei,sayehs,wyazar,twebb,xwang}@d-matrix.ai

ABSTRACT

Large language models of high parameter counts are computationally expensive, yet can be made much more efficient by compressing their weights to very low numerical precision. This can be achieved either through post-training quantization by minimizing local, layer-wise quantization errors, or through quantization-aware fine-tuning by minimizing the global loss function. In this study, we discovered that, under the same data constraint, the former approach nearly always fared worse than the latter, a phenomenon particularly prominent when the numerical precision is very low. We further showed that this difficulty of post-training quantization arose from stark misalignment between optimization of the local and global objective functions. Our findings suggested limited utility in minimization of local quantization error and the importance of direct quantization-aware fine-tuning, in the regime of large models at very low precision.

1 INTRODUCTION

Large language models (LLMs) are remarkably powerful and increasingly deployed in real-world applications in recent years (Xu et al., 2023; Wang et al., 2024; Zheng et al., 2023; Wu et al., 2023). Despite their effectiveness, LLMs are typically trained with weights in `float16`, making post-training compression techniques such as quantization crucial for efficient inference (Yao et al., 2022; Kim et al., 2023a;b; Park et al., 2022; Frantar et al., 2022a; Lin et al., 2023).

Two distinct methods are prevalent for LLM weight quantization. The first one, *quantization-aware fine-tuning* (QAFT) optimizes a differentiable global objective just like at pretraining time, with quantization operations acting on the model weights, requiring backpropagation and gradient updates (Chee et al., 2024; Dettmers et al., 2023; 2024; Hu et al., 2021; Li et al., 2023; Huang et al., 2024). The second method minimizes layer-wise local quantization errors, with no backpropagation needed (Frantar et al., 2022b; Xiao et al., 2023; Lee et al., 2024; Lin et al., 2023). Both methods typically require a small amount of data and should theoretically achieve similar result since minimizing the local losses should in turn minimize the global loss, and *vice versa*.

Let us first introduce some formal notations. Denote the LLM network by $f_{\boldsymbol{W}} : \mathbb{D} \to \mathbb{R}^{|\mathcal{V}|}$, which takes text input $\boldsymbol{x} \in \mathcal{D} \subset \mathbb{D}$ and generates logits $f_{\boldsymbol{W}}(\boldsymbol{x})$ across vocabulary $\mathcal{V}$. Assume it is well-pretrained on language-modeling objective, *i.e.* $\boldsymbol{W} = \arg\min_{\boldsymbol{W}'} \mathrm{NLL}(\boldsymbol{x}|f_{\boldsymbol{W}'})$ on a training data set $\mathcal{D}$. Here $\boldsymbol{W} \triangleq (\boldsymbol{W}_1, \cdots, \boldsymbol{W}_L)$ collects all layer-wise weights $\boldsymbol{W}_l$ with $l \in \{1, \cdots, L\}$ indexing layers. In the following we will also use vector notation $\boldsymbol{w} \in \mathbb{R}^D$ to represent an equivalent, flattened version of $\boldsymbol{W}$ in the D-dimensional weight space. We denote $Q : \mathbb{R}^D \to \mathbb{R}^D$ as the weight (fake-)quantization function (for procedural details see Appendix B.2). We refer to the fake-quantized weights as *round-to-nearest* (RTN), $\boldsymbol{W}_{\text{RTN}} \triangleq Q(\boldsymbol{W})$.

QAFT methods essentially keep optimizing the global objective with quantization in the loop, *i.e.*

$$\boldsymbol{W}_{\text{QAFT}} = \arg\min_{\boldsymbol{W}'} \mathrm{NLL}(\boldsymbol{x}|f_{Q(\boldsymbol{W}')}), \tag{1}$$

where $\boldsymbol{x} \in \mathcal{D}$; due to the non-differentiability of $Q(\cdot)$, straight-through estimator is commonly used in gradient back-propagation.

In contrast, layer-wise quantization error minimization techniques seek to solve L distinct layer-wise optimization problems:

$$\boldsymbol{W}_l = \arg\min_{\boldsymbol{W}_l'} \mathrm{MSE}\left(Q(\boldsymbol{W}_l')\boldsymbol{x}_l, \boldsymbol{W}_l\boldsymbol{x}_l\right) \tag{2}$$

$$= \arg\min_{\boldsymbol{W}_l'} \|Q(\boldsymbol{W}_l')\boldsymbol{x}_l - \boldsymbol{W}_l\boldsymbol{x}_l\|^2, \tag{3}$$

where $\boldsymbol{x}_l$ is the input to the l-th layer when $\boldsymbol{x} \in \mathcal{D}$ is passed through the network f. One popular method of this kind, namely GPTQ (Frantar et al., 2022a), derived from *optimal brain compression* (OBC) (Frantar & Alistarh, 2022), seeks to solve the layer-wise mean squared error (MSE) minimization problem by finding a less steeply rising direction in this local quadratic loss landscape, through efficient computation of the Hessian of the local MSE losses. We denote a GPTQ-optimized network's weights by $\boldsymbol{W}_{\mathrm{GPTQ}}$.

While intuitively, both approaches should produce well-generalizing quantized networks, our systematic study reveals a misalignment. Local loss minimization often leads to suboptimal global loss, and *vice versa*, especially at low quantization precisions. Through loss-landscape analysis, we offered an explanation of why the post-training quantization by local loss minimization struggles to produced well-generalizing quantized networks, guiding future LLM quantization practices.

2 METHODS

We experimented with 11 models from 3 major model families: GPT-2 (Radford et al., 2019), OPT (Zhang et al., 2022), and Llama 2 (Touvron et al., 2023), using the WikiText-2 dataset (Merity et al., 2016). The training set consisted of 128 examples at the model's maximum sequence length. For weight quantization, we tested 5 integer data types (`int8`, `int6`, `int4`, `int3`, `int2`), applying PyTorch's per-tensor symmetric quantization calibrated via the HistogramObserver mechanism. GPTQ was implemented with layer-wise tuning of the dampening factor, and QAFT was conducted for 8 epochs using a grid search for learning rate optimization. Details on datasets, calibration, and hyperparameter tuning are provided in Appendix B.

3 RESULTS

3.1 GPTQ VS. QAFT UNDER THE SAME DATA CONSTRAINT

First, we investigated whether minimization of the local MSE losses through GPTQ aligned with minimization of the global NLL loss through QAFT on test dataset. Figure 1 compares the global NLL and local MSE losses for `gpt2-xl`, `opt-6.7b` and `llama2-7b`, which were the largest models among each model family we experimented with. In the upper rows, global NLL loss for QAFT are consistently lower than those for GPTQ, as indicated by the position of the colored dots below the diagonal identity line. In some cases, GPTQ even resulted in higher NLL losses than RTN. In contrast, the lower row shows that GPTQ always reduced layer-wise MSE losses as designed, whereas QAFT maintains or even increases the MSE losses from RTN.

These results indicate a misalignment: minimizing global NLL loss via QAFT does not necessarily reduce local MSE, and minimizing local MSE loss via GPTQ does not necessarily reduce global NLL. This misalignment is particularly evident in low-precision formats, where QAFT significantly outperforms GPTQ. Since the generalization capability of a quantized model is measured by the global NLL loss, minimizing layer-wise MSEs seems to be an ineffective surrogate, in light of the observed misalignment. In Appendix F, we showed that even a few QAFT iterations can produce better generalizing quantized model than GPTQ.

3.2 EXPLANATION OF THE MISALIGNMENT FROM A LOSS LANDSCAPE PERSPECTIVE

Next, we investigated why minimization of local layer-wise MSE losses did not align with minimizing the global NLL loss, especially at low numerical precision. To address this, we analyzed global NLL loss landscape in the D-dimensional weight space around the pretrained weights $\boldsymbol{w} \in \mathbb{R}^D$.

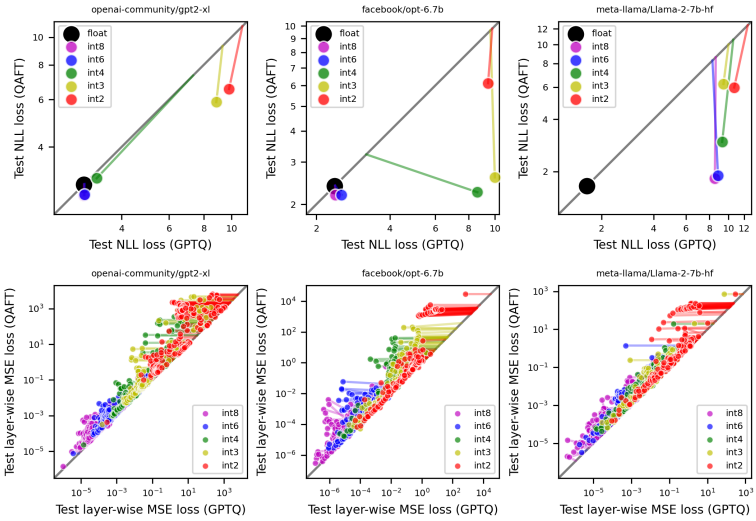

Figure 1: **Misalignment between minimization of the global** NLL **loss** (QAFT) **and minimization of the local layer-wise** MSE **losses** (GPTQ). The upper row shows global NLL losses, and the lower row presents layer-wise MSE losses for three models (one per column). Data points compare QAFT (vertical axis) to GPTQ (horizontal axis). The gray diagonal indicates identity. Black dots (if present) represent full-precision models, while colored dots mark losses after QAFT and GPTQ. Colored lines originating from each dot intersect the diagonal, showing RTN-quantized model losses for the corresponding format. In the lower row, symbols represent individual quantized layers.

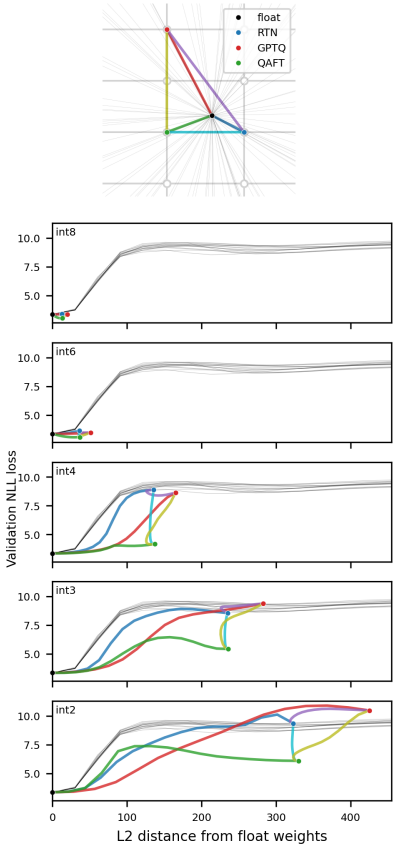

Figure 2: **Loss landscape analysis of quantized model weights.** Data illustrated here are from `opt-125m`, a network small enough for numerous loss evaluations. In the legend at the top, we illustrate the mapping strategy in a 2-dimensional cartoon, which captures key concepts in the D-dimensional weight space. The black dot in the middle marks the pretraining convergence $\boldsymbol{w}$. The continuous loss landscape is probed first by measuring loss at $\boldsymbol{w} + \lambda\hat{\boldsymbol{e}}$, *i.e.* pretrained weight subject to random perturbation $\hat{\boldsymbol{e}} \sim \mathcal{S}^D$ sampled uniformly from the D-dimensional unit sphere. We sweep $\lambda \in \mathbb{R}^+$ (thin, light gray lines emanating from the black circle) to map the radial loss landscape along a specific random direction $\hat{\boldsymbol{e}}$. The gray grid represents the representable weight values prescribed by the weight quantizer $Q(\cdot)$, out of which we show three key quantized weights under question: $\boldsymbol{w}_{\text{RTN}} = Q(\boldsymbol{w})$ (blue circle), $\boldsymbol{w}_{\text{QAFT}}$ (green circle), and $\boldsymbol{w}_{\text{GPTQ}}$ (red circle). We measure the loss function at these key points as well as those along the linear segment resulting from a convex combination of two of these (colored lines). We plot the radial loss landscape (NLL loss against ℓ_2 distance from $\boldsymbol{w}$) for validation loss (training loss landscape exhibits similar trend and can be found in Appendix E). Graphical symbols of points and segments are consistent with the legend at the top.

We measured NLL loss along a number of random directions emanating from $\boldsymbol{w}$ (the thin, light gray lines in Figure 2). The radial mapping revealed that $\boldsymbol{w}$ sat near the bottom of an attractive basin. Within this near-convergence region, the loss landscape appeared quadratic, with similar profiles across various random directions. This aligns with prior analysis of the loss Hessian spectra showing a dominant bulk subspace with a few outliers (Sagun et al., 2017; 2018; Ghorbani et al., 2019). Hence, a random linear combination of Hessian eigenvectors would likely stay within the bulk subspace, resulting in similar radial profiles. Beyond the near-convergence locality, the loss landscape deviated from a quadratic approximation and plateaued at a high level, defining the attractive basin's radius $R(\boldsymbol{w})$.

We charted the loss landscape of the quantized weights via RTN, QAFT and GPTQ, and of linear interpolation segments between them (colored circles and line segments in Figure 2). Key observations include:

- $\boldsymbol{w}_{\text{RTN}}$ (blue circle): The closest quantized weight to $\boldsymbol{w}$, It was within the attractive basin for `int8` and `int6`, near the border for `int4`, and outside it for `int3` and `int2`.
- $\boldsymbol{w}_{\text{GPTQ}}$ (red circle): Further from $\boldsymbol{w}$ than $\boldsymbol{w}_{\text{RTN}}$, but along a flatter loss direction (red segment). The loss values at $\boldsymbol{w}_{\text{GPTQ}}$ were low for `int8` and `int6`, but high for `int4`, `int3` and `int2`, similar to $\boldsymbol{w}_{\text{RTN}}$ at the same precision.
- $\boldsymbol{w}_{\text{QAFT}}$ (green circle): Slightly further from $\boldsymbol{w}$ than $\boldsymbol{w}_{\text{RTN}}$, yet achieving significantly lower loss level.
- Connectivity: Quantized weights beyond the basin's radius were not simply connected to $\boldsymbol{w}$. Linear interpolation between $\boldsymbol{w}$ and these weights showed non-monotonic loss profiles with ridges in the middle.

Based on these observations, we are able to explain experimental findings from previous sections. Minimizing layer-wise local MSE by GPTQ effectively identified less steeply rising directions from $\boldsymbol{w}$, resulting in lower loss levels when $\boldsymbol{w}_{\text{GPTQ}}$ remained within the attractive basin (e.g. `int8` and `int6`). However, for larger quantization-induced perturbations, the shallow rise near $\boldsymbol{w}$ is insufficient, resulting in high loss values at $\boldsymbol{w}_{\text{GPTQ}}$ (e.g. `int3` and `int2`). In contrast, QAFT consistently follow less steep directions from $\boldsymbol{w}_{\text{RTN}}$, even when $\boldsymbol{w}_{\text{QAFT}}$ was distance from $\boldsymbol{w}$. This led to significantly lower loss levels, albeit in a separate attractive basin, similar to patterns observed in sparse networks (Evci et al., 2020).

Our results suggested that the alignment between local and global loss minimization depends on the relationship between the attractive basin size $R(\boldsymbol{w})$ and the quantization-induced weight perturbation $\|\Delta\boldsymbol{w}\| = \|\boldsymbol{w}_{\text{RTN}} - \boldsymbol{w}\|$. When $\|\Delta\boldsymbol{w}\|$ is substantially greater than $R(\boldsymbol{w})$, misalignment occurs.

4 Discussion

In this work, we systematically compared the effectiveness of post-training quantization GPTQ and quantization-aware fine-tuning (QAFT), where GPTQ minimized layer-wise local quantization errors and QAFT minimized the global training loss. Under the same low training data constraint (128 training examples), QAFT consistently outperformed GPTQ.

Surprisingly, we observed that GPTQ's local MSE minimization and QAFT's global NLL minimization often misalign, especially pronounced at very low precision quantization. Through loss landscape analysis, we elucidated that such misalignment was prominent when the weight perturbation due to quantization was significantly larger than the radius of the attractive basin at the pretraining convergence.

Our findings reveal a lack of correlation between a quantized network's generalization ability and its local quantization errors, challenging the common reliance on local error minimization metrics for evaluating quantization schemes. We urge caution in generalizing the utility of local-error-based post-training quantization and provided a new perspective for understanding the difficulty and identification of conditions where these methods are effective.

REFERENCES

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv preprint arXiv:2104.08758*, 2021.

Utku Evci, Fabian Pedregosa, Aidan Gomez, and Erich Elsen. The difficulty of training sparse neural networks. *arXiv preprint arXiv:1906.10732*, 2020.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488, 2022.

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

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2022b.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. *arXiv preprint arXiv:1901.10159*, 2019.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and Xiaojuan Qi. BiLLM: Pushing the limit of post-training quantization for llms. *arXiv preprint arXiv:2402.04291*, 2024.

Hyesung Jeon, Yulhwa Kim, and Jae-joon Kim. L4Q: Parameter efficient quantization-aware training on large language models via lora-wise lsq. *arXiv preprint arXiv:2402.04902*, 2024.

Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joonsuk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. *Advances in Neural Information Processing Systems*, 36, 2024.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023a.

Young Jin Kim, Rawn Henry, Raffy Fahim, and Hany Hassan Awadalla. Finequant: Unlocking efficiency with fine-grained weight-only quantization for llms. *arXiv preprint arXiv:2308.09723*, 2023b.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. OWQ: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13355–13364, 2024.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2018.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. LoftQ: Lora-fine-tuning-aware quantization for large language models. *arXiv preprint arXiv:2310.08659*, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. AWQ: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*, 2024.

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

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

Gunho Park, Baeseong Park, Minsub Kim, Sungjae Lee, Jeonghoon Kim, Beomseok Kwon, Se Jung Kwon, Byeongwook Kim, Youngjoo Lee, and Dongsoo Lee. Lut-gemm: Quantized matrix multiplication based on luts for efficient inference in large-scale generative language models. *arXiv preprint arXiv:2206.09557*, 2022.

Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2017.

Levent Sagun, Utku Evci, V. Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2018.

Yuzhang Shang, Zhihang Yuan, Qiang Wu, and Zhen Dong. PB-LLM: Partially binarized large language models. *arXiv preprint arXiv:2310.00034*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. BitNet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*, 2023.

Tiannan Wang, Jiamin Chen, Qingrui Jia, Shuai Wang, Ruoyu Fang, Huilin Wang, Zhaowei Gao, Chunzhao Xie, Chuou Xu, Jihong Dai, et al. Weaver: Foundation models for creative writing. *arXiv preprint arXiv:2401.17268*, 2024.

Tianyu Wu, Shizhu He, Jingping Liu, Siqi Sun, Kang Liu, Qing-Long Han, and Yang Tang. A brief overview of chatgpt: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1122–1136, 2023.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. A paradigm shift in machine translation: Boosting translation performance of large language models. *arXiv preprint arXiv:2309.11674*, 2023.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhonghua Zheng, Lizi Liao, Yang Deng, and Liqiang Nie. Building emotional support chatbots in the era of llms. *arXiv preprint arXiv:2308.11584*, 2023.

A Related work

Post-training quantization Quantization reduces the memory and computational demands of neural networks by converting weights or activations from full precision to lower precision formats, like 8-bit integers. Post-training quantization (PTQ) techniques achieve this without retraining the network. Several PTQ techniques have emerged to reduce deployment cost of LLMs. Some methods focus on identifying outlier features that are difficult to quantize and either represent them with a higher precision, *e.g.* LLM.int8() (Dettmers et al., 2022), or mitigate their quantization error by adding additional operations to the network, such as SmoothQuant (Xiao et al., 2023), AWQ (Lin et al., 2023), and OWQ (Lee et al., 2024).

Other PTQ methods employ adaptive rounding techniques to reduce quantization errors. For instance, OBC (Frantar & Alistarh, 2022) quantizes weights one-by-one in a specific order based on the approximate second-order information of the weights, and adjusts the remaining weights to minimize the quantization error. GPTQ (Frantar et al., 2022a), also known as OPTQ (Frantar et al., 2022b), extends OBC by enabling parallel quantization of weight matrices, applying the same quantization order to all rows of the weight matrix. Similarly, QuIP (Chee et al., 2024) uses adaptive rounding to minimize a quadratic proxy objective of the quantization error.

Quantization-aware fine-tuning Fine-tuning LLMs ensures task-specific adapdations but is computationally expensive. Parameter efficient fine-tuning (PEFT) reuses some of the pretrained model's parameters and selectively fine-tune a subset of parameters for the downstream tasks. Common PEFT methods include LoRA (Hu et al., 2021), QLoRA (Dettmers et al., 2024), L4Q (Jeon et al., 2024), LoftQ (Li et al., 2023), Prefix and Prompt Tuning (Li & Liang, 2021; Lester et al., 2021; Qin & Eisner, 2021; Liu et al., 2023), IA3 (Liu et al., 2022), and PEQA (Kim et al., 2024).

LoRA, QLoRA, L4Q, and LoftQ freeze pretrained model parameters and fine-tune on inserted task-specific adapters. Adapters undergo low rank decomposition to further reduce the trainable parameters (Hu et al., 2021; Dettmers et al., 2024; Jeon et al., 2024; Li et al., 2023). In prefix and prompt tuning, the parameters of an original large language model are frozen, and only the trainable prompt embeddings are fine-tuned (Li & Liang, 2021; Lester et al., 2021; Qin & Eisner, 2021; Liu et al., 2023). Similarly, in IA3 only the hidden state parameters are fine-tuned (Liu et al., 2022).

PEQA is a memory-efficient fine-tuning method for quantized LLMs that updates only the quantization scale, keeping the integer matrix frozen (Kim et al., 2024).

Ultra-low precision pretraining Recent efforts aim to binarize LLMs by quantizing them to ultra-low bit-widths. For instance, PB-LLM (Shang et al., 2023) and SpQR (Dettmers et al., 2023) employ a mixed-precision quantization technique, representing the majority of the weights with a single bit while retaining a small portion of the weight in the original high precision or `int8`. BiLLM (Huang et al., 2024) utilize Hessian information to identify salient and non-sailents weights, employing binary residual approximation of salient weights and grouped quantization of non-salient weights. BitNet (Wang et al., 2023) replaces transformer linear layers by a binary linear layer, retaining other components in high-precision. BitNet b1.58 (Ma et al., 2024) is an extension of BitNet that utilizes ternary quantization for its weights, achieving better accuracy in downstream tasks compared to BitNet.

Loss landscape analysis Analysis of deep neural networks' loss landscape has long been a tool toward understanding of the generalization properties of the optimized model (Li et al., 2018; Ghorbani et al., 2019; Sagun et al., 2017; 2018), as well as in explaining difficulties arising in efficient network optimization processes (Evci et al., 2020).

B METHODS

B.1 MODELS AND DATA SET

We experimented with 11 models from 3 model families, namely GPT-2 (Radford et al., 2019), OPT (Zhang et al., 2022) and Llama 2 (Touvron et al., 2023). All models were served by the Hugging Face Model Hub [1].

We used the WikiText-2 (Merity et al., 2016) data set in all experiments, see Appendix C for generalization to other datasets. Unless noted otherwise, the training split used in all procedures was of 128 examples, each of the maximum sequence length supported by the model being experimented. Entire split was used for validation and test data.

B.2 NUMERICAL DATA TYPES AND QUANTIZER CALIBRATION

We experimented with 5 integer data types for weight quantization, namely `int8`, `int6`, `int4`, `int3` and `int2`, of varied numerical precision. For integer with B-bit precision, encoding range is symmetric, *i.e.* $\{-2^{B-1}+1, \cdots, 2^{B-1}-1\}$, excluding -2^{B-1}; for example, `int2` quantization is effectively ternary, $\{-1, 0, 1\}$.

We used PyTorch's quantization API [2] to obtain the weight fake-quantization functions $Q(\cdot)$. The quantization scheme used was per-tensor symmetric [3]. The scaling factor a_l's of the fake-quantizer was determined by mean squared quantization error minimization, *i.e.* $a_l = \arg\min_a \|Q_a(\boldsymbol{W}_l) - \boldsymbol{W}_l\|^2, \forall l \in \{1, \cdots L\}$, by means of PyTorch's `HistogramObserver` mechanism. For simplicity, only weights of layers in the transformer stack were quantized, sparing other weights such as those in embedding and prediction head layers.

[1] All models were accessed from official repositories hosted by `https://huggingface.co/` in April 2024.

[2] See `https://pytorch.org/docs/stable/quantization.html`.

[3] Note that different quantization schemes and/or different data types of the same precision often result in different generalization quality of the quantized network, determined by the granularity of a channel/group/block-wise scheme. Here we choose to use the simplest scheme in order to conduct a controlled scientific study with fewest confounding factors.

B.3 GPTQ

We followed the original GPTQ procedure (Frantar et al., 2022a) exactly except for one change described below. As reported by other adopters of GPTQ, *e.g.* `https://huggingface.co/TheBloke`, different choices of the dampening factor as a hyperparameter in the GPTQ procedure could lead to outcomes of varied qualities. In order to eliminate this confounding factor, we performed hyperparameter tuning for the GPTQ dampening factor over a search space of $\{10^{-3}, \cdots, 10^{4}\}$, in a layer-wise manner. As mentioned above, 128 examples from the training split were used for the GPTQ procedure, consistent with the original work (Frantar et al., 2022a).

B.4 Quantization-aware fine-tuning (QAFT)

To perform QAFT, the exact same 128 training examples were used as above, for a fair comparison. We ran QAFT for 8 epochs, *i.e.* in total 1024 training iterations (see Appendix F for a study on the effect of number of iterations). Straight through estimator (STE) (Hinton et al., 2012; Bengio et al., 2013) was employed in back-propagation through quantization functions. Only quantized weights in the transformer network were subject to gradient updates. We used the AdamW optimizer (Loshchilov & Hutter, 2017) without weight decay, and with a linear learning rate decay schedule of 1 order of magnitude. We ran a hyperparameter grid search over 4 initial learning rate values, $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$ for all models except for `Llama-2-7b-hf`, in which case it was $\{10^{-6}, 10^{-3}\}$ (see Appendix G for more details on hyperparameter tuning); best NLL loss over the validation split was chosen.

C Generalization on different datasets

To assess the generalizability of our findings, we repeated a subset of experiments on C4 (Dodge et al., 2021) and LAMBADA (Paperno et al., 2016) datasets. In both cases, we observed consistent misalignment between the minimization of global and local loss (Figure 3), as well as similar trends in the loss landscape (Figure 4).

D Scaling with LLM size and numerical precision

We examined how the generalization abilities of quantized models produced by RTN, QAFT and GPTQ, scaled with model size and numerical precision. To capture this, we plotted test NLL loss against total transformer weight size (in gigabytes) for all models on a single trade-off graph (Figure 5).

As shown in Figure 5, QAFT consistently dominated the Pareto front across model families. For GPT-2 models, QAFT quantized solutions at `int6` and `int8` occupied the Pareto front, outperforming both smaller full-precision models and larger models quantized with lower precisions. Similar trends were observed for the OPT and Llama 2 families, where QAFT quantized models at `int4`, `int6`, and `int8` dominated.

Colored vertical strips in Figure 5 highlight test NLL differences between QAFT and GPTQ. Models quantized by QAFT outperforms GPTQ more significantly at lower precisions, such as `int2`, `int3` and `int4`, aligning with the earlier observation that misalignment between global NLL and local MSE losses is more pronounced at low numerical precision.

E Loss Landscape for training loss

Loss landscape analysis of quantized model weights for training loss exhibits similar trends to validation loss, as shown in Figure 6

F Effect of number of QAFT iterations

Although we used 8 epochs (1024 iterations) to achieve the most optimal performance, we discovered that QAFT for 1 epoch is sufficient to outperform GPTQ, as shown in Figure 7.

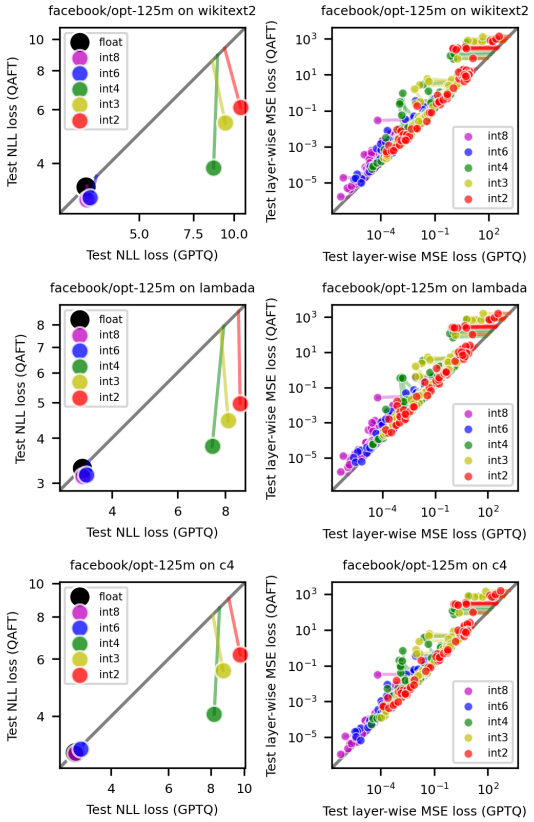

Figure 3: **Misalignment between minimization of the global NLL loss (by QAFT) and minimization of the local layer-wise MSE losses (by GPTQ) on different datasets.** Follows the same conventions as Figure 1.

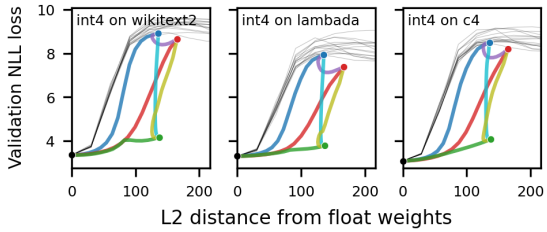

Figure 4: **Loss landscape analysis of quantized model weights on different datasets**. Plots showing results for `opt-125m` quantized in `int4`. Follows the same conventions as Figure 2.

G Optimal learning rate for QAFT

While performing hyperparameter grid search on initial learning rate over the set $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$, we discovered that the optimal choices strongly correlated with the model size and the quantization precision, which is supported by Figure 8. For a particular model, we observe that QAFT at higher precision such as `int8`, `int6` and `int4` favored low learning rates, whereas QAFT at lower precision such as `int3` and `int2` preferred high learning rates. For a specific format, larger models in the same model family preferred lower learning rates while smaller models in the model family prefers higher learning rates.

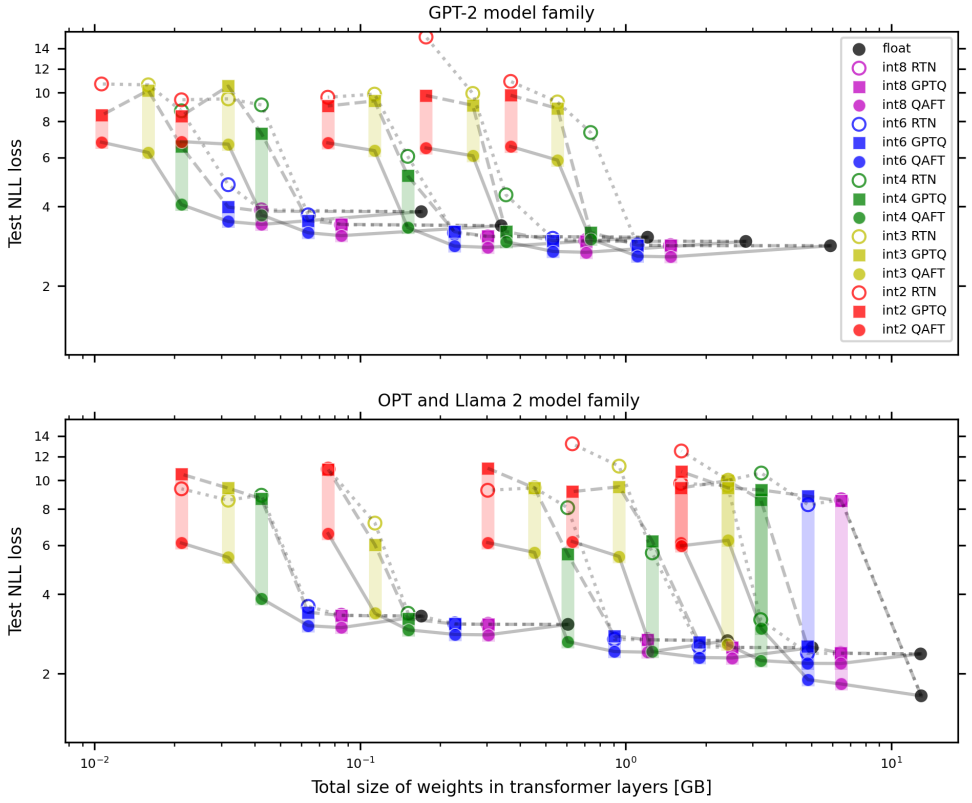

Figure 5: **Tradeoff between quantized model generalization and its weight size.** **Upper**: models from the GPT-2 model family: `distilgpt2`, `gpt2`, `gpt2-medium`, `gpt2-large` and `gpt2-xl`. **Lower**: models from the OPT and Llama 2 families: `opt-250m`, `opt-350m`, `opt-1.3b`, `opt-2.7b`, `opt-6.7b` and `Llama-2-7b-hf`. Black circles represent the full-precision models. Hollow colored circles are RTN-quantized models, solid colored circles QAFT-quantized models, and solid colored squares GPTQ-quantized models. Dotted, dashed and solid gray lines connect quantized solutions from the same model produced by RTN, GPTQ and QAFT, respectively. We highlight the difference between GPTQ- and QAFT-quantized models with colored, transparent, vertical strips, for each quantized model.

H Optimal GPTQ dampening factor

In GPTQ, dampening factor is multiplied with the average diagonal value in Hessian matrix H and added to the diagonal entries of H to achieve better numerical stability (Frantar et al., 2022a). In our experiments, we did a grid search on dampening factor for each layer with the objective of minimizing layer-wise MSE. We visualized the best dampening factor for each quantized layer but failed to find any meaningful patterns.

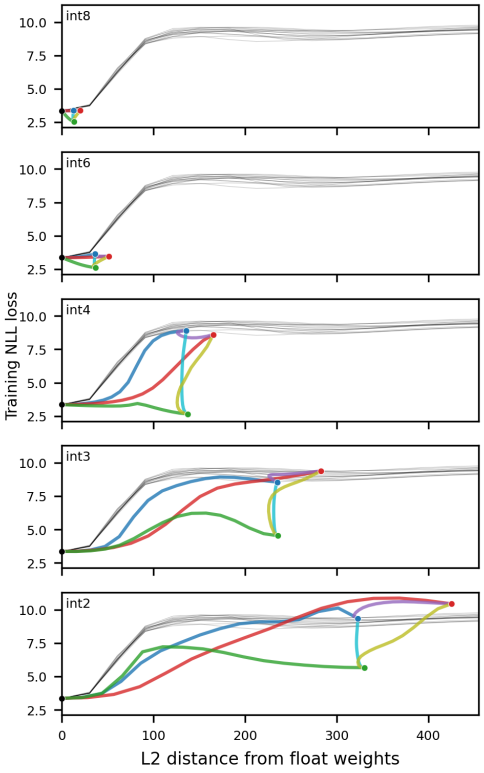

Figure 6: **Loss landscape analysis of quantized model weights on training loss.** Follows same conventions as Figure 2

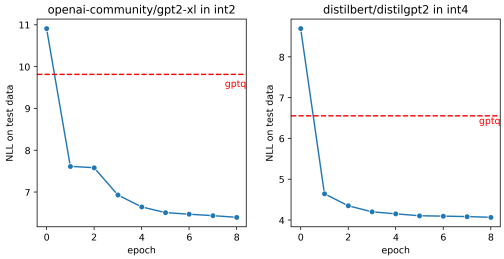

Figure 7: NLL on test data after each epoch of fine-tuning. Epoch 0 represents the NLL of RTN, which is the starting point for fine-tuning. The horizontal red line marks the NLL after GPTQ.

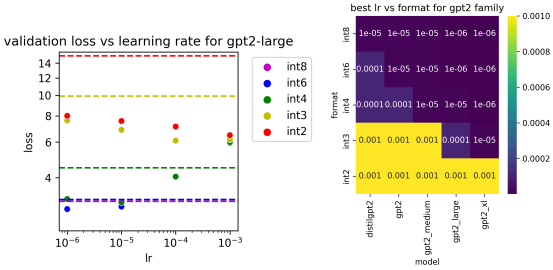

Figure 8: Left: validation NLL loss of `gpt2-large` after fine-tuning. Dashed lines represent the loss after RTN. Right: best learning rates for GPT-2 family.