CafeQ: Calibration-free Quantization via Learned Transformations and Adaptive Rounding

Anonymous Authors[1]

Abstract

Post-training weight quantization is an effective method for reducing the serving cost of large language models. However, standard round-to-nearest quantization often introduces large errors due to outliers in the weights. Proposed mitigation mechanisms include applying adaptive rounding, random rotation transformations or committing to a post-training target using calibration data. Unfortunately, this reliance on calibration data can be severely limiting due to data inavailability, or additional computational overhead. In this paper, we propose algorithms to optimize transformations and adaptive rounding without access to *any* calibration data. The optimization is achieved by designing a suitable proxy function for the quantization loss without calibration data. To maintain inference efficiency, we perform structured matrix transformations for single matrices. For paired weights that interact directly in the computation graph, we use dual matrix transformations and adaptive rounding methods. We conduct experiments on Gemma and Qwen models. Compared to baseline methods without calibration data, we observe consistent improvement across various benchmarks and quantization levels. Compared to SOTA calibration-based methods, for 4-bit quantization on Qwen2.5-7B model, our method achieves comparable performance with < 0.2 reduction in both WikiText-2 and C4 perplexities and even better average score on zero-shot benchmarks, demonstraing the effectiveness of calibration-free quantization.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

Large language models (Brown et al., 2020; Chowdhery et al., 2022; Thoppilan et al., 2022; Touvron et al., 2023) contain billions or trillions of parameters, which require up to terabytes of memory. This massive model size far exceeds the on-chip high bandwidth memory (HBM) available on a single accelerator (e.g., GPU), and is complicated by the fact that each forward pass requires reading the entire set of model weights from memory (Davies et al., 2025). The time spent on loading these parameters into memory often exceeds the time spent on tensor computations, causing memory to be the bottleneck of LLM inference.

Quantizing weight matrices is a common strategy to reduce the memory loading time, which improves inference efficiency. For example, instead of using 2 bytes for each parameter at standard 16-bit precision (e.g., bfloat16), quantization can represent the parameter using just 1 byte (8 bits) or even as few as 4 bits, effectively halving or quartering the memory requirements, which leads to significant reductions in memory loading time and overall inference latency as well. The standard method for quantizing weight matrices in LLMs is *uniform* quantization. In uniform quantization with N bits, the weights are divided into blocks (channels) $\vec{w} = (w_1, \ldots, w_d)$, and then the range of these weights is computed to determine the quantization scale $s = (w_{\max} - w_{\min})/(2^N - 1)$, where $w_{\max} = \max_i w_i$ and $w_{\min} = \min_i w_i$. Then, each quantized value is computed as

$$\hat{w_i} = s \cdot \left\lceil \frac{w_i - w_{\min}}{s} \right\rfloor + w_{\min}, \tag{1}$$

where $\lceil x \rfloor$ denotes the nearest integer to x. The error of uniform quantization over a set of weights indexed by i is upper bounded by

$$|w_i - \hat{w}_i| \leq \frac{w_{\max} - w_{\min}}{2(2^N - 1)} \leq \frac{\|w\|_\infty}{2^N - 1}. \tag{2}$$

Despite being widely adopted, uniform quantization suffers from the following drawbacks: (1) The error

of uniform quantization is determined by the quantization scale, which is heavily affected by the outlier weights within a block. (2) Each weight matrix in the transformer network is independently quantized. This does not take advantage of the computational structure in the transformer block, particularly in the attention modules. which in contrast, aims to quantize an *already-trained* model with little to no retraining. PTQ techniques can be differentiated along two broad axes. The first axis is the complexity of the quantization scheme, which ranges from the simplest scalar uniform quantization (Frantar et al., 2023) to more complex methods like scalar look-up table (LUT) quantization (Kim et al., 2024), vector uniform quantization (Tseng et al., 2024a), and vector LUT quantization (Tseng et al., 2024c). The second axis pertains to whether calibration data is needed at all.

Goals of this paper. We focus on calibration data-free scalar uniform quantization. We choose uniform quantization for its high efficiency and broad support on modern hardware accelerators, though our techniques can extend to other quantization formats such as floating-point quantization (Abecassis et al., 2025; Mishra et al., 2025). Calibration data-free (or *calibration-free*, for short) quantization is useful in several scenarios. First, in some applications, representative data for calibration is unavailable and even when data exists, its use may be prohibited due to privacy and security concerns, e.g., in applications involving medical or biometric data. Secondly, calibration-based methods (Ashkboos et al., 2024b; Chee et al., 2024; Liu et al., 2024; Sun et al., 2025) often require forward-pass or backward passes through the model to get intermediate activations, requiring a significant amount of computational resources and time, which can be prohibitive in real-world scenarios. Thirdly, relying on a static calibration set introduces a significant vulnerability to domain shift. A model may need to be deployed for multiple or unknown downstream tasks, and specifically quantizing for each task may be prohibitive. A model quantized based on a specific dataset can see a degradation in performance when the real-world data it encounters evolves or differs from the calibration set (Tang et al., 2023; Williams & Aletras, 2024). These motivations underscore the necessity of robust quantization techniques that are *data-independent*, thereby ensuring broader applicability, enhanced privacy, and robustness to domain shifts, in downstream applications.

The central questions in scalar PTQ for LLMs revolve around two main challenges:

(a) Handling outliers: How do we prevent a few large magnitude values from dominating the quantization range and destroying precision for all other inliers?

(b) Rounding: How to map high-precision values to their low-precision counterparts?

2. Our contributions

Modern LLMs consist of stacked layers of transformer blocks, and an embedding layer. Each transformer block includes an attention block and a feedforward-network (FF) block, parameterized by the corresponding weight matrices, and scaling vectors in layer norms. The embedding layer is parameterized by an embedding matrix. Most of the computation in these layers is the matrix-vector multiplication $y = Wx$. In this paper, we focus on weight-only quantization that quantizes weight matrices to preserve the result of the output multiplication operations. With this background in mind, we propose a calibration-free PTQ framework that addresses the above mentioned challenges through four primary contributions:

Handling outliers via proxy-loss minimization A central challenge in calibration-free quantization is the inability to measure the impact of quantization error on downstream task performance. We first establish that the Frobenius norm of the quantization error, $\|W - \widehat{W}\|_F$ (where W is the original matrix and $\hat{W}$ is the quantized matrix), serves as a good proxy for final task accuracy. Since the quantization operator, $\widehat{\cdot}$, makes $\|W - \widehat{W}\|_F$ non-differentiable, we bound $\|W - \widehat{W}\|_F$ with surrogate losses to smooth the optimization landscape. Combining the above observations, we reframe the problem of mitigating outliers as an optimization problem with the goal of minimizing designed surrogate losses. We focus on methods that apply affine transforms to the weight matrices to mitigate outlier effects. Therefore the question becomes: can we learn an affine transformation that minimizes this Frobenius norm quantization error?

Structured transformations for single matrices For layers that operate independently, such as feed-forward layers, following earlier works of Adepu et al. (2024); Ashkboos et al. (2024b), we propose to apply an efficient transformation M to the matrix W before quantization and then apply the inverse of the transformation during inference. The effective layer parameter is thus $M^{-1}\widehat{MW}$, where as before, $\widehat{\cdot}$ denotes the quantization operation. However, unlike prior works that use fixed or randomized transformations, we learn a structured transformation. This structured transformation is obtained by minimizing a proxy loss for the quantization error. We find that block-diagonal matrices perform best in our experiments, in contrast to the

Walsh-Hadamard transform used in earlier works.

Structured transformations can be applied to all weight matrices, including the attention block, the feedforward block, and the embedding layer. We will discuss this technique in detail in Section 3.1, and how one can choose M such that the M^{-1} can be applied efficiently.

Transformations for coupled matrices. For matrices that are coupled in the computation graph (i.e., they are applied sequentially *without* an intermediate non-linearity), we propose learning an arbitrary matrix M. This incurs no additional inference overhead, as the transformation can be absorbed into the coupled weight matrices. If W_1 and W_2 are coupled matrices, we propose to find M such that the product of the quantized transformed matrices $\widehat{W_1M^{-1}}\widehat{MW_2}$ closely approximates the original product W_1W_2.

To illustrate the usage of this technique, consider attention score computation between locations i, j below:

$$Attn_{i,j} = \text{Softmax}\left(\frac{X_j^T \cdot W_q \cdot W_k \cdot x_i}{\sqrt{D}}\right) \odot (W_o \cdot W_v \cdot X_i).$$

Note that the attention computation remains unchanged if the product of certain pairs of matrices remain the same, such as the (W_v, W_o) pair and the (W_q, W_k) pair. Note that for any pair of matrices W_1, W_2, and any invertible matrix M, we have

$$W_1W_2 = (W_1M^{-1})(MW_2).$$

Hence no additional online operation is needed since the output of the operation is already preserved. While this technique is cannot be applied to (W_q, W_k) when rotary positional embeddings is used due to non-commutativity, recent architectures are moving towards using "NoPE", or eliminating positional embeddings altogether (Kazemnejad et al., 2023). For example, Llama 4's attention layers alternate between RoPE and NoPE (Meta, 2025). However, since many architectures still use RoPE, we focus on applying our paired quantization technique to the output-value projection pair (W_v, W_o) and leave applications to the query-key pair, (W_q, W_k), as future work.

Better adaptive rounding techniques for coupled matrices. For coupled matrices, we also propose a new alternating adaptive rounding technique that accounts for the matrix product structure. Intuitively, if a value in the first matrix is rounded down, our method attempts to compensate by rounding up corresponding values in the second matrix that interact with it during multiplication. Similar to our previous contribution, this can be applied to W_o and W_v matrices in all transformer architectures, and for the query-key pair, (W_q, W_k), in some architectures.

By using a combination of these methods, we observe consistent improvement over other calibration-free baselines on Gemma 2 models. For Gemma 2 9B quantization, our method improves the average benchmark score from 61.9 to 62.4 for 4-bit quantization and from 52.0 to 60.6 for 3-bit quantization on standard benchmarks, while adding less than 3% of computation overhead. We also compare our method with SOTA calibration-based methods and observe that our method can achieve comparable performance without using any calibration data. E.g., we observe less than 0.2 reduction in both WikiText-2 and C4 perplexities and even better average score on zero-shot benchmarks for 4-bit quantization on Qwen2.5-7B model. This demonstrates the effectiveness of calibration-free quantization.

The rest of the paper is organized as follows. In Section 3.1, we propose structured transformations for single matrices. Due to space constraints we defer the learning of coupled transformations to Section A and propose the adaptive rounding technique for coupled matrices to Section B. Finally, in Section 4, we provide ablation studies and results on the Gemma and Qwen family of models.

3. Removing outliers through learned linear transformations

Our quantization approach is motivated by the fact that random rotations remove outliers (Adepu et al., 2024; Tseng et al., 2024a; Ashkboos et al., 2024b), which can reduce quantization error when applied to the weight matrix before uniform quantization. To further remove outliers beyond random rotations, we propose to learn weight-dependent linear transformations.

More precisely, for a weight matrix of shape $d_1 \times d_2$, we learn a weight-dependent invertible transformation matrix $M : d_1 \times d_1$, and perform uniform quantization on MW to yield $\widehat{MW}$. During inference, we apply the inverse transformation M^{-1} to a single input feature vector Y of shape $1 \times d_1$, to get XM^{-1}, yielding quantized layer output of

$$\hat{Y} = XM^{-1}\widehat{MW}.$$

Compared to the desired output $Y = XW$, the output error introduced by the quantization step is

$$\hat{Y} - Y = XM^{-1}\widehat{MW} - XW = X(M^{-1}\widehat{MW} - W).$$

When quantization is performed, we want to learn M such that the expected error on the outputs

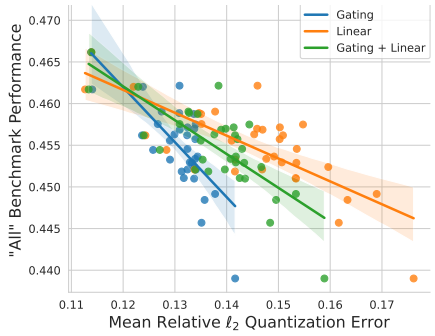


Figure 1. Downstream performance as a function of quantization error of the feedforward weights. All models were Gemma 2 2B pretrained models whose gating and linear weights were quantized. *Gating + Linear* corresponds to computing the mean quantization error over both sets of weights. We obtain Spearman's rank correlation coefficient of -0.640 (p=4.6e-5) and -0.679 (p=10.0e-6) for the gating and linear layers, respectively.

$\mathbb{E}_Y[\|\hat{Y} - Y\|_F^2]$ is minimized. With a calibration set, we could approximate the expected error with an empirical average. However, this method cannot be applied in the calibration-free setting. Our approach for learning M relies on the following inequality, which follows from the Cauchy–Schwarz inequality:

$$\begin{aligned}\mathbb{E}[\|\hat{Y} - Y\|_F^2] &= \mathbb{E}[\|X(M^{-1}\widehat{MW} - W)\|_F^2] \\ &\leq \mathbb{E}[\|X\|_F^2 \|M^{-1}\widehat{MW} - W\|_F^2] \\ &= \|M^{-1}\widehat{MW} - W\|_F^2 \mathbb{E}[\|X\|_F^2].\end{aligned} \tag{3}$$

Assuming that the change of the expectation on the input norm $\mathbb{E}[\|X\|_F^2]$ due to quantization is small, we minimize the expected error on the outputs by using $\|M^{-1}\widehat{MW} - W\|_F$ as a proxy. Note that this is the same as the ℓ_2 loss on the vectorized version of $M^{-1}\widehat{MW} - W$. In this paper, we also refer to this proxy loss as the ℓ_2 loss.

We perform an empirical study on the feed-forward (FF) block and observe strong Spearman's rank correlation between the proposed ℓ_2 proxy loss and the scores on downstream evaluations (see Appendix E for details on this analysis). The results are presented in Fig. 1. The above finding suggests that the Frobenius norm of the reconstruction error on weight matrices is a reasonable proxy loss to use for calibration-free quantization. A different challenge we face with this transformation-based approach is the extra computation required due to applying M^{-1} online. In Section 3.1 and Appendix A, we discuss how this computation can be mitigated for single and coupled matrices.

3.1. Learned structured transformation for a single matrix

To reduce the extra inference-time computation, we consider learning structured matrices that support fast matrix-vector multiplication. More specifically, we use block diagonal matrices. A block diagonal matrix of dimension d and block size k can be written as

$$M = \begin{pmatrix} M_1 & & \\ & \ddots & \\ & & M_{d/k} \end{pmatrix}$$

where all M_i's are $k \times k$ matrices and the neglected entries in M are all zeros. The number of free parameters in a block diagonal matrix is $d \cdot k$. We note that the block diagonal matrices can be multiplied efficiently. During inference, the cost for multiplying such a matrix with a d-dimension vector will be $C_{d,k} = O(d \cdot k)$.

Optimization of M. $\|M^{-1}\widehat{MW} - W\|_F^2$ is not differentiable due to the quantization operator. Hence, for optimizing M, we approximate $\|M^{-1}\widehat{MW} - W\|_F^2$ by the expected reconstruction error under the stochastic quantization algorithm. In stochastic quantization, instead of rounding $\frac{w_i - w_{\min}}{s}$ to the nearest integer as in Eq. (1), we round to the nearest two integers, each with probability proportional to their distance from the current value. Let $\Delta = \widehat{MW} - MW$. With stochastic rounding, entries of Δ are independent of each other with mean zero and variance at most $R^2/(4(2^N - 1)^2)$ where R is the range of their corresponding group of weights.

$$\begin{aligned}&\mathbb{E}\left[\|M^{-1}\widehat{MW} - W\|_F^2\right] \\ &= \mathbb{E}\left[\|M^{-1}(\widehat{MW} - MW)\|_F^2\right] \\ &= \mathbb{E}\left[\sum_{i=1}^{d_1}\sum_{k=1}^{d_2}\left(\sum_{j=1}^{d_1}(M^{-1})_{i,j}\Delta_{j,k}\right)^2\right] \\ &= \sum_{i=1}^{d_1}\sum_{k=1}^{d_2}\sum_{j=1}^{d_1}((M^{-1})_{i,j})^2\mathbb{E}\left[\Delta_{j,k}^2\right] \\ &\leq \frac{d_2}{(2^N-1)^2}\sum_{i=1}^{d_1}\sum_{j=1}^{d_1}((M^{-1})_{i,j})^2\|(MW)_j\|_\infty^2.\end{aligned} \tag{4}$$

Hence, we initialize M as a block diagonal matrix with each block being a random rotation matrix, and perform gradient descent with Eq. (4) as the loss. Even though we do not perform stochastic quantization in our experiments, the minimizer of Eq. (4) yields small quantization error. In Table 1, we present the average

Table 1. The effect of block diagonal size on the down projection layer of Gemma 2 2B (input dimension: 9216, output dimension: 2304). RANDOM denotes applying a block diagonal matrix with block size 1024, where each block is a random Hadamard matrix. *Extra FLOPs (%)* and *Extra wall-clock time (%)* respectively indicate the extra compute and time required compared to the original matmul operation.

Method	UNIFORM	RANDOM	Learned block diagonal			
Diagonal block size	-	1024	32	64	128	256
Avg. relative ℓ_2 loss	0.176	0.155	0.113	0.103	0.094	0.085
Extra FLOPs (%)	0	0.11	0.35	0.69	1.39	2.78
Extra wall-clock time (%)	0	-	2.4	2.9	3.3	4.9

relative ℓ_2 loss we get with different diagonal block sizes on the Gemma 2 2B model and compare it against uniform quantization and transformed quantization using a random structured matrix. As we can see, we are able to get lower average ℓ_2 loss compared to both the UNIFORM and RANDOM* settings, with a small additional penalty in terms of FLOPs, with diagonal block size 32. The error is also significantly reduced as the diagonal block size increases. Furthermore, block diagonal matrices can be applied efficiently on GPUs. In Table 1, we also present the extra FLOPs and wall-clock time needed in a GPU implementation compared to the original computation in the down projection layer. We note that applying a block diagonal matrix with diagonal block size 128 requires 1.4% extra FLOPs and 3.3% extra wall clock time relative to the original computation. This is the setting we use in our experiments.

4. Experiments

We perform experiments with pretrained Gemma 2 2B and 9B models (Team, 2024). We consider both 3 and 4 bit quantization with each quantization block being a channel along the contraction dimension (per-channel quantization), as well as subchannel quantization with block size 256 and 128. We perform quantization only on the weights and leave the embedding table and activations unquantized. For CafeQ quantization, we use paired transformation for the pair of W_v and W_o. For all the remaining matrices, we use block diagonal matrices along the contraction dimension with diagonal size 128 for Gemma 2 2B and 256 for Gemma 2 9B. In total, this adds $< 3\%$ of additional FLOPs for both models.

We evaluate the model on standard Gemma 2 benchmarks (Team, 2024), as described in Table 7 in the appendix. We present the results in Table 2. As shown, CafeQ quantization achieves superior performance compared to the uniform quantization baselines with no rotation or random Hadamard rotations. The improvement is consistent for both the 4-bit case where the quantization loss is smaller and the 3-bit case where the quantization loss is large. We also observe consistent improvement across different quantization block sizes, and find that improvements on individual tasks are also consistent with overall performance (Appendix G).

Next we present ablation studies we performed on the Gemma 2 2B model. All ablations were done with per-channel 4 bit quantization.

Ablation on each separate weight matrices. We study the effect of quantization on each of the components in a transformer block, including FF matrices, QK matrices, and VO matrices. For ablation on each of the component, we keep other components unquantized. The results are presented in Table 3. We see that our method achieves consistent improvement for each individual component in the transformer block. We see that quantization leads to the most extreme score drop on FF blocks, and the least drop on QK blocks. This could be explained by the large parameter count of the FF layers. While QK and VO have similar parameter counts, QK only affects the attention scores, which may have less impact on the overall performance.

The effect of diagonal block size. Next we study how the diagonal block size in the transformation matrix would affect the downstream evaluations. We observe in Table 4 that as the block downstream performance improves overall due to the increasing expressiveness of the optimization space. With a diagonal block size of 32, we are already improving over RANDOM transformation-based method.

Comparison to calibration-based methods In Table 4, we also compare our method with state-of-the-art calibration-based methods, including GPTQ (Frantar et al., 2023), a canonical calibration-based method, which does not use a transformation matrix and only relies on calibration data to perform adaptive rounding, as well as FlatQuant (Sun et al., 2025). FlatQuant learns transformation matrices to reduce quantization error similar to our paper but with calibration data, and it achieves the SOTA performance among

Table 2. Average benchmark scores for Gemma 2 2B and 9B quantization. Number of quantization bits and subchannel block sizes are listed along the columns. All evaluations of CafeQ method were conducted on models where the V and O matrices were reverse-transformed prior to inference (even though this was not strictly necessary). * indicates runs (2B 4-bit quantization) where the VO matrices were not transformed back before quantized models were evaluated. Applying the reverse transformation for those runs resulted in similar, but slightly worse overall performance: 45.3, 45.8, 45.9.

Method	Gemma 2 2B						Gemma 2 9B					
	3 bits			4 bits			3 bits			4 bits		
	N/A	256	128	N/A	256	128	N/A	256	128	N/A	256	128
Unquantized	48.0						63.9					
UNIFORM	23.3	33.1	34.9	41.3	44.7	45.4	28.0	50.4	58.0	58.1	61.1	62.3
RANDOM	26.1	34.0	35.3	43.6	44.5	45.3	40.4	52.0	59.0	60.3	61.9	62.3
CafeQ **(ours)**	**35.1**	**38.7**	**39.1**	**45.6***	**46.0***	**46.1***	**53.7**	**60.6**	**61.7**	**61.9**	**62.4**	**62.4**

Table 3. Overall downstream performance after applying uniform/random rotation/CafeQ 4-bit quantization to each set of model weights for pretrained Gemma 2 2B, leaving other weights unquantized. For CafeQ, we use a diagonal block size of 128.

Method \ Weights	FF	QK	VO
Unquantized		48.0	
Uniform	43.9	47.2	45.9
Random	45.3	47.4	46.4
CafeQ	**46.6**	**47.6**	**46.8**

Table 4. The effect of block diagonal size d_{block} on average scores on downstream evaluations. We perform per-channel 4-bit quantization on Gemma 2 2B model for all methods. UNI and RAN* are the uniform and random transformation baselines.

Method	UNI	RAN*	Learned block diagonal			
d_{block}	-	1024	32	64	128	256
Avg. score	41.3	43.6	44.7	45.4	45.3	46.2

calibration-based learning methods as reported in the paper. We use the code released in the FlatQuant codebase for both methods and perform per-channel 4-bit weight quantization on Qwen2.5-7B model with the recommended configurations. We set the diagonal block size to be 128 to match the additional computation overhead required by FlatQuant. We use the same evaluation setup and evaluation benchmarks as in (Frantar et al., 2023) and measure WikiText-2 perplexity, C4 perplexity, and the score across 6 downstream benchmarks (ARC-C, ARC-E, HellaSwag, LAMBADA, PIQA, Winogrande) for all methods. The results are presented in Table 5.

We find that CafeQ achieves much better performance comapred to GPTQ, demonstrating the importance of using transformation matrices to remove outliers and reduce quantization error. We also find that CafeQ is competitive with FlatQuant, with a 0.1 gap in WikiText-2 perplexity. We note that FlatQuant acheives lower perplexity on C4 even compared to the full precision model, which is not obtained through better quantization scheme. Hence we measure the perplexity drop of our method on C4 against the full precision model, which is only 0.16. We also find that CafeQ achieves better performance on the downstream benchmarks, with an average score of 70.63 compared to 68.62 for FlatQuant. Overall, the results demonstrate that CafeQ is able to achieve competitive performance compared to calibration-based methods without using any calibration data. Moreover, this comes with a significant computational advantage. This showcases the promise of calibration-free learning methods for quantizing LLMs.

Table 5. Comparison of FLATQUANT to calibration-based methods with per-channel 4-bit quantization on Qwen2.5-7B.

	WikiText-2	C4	ARC-C	ARC-E	HellaSwag	LAMBADA	PIQA	Winogrande	Avg
FP16	8.36	14.37	79.82	76.01	79.53	51.88	67.9	69.22	70.74
FlatQuant	8.46	13.94	51.71	77.69	78.42	57.46	76.93	69.53	68.62
GPTQ	9.27	15.78	78.24	77.28	72.77	51.79	66.3	60.49	67.81
CafeQ	8.56	14.53	79.27	79.32	77.15	51.62	70.24	66.16	70.63

References

Abecassis, F., Agrusa, A., Ahn, D., Alben, J., Alborghetti, S., Andersch, M., Arayandi, S., Bjorlin, A., Blakeman, A., Briones, E., et al. Pretraining large language models with nvfp4. *arXiv preprint arXiv:2509.25149*, 2025.

Adepu, H., Zeng, Z., Zhang, L., and Singh, V. FrameQuant: Flexible low-bit quantization for transformers, 2024. URL https://arxiv.org/abs/2403.06082.

Ashkboos, S., Croci, M. L., do Nascimento, M. G., Hoefler, T., and Hensman, J. SliceGPT: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=vXxardq6db.

Ashkboos, S., Mohtashami, A., Croci, M. L., Li, B., Cameron, P., Jaggi, M., Alistarh, D., Hoefler, T., and Hensman, J. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=dfqsW38v1X.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., and Sutton, C. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

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

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Chee, J., Cai, Y., Kuleshov, V., and Sa, C. D. QuIP: 2-bit quantization of large language models with guarantees, 2024. URL https://arxiv.org/abs/2307.13304.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Ponde de Oliveira Pinto, H., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Petroski Such, F., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Hebgen Guss, W., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Burstein, J., Doran, C., and Solorio, T. (eds.), *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)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

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

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Davies, M., Crago, N., Sankaralingam, K., and Kozyrakis, C. Efficient llm inference: Bandwidth, compute, synchronization, and capacity are all you need. *arXiv preprint arXiv:2507.14397*, 2025.

Dettmers, T., Svirschevski, R., Egiazarian, V., Kuznedelev, D., Frantar, E., Ashkboos, S., Borzunov, A., Hoefler, T., and Alistarh, D. Spqr: A sparse-quantized representation for near-lossless LLM weight compression. In *International Conference on Learning Representations, ICLR*. OpenReview.net, 2024.

Egiazarian, V., Panferov, A., Kuznedelev, D., Frantar, E., Babenko, A., and Alistarh, D. Extreme

compression of large language models via additive quantization, 2024. URL https://arxiv.org/abs/2401.06118.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. GPTQ: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Golub, G. H. and Van Loan, C. F. *Matrix computations*. JHU press, 2013.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. In *International Conference on Learning Representations, ICLR*, 2021a.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021b.

Joshi, M., Choi, E., Weld, D., and Zettlemoyer, L. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Barzilay, R. and Kan, M.-Y. (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Kazemnejad, A., Padhi, I., Natesan Ramamurthy, K., Das, P., and Reddy, S. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.

Kim, J., young Kim, H., Cho, E., Lee, C., Kim, J., and Jeon, Y. BoA: Attention-aware post-training quantization without backpropagation, 2025. URL https://arxiv.org/abs/2406.13474.

Kim, S., Hooper, C., Gholami, A., Dong, Z., Li, X., Shen, S., Mahoney, M. W., and Keutzer, K. Squeezellm: Dense-and-sparse quantization. In *International Conference on Machine Learning, ICML*. OpenReview.net, 2024.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Li, X., Hanna, O. A., Fragouli, C., and Diggavi, S. N. Icquant: Index coding enables low-bit LLM quantization. *CoRR*, abs/2505.00850, 2025. URL https://doi.org/10.48550/arXiv.2505.00850.

Liu, Z., Zhao, C., Fedorov, I., Soran, B., Choudhary, D., Krishnamoorthi, R., Chandra, V., Tian, Y., and Blankevoort, T. SpinQuant: LLM quantization with learned rotations, 2024. URL https://arxiv.org/abs/2405.16406.

Ma, Y., Li, H., Zheng, X., Ling, F., Xiao, X., Wang, R., Wen, S., Chao, F., and Ji, R. Affinequant: Affine transformation quantization for large language models, 2024. URL https://arxiv.org/abs/2403.12544.

Malinovskii, V., Mazur, D., Ilin, I., Kuznedelev, D., Burlachenko, K., Yi, K., Alistarh, D., and Richtárik, P. Pv-tuning: Beyond straight-through estimation for extreme LLM compression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Meta. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, April 2025. Accessed: 2025-09-24.

Mishra, A., Stosic, D., Layton, S., and Micikevicius, P. Recipes for pre-training llms with mxfp8. *arXiv preprint arXiv:2506.08027*, 2025.

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

Sap, M., Rashkin, H., Chen, D., Le Bras, R., and Choi, Y. Social IQa: Commonsense reasoning about social interactions. In Inui, K., Jiang, J., Ng, V., and Wan, X. (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Shao, W., Chen, M., Zhang, Z., Xu, P., Zhao, L., Li, Z., Zhang, K., Gao, P., Qiao, Y., and Luo, P. Omniquant: Omnidirectionally calibrated quantization for

large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8Wuvhh0LYW.

Sun, Y., Liu, R., Bai, H., Bao, H., Zhao, K., Li, Y., Hu, J., Yu, X., Hou, L., Yuan, C., Jiang, X., Liu, W., and Yao, J. Flatquant: Flatness matters for llm quantization, 2025. URL https://arxiv.org/abs/2410.09426.

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q., Chi, E., Zhou, D., and Wei, J. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Tang, H., Sun, Y., Wu, D., Liu, K., Zhu, J., and Kang, Z. Easyquant: An efficient data-free quantization algorithm for llms. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9119–9128, 2023.

Team, G. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and Sa, C. D. QuIP$\#$: Even better LLM quantization with hadamard incoherence and lattice codebooks. In *Forty-first International Conference on Machine Learning*, 2024a. URL https://openreview.net/forum?id=9BrydUVcoe.

Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and Sa, C. D. QuIP#: Even better LLM quantization with hadamard incoherence and lattice codebooks, 2024b. URL https://arxiv.org/abs/2402.04396.

Tseng, A., Sun, Q., Hou, D., and Sa, C. D. QTIP: Quantization with trellises and incoherence processing. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024c. URL https://openreview.net/forum?id=7sdkLVuYCU.

van Breugel, B., Bondarenko, Y., Whatmough, P., and Nagel, M. Fptquant: Function-preserving transforms for llm quantization, 2025. URL https://arxiv.org/abs/2506.04985.

Williams, M. and Aletras, N. On the impact of calibration data in post-training quantization and pruning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10100–10118, 2024.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In Korhonen, A., Traum, D., and Màrquez, L. (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Zhong, W., Cui, R., Guo, Y., Liang, Y., Lu, S., Wang, Y., Saied, A., Chen, W., and Duan, N. AGIEval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2299–2314, 2024.

A. Learned paired quantization for coupled matrices

As discussed in Section 2, in the attention computation there are pairs of weight matrices that would not affect the attention output as long as their product is preserved. For these cases, we can couple the transformations of the two matrices so that no online operation is needed. Let $W_1 : d_1 \times d_2$ and $W_2 : d_2 \times d_3$ be the weight matrices of two consecutive linear layers, and M be an invertible matrix; then we have for any input to the first layer $X \in 1 \times d_1$,

$$XW_1MM^{-1}W_2 = XW_1W_2.$$

Hence, motivated by Eq. (3), we aim to learn $M : d_2 \times d_2$ such that applying quantization on W_1M and $M^{-1}W_2$ results in a small ℓ_2 error on matrix products, described below.

$$\min_{M,\ \widehat{\cdot}} \quad \|\widehat{W_1M}\widehat{M^{-1}W_2} - W_1W_2\|_F. \tag{5}$$

We refer to this quantity as the *paired quantization error* (PQE). Another design choice in minimizing PQE is the rounding mechanism, which exploits a joint design. In this section, we focus on learning the transformation matrix M with the aim of removing outliers, and discuss joint design of the rounding mechanism in Appendix B.

Similar to the Frobenius norm in Section 3.1, PQE is also not differentiable. Hence, we propose to minimize a surrogate loss. Let $U = W_1M$ and $V = M^{-1}W_2$. Let the following be channel-wise maximum absolute values of U and V, respectively:

$$m_u(i) = \max_j |U_{i,j}|, \qquad m_v(j) = \max_i |V_{i,j}|$$

$$\begin{aligned}
&\|\widehat{W_1M} - W_1M\|_F + \|\widehat{M^{-1}W_2} - M^{-1}W_2\|_F \\
&\quad \le \|W_1M\|_\infty + \|M^{-1}W_2\|_\infty \\
&\quad = \|U\|_\infty + \|V\|_\infty \\
&\quad \le \frac{1}{t}\log\left(\sum_{i=1}^{d_1} e^{tm_u(i)} + \sum_{j=1}^{d_3} e^{tm_v(j)}\right) \\
&\quad \triangleq \mathrm{LogSumExp}(U, V),
\end{aligned}$$

where the first inequality follows from Eq. (2) and the second inequality follows from the standard observation that maximum is smaller than log-sum-exp. Note that $\mathrm{LogSumExp}(U, V)$ is a 1-Lipschitz and t-smooth function. We minimize $\mathrm{LogSumExp}(U, V)$ over all invertible matrices M. Note that since we do not need to do any online rotations during inference, structured matrices are not necessary. Details on various optimization algorithms, regularization parameters, and analysis on optimization dynamics can be found in Appendix F.

B. Joint quantization of weight matrices

In Appendix A, we described how one can learn a transformation M such that the PQE with individual rounding is minimized. In this section, we modify the quantization operator itself ($\widehat{\cdot}$), to additionally minimize the PQE defined in Eq. (5).

The algorithm takes as inputs two weight matrices and a base rounding map Q, which permits fast inference-time dequantization. For example, Q can be uniform quantization or uniform quantization after applying a transformation learned from the previous step. The algorithm starts by quantizing W_1 independently. Then it proceeds iteratively, updating each quantized matrix to compensate for the quantization error in the other matrix. More specifically, given a quantized version $\widehat{W}_1$ of W_1, instead of quantizing W_2, it instead quantizes:

$$W_2' = \widehat{W}_1^{\,\dagger} W_1 W_2,$$

where $\widehat{W}_1^{\,\dagger}$ is the pseudoinverse of $\widehat{W}_1$, with the goal of having

$$W_1W_2' = W_1\widehat{W}_1^{\,\dagger}W_1W_2 \approx W_1W_2.$$

Table 6. Average relative ℓ_2 reconstruction error comparison on $W_v W_o$ with 4 bit per-channel quantization. I stands for independent rounding, and A stands for adaptive rounding. Averages are computed over all Gemma 2 2B layers.

VO Quantization method	UNIFORM (I)	RANDOM (I)	Learned (I)	Learned (A)
Average ℓ_2 loss	0.182	0.179	0.149	**0.117**

Then we update the quantized version of W_2 as $\widehat{W_2} = \mathrm{Q}(W_2') = \mathrm{Q}(\widehat{W_1}^{\dagger} W_1 W_2)$.

Algorithm 1 Adaptive rounding for matrix product quantization.

Input: Matrices to be quantized $W_1 : d_1 \times h$ and $W_2 : h \times d_2$, number of iterations I, base quantization function Q.

$\widehat{W_1} \leftarrow \mathrm{Q}(W_1)$
for $i \leftarrow 1$ **to** I **do**
  $\widehat{W_2} \leftarrow \mathrm{Q}(\widehat{W_1}^{\dagger} W_1 W_2)$.
  $\widehat{W_1} \leftarrow \mathrm{Q}(W_1 W_2 \widehat{W_2}^{\dagger})$.
end for
return $\widehat{W_1}, \widehat{W_2}$.

The above iterative quantization scheme still permits fast inference-time dequantization since both $\widehat{W_1}, \widehat{W_2}$ are obtained by the basic rounding scheme Q. Moreover, the iterative procedure can produce even lower ℓ_2 quantization error on the matrix product by explicitly compensating for the quantization loss in the other matrix.

We provide a toy example to illustrate this. Suppose one wants to quantize 2×2 matrices:

$$W = W_1 = W_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0.6 \end{bmatrix}$$

to 1 bit, such that the PQE of W_1 and W_2 is minimized. Uniform row-wise or column-wise quantization of either matrix will produce: $\widehat{W} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$, yielding $\widehat{W_1}\widehat{W_2} - W_1 W_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0.64 \end{bmatrix}$ and a PQE of $\|\widehat{W_1}\widehat{W_2} - W_1 W_2\|_F = 0.64$. Alternatively, applying even a partial iteration of Algorithm 1 yields a reduction in the PQE to 0.36. To see this, in our iterative quantization scheme, we first quantize $\widehat{W_1} \leftarrow \mathrm{Q}(W_1) = I$, according to uniform quantization. Updating $\widehat{W_2} \leftarrow \mathrm{Q}(\widehat{W_1}^{\dagger} W_1 W_2) = \mathrm{Q}(I W_1 W_2) = Q\left(\begin{bmatrix} 1 & 0 \\ 0 & 0.36 \end{bmatrix}\right) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, yielding a PQE of $\|\widehat{W_1}\widehat{W_2} - W_1 W_2\|_F = 0.36$.

We conduct experiments to validate the gain of learned matrix and adaptive rounding for a Gemma 2 2B model, listing the results in Table 6. The learned transformation reduces ℓ_2 error by over 16% relative to both random rotation and independent rounding. Adaptive rounding further reduces ℓ_2 error by 21%.

The time complexity of our iterative algorithm at each step is dominated by computing the Moore-Penrose inverse, which itself is dominated by the singular value decomposition (SVD) of each constituent matrix. The time complexity of the SVD of a $d \times h$ matrix is $O\left(d \cdot h \cdot \min(d, h)\right)$ (Golub & Van Loan, 2013). Therefore, the time complexity of the whole algorithm on input matrices of $d \times h$, number of iterations I, and with Q as the uniform quantization operation is $O\left(I \cdot d \cdot h \cdot \min(d, h)\right)$.

C. Related work

The idea of transforming weight matrices to make them easier to quantize has been widely explored in the literature. SmoothQuant (Xiao et al., 2023) and OmniQuant (Shao et al., 2024) use calibration data to apply activation-dependent scaling on weight matrices.

Applying random rotations to weight matrices has been shown to be an effective way of mitigating outliers prior to quantization. This idea has been explored in recent works to improve the performance of uniform quantization (Adepu et al., 2024; Ashkboos et al., 2024b; Chee et al., 2024; Liu et al., 2024). This has been

extended to use vector quantizers instead of scalar quantizers (Tseng et al., 2024c;a). Using non-uniform scalar quantizers has been studied in (Malinovskii et al., 2024). All these methods use calibration data to design the quantizers. There are methods that do not use rotations, but try to index outliers directly (Dettmers et al., 2024; Kim et al., 2024; Li et al., 2025), but these methods require one to index outliers and also require calibration data for good performance. To reduce the inference-time computation to rotate the matrices back, two main techniques have been used: (1) Exploiting the computational invariance property (Ashkboos et al., 2024a;b; Liu et al., 2024; Ma et al., 2024) in transformer blocks to avoid online rotations for "paired", composed, matrices, (2) Rotate the weight matrices using a restricted class of structured matrices (Adepu et al., 2024; Tseng et al., 2024b; Ashkboos et al., 2024b) such as Hadamard matrices, that admit fast online rotations.

Methods have also been proposed to introduce correlation between the quantization for different weights and channels to further reduce quantization error. For example, Frantar et al. (2023); Chee et al. (2024); Kim et al. (2025) use a calibration dataset to calculate the Hessian matrix of the output error with respect to each model weight, and use the Hessian matrix to introduce correlations between the quantization function of different weights. While most works quantize each layer independently, Liu et al. (2024); Tseng et al. (2024b); Egiazarian et al. (2024) use finetuning to introduce cross-weight dependencies to further reduce the quantization error.

Similar to our work, FlatQuant (Sun et al., 2025) also considers learned structured matrices for better quantization of non-paired matrices. They use the Kronecker product of two lightweight matrices for faster online reverse transformation. The concurrent work of FPTQuant (van Breugel et al., 2025) considers learning non-rotation matrices for the transformation matrix on (W_v, W_o) pair. However, all these learning-based methods rely on calibration data for optimizing the transformation.

D. Evaluation datasets

Table 7. Downstream benchmarks considered in this work. We report the average performance over *All* of these tasks, along with average performance over the *Core*, *Math*, and *Code* subsets. We use top-1 accuracy as the metric for all non-coding tasks, where the generated prefix for sampling tasks must match the reference. For coding tasks, we consider Pass@1 as the metric.

Task	Core	Math	Code	Notes	Eval Setting
MMLU (Hendrycks et al., 2021a)	x	x		5-shot	scoring
ARC (Challenge) (Clark et al., 2018)	x	x		0-shot	scoring
GSM8K (Cobbe et al., 2021)	x	x		8-shot	sampling
AGIEval (English) (Zhong et al., 2024)	x			3-5-shot	sampling
BBH (Suzgun et al., 2023)	x	x		0-shot	sampling
Winogrande (Sakaguchi et al., 2021)	x	x			scoring
HellaSwag (Zellers et al., 2019)	x			0-shot	scoring
MATH (Hendrycks et al., 2021b)		x		4-shot	sampling
ARC (Easy) (Clark et al., 2018)				0-shot	scoring
PIQA (Bisk et al., 2020)					scoring
SIQA (Sap et al., 2019)					scoring
BoolQ (Clark et al., 2019)				0-shot	scoring
TriviaQA (Joshi et al., 2017)				5-shot	sampling
NQ (Kwiatkowski et al., 2019)				5-shot	sampling
HumanEval (Chen et al., 2021)			x		sampling
MBPP (Austin et al., 2021)			x	3-shot	sampling

Table 7 lists the evaluation tasks considered in this work, along with how they are grouped into task suites. Performance on individual tasks and suites are presented in Appendix G.

E. Details for feedforward parameter sweep

Here we provide sweep details for producing Fig. 1. In this case we considered 4-bit quantization of the gating and linear weights of the feedforward network. We report model performance averaged over *All* downstream tasks,

and swept over various quantization hyperparameters in order to produce a range of models with varying intrinsic and extrinsic performance. We considered models without any quantization, vanilla uniform quantization, random rotation/Hadamard transformation prior to quantization, in addition to a range of models with learned block Hadamard transformations. For all learning runs, we set the number of iterations to 20,000, and swept over the following parameters for learning transformations of the FFW:

- Learning rate: $\{0.1, 0.5, 1.0\}$
- Block diagonal size: $\{2, 4, 8, 16, 64, 256\}$
- Number of Hadamard matrices: $\{0, 1, 2\}$

Each point in Fig. 1 corresponds to the relative quantization error with respect to a particular subset of FFW (gating, linear, or the mean of the two) against the downstream performance for one of these models.

F. Paired quantization hyperparameter details

F.1. Alternative choices for the pseudo-loss

Initially we considered other pseudo-loss choices for learning paired transformations. Using the notation described in Appendix A, we considered (1) the mean of squared channel-wise maxima:

$$\frac{\sum_{i=1}^{m}(M_i^u)^2}{m} + \frac{\sum_{j=1}^{n}(M_j^v)^2}{n} \frac{\sum_{i=1}^{d_1} m_u(i)^2}{d_1} + \frac{\sum_{j=1}^{d_3} m_v(j)^2}{d_3}$$

and (2) the weighted mean of squared channel-wise maxima:

$$\frac{\|V'\|_F \sum_{i=1}^{d_1} m_u(i)^2}{d_1} + \frac{\|U'\|_F \sum_{j=1}^{d_3} m_v(j)^2}{d_3}$$

This latter objective is motivated by the intuition that quantization error in one weight matrix has the potential to be magnified if the norm of its paired weight is large. Thus, quantization errors should be penalized accordingly. This objective explicitly acknowledges the fact that U' and V' directly interact in the computation graph. We sweep over the choice of this pseudo-loss in the following subsection.

F.2. Hyperparameter tuning

We explored several hyperparameters for learning paired VO transformations. In this section, we present sweeps over these hyperparameters, noting which ones the PQE happened to be sensitive to. We fix number of iterations to 100,000 for all runs, and consider Cayley SGD (with $\beta = 0.1$ momentum) as well unconstrained optimization with Adam ($\beta = 0.1$), learning M to transform V/O for all layers in a Gemma2-2B-PT model. We compute the mean PQE across all layers, assuming uniform channel-wise 4-bit quantization for $\hat{\cdot}$. Note that we report the relative PQE in this section – relative to $\|W_v^T W_o\|_F$. This ensures that the average PQE is not unduly influenced by the magnitude of $W_v^T W_o$ for any given layer.

We varied:

- base learning rate $lr \in 10^{\{-4,-3,-2,-1,0\}}$
- orthonormal regularization weight $\lambda_{orth} \in \{0, 0.01, 0.1, 1\}$
- LogSumExp temperature $t \in \{1, 5, 10\}$
- Pseudo-loss $\in$ {LogSumExp, Sum of Squares, Weighted Sum of Squares}

LogSumExp temperature PQE as a function of t, the temperature of the LogSumExp pseudo-loss, is shown in Fig. 2. While the mean is relatively unchanged across settings of t, the variance of the ultimate PQE increases with t. We chose $t = 5$ for LogSumExp pseudo-loss experiments, as PQE was relatively insensitive to this hyperparameter.

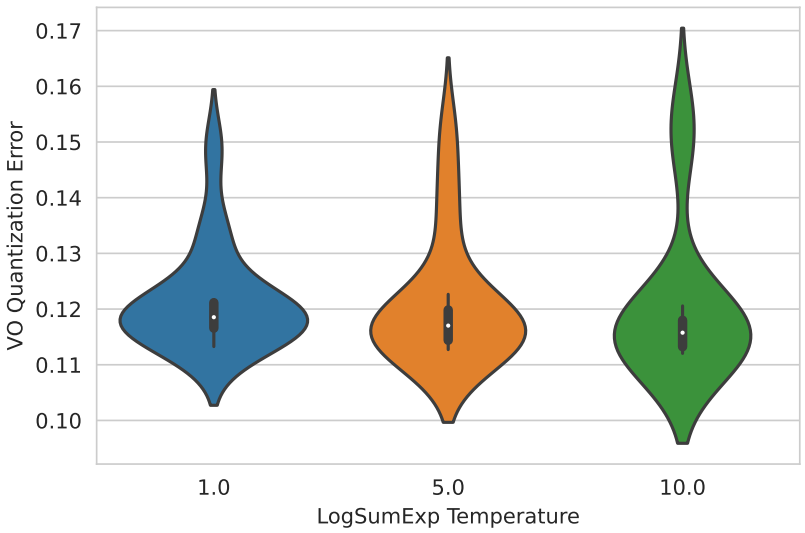


Figure 2. Distribution over the VO product relative PQE as a function of the temperature on the LogSumExp pseudo-loss, optimized with Adam. Each violin encompasses a sweep over learning rate in $10^{\{-4,-3,-2,-1\}}$, and orthogonal regularization weight in $\{0, 0.1, 1\}$ for the given value of t. Note that we exclude runs with learning rate of 1.0, as these runs diverged (Fig. 3).

Learning rate & Pseudo-loss While t had little effect on PQE, both the base learning rate and choice of pseudo-loss affected the stability of the learning curve. Fig. 3 displays learning curves for Cayley SGD and Adam as a function of learning rate and pseudo-loss.

PQE tends to converge quickly with Adam assuming that the learning rate is small enough (less than 0.1). This is true irrespective of the choice of pseudo-loss. While Cayley SGD also smoothly reduces PQE, it benefits from a higher learning rate for PQE to converge in the allotted iterations.

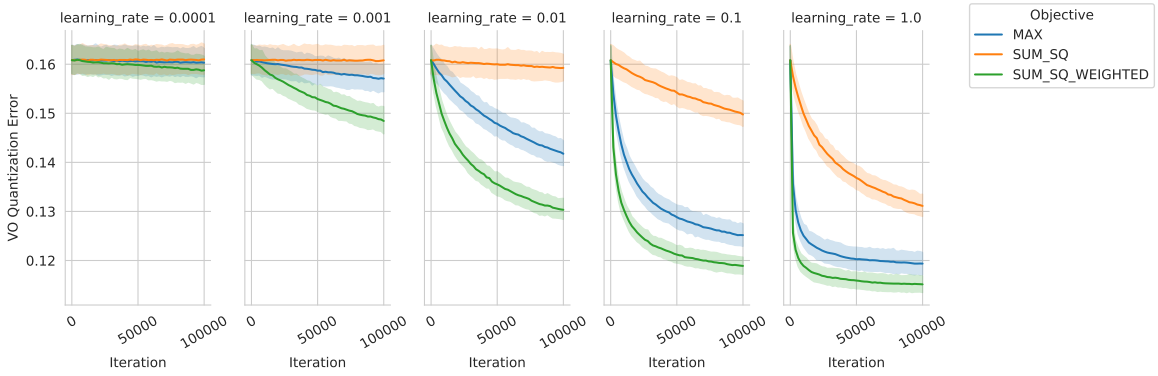
Constraints on M We observe that M need only be invertible to ensure that the transformation is computationally invariant. Prior work learns M such that M is strictly a rotation matrix (Liu et al., 2024). In this work, we varied the degree to which M is orthonormal by minimizing a pseudo-loss directly with Adam without strictly enforcing orthonormality. We experimented with adding a regularization term to the loss to encourage M to not stray too far from orthonormality:

$$\frac{\|MM^T - I\|_F}{\sqrt{d}}.$$

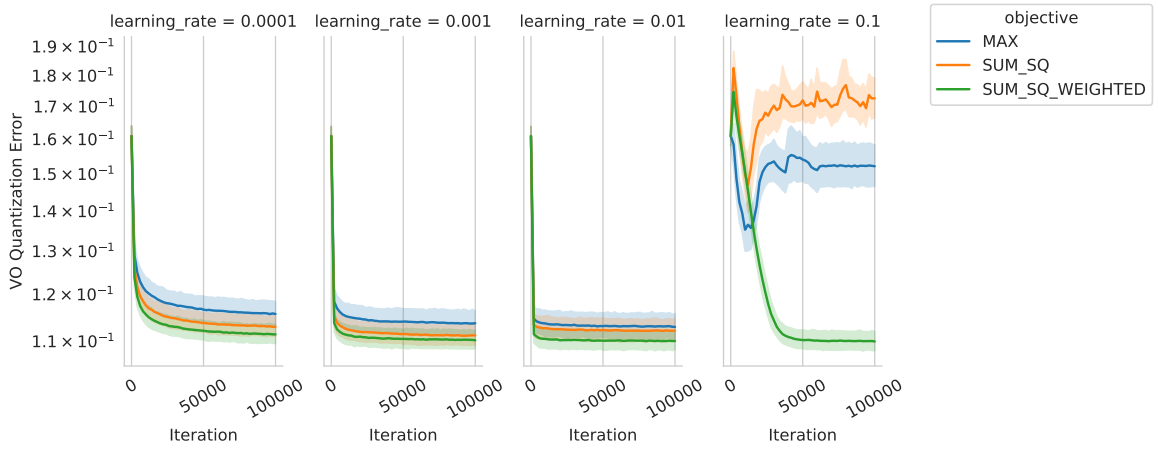
We controlled the weight of this term with λ_{orth}

Importance of learning an orthonormal M Although prior work has learned M such that M is a rotation, in practice, M need only be invertible to ensure computational invariance. To determine how important it is that M be orthonormal, we select the minimum PQE run per objective, optimization method, and orthonormal regularization weight, while sweeping over base learning rate, as well as t for the LogSumExp runs.

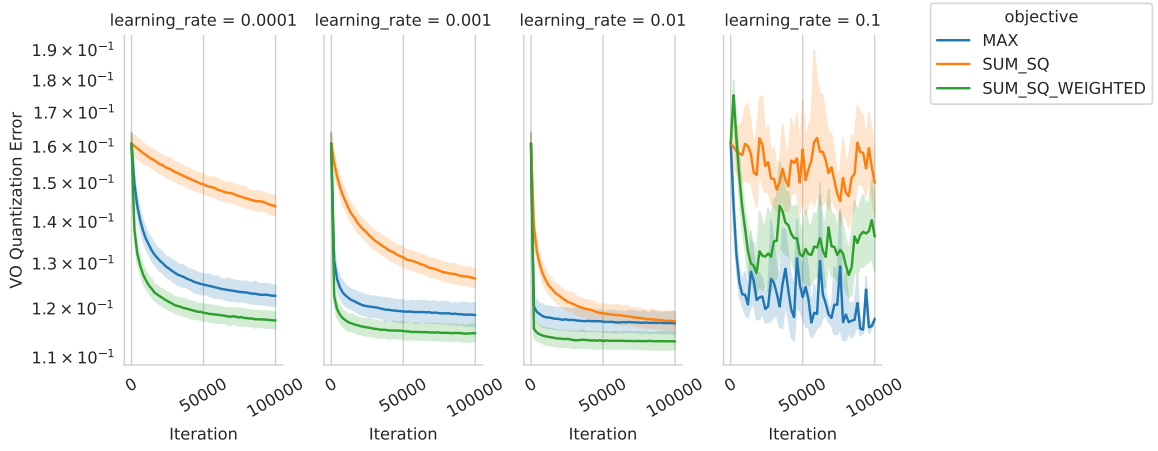
Table 8 displays the relative PQE for each of these runs along with the minimum and maximum eigenvalue, averaged across all layers, for the learned M. The lowest PQE transformations are learned through unconstrained, Adam, optimization without any weight on the orthonormal regularization term, with a mean relative PQE of 0.110 vs. 0.118 for the next best run. Note also that the average minimum and maximum eigenvalue for these transformations are quite large, suggesting the learned transformations are far from rotations.It is important to also note that learning strict rotation transformations via Cayley SGD yields mean PQE which is at least 21.7% higher than the best unconstrained learning run.



(a) Cayley SGD.



(b) Adam; $\lambda_{orth} = 0$.



(c) Adam; $\lambda_{orth} = 1$.

Figure 3. Learning curves for Cayley SGD (top) and Adam with various orthonormal regularization weights (center and bottom), with each pseudo-loss a separate line. Learning rate is varied along columns. PQE is on the y-axis and iteration count on the x-axis. Each line corresponds to the mean PQE across all model layers for a given pseudo-loss, with the 95% bootstrap confidence interval indicated by the shaded region.

Table 8. Relative mean PQE across all layers after 100K iterations, as a function of objective, optimization method, and λ_{orth}. The average minimum/maximum eigenvalue across all layers for the learned M are given as well.

Objective	Opt	λ_{orth}	PQE	$\min \sigma$	$\max \sigma$
SumSqWted	Adam	0.00	0.110	338.848	894.407
LogSumExp	Adam	0.10	0.118	0.978	1.062
LogSumExp	Adam	0.01	0.118	0.845	2.175
LogSumExp	Adam	0.00	0.120	0.876	2.508
LogSumExp	Adam	1.00	0.120	0.981	1.049
SumSqWted	Cayley	0.00	0.134	0.992	1.000
LogSumExp	Cayley	0.00	0.142	0.997	1.001
SumSq	Cayley	0.00	0.152	0.999	1.001
SumSqWted	Adam	0.01	0.166	2.503	912.959
SumSqWted	Adam	0.10	0.185	1.915	905.954
SumSqWted	Adam	1.00	0.200	1.460	730.075
SumSq	Adam	0.10	0.255	0.867	165.328
SumSq	Adam	0.01	0.261	0.886	222.902
SumSq	Adam	1.00	0.462	0.869	138.006
SumSq	Adam	0.00	0.639	0.551	971.175

Note that PQE is an intrinsic measure of quality, and one must also evaluate these quantized models downstream to note any degradation. In terms of stability with respect to λ_{orth}, learning transformations that are close to rotation, and achieving low PQE, unconstrained optimization with the LogSumExp pseudo-loss yields a better solution than weighted sum of squares or learning M using natural gradient descent. With that in mind, we selected unconstrained optimization with LogSumExp pseudo-loss unless mentioned otherwise.

G. Downstream performance on individual tasks

Table 9 displays the performance of CafeQ against competing quantization methods on each individual task. CafeQ outperforms all other quantization methods on each task suite: all, core, math, and coding tasks. It also achieves the highest quantized performance on 12 out of 16 of the individual tasks, suggesting that overall performance is not driven by outperformance on a small subset of tasks.

Table 9. Individual task performance along with task suite averages for various Gemma 2 2B quantized models. The best-performing quantized model on each task/suite is noted in bold. Performance of the original Gemma 2 2B is given in the *None* column.

Benchmark	None	GPTQ	Uniform	Random	CafeQ
MMLU	51.9	48.2	43.9	47.5	**50.1**
ARC-C	50.3	47.9	46.3	**49.7**	49.0
GSM8K	21.2	12.4	9.5	13.3	**18.6**
AGIEval	31.7	26.9	26.5	27.2	**28.9**
BBH	42.6	37.8	35.0	34.3	**38.8**
Winogrande	68.6	**67.8**	67.4	67.5	66.9
HellaSwag	73.9	71.3	69.5	70.5	**71.9**
MATH	16.0	9.0	8.5	10.0	**13.2**
ARC-e	80.7	78.2	76.3	78.7	**78.9**
PIQA	78.5	77.0	76.9	77.6	**78.0**
SIQA	51.6	50.8	49.5	51.2	**51.4**
Boolq	73.0	**73.4**	63.6	72.4	67.1
TriviaQA	60.3	52.3	48.5	49.4	**55.2**
NQ	17.3	13.4	11.5	12.8	**15.2**
HumanEval	19.5	14.6	9.8	**15.2**	14.6
MBPP	30.4	18.0	17.6	18.8	**26.0**
Average (All)	48.0	43.7	41.3	43.5	**45.2**
Average (Core)	48.6	44.6	42.6	44.3	**46.3**
Average (Math)	41.8	37.2	35.1	37.0	**39.4**
Average (Code)	25.0	16.3	13.7	17.0	**20.3**