Interactions Across Blocks in Post-Training Quantization of Large Language Models

Khasmamad Shabanovi
Recogni, Technical University of Munich
khasmamad.shabanovi@recogni.com

Lukas Wiest
Recogni
lukas.wiest@recogni.com

Vladimir Golkov
Technical University of Munich
vladimir.golkov@tum.de

Daniel Cremers
Technical University of Munich
cremers@tum.de

Thomas Pfeil
Recogni
thomas.pfeil@recogni.com

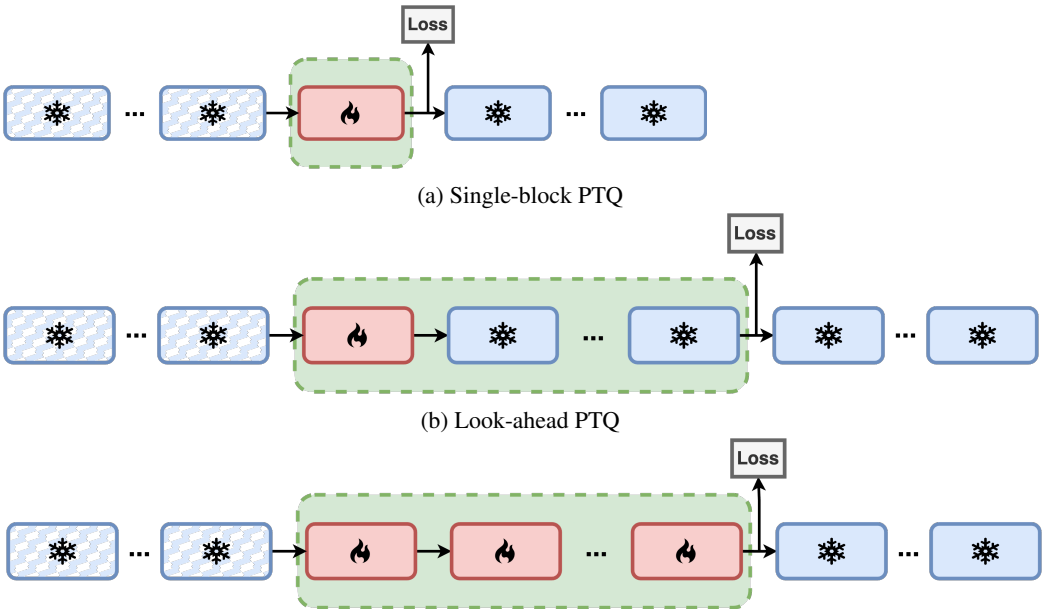
Abstract

Post-training quantization is widely employed to reduce the computational demands of neural networks. Typically, individual substructures, such as layers or blocks of layers, are quantized with the objective of minimizing quantization errors in their pre-activations by fine-tuning the corresponding weights. Deriving this local objective from the global objective of minimizing task loss involves two key simplifications: assuming substructures are mutually independent and ignoring the knowledge of subsequent substructures as well as the task loss. In this work, we assess the effects of these simplifications on weight-only quantization of large language models. We introduce two multi-block fine-tuning strategies and compare them against the baseline of fine-tuning single transformer blocks. The first captures correlations of weights across blocks by jointly optimizing multiple quantized blocks. The second incorporates knowledge of subsequent blocks by minimizing the error in downstream pre-activations rather than focusing solely on the quantized block. Our findings indicate that the effectiveness of these methods depends on the specific network model, with no impact on some models but demonstrating significant benefits for others.

1 Introduction

Recently, large language models (LLMs) [Zhang et al., 2022, Touvron et al., 2023, Jiang et al., 2023] have transformed the field of natural language processing, achieving impressive results on various challenging language tasks [Wei et al., 2022, Bubeck et al., 2023]. Nevertheless, these models, often containing billions of parameters, typically demand substantial computational power. Post-training quantization (PTQ) has emerged as a practical method for reducing the size and computational requirements of LLMs without the need for retraining and requiring only a small set of calibration data [Frantar et al., 2022, Xiao et al., 2023, Lin et al., 2024, Shao et al., 2024]. By converting high-precision weights and activations to lower-precision representations, PTQ enables the deployment of LLMs on resource-constrained devices, expanding their applicability in real-world scenarios. In this study, we focus on weight-only quantization, as model weights are the primary factor impacting memory bandwidth and, consequently, the runtime of LLM inference [Kim et al., 2023].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) Single-block PTQ

(b) Look-ahead PTQ

(c) Multi-block PTQ

Figure 1: Commonly, for SB-PTQ, each block is independently optimized with the loss attached to its output (a). We propose LA-PTQ (b) and MB-PTQ (c), where the reconstruction loss is attached to a subsequent block. For LA-PTQ, still a single block is optimized (red) while all other blocks are not modified (blue). The blocks that contribute to the computation of the gradient are highlighted in green. For MB-PTQ, multiple blocks are jointly optimized. All the previous blocks are already quantized and fine-tuned (indicated with zig-zag lines).

Current PTQ methods independently optimize layers [Hubara et al., 2021, Frantar et al., 2022, Xiao et al., 2023, Lin et al., 2024] or blocks of layers [Li et al., 2021, Cheng et al., 2023, Shao et al., 2024]. Such block-wise optimization, although computationally efficient, ignores correlations across blocks and disregards the knowledge of subsequent blocks and the task loss. Despite the potential significance of these limitations, there has been little research addressing them.

Nagel et al. [2020] derived the local, layer-wise objective from the global, task-based objective. Their study investigates how leaving out information about the later layers and task loss affects optimizing the first layer of ResNet-18. They discover that optimizing without this information works better than with it. Li et al. [2021] extended this study to blocks with an arbitrary number of layers. Based on their theoretical analysis, they suggest that incorporating the squared derivative of the task loss with respect to pre-activations into the optimization objective can improve performance. Additionally, their empirical results show that fine-tuning multiple layers together, such as in Residual Bottleneck Blocks, produces better outcomes compared to fine-tuning individual layers in CNN architectures. However, they note that increasing the number of layers increases the generalization error, likely due to the limited number of calibration samples. Ding et al. [2023] increase the scope of fine-tuning to multiple transformer blocks. While they show that this improves the task performance, their results rely on additional techniques that obscure the direct effect of increasing cross-block dependencies.

In this work, we propose two methods that allow us to evaluate the effect of the two key simplifications in the derivation of the local from the global objective. The term block refers to a transformer block unless stated otherwise. The first method bundles multiple blocks together during optimization enabling second-order interactions across these blocks. We name this method multi-block PTQ (MB-PTQ). The second method fine-tunes each block with the target of minimizing the error in the output of a downstream block. This effectively allows each block to "look ahead" and inform its optimization about the effect on this downstream block. We name this method look-ahead PTQ (LA-PTQ). We formalize MB-PTQ and LA-PTQ in Section 2, compare them with the baseline of single-block PTQ (SB-PTQ) in Section 3, and discuss the results in Section 4.

2 Methods

In this section, we revisit the derivation of the local objective from the global one and define quantization. Building upon the visualizations in Figure 1, we formally introduce SB-PTQ, LA-PTQ, and MB-PTQ.

Global to Local Objective Finding the optimal quantization can be formulated as the following global optimization objective that minimizes the error in the task loss caused by quantization:

$$\arg\min_{\Delta\mathbf{w}} \mathbb{E}\left[\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w} + \Delta\mathbf{w}) - \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w})\right], \tag{1}$$

where $\mathcal{L}$ is the task loss, $\mathbf{x}$ are the inputs, $\mathbf{y}$ are the true labels, $\mathbf{w}$ is the flattened vector of all model weights, and $\Delta\mathbf{w}$ is the perturbations to the weights introduced by quantization. The expectation is over $\mathbf{x}$ and $\mathbf{y}$.

Nagel et al. [2020] approximated this objective by a second-order Taylor expansion around $\mathbf{w}$ where $\mathbf{H}^{(\mathbf{w})}$ is the Hessian of the task loss with respect to weights $\mathbf{w}$ and the first-order term is ignored since the model is assumed to have converged:

$$\arg\min_{\Delta\mathbf{w}} \mathbb{E}\left[\Delta\mathbf{w}^T \mathbf{H}^{(\mathbf{w})} \Delta\mathbf{w}\right]. \tag{2}$$

To obtain the local, layer-wise objective two simplifications are applied. First, the layers are assumed to be mutually independent, resulting in a block-diagonal structure for the Hessian matrix. Consequently, each layer ℓ can be optimized separately in Equation 2:

$$\arg\min_{\Delta\mathbf{w}^{(\ell)}} \mathbb{E}\left[\Delta\mathbf{w}^{(\ell)^T} \mathbf{H}^{(\mathbf{w}^{(\ell)})} \Delta\mathbf{w}^{(\ell)}\right]. \tag{3}$$

However, this objective remains computationally challenging, as calculating $\mathbf{H}^{(\mathbf{w}^{(\ell)})}$ requires computing the Hessian $\nabla^2_{\mathbf{z}^{(\ell)}}\mathcal{L}$ of the task loss with respect to the pre-activations $\mathbf{z}^{(\ell)}$:

$$\mathbf{H}^{(\mathbf{w}^{(\ell)})} = \mathbb{E}\left[\mathbf{x}^{(\ell-1)}\mathbf{x}^{(\ell-1)^T} \otimes \nabla^2_{\mathbf{z}^{(\ell)}}\mathcal{L}\right], \tag{4}$$

where $\otimes$ denotes the Kronecker product. To simplify this objective, we further assume this Hessian to be a constant diagonal matrix, i.e. $\nabla^2_{\mathbf{z}^{(\ell)}}\mathcal{L} = c \times \mathbf{I}$, effectively ignoring the knowledge of downstream layers and the task loss. Hence, we arrive at the local, layer-wise optimization objective that minimizes the error in the pre-activations caused by quantization:

$$\arg\min_{\Delta\mathbf{w}^{(\ell)}} \mathbb{E}\left[\Delta\mathbf{w}^{(\ell)^T}\mathbf{x}^{(\ell-1)}\mathbf{x}^{(\ell-1)^T}\Delta\mathbf{w}^{(\ell)}\right] = \arg\min_{\Delta\mathbf{w}^{(\ell)}} \mathbb{E}\left[\left(\Delta\mathbf{w}^{(\ell)^T}\mathbf{x}^{(\ell-1)}\right)^2\right]. \tag{5}$$

Li et al. [2021] extended the previous objective to encompass blocks containing any number of layers, demonstrating that the global objective can be effectively approximated by locally minimizing the error in block outputs.

Weight Quantization Following Cheng et al. [2023] we quantize the high-precision weights $\mathbf{W}$ to b-bit precision by

$$\tilde{\mathbf{W}} = s \cdot \mathrm{clip}\left(\left\lceil \frac{\mathbf{W}}{s} + \mathbf{V} \right\rfloor, 0, 2^b - 1\right), \tag{6}$$

where $\mathbf{V}$ is a learnable parameter to adjust rounding, $\lceil \cdot \rfloor$ is the round-to-nearest (RTN) operation, and $\mathrm{clip}(x, n, m)$ restricts the value of x to lie within the range $[n, m]$. The scaling factor is defined as

$$s = \frac{\max(\mathbf{W}) \cdot \alpha - \min(\mathbf{W}) \cdot \beta}{2^b - 1}, \tag{7}$$

where $\alpha, \beta \in [0, 1]$ are learnable parameters.

LA-PTQ In LA-PTQ, the learnable parameters α, β and $\mathbf{V}$ of the k-th transformer block are optimized, using the outputs of the $k+n$-th block as reconstruction target. To facilitate the discussion, we refer to n as the number of look-ahead blocks. In practice, if the network has L blocks, the reconstruction target is set to block $\min(k+n, L)$, though we omit this detail here for simplicity. In this optimization setting, only the parameters of the block k are tuned while the parameters of the other blocks are kept frozen. Let $(\cdot)^{(k)}$ denote the parameters of the k-th block. Then, the optimization target is expressed as follows:

$$\underset{\alpha^{(k)},\beta^{(k)},\mathbf{V}^{(k)}}{\arg\min} \mathbb{E}\Big[||\mathcal{T}(\mathbf{X}, \mathbf{W}^{(k)}, ..., \mathbf{W}^{(k+n)}) - \mathcal{T}(\mathbf{X}, \tilde{\mathbf{W}}^{(k)}, \mathbf{W}^{k+1}, ..., \mathbf{W}^{(k+n)})||_F\Big], \tag{8}$$

where $\mathcal{T}(\mathbf{X}, \mathbf{W}^{(i)}, ..., \mathbf{W}^{(i+n)})$ denotes the transformation applied to the input $\mathbf{X}$ over n transformer blocks with their respective weights $\mathbf{W}^{(i)}, ..., \mathbf{W}^{(i+n)}$. The expectation is over the input $\mathbf{X}$ and $||\cdot||_F$ denotes the Frobenius norm. Starting with the first block ($k = 1$), we sequentially optimize one block at a time, progressively quantizing the neural network's weights. For each block k, the output from the already quantized part of the network serves as the input $\mathbf{X}$. Single block PTQ is the special case of $n = 0$.

MB-PTQ In MB-PTQ, the learnable parameters of n blocks are jointly optimized. More formally, for blocks k to $k+n-1$ to be optimized this is expressed as follows:

$$\underset{\alpha,\beta,\mathbf{V}}{\arg\min}\, \mathbb{E}\Big[||\mathcal{T}(\mathbf{X}, \mathbf{W}^{(k)}, ..., \mathbf{W}^{(k+n-1)}) - \mathcal{T}(\mathbf{X}, \tilde{\mathbf{W}}^{(k)}, ..., \tilde{\mathbf{W}}^{(k+n-1)})||_F\Big], \tag{9}$$

where α, β, and $\mathbf{V}$ denote the parameters of all n blocks. In contrast to LA-PTQ, after quantizing the parameters of the current n blocks, we move on to the next set of n blocks, without overlap between the sets of blocks. This approach differs from that of Ding et al. [2023], who permit consecutive sets of n blocks to overlap, thereby optimizing the overlapping blocks multiple times.

3 Experiments

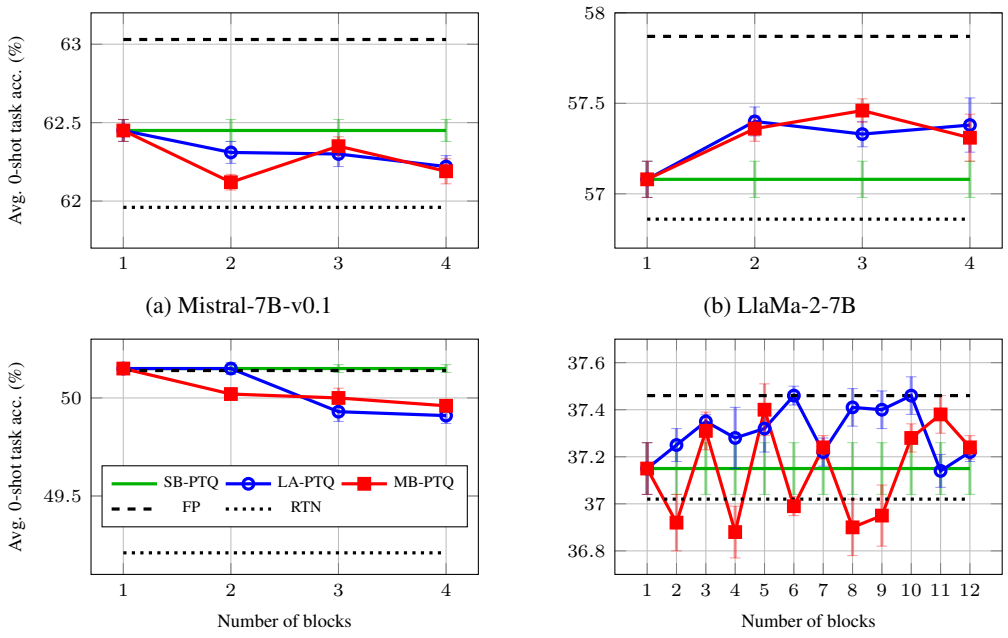
In this section, we first describe the details of our experimental setup, followed by a comparison of our proposed approaches, LA-PTQ and MB-PTQ, against the baseline method, SB-PTQ.

3.1 Setup

We use the abbreviations LA-n and MB-n to refer to LA-PTQ and MB-PTQ with n blocks, respectively. LA-n refers to the setting where one block is fine-tuned with $n-1$ look-ahead blocks and MB-n refers to the setting where n blocks are jointly fine-tuned. Note that in both cases there are a total of n blocks involved in each optimization round (compare Figure 1b to 1c). As reference, we also provide the accuracy for the full precision (FP) and round-to-nearest (RTN) cases. In the RTN scenario, weights are quantized, but not fine-tuned.

We evaluate our methodology on LLaMa-2-7B [Touvron et al., 2023], Mistral-7B-v0.1 [Jiang et al., 2023], OPT-6.7B, and OPT-125M [Zhang et al., 2022]. Our primary metric is the average accuracy across 11 zero-shot tasks, including HellaSwag [Zellers et al., 2019], WinoGrande [Sakaguchi et al., 2021], PIQA [Bisk et al., 2020], LAMBADA [Paperno et al., 2016], TruthfulQA [Lin et al., 2022], OpenBookQA [Mihaylov et al., 2018], BoolQ [Clark et al., 2019], RTE [Dagan et al., 2010], ARC-Easy, ARC-Challenge [Clark et al., 2018], and MMLU [Hendrycks et al., 2021], which we compute by using the `lm-evaluation-harness` [Gao et al., 2024]. We report the average accuracy across these tasks in the main body of the paper and provide the individual accuracy results for each task in Appendix B.

In general, our experimental setup follows that of Cheng et al. [2023] if not specified otherwise. Quantization is limited to the weights of the linear layers in transformer blocks excluding the embedding and the final linear layer. Weights are quantized down to 4 bits, where each group of 128 weights share a learnable scaling factor (see Equation 6 and 7). Calibration data is randomly sampled using the same seed from the publicly available pile-10k dataset, which consists of the first



(a) Mistral-7B-v0.1

(b) LlaMa-2-7B

(c) OPT-6.7B

(d) OPT-125M

Figure 2: Comparison of task accuracy for look-ahead (LA-) and multi-block (MB-) PTQ against single-block (SB-) PTQ, across varying numbers of blocks n and different network models. For all models, we present the average and standard error over 4 trials.

10k samples from the Pile dataset [Gao et al., 2020]. For fine-tuning, 512 calibration samples with a sequence length of 2048 tokens are used. SignSGD is used for optimization with a linear learning rate decay and a batch size of 8. Unlike Cheng et al. [2023], we decrease the learning rate from 5×10^{-3} to 1×10^{-3} to ensure the stability of convergence during fine-tuning. To account for this lower learning rate and potentially more challenging optimization objectives, we increase the number of fine-tuning steps from 200 to 1000.

3.2 Results

Evaluation on Zero-Shot Tasks We evaluate MB-PTQ and LA-PTQ with an increasing number of blocks to capture the impact of progressively introducing more cross-block dependencies and incorporating knowledge from blocks further ahead. Specifically, we fine-tune Mistral-7B-v0.1, LlaMa-2-7B, and OPT-6.7B using LA-PTQ with up to 3 look-ahead blocks and MB-PTQ with substructures of up to 4 blocks. For OPT-125M, as the end-to-end optimization of this model fits into the memory of a single GPU, we iterate over all possible configurations.

We observe that the effect of LA-PTQ and MB-PTQ on the task accuracy depends on the model (see Figure 2). While for Mistral-7B-v0.1 and OPT-6.7B the accuracy of both LA-PTQ and MB-PTQ does not improve compared to SB-PTQ, LlaMa-2-7B shows an improvement with an increasing number of blocks that saturates at 2 blocks. The absence of improvement for OPT-6.7B can be likely explained by SB-PTQ already being sufficient to recover the full-precision performance. For OPT-125M, we observe that both LA-PTQ and MB-PTQ achieve higher accuracy compared to the baseline SB-PTQ for certain block configurations, with LA-PTQ showing more consistent improvement than MB-PTQ. However, the overall differences in accuracy for this model are small (compare FP to RTN in Figure 2d), and these trends are not statistically significant. In contrast to the other models, for OPT-125M, we do not significantly benefit from fine-tuning single blocks in isolation (compare SB-PTQ to RTN in Figure 2b and 2d).

Ablations on Hyperparameters We validate the choice of our learning rate on the example of MB-3 and LA-4 (for details, see Table A.1). Generally, we observe a low sensitivity of the accuracy on the learning rate. However, fine-tuning Mistral-7B-v0.1 with MB-3 using a learning rate of

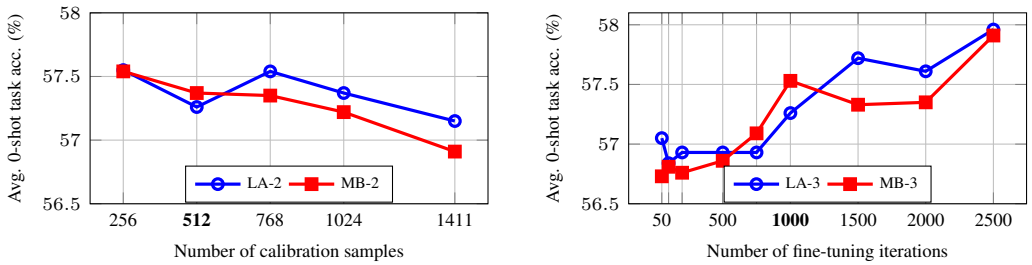


Figure 3: The dependence of task accuracy on the size of the calibration dataset (left) and the number of fine-tuning iterations (left) is shown on the example of LlaMa-2-7B. Default values are highlighted in bold.

5×10^{-3} diverges and results in a significant performance drop. Hence, our default learning rate (1×10^{-3}) serves as an effective middle ground, ensuring smooth convergence across various models and configurations.

For LA-PTQ, the number of free parameters is the same as in SB-PTQ. However, for MB-PTQ, the number of free parameters increases with the number of blocks, n. To assess the potential for overfitting in this case, we examine the relationship between accuracy and the number of fine-tuning iterations, as well as the size of the calibration dataset using LlaMa-2-7B (see Figure 3). Since performance does not improve with an increasing number of calibration samples or a decreasing number of fine-tuning iterations, we can rule out overfitting as the reason why LA-PTQ and MB-PTQ fail to outperform SB-PTQ in certain models. In general, these experiments further validate our selection of default hyperparameters, as they demonstrate comparable or superior performance compared to alternative configurations. However, the performance continuous to improve, albeit at slow pace, as the number of fine-tuning iterations increases. This underscores the delicate trade-off between enhanced performance and the computational resources invested.

4 Discussion

We investigated how incorporating knowledge of subsequent transformer blocks and interactions across blocks effect the fine-tuning of quantized weights in LLMs. We found that the effectiveness of these approaches is model-specific and cannot be generalized across all models. While we do not observe improvements for the Mistral-7B-v0.1, OPT-125M and OPT-6.7B models, the LlaMa-2-7B model shows enhanced task accuracy. However, it is important to note that including more blocks in the optimization process increases computational costs. Further research is needed to explore how the effectiveness of our methods depends on different network models and to extend this investigation to larger LLMs.

In both LA-PTQ and MB-PTQ, the reconstruction loss is applied to the output of downstream blocks. This increases the complexity of the optimization landscape due to the additional blocks and their inherent non-linearities. While MB-PTQ may alleviate this increased complexity through a greater number of free parameters and cross-block optimization compared to LA-PTQ, it does not show superior performance (see Figure 2). This holds true even after ensuring that overfitting, either from a small calibration dataset or excessive training iterations, does not occur (see Figure 3). While we observe a significant performance improvement with LA-2 compared to LA-1 for LlaMa-2-7B (see Figure 2b), further increasing n to extend the look-ahead does not yield additional benefits. Therefore, setting $n = 2$ appears to offer a favorable balance between enhanced accuracy and computational efficiency.

Extending the fine-tuning to the full network, in combination with the original training pipeline and dataset, should ideally yield the best task accuracy. However, in our study, increasing the number of calibration samples does not enhance performance and may even be detrimental (see Figure 3 left). This could indicate a co-variate shift, i.e. a mismatch between the distributions of the calibration and test datasets [Moreno-Torres et al., 2012]. On the other hand, increasing the number of fine-tuning iterations improves the results (see Figure 3 right). Nevertheless, this significant increase in computational demands, especially in comparison to the 200 iterations used by Cheng et al. [2023],

may not justify the performance improvements over SB-PTQ, especially for LLMs. Exploring strategies to reduce or accelerate fine-tuning iterations, such as using LORA adapters [Ding et al., 2023, Bondarenko et al., 2024], is a promising direction for future research.

Ding et al. [2023] demonstrate the advantages of MB-PTQ over SB-PTQ (see their Table 3), which they attribute to the overlap between blocks during the joint optimization of multiple blocks. This raises the question of whether the observed benefits arise from the increased interaction between overlapping and additional blocks, or if they result from the effective increase in fine-tuning iterations, as each overlapping block is optimized multiple times. However, a direct comparison to their results is challenging, since they combine MB-PTQ with several other advanced compression techniques and do not isolate the specific effects of MB-PTQ.

Acknowledgments and Disclosure of Funding

KS, LW, and TP conceptualized and designed the study. KS carried out the experiments and performed data analysis. KS and TP drafted the manuscript, while VG and DC provided valuable feedback on both the study design and the manuscript preparation. We thank Thomas Elsken, Maximilian Frühauf, and Jan Hansen-Palmus for proof reading and Lukas Rinder for his support.

References

Y. Bisk, R. Zellers, J. Gao, Y. Choi, et al. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

Y. Bondarenko, R. Del Chiaro, and M. Nagel. Low-rank quantization-aware training for LLMs. *arXiv preprint arXiv:2406.06385*, 2024.

S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

W. Cheng, W. Zhang, H. Shen, Y. Cai, X. He, K. Lv, and Y. Liu. Optimize weight rounding via signed gradient descent for the quantization of LLMs. *arXiv preprint arXiv:2309.05516*, 2023.

C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL `https://aclanthology.org/N19-1300`.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

I. Dagan, B. Dolan, B. Magnini, and D. Roth. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(1):105–105, 2010.

X. Ding, X. Liu, Y. Zhang, Z. Tu, W. Li, J. Hu, H. Chen, Y. Tang, Z. Xiong, B. Yin, et al. CBQ: Cross-block quantization for large language models. *arXiv preprint arXiv:2312.07950*, 2023.

E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou. A framework for few-shot language model evaluation, 07 2024. URL `https://zenodo.org/records/12608602`.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=d7KBjmI3GmQ`.

I. Hubara, Y. Nahshan, Y. Hanani, R. Banner, and D. Soudry. Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, pages 4466–4475. PMLR, 2021.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

S. Kim, C. Hooper, T. Wattanawong, M. Kang, R. Yan, H. Genc, G. Dinh, Q. Huang, K. Keutzer, M. W. Mahoney, et al. Full stack optimization of transformer inference: a survey. *arXiv preprint arXiv:2302.14017*, 2023.

Y. Li, R. Gong, X. Tan, Y. Yang, P. Hu, Q. Zhang, F. Yu, W. Wang, and S. Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=POWv6hDd9XH.

J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han. AWQ: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

S. Lin, J. Hilton, and O. Evans. TruthfulQA: Measuring how models mimic human falsehoods. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.acl-long.229.

T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. URL https://aclanthology.org/D18-1260.

J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera. A unifying view on dataset shift in classification. *Pattern recognition*, 45(1):521–530, 2012.

M. Nagel, R. A. Amjad, M. Van Baalen, C. Louizos, and T. Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pages 7197–7206. PMLR, 2020.

D. Paperno, G. Kruszewski, A. Lazaridou, N. Q. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In K. Erk and N. A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. URL https://aclanthology.org/P16-1144.

K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

W. Shao, M. Chen, Z. Zhang, P. Xu, L. Zhao, Z. Li, K. Zhang, P. Gao, Y. Qiao, and P. Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8Wuvhh0LYW.

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? In A. Korhonen, D. Traum, and L. Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://aclanthology.org/P19-1472.

S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

A The effect of learning rate on task accuracy

Model	Config	Learning Rate	avg. 0-shot task acc.
LlaMa-2-7B	MB-3	1e-4	56.74
	MB-3	1e-3 (default)	57.53
	MB-3	5e-3	57.77
	LA-4	1e-4	56.45
	LA-4	1e-3 (default)	57.0
	LA-4	5e-3	57.77
Mistral-7B-v0.1	MB-3	1e-4	62.19
	MB-3	1e-3 (default)	62.42
	MB-3	5e-3	37.54
	LA-4	1e-4	62.19
	LA-4	1e-3 (default)	62.4
	LA-4	5e-3	62.51
OPT-6.7B	MB-3	1e-4	49.85
	MB-3	1e-3 (default)	49.88
	MB-3	5e-3	50.36
	LA-4	1e-4	49.55
	LA-4	1e-3 (default)	49.94
	LA-4	5e-3	50.37

Table A.1: The effect of different settings of learning rates on the average 0-shot task accuracy of LlaMa-2-7B and OPT-6.7B fine-tuned with MB-3 and LA-4. Our default learning rate ($1e-3$) stands out as a good compromise to ensure favorable convergence of different models for both LA-PTQ and MB-PTQ configurations.

B Raw accuracy numbers across zero-shot tasks

config	mmlu	lambada_openai	hellaswag	winogrande	piqa	truthfulqa_mc1	openbookqa	boolq	rte	arc_easy	arc_challenge	avg
FP	41.31	73.92	57.13	69.22	78.07	25.21	31.4	77.74	62.82	76.26	43.52	57.87
RTN	40.21	72.58	56.88	68.82	77.53	24.85	31.4	77.52	56.32	76.26	43.09	56.86
LA-1	39.23	71.26	56.13	68.35	77.09	25.46	31.6	75.29	**66.06**	74.45	41.47	56.95
LA-2	40.74	72.66	**56.57**	68.9	**77.48**	**25.83**	**32.6**	76.24	60.65	75.34	**42.83**	57.26
LA-3	**41.33**	**73.37**	56.39	**69.14**	77.31	25.58	31.0	77.19	62.45	**75.51**	42.75	**57.46**
LA-4	41.08	72.99	56.5	68.98	77.8	24.72	30.8	**77.43**	58.84	75.42	42.41	57.0
MB-1	39.23	71.26	56.13	68.35	77.09	25.46	31.6	75.29	**66.06**	74.45	41.47	56.95
MB-2	40.81	72.42	56.4	68.75	78.07	25.46	31.6	75.78	63.18	**75.88**	42.66	57.37
MB-3	41.06	**73.24**	56.47	**68.98**	77.91	**25.83**	**32.6**	76.18	62.82	75.42	42.32	**57.53**
MB-4	**41.62**	73.04	**56.73**	68.43	**78.13**	24.85	**33.2**	**78.35**	59.21	75.88	**42.92**	57.49

Table A.2: LlaMa-2-7B accuracy across 0-shot tasks for both LA-PTQ and MB-PTQ.

config	mmlu	lambada_openai	hellaswag	winogrande	piqa	truthfulqa_mc1	openbookqa	boolq	rte	arc_easy	arc_challenge	avg
FP	58.78	75.61	61.29	73.95	80.69	28.03	32.8	83.76	67.15	80.89	50.34	63.03
RTN	56.62	74.46	60.97	73.56	80.14	27.05	32.2	83.21	63.9	79.84	49.66	61.96
LA-1	**58.16**	75.12	**61.05**	73.32	**80.74**	**28.15**	**33.4**	83.12	62.45	80.22	**50.34**	62.37
LA-2	58.04	75.08	60.84	**75.14**	80.47	27.66	32.2	82.78	62.09	**80.3**	49.74	62.21
LA-3	57.9	**75.24**	60.89	74.11	80.36	27.42	**33.4**	**83.27**	61.37	79.92	49.49	62.12
LA-4	57.68	75.2	60.84	73.88	80.2	27.17	32.6	83.12	**66.43**	79.34	50.0	**62.4**
MB-1	**58.16**	75.12	**61.05**	73.32	**80.74**	**28.15**	**33.4**	83.12	62.45	80.22	**50.34**	62.37
MB-2	58.28	75.33	60.77	73.4	80.63	27.54	32.6	82.72	63.54	79.84	49.74	62.22
MB-3	**58.65**	74.79	60.76	**73.72**	80.03	**28.15**	**33.2**	83.06	**64.26**	**80.3**	49.74	**62.42**
MB-4	57.76	**75.41**	**61.01**	73.09	**79.98**	27.17	30.6	**83.61**	63.54	79.88	50.0	62.0

Table A.3: Mistral-7B-v0.1 accuracy across 0-shot tasks for both LA-PTQ and MB-PTQ.

config	mmlu	lambada_openai	hellaswag	winogrande	piqa	truthfulqa_mc1	openbookqa	boolq	rte	arc_easy	arc_challenge	avg
FP	24.89	67.69	50.52	65.27	76.28	21.79	27.6	66.06	55.23	65.61	30.63	50.14
RTN	25.39	66.1	49.34	64.25	76.17	21.3	27.2	61.1	53.43	65.99	31.06	49.21
LA-1	**25.16**	67.22	50.04	65.19	76.44	21.66	27.0	66.02	56.32	65.49	30.72	50.12
LA-2	24.98	**67.48**	**50.1**	65.98	76.17	**22.03**	27.6	**66.33**	55.23	**65.95**	**30.8**	**50.24**
LA-3	24.9	67.03	49.94	65.59	76.33	21.42	27.6	65.5	**55.6**	65.4	30.12	49.95
LA-4	24.85	66.95	50.06	65.35	**76.5**	21.05	**27.2**	66.15	**55.6**	65.66	29.95	49.94
MB-1	**25.16**	67.22	50.04	65.19	**76.44**	21.66	27.0	66.02	56.32	65.49	30.72	50.12
MB-2	24.98	**67.53**	**50.14**	65.43	76.44	21.05	26.4	65.96	**56.68**	65.7	30.38	50.06
MB-3	**25.22**	67.42	49.94	65.19	76.22	21.54	**27.0**	65.6	54.51	**65.99**	30.03	49.88
MB-4	24.98	67.09	**50.17**	65.51	76.22	20.81	25.8	65.72	55.6	65.87	**31.4**	49.92

Table A.4: OPT-6.7B accuracy across 0-shot tasks for both LA-PTQ and MB-PTQ.

config	mmlu	lambada_openai	hellaswag	winogrande	piqa	truthfulqa_mc1	openbookqa	boolq	rte	arc_easy	arc_challenge	avg
FP	22.88	37.86	29.18	50.36	62.95	23.99	16.6	55.44	50.18	43.52	19.11	37.46
RTN	22.87	36.76	28.81	51.7	63.33	24.85	18.4	49.42	49.82	42.38	18.86	37.02
LA-1	22.95	37.03	28.81	49.49	62.84	23.75	16.8	55.63	46.21	43.48	19.45	36.95
LA-2	23.0	37.2	28.85	49.57	63.0	**24.36**	17.0	**57.03**	45.49	**43.56**	19.11	37.11
LA-3	**23.02**	36.91	29.08	50.28	62.46	23.99	16.8	55.78	49.82	43.41	**20.05**	37.39
LA-4	22.85	36.97	28.87	50.36	62.79	23.99	16.4	56.3	46.57	43.39	19.45	37.09
LA-5	22.75	37.76	28.87	50.91	62.57	23.87	16.8	55.87	49.82	43.28	18.77	37.33
LA-6	22.87	37.51	28.86	50.75	62.51	23.87	16.6	55.69	50.54	43.43	**20.05**	37.52
LA-7	22.94	37.71	28.89	51.3	62.73	23.99	16.6	55.78	47.65	43.18	19.62	37.31
LA-8	22.86	37.45	**29.03**	50.91	63.28	23.99	17.2	54.25	**51.62**	43.31	19.62	**37.59**
LA-9	22.87	**38.07**	28.93	50.59	63.0	24.11	16.0	55.02	49.82	42.72	19.28	37.31
LA-10	22.85	38.04	28.99	**51.14**	**63.06**	24.24	15.6	55.63	48.01	42.93	19.71	37.29
LA-11	22.87	**38.15**	28.88	50.51	62.51	24.11	17.4	54.46	46.21	43.48	19.43	37.11
LA-12	22.87	37.43	**29.06**	**51.54**	62.79	23.87	16.4	54.98	46.57	42.93	**20.05**	37.14

Table A.5: OPT-125M accuracy across 0-shot tasks for LA-PTQ.

config	mmlu	lambada_openai	hellaswag	winogrande	piqa	truthfulqa_mc1	openbookqa	boolq	rte	arc_easy	arc_challenge	avg
FP	22.88	37.86	29.18	50.36	62.95	23.99	16.6	55.44	50.18	43.52	19.11	37.46
RTN	22.87	36.76	28.81	51.7	63.33	24.85	18.4	49.42	49.82	42.38	18.86	37.02
MB-1	22.95	37.03	28.81	49.49	62.84	**24.36**	16.8	55.63	46.21	43.48	19.45	36.95
MB-2	22.96	37.12	28.75	48.93	63.11	**24.36**	16.4	54.83	46.21	**43.69**	19.11	37.61
MB-3	22.9	**37.8**	28.86	50.28	62.84	**24.24**	16.4	55.86	48.43	43.6	**20.31**	37.31
MB-4	22.82	37.34	28.84	49.8	62.73	23.75	17.0	55.38	46.57	42.63	18.86	36.88
MB-5	22.93	37.36	28.94	50.36	**63.28**	23.62	16.2	56.48	49.46	42.72	**20.05**	37.4
MB-6	22.93	37.18	**29.03**	50.04	62.73	24.11	16.2	55.29	47.29	42.93	19.11	36.99
MB-7	**23.0**	36.87	28.98	50.67	62.84	24.24	**17.0**	55.81	48.38	42.59	19.28	37.24
MB-8	22.97	36.77	28.9	50.91	62.57	24.24	**17.0**	54.34	46.57	42.21	19.45	36.9
MB-9	22.99	36.74	28.87	50.2	62.35	**24.36**	15.8	53.91	**48.01**	42.76	**20.48**	36.95
MB-10	22.95	36.64	28.81	49.57	62.62	24.11	16.8	**57.0**	49.46	42.63	19.54	37.28
MB-11	**23.0**	37.36	28.98	51.3	62.4	**24.36**	15.8	55.63	49.46	43.01	19.88	37.38
MB-12	22.95	36.99	28.86	**51.38**	62.79	23.87	16.0	54.71	**50.18**	42.59	19.37	37.24

Table A.6: OPT-125M accuracy across 0-shot tasks for MB-PTQ.