CAGE: Curvature-Aware Gradient Estimation For Accurate Quantization-Aware Training

Rush Tabesh [*]
ISTA
stabesh@ista.ac.at

Mher Safaryan [*]
ISTA
msafaryan@ista.ac.at

Dan Alistarh
ISTA & Red Hat AI
dalistarh@ista.ac.at

Abstract

Despite significant work on low-bit quantization-aware training (QAT), there is still an accuracy gap between such techniques and native training. To address this, we introduce **CAGE** (Curvature-Aware Gradient Estimation), a new QAT method that augments the straight-through estimator (STE) gradient with a curvature-aware correction designed to counteract the loss increase induced by quantization. CAGE is derived from a multi-objective view of QAT that balances loss minimization with adherence to quantization constraints, yielding a principled correction term that depends on local curvature information. On the theoretical side, we introduce the notion of Pareto-optimal solutions for quantized optimization, and establish that CAGE yields strong convergence guarantees in the smooth non-convex setting. In terms of implementation, our approach is optimizer-agnostic, but we provide a highly-efficient implementation that leverages Adam statistics. When pre-training Llama-style models of up to 800M-parameters, CAGE *recovers over 10% of the quantization-induced loss increase* in the W4A4 regime over outlier-mitigation methods. These results indicate that curvature-aware gradient corrections can bridge the remaining performance gap beyond current outlier-handling methods.

1 Introduction

Quantization has emerged as a standard technique for improving the computational efficiency of large language model (LLM) deployments, as prominent open-source models such as Llama, Gemma, and GPT-OSS are released in open-source formats [30; 29; 25]. Yet, the vast majority of research in this area has concentrated on *post-training quantization (PTQ)*, where a fully trained model's weights are quantized by applying numerical algorithms over a small calibration dataset [13].

Relatively less is known about quantization-aware training (QAT), where the model learns to adapt to the constraints of quantization during the training process itself. QAT is more computationally-expensive, but can yield better accuracy. The state of the art for QAT methods has long been dominated by the straight-through estimator (STE), a method first suggested by Hinton [15] and formalized by Bengio et al. [5], integrated by default in deep learning frameworks like PyTorch [28]. This approach bypasses the non-differentiable quantization function in the backward pass, by approximating its gradient with the identity. While versatile, STE is known to suffer from slow convergence and instability. A body of work, such as LSQ [12], LSQ+ [6], and the recently proposed QuEST [11; 6; 26], has introduced improved approaches to stabilize and accelerate this process learnable quantization parameters or more sophisticated gradient scaling. Yet, these methods are heuristic, and do not offer formal convergence guarantees.

[*]Equal contribution

Contribution. In this paper, we address this gap by examining QAT from both a theoretical and a practical standpoint. We reframe QAT as a multi-objective optimization problem, where the goal is to simultaneously minimize the task loss and the quantization error. From this perspective, we identify a condition for Pareto-optimality, where any improvement in one component of the objective necessarily leads to a degradation in the other. This formulation inspires a new algorithm, called CAGE, for **C**urvature-**A**ware **G**radient **E**stimation, which provides theoretical guarantees with high accuracy and efficiency in practice. More precisely, our technical contributions are as follows:

- The CAGE we introduce provides a principled method for augmenting the standard STE gradient with a curvature-aware correction term, derived directly from our Pareto-optimality condition. This explicitly counteracts the increase in loss induced by the quantization step, by leveraging local second-order information about the loss landscape (i.e., the Hessian).
- From the theoretical perspective, we prove that CAGE possesses strong ergodic convergence guarantees to a Pareto-optimal point in the smooth non-convex setting, directly addressing the lack of guarantees that is a primary limitation of prior work. Our approach is optimizer-agnostic, and we provide a highly efficient implementation that utilizes statistics readily available in adaptive optimizers such as Adam.
- We validate the effectiveness of CAGE through extensive experiments, including pre-training Llama-style models of up to 800M parameters from scratch. Our results demonstrate that in the challenging low-bit W4A4 (4-bit weights and 4-bit activations) regime, CAGE *recovers over 10% of the quantization-induced loss increase* when compared to strong baselines that incorporate modern outlier-mitigation techniques.

In summary, our findings show that incorporating curvature-aware gradient corrections is a key step toward bridging the remaining performance gap between low-bit quantized models and their full-precision counterparts.

2 Related Work

The central challenge in QAT is the non-differentiable quantization function, with zero gradients almost everywhere. The standard is the STE, initially suggested by Hinton [15] and formalized by Bengio et al. [5], which bypasses the quantization operator during the backward pass. Yet, STE is known to often lead to instability and suboptimal convergence [32].

Thus, substantial research has focused on STE refinements, via e.g. learnable quantization parameters, e.g. [12], or improving the accuracy of the gradient itself, such as AdaSTE [20] (working via complex bi-level optimization), or ReSTE [31] which uses a rectified estimator which balances the estimation error and the gradient stability. In addition, ProxQuant [3] reformulated QAT as a regularized learning problem solved via proximal gradient methods. Instead of relying solely on STE, ProxQuant applies a prox operator between stochastic gradient steps to encourage quantization. Liu et al. [23] proposed Reinmax, based on the observation that the STE acts as a first-order approximation of the gradient and proposed a second-order estimator via Heun's method. In addition, Markov Random Field (MRF) and lifted probability space formulations [1] represent each parameter of the model as weighted average of quantization nodes and minimizes the loss with respect to these probabilistic weights.

The Mirror Descent formulation interprets full-precision parameters as the dual of quantized ones and analyses the cases where quantization operator generates a valid mirror map. Particularly, if we take softmax projection as quantization operator, then the dual iterates of STE follow Exponentiated Gradient Descent (EGD) [2]. By contrast, CAGE introduces a new approach rooted in a multi-objective optimization perspective of QAT. Instead of refining the first-order approximation or employing standard regularization, CAGE introduces a principled, curvature-aware correction term to the gradient, with theoretical guarantees.

3 CAGE: Curvature-Aware Gradient Estimation

3.1 Problem Description and Motivation

Intuitively, the key question in quantization-aware training (QAT) is to construct a gradient estimator that updates the parameters towards the an "optimal" quantized model, i.e. a constrained model which minimizes the loss. More precisely, the optimization problem underlying QAT can be described by the following constrained optimization:

$$\min_{x \in \mathcal{Q}} f(x), \tag{1}$$

where $\mathcal{Q} = \{x \in \mathbb{R}^d \colon x = Q(x)\}$ is the constraint set of quantized parameters and $Q \colon \mathbb{R}^d \to \mathbb{R}^d$ is the quantization operator. An alternative way of looking at the same problem is to consider its unconstrained reformulation and minimizing the objective $f(Q(x))$ with respect to $x \in \mathbb{R}^d$ [21]. The key issue here is the non-differentiability of quantization operator Q. As such, the straight-through estimator [4] is the default choice to approximate the gradient of $f(Q(x)$. Specifically, $\nabla_x[f(Q(x))] = JQ(x)^\top \cdot \nabla f(Q(x))$ is approximated by $\nabla f(Q(x))$. Effectively, this replaces the Jacobian of the quantization operation by the identity matrix [7; 17; 21].

Error feedback reformulation. One way to gain more insight into the dynamics of STE is to examine it through the lens of error feedback. It is easy to observe that SGD-based training with STE over the iterate x_t is equivalent to an instance of error-feedback [8] with quantized parameters $w_t = Q(x_t)$ and quantization error $e_t = x_t - Q(x_t)$, as described over iterations $t \geq 0$ below:

$$x_{t+1} = x_t - \alpha \widetilde{\nabla} f(Q(x_t)) \quad \iff \quad \begin{cases} g_t = \alpha \widetilde{\nabla} f(w_t) - e_t \\ w_{t+1} = Q(w_t - g_t) \\ e_{t+1} = (w_t - g_t) - w_{t+1}. \end{cases}$$

Effectively, the equivalent right-hand-side formulation says that STE is equivalent to a process where the quantization error e_{t+1} due to weight quantization at a given step $(w_t - g_t) - w_{t+1}$ is fed into the *gradients* at the next step. This would be "correct" if the loss function f were an isotropic quadratic function, but is not the right update in general.

Multi-objective perspective. Prior work on convergence guarantees for the problem (1) in the non-convex setting attempt to find quantized point $Q(x^*)$ that is a stationary point for the loss f, namely $\nabla f(Q(x^*)) = 0$. However, as quantization is not an invertible transformation, this goal cannot be achieved in general, and current convergence guarantees have non-vanishing terms in their rate, proportional to quantization error [22; 8; 27]. For the sake of illustration, consider a toy example where our scalar loss function is $f(x) = \frac{1}{2}(x - \frac{1}{2})^2$, and the quantization operator $Q(x) = \lfloor x \rfloor$ gives the integer part of the one dimensional input $x \in \mathbb{R}$. Clearly, $\nabla f(Q(x)) = \lfloor x \rfloor - \frac{1}{2}$ does not vanish at any point, with the smallest absolute value being $\frac{1}{2}$, coming from the quantization/rounding error.

Instead, we propose to view the QAT problem in Equation (1) from the perspective of multi-objective optimization. The first objective is to minimize the loss $f(x)$ (in non-convex setting this becomes finding a stationary point, namely solving $\nabla f(x) = 0$), while the second is to satisfy quantization constraints $x \in \mathcal{Q}$. In other words, we aim to find a solution x^* for which $\nabla f(x^*) = 0$ and the distance between x^* and $Q(x^*)$ is minimized. As noted before, it is easy to see that, these two conditions do not have to hold in general, meaning $\nabla f(x)$ can be non-zero for any quantized point $x \in \mathcal{Q}$. Instead, we define solutions as *Pareto optimal* points that balance these two conditions.

Let $x \in \mathbb{R}^d$ be the current state of parameters. If we update towards $-\nabla f(x)$ (or any other direction that aligns with it) with small enough step-size, then we would minimize the loss. Clearly, the opposite direction would maximize the loss. Analogously, if we update towards $Q(x) - x$, then we would reduce quantization error. To have those two objectives in balance, we want our iterates to converge to some Pareto-optimal state x^* such that $\nabla f(x^*) = \lambda(Q(x^*) - x^*)$ for some scalar $\lambda > 0$. In this case, any sufficiently small update applied to x^* would hurt at least one of the objectives. In other words, any local improvement of one objective would be at the cost of other objective. Therefore, we define λ-*Pareto optimal* solution x^* for the problem (1) as

$$\nabla_{\lambda \mathrm{P}} f(Q(x^*)) \coloneqq \nabla f(x^*) + \lambda(x^* - Q(x^*)) = 0, \tag{2}$$

where $\lambda > 0$ is a parameter balancing the two objectives. Recalling the toy example mentioned earlier, the value $x^*(\lambda) = \frac{1}{2(1+\lambda)}$ is λ-Pareto optimal for any $\lambda > 0$. Note that there can be many Pareto optimal solutions for the same λ. For example, $x^* = -1/4$ is another 1-Pareto optimal value.

Motivated by this multi-objective viewpoint and the new optimality condition in Equation (2), we propose incorporating the quantization error $e_t = x_t - Q(x_t)$ directly into the training dynamics. By doing so, we implicitly regularize the initial task loss, which can be made explicit if the quantization scheme is smooth (as we demonstrate in the theoretical analysis). Indeed, under the smoothness assumption of Q (Assumption 3), the quantization error can be expressed as $x - Q(x) = \nabla\phi(x)$ for some smooth regularizer $\phi(x)$. Notably, our approach naturally avoids the need to handle the (non-existent) Jacobian $JQ(x)$ of the quantization operator and bypasses the computation of a proximal step for the explicit "quadratic" regularizer $\frac{\lambda}{2}\|x - Q(x)\|_2^2$, which would be infeasible for the dynamic quantization used in our experiments [3].

Noting that our approach is optimizer-agnostic, we propose two ways to incorporate the quantization error, referred to as the *coupled* and *decoupled* corrections. The distinction lies in whether the quantization error is added to the current gradient before it is passed to the optimizer (coupled correction) or added on top of the model update computed by the optimizer (decoupled correction).

If the base optimizer is SGD, then these two corrections are identical and we provide theoretical convergence guarantees to a Pareto-optimal point in the smooth non-convex setting. However, for statefull optimizers, such as Adam, these two corrections produce conceptually different schemes, which we discuss below (subsuming bias correction terms into the learning rate α):

$$\begin{aligned}
\text{Mini-batch gradient:}\quad g_t &= \widetilde{\nabla} f(x_t) + \lambda_t e_t, \;\; \text{(coupled correction)} \\
\text{Optimizer states:}\quad m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t,\; v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \\
\text{Optimizer update:}\quad \Delta_t &= m_t/(\sqrt{v_t} + \varepsilon) + \lambda_t e_t, \;\; \text{(decoupled correction)} \\
\text{Model update:}\quad x_{t+1} &= x_t - \alpha \Delta_t,
\end{aligned}$$

To compare these two correction variants in the context of the Adam optimizer, note that the coupled correction term $\lambda_t e_t$ becomes $(1-\beta_1)\lambda_t e_t/(\sqrt{v_t} + \varepsilon)$ after being processed by the optimizer states. In contrast to the decoupled correction, the coupled version introduces diagonally-preconditioned curvature-aware correction term, where the curvature is approximated using the Adam statistics already maintained within the optimizer states.

Relationship to LOTION. Concurrent work [18] proposes an approach called LOTION, which smooths the quantized loss using unbiased randomized rounding (RR) via $f(Q(x)) \approx \mathbb{E}_{\epsilon\sim\mathrm{RR}}[f(x+\epsilon)]$, and considers its second-order expansion as a surrogate loss to minimize, i.e., $f(x+\epsilon) \approx f(x) + \epsilon^\top \nabla f(x) + \frac{1}{2}\epsilon^\top \mathbf{H}(x)\epsilon$. Due to the unbiasedness of rounding (i.e., $\mathbb{E}[\epsilon] = 0$), the first-order term of the expansion vanishes, and the smoothed loss becomes the original loss $f(x)$ with an additional curvature-aware regularization term $\frac{1}{2}\mathrm{Tr}\left(\mathbf{H}(x)\,\mathrm{Cov}[\epsilon]\right)$, where the exact Hessian matrix is replaced by its Gauss–Newton approximation. If we further apply the empirical Fisher approximation and approximate the diagonal entries of the Gauss–Newton matrix using Adam statistics, this regularization corresponds to our coupled correction variant. However, computing the gradient estimator for the regularized loss requires differentiating the regularizer, which involves third-order derivatives of the loss.

In contrast to LOTION, we propose to regularize the training dynamics directly rather than the loss itself. The theoretical advantage of our approach is that the regularized dynamics naturally arise from our Pareto-optimality condition, while the practical benefit is that it avoids the need to differentiate or compute the proximal operator of a loss regularizer. In addition, LOTION requires a full-precision forward pass, and the final model is a weight-only quantized model, while CAGE training is done with weights and activations quantized, where it can also benefit from high-performance low-precision GEMM kernels for a more efficient training and inference.

3.2 Theoretical Analysis

In this section, we present our main convergence result for CAGE when the base optimizer is SGD. In this case, coupled and decoupled versions coincide with the following iterates:

$$x_{t+1} = x_t - \alpha(\widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))), \quad t = 0, 1, \ldots. \tag{3}$$

We use the following assumptions regarding the smoothness of loss and nature of stochastic noise.

Assumption 1 *The loss function $f\colon \mathbb{R}^d \to \mathbb{R}$ is lower bounded by some $f^* \in \mathbb{R}$ and is L_f-smooth, namely, $\|\nabla f(x) - \nabla f(y)\|_2 \le L_f \|x - y\|_2$, for any $x, y \in \mathbb{R}^d$.*

Assumption 2 *For all iterates t, the stochastic gradient $\widetilde{\nabla} f(x_t)$ is unbiased, namely $\mathbb{E}[\widetilde{\nabla} f(x_t)] = \nabla f(x_t)$, and the variance is bounded $\mathbb{E}[\|\widetilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2] \le \sigma^2$ for some constant $\sigma^2 \ge 0$.*

Both assumptions are quite standard in the optimization literature and, in some sense, minimal to get meaningful convergence guarantees. Additionally, to handle quantization operator Q in the analysis, it is common to approximate the actual non-smooth quantization operator Q with smooth surrogates with annealing hyperparameters.

Assumption 3 *There exists some L_ϕ-smooth function $\phi\colon \mathbb{R}^d \to \mathbb{R}$ such that the quantization error $x - Q(x) = \nabla \phi(x)$.*

Note that Assumption 3 is essentially equivalent to the Lipschitz continuity of the quantization operator Q. The purpose of formulating this condition using an auxiliary function $\phi(x)$ is to hint that the iterates in (3) aim to minimize the regularized loss $f(x) + \lambda\phi(x)$, as we elaborate on later in the analysis. Next, we present the convergence result.

Theorem 1 *Let Assumptions 1, 2, 3 hold and $L = L_f + \lambda L_\phi$. Then, for any $\lambda \ge 0$, the iterates (3) with step-size $\alpha = \min(\frac{1}{L}, \frac{1}{\sqrt{T}})$ satisfy*

$$\mathbb{E}\|\nabla_{\lambda \mathrm{P}} f(Q(\hat{x}))\|^2 \le \frac{2}{\sqrt{T}} \left(f(x_0) - f^* + \lambda \max_{x \in [x_0, x_T]} \|x - Q(x)\| \|x_0 - x_T\| + \frac{L\sigma^2}{2} \right) \max(1, \tfrac{L}{\sqrt{T}}),$$

where $\hat{x}$ is a random iterate drawn from $\{x_0, x_1, \ldots, x_{T-1}\}$ uniformly at random.

Discussion. First, observe that the convergence bound shows strong ergodic convergence to a Pareto-optimal state of problem 1, with a convergence rate that is optimal in the unquantized case. The convergence measure is defined with respect to our proposed Pareto-gradients (2) and, importantly, there is no non-vanishing term in the upper bound because of quantization error. In practical settings, both the distance $\|x_0 - x_T\|$ and the quantization error $\|x - Q(x)\|$ are expected to remain bounded for any total number of training steps T. Therefore, this implies that the iterates (3) achieve $\mathcal{O}(\frac{1}{\sqrt{T}})$ ergodic convergence to the Pareto-optimal solution.

3.3 Practical Implementation

Since our main application is in training and fine-tuning LLMs, we implement CAGE on top of AdamW [24]. In contrast to LOTION [18], we focus on the *decoupled* update rule and keep the correction term outside the preconditioning path (Alg. 1), which we have found to work better in practice for this application. The base optimizer AdamW processes the mini-batch gradients g_t obtained from STE, and the CAGE correction acts as a lightweight, elementwise post-step on the parameters. This decoupling has multiple practical advantages: (1) it preserves the optimizer's well-understood behavior, (2) it keeps the Pareto stationarity direction $x - Q(x)$ from being distorted by preconditioning, which empirically stabilizes training in low-bit regimes, and (3) it avoids numerical errors by decoupling terms with different behavior.

Decoupled update. Given current set of parameters x_t and STE gradient estimator g_t through backpropagation, AdamW performs

$$\tilde{x}_{t+1} = x_t - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} \quad \text{with} \quad \hat{m}_t = \frac{m_t}{1 - \beta_1^t},\ \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

with additional decoupled weight decay regularization in the form of $x_t \leftarrow (1 - \alpha\omega)x_t$.[1] The CAGE correction term then pushes the parameters toward the quantized support via the instantaneous quantization error:

$$e_t := x_t - Q(x_t), \qquad x_{t+1} = \tilde{x}_{t+1} - \alpha\lambda_t \bar{e}_t$$

[1]We use the common formulation where weight decay is applied before the Adam step; other equivalent placements are fine so long as decay is not mixed into g_t.

Algorithm 1 CAGE-AdamW (decoupled)

```
Require: Initial parameters $x_0$; total steps $T$; AdamW hyperparameters $\beta_1, \beta_2, \alpha, \omega, \varepsilon$; Quantization function $Q$;
Require: CAGE coefficient $\lambda$, silence ratio $s$
Ensure: Parameters $x_T$
 1: Initialize $m_0 \leftarrow 0,\ v_0 \leftarrow 0,\ \widehat{e}_0 \leftarrow 0$
 2: for $t = 1, 2, \dots, T$ do
 3:     $r_t \leftarrow t/T$                                          ▷ training progress ratio
 4:     if $r_t \le s$ then
 5:         $\lambda_t \leftarrow 0$
 6:     end if
 7:     $\lambda_t \leftarrow \lambda \cdot \frac{r_t - s}{1 - s}$
 8:     Sample minibatch and compute stochastic gradient $g_t$ with quantized forward pass.
 9:     $x_t \leftarrow (1 - \alpha\omega)\, x_t$                          ▷ decoupled weight decay
10:     $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$
11:     $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2)\, g_t \odot g_t$
12:     $\hat{m}_t \leftarrow m_t/(1 - \beta_1^t);\quad \hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$
13:     $\tilde{x}_{t+1} \leftarrow x_t - \alpha\, \hat{m}_t / \left(\sqrt{\hat{v}_t} + \varepsilon\right)$
14:     $e_t \leftarrow x_t - Q(x_t)$                            ▷ quantization error (no grad)
15:     $x_{t+1} \leftarrow \tilde{x}_{t+1} - \alpha\, \lambda_t\, e_t$                ▷ decoupled correction
16: end for
17: return $x_T$
```

Warmup schedule. In practice, we have found that activating the correction from the very beginning of training can over-constrain the dynamics before the model settles into the loss landscape. Similar to standard learning-rate warmup schedules, we introduce a *silence period* ratio $s \in [0, 1)$, which skips the correction for the initial sT steps and then linearly ramps it up to its target magnitude:

$$r_t := \tfrac{t}{T}, \qquad \lambda_t = \begin{cases} 0, & r_t \le s, \\ \lambda \cdot \dfrac{r_t - s}{1 - s}, & r_t > s. \end{cases}$$

In practice, we have found parameters $s \in [0.8, 0.9]$ with a linear ramp and $\lambda \in [0.5, 2]$ work robustly; we will ablate these choices in §4. The ramp prevents abrupt shifts in the optimization objective and reduces loss spikes in training.

4 Experiments

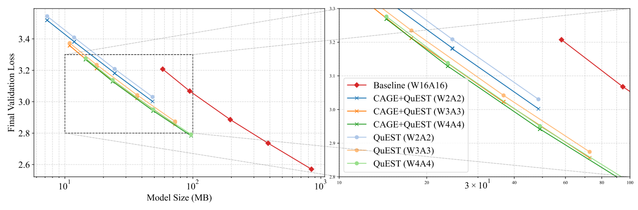

Figure 1: Validation loss versus model size (bytes) for weight-only quantization using W2/W3/W4, comparing CAGE+QuEST to baseline QuEST and BF16. Observe that CAGE yields consistently lower loss.

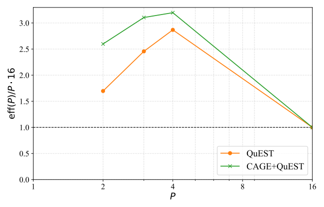

Figure 2: Fitted parameter efficiency eff(P) across precisions: CAGE increases effective model capacity over QuEST, peaking near 4-bit.

Quantization pipeline. Our experiments use a *row-wise* quantizer as in QuEST [26]. Let $x \in \mathbb{R}^d$ denote a tensor row , and let $H \in \{\pm\frac{1}{\sqrt{d}}\}^{d\times d}$ be a (Walsh-)Hadamard transform. We define the

Table 1: Pretraining results (final validation perplexity; lower is better) for W4A4 across model sizes. HT = Hadamard transform. BF16 is a full-precision reference.

Method	30M	50M	100M	200M	800M
CAGE + HT	**26.277**	**22.747**	**18.944**	**16.166**	**12.049**
QuEST + HT	26.475	23.062	19.123	16.311	12.203
CAGE (no HT)	27.287	23.781	19.630	16.596	13.089
QuEST (no HT)	27.401	23.991	19.799	16.701	13.852
BF16 (reference)	24.715	21.491	17.923	15.422	11.541

pre-quantized tensor

$$z = Hx.$$

For bit-width b, we quantize symmetrically with integer bounds $q_{\max} = 2^{b-1} - 1$ and $q_{\min} = -2^{b-1}$. Using the pre-computed MSE-optimal Gaussian clipping factor k_b (per bit-width), with $\sigma = \sqrt{\frac{1}{d}\sum_i z_i^2}$, we set the scale

$$s = \frac{k_b\,\sigma}{q_{\max}},$$

then apply

$$q = \operatorname{clip}\left(\left\lfloor \tfrac{z}{s} \right\rceil, q_{\min}, q_{\max}\right), \qquad \hat{z} = sq, \qquad \hat{x} = H^\top \hat{z},$$

where $\lfloor\cdot\rceil$ denotes round-to-nearest. The backward pass uses QuEST's trust-masked STE in the transform domain.

Note that CAGE itself is *quantizer-agnostic*. It augments the optimizer update via the instantaneous quantization error $e_t = x_t - Q(x_t)$ (see §3.3). We choose the QuEST quantizer here because it is a strong SOTA baseline for low-bit training. Other quantizers can be used without changing CAGE.

Pretraining experiments. We test our method by pre-training Llama-style transformers with parameter counts $N \in \{30\text{M}, 50\text{M}, 100\text{M}, 200\text{M}, 800\text{M}\}$ under QAT with weight/activation bit-widths $b \in \{2, 3, 4\}$, using QuEST-style pre-quantization transforms as our baseline pipeline [26]. Training uses C4 with a fixed budget of $D = 100 \times N$ tokens (TPP), where D is the number of tokens and N is the number of parameters in the model. Master weights are stored FP32, aupdates follow AdamW with decoupled weight decay[24] as in 1. Experiments are done on 8xH100 GPUs and common hyperparameters (LR schedule, warmup, clipping, etc.) are in Appendix B. Each pre-training average is done over three seeds and the mean and standard deviation are reported in Table 1. Models up to $200M$ are visualized in Figure 1, although the standard deviation is too small to be visible in Figure 1.

We compare CAGE against (i) QuEST with Hadamard outlier mitigation (our strong baseline) and (ii) a high-precision BF16 reference. Figure 1 shows validation loss vs. model size for $b \in 2, 3, 4$, with CAGE, QuEST baseline, and BF16 reference values.

Scaling law fitting. To compare methods beyond pointwise losses, we fit a separable data/model/precision-scaling law to the validation loss. Following references [14; 26; 16], we fit a law of the form:

$$\mathcal{L}(N, D, P) \;=\; \frac{A}{(N \cdot \operatorname{eff}(P))^{\alpha}} \;+\; \frac{B}{D^{\beta}} \;+\; E,$$

where N is parameter count, D is the seen token count, and P denotes the quantization bits. The factor $\operatorname{eff}(P)$ captures the *effective capacity* penalty due to quantization. $\operatorname{eff}(\text{FP}) \equiv 1$.

We jointly fit A, α, B, β, E *shared* across methods and learn a separate $\operatorname{eff}(P)$ per method/bit-width via nonlinear least squares on the grid of (N, D) we trained (Appendix B). We regularize with weak log-priors to keep $\alpha, \beta > 0$. Figure 2 plots $\operatorname{eff}(P)$. CAGE consistently increases $\operatorname{eff}(P)$ vs. QuEST, with the largest gains at $b \approx 3, 4$, indicating improved *effective* capacity for a given parameter budget.

QAT fine-tuning on MXFP4. We further evaluate CAGE under an MXFP4-style 4-bit floating format [9] on a larger model, Llama-3.2-3B [10]. We fine-tune on Tulu-SFT [19] with QAT (master

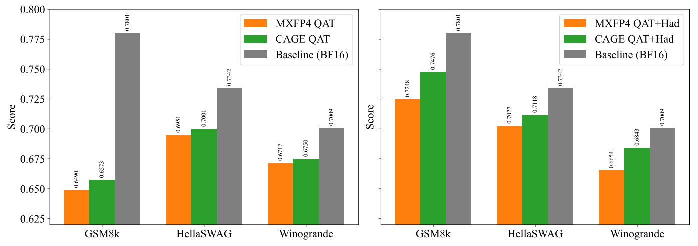

Figure 3: MXFP4 QAT on Llama-3.2-3B (Tulu-SFT): CAGE vs. MXFP4 baseline, with/without Hadamard. CAGE improves GSM8K/HellaSwag/WinoGrande, with larger gains when Hadamard is enabled.

FP32, forward W4A4 in MXFP4) with and without CAGE applied, and report zero-shot or few-shot scores on GSM8K (exact-match), HellaSwag (accuracy), and WinoGrande (accuracy). We test with and without the Hadamard pre-transform to isolate its interaction with MXFP4. Figure 3 summarizes results. Hyperparameters for the experiments can be found in Appendix B

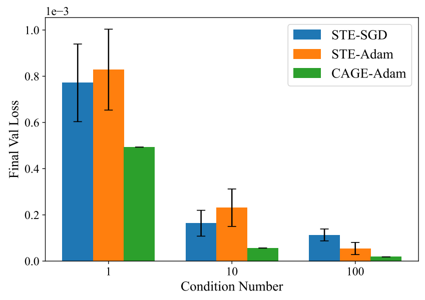

Figure 4: Quadratic with non-isotropic Hessian: final validation loss across various condition numbers. CAGE reduces stationary error.

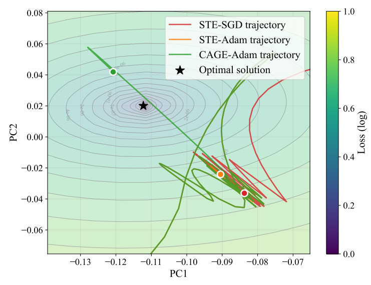

Figure 5: Representative trajectories (PC1/PC2) for the quadratic task; CAGE follows curvature and converges closest to the optimum.

Quadratic experiments. To isolate dynamics, we study a quadratic objective $f(x) = \frac{1}{2}x^\top Ax - b^\top x$ with $\kappa(A) \in \{1, 10, 100\}$ (non-isotropic curvature), comparing SGD, Adam, and CAGE-Adam under a 4-bit quantization of x using the same pipeline as above. We initialize $x_0 \sim \mathcal{N}(0, \sigma_0^2 I)$, run a fixed budget of T steps, and measure (i) final validation loss over 10 seeds and (ii) trajectories in the $(x^\top Ax)^{1/2}$ vs. iteration plane relative to the true minimizer $x^\star = A^{-1}b$. Figure 4 reports the distribution of final losses (mean$\pm$std across 10 seeds) Figure 5 shows representative trajectories and end-points over a reduced-dimension loss landscape around the optimum.

5 Conclusion and Discussion

We introduced CAGE (Curvature-Aware Gradient Estimation), a novel method for quantization-aware training (QAT) designed to mitigate the accuracy degradation inherent in low-bit quantization. CAGE addresses the limitations of the straight-through estimator (STE) by augmenting the gradient with a principled, curvature-aware correction term. This approach is rooted in a reformulation of QAT as a multi-objective optimization problem, balancing the minimization of the task loss with the satisfaction of quantization constraints.

The key theoretical contribution is the definition of Pareto-optimal solutions for quantized optimization. Moreover, CAGE offers strong ergodic convergence guarantees to a Pareto-optimal point in the smooth non-convex setting, providing theoretical grounding often lacking in STE heuristics. While the approach is optimizer-agnostic, we developed a highly efficient, decoupled implementation on top of AdamW that leverages existing optimizer statistics for stability and performance. Our formulation also generalizes the concurrent work of [18].

Empirical validation on both pre-training and post-training QAT demonstrates the effectiveness of CAGE through extensive experiments involving the pre-training of Llama-style models up to 800M parameters. CAGE consistently outperformed strong baselines that employ modern outlier-mitigation techniques. Specifically, scaling law analysis confirms that CAGE improves the effective capacity of models compared to existing QAT methods.

The success of CAGE indicates that accurately estimating the gradient by incorporating local curvature information is crucial for advancing QAT performance. By explicitly counteracting the impact of quantization error on the optimization dynamics, CAGE provides a robust and theoretically sound framework for training accurate low-bit models, helping to bridge the gap between quantized efficiency and full-precision performance. Future work may explore the application of CAGE in ultra-low-bit regimes and its integration with other compression techniques, such as sparsification and vector quantization.

REFERENCES

[1] Thalaiyasingam Ajanthan, Puneet K. Dokania, Richard Hartley, and Philip H. S. Torr. Proximal mean-field for neural network quantization. *arXiv preprint arXiv:1812.04353*, 2019.

[2] Thalaiyasingam Ajanthan, Kartik Gupta, Philip H. S. Torr, Richard Hartley, and Puneet K. Dokania. Mirror Descent View for Neural Network Quantization. *arXiv preprint arXiv:1910.08237*, 2021.

[3] Yu Bai, Yu-Xiang Wang, and Edo Liberty. Proxquant: Quantized neural networks via proximal operators. *arXiv preprint arXiv:1810.00861*, 2018.

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

[5] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. URL `https://arxiv.org/abs/1308.3432`.

[6] Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[7] Congliang Chen, Li Shen, Haozhi Huang, and Wei Liu. Quantized Adam with Error Feedback. *arXiv preprint arXiv:2004.14180*, 2021.

[8] Congliang Chen, Li Shen, Haozhi Huang, and Wei Liu. Quantized adam with error feedback, 2021. URL `https://arxiv.org/abs/2004.14180`.

[9] Bita Darvish Rouhani, Nitin Garegrat, Tom Savell, Ankit More, Kyung-Nam Han, Mathew Zhao, Ritchie amd Hall, Jasmine Klar, Eric Chung, Yuan Yu, Michael Schulte, Ralph Wittig, Ian Bratt, Nigel Stephens, Jelena Milanovic, John Brothers, Pradeep Dubey, Marius Cornea, Alexander Heinecke, Andres Rodriguez, Martin Langhammer, Summer Deng, Maxim Naumov, Paulius Micikevicius, Michael Siu, and Colin Verrilli. OCP Microscaling (MX) Specification. *Open Compute Project*, 2023.

[10] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[11] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019.

[12] Steven K. Esser, Jeffrey L. McKinstry, Dhruv Bablani, Raja Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *International Conference on Learning Representations (ICLR)*, 2020. URL `https://openreview.net/pdf?id=rkgO66VKDS`.

[13] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022. URL `https://arxiv.org/abs/2210.17323`.

[14] Elias Frantar, Utku Evci, Wonpyo Park, Neil Houlsby, and Dan Alistarh. Compression scaling laws: Unifying sparsity and quantization, 2025.

[15] Geoffrey Hinton. Neural networks for machine learning, lecture 9c. Coursera, 2012.

[16] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL `https://arxiv.org/abs/2203.15556`.

[17] Lu Hou, Ruiliang Zhang, and James T. Kwok. Analysis of Quantized Models. *ICLR*, 2019.

[18] Mujin Kwun, Depen Morwani, Chloe Huangyuan Su, Stephanie Gil, Nikhil Anand, and Sham Kakade. Lotion: Smoothing the optimization landscape for quantized training, 2025. URL `https://arxiv.org/abs/2510.08757`.

[19] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

[20] Huu Le, Rasmus Kjær Høier, Che-Tsung Lin, and Christopher Zach. AdaSTE: An Adaptive Straight-Through Estimator to Train Binary Neural Networks. *arXiv preprint arXiv:2112.02880*, 2021.

[21] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. Training Quantized Nets: A Deeper Understanding. *arXiv preprint arXiv:1706.02379*, 2017.

[22] Tao Lin, Sebastian U. Stich, Luis Barba, Daniil Dmitriev, and Martin Jaggi. Dynamic model pruning with feedback, 2020. URL `https://arxiv.org/abs/2006.07253`.

[23] Liyuan Liu, Chengyu Dong, Xiaodong Liu, Bin Yu, and Jianfeng Gao. Bridging discrete and backpropagation: Straight-through and beyond. *Advances in Neural Information Processing Systems*, 36:12291–12311, 2023.

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

[25] OpenAI. Introducing gpt-oss. `https://openai.com/index/introducing-gpt-oss/`, 2025. Accessed 2025-09-21.

[26] Andrei Panferov, Jiale Chen, Soroush Tabesh, Mahdi Nikdan, and Dan Alistarh. Quest: Stable training of llms with 1-bit weights and activations. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. URL `https://openreview.net/forum?id=I0Ux2nAN6u`.

[27] Andrei Panferov, Alexandra Volkova, Ionut-Vlad Modoranu, Vage Egiazarian, Mher Safaryan, and Dan Alistarh. Unified scaling laws for compressed representations, 2025. URL `https://arxiv.org/abs/2506.01863`.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[29] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[30] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL `https://arxiv.org/abs/2307.09288`.

[31] Xiao-Ming Wu, Dian Zheng, Zuhao Liu, and Wei-Shi Zheng. Estimator meets equilibrium perspective: A rectified straight through estimator for binary neural networks training, 2023. URL `https://arxiv.org/abs/2308.06689`.

[32] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations (ICLR)*, 2019.

A Convergence Analysis: Proof of Theorem 1

Let $F(x) = f(x) + \lambda\phi(x)$ be the overall regularized loss with smoothness constant $L = L + \lambda L_\phi$. Then, due to smoothness inequality, we have

$$\begin{aligned} F(x_{t+1}) &\leq F(x_t) + \langle \nabla F(x_t), x_{t+1} - x_t \rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2 \\ &= F(x_t) - \alpha\langle \nabla f(x_t) + \lambda(x_t - Q(x_t)), \widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\rangle \\ &\quad + \frac{L\alpha^2}{2}\|\widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\|^2. \end{aligned}$$

Applying conditional expectation $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid x_t]$, using unbiasedness of stochastic gradient and boundedness of variance, we get

$$\begin{aligned} \mathbb{E}_t[F(x_{t+1}) - F(x_t)] &\leq -\alpha\|\nabla f(x_t) + \lambda(x_t - Q(x_t))\|^2 + \frac{L\alpha^2}{2}\mathbb{E}_t\left[\|\widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\|^2\right] \\ &= -\alpha\|\nabla_{\lambda\mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\left(\|\nabla_{\lambda\mathrm{P}} f(Q(x_t))\|^2 + \sigma^2\right) \\ &= -\alpha\left(1 - \frac{L\alpha}{2}\right)\|\nabla_{\lambda\mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\sigma^2 \\ &\leq -\frac{\alpha}{2}\|\nabla_{\lambda\mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\sigma^2, \end{aligned}$$

where we enforced $\alpha \leq \frac{1}{L}$ step-size restriction. Applying full expectation and summing the obtained inequalities from $t = 0, \ldots, T-1$, we get

$$\begin{aligned} \mathbb{E}\|\nabla_{\lambda\mathrm{P}} f(Q(\hat{x}))\|^2 &\leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla_{\lambda\mathrm{P}} f(Q(x_t))\|^2 \leq \frac{2(F(x_0) - F(x_T))}{\alpha T} + \alpha L\sigma^2 \\ &= \frac{2(f(x_0) - f(x_T))}{\alpha T} + \frac{2\lambda(\phi(x_0) - \phi(x_T))}{\alpha T} + \alpha L\sigma^2 \\ &\leq \frac{1}{\sqrt{T}}\left(2(f(x_0) - f^*) + 2\lambda(\phi(x_0) - \phi(x_T)) + L\sigma^2\right)\max\left(1, \frac{L}{\sqrt{T}}\right) \end{aligned}$$

with $\alpha = \min(\frac{1}{L}, \frac{1}{\sqrt{T}})$. It remains to bound the term $\phi(x_0) - \phi(x_T)$ stemming from the quantization.

Due to the assumption 3 on quantization we can represent the scalar function ϕ as path-independent line integral:

$$\phi(x) = \int_{x_0}^{x}(I - Q)\cdot d\mathbf{r},$$

where I is the identity map, i.e. $I(x) = x$ for any $x \in \mathbb{R}^d$. Since $I - Q$ is a gradient field (i.e., $x - Q(x) = \nabla\phi(x)$), the line integral above does not depend on how the endpoints x_0 and x are connected. Therefore, we choose the direct path $r(t) = x_0 + (x - x_0)t$ for $t \in [0, 1]$ and simplify the integral into usual Riemannian integral as

$$\phi(x) = \int_{x_0}^{x}(I - Q)\cdot d\mathbf{r} = \int_0^1 \langle r(t) - Q(r(t)), r'(t)\rangle\, dt.$$

Using this relation, we can bound the term as follows

$$\begin{aligned} \phi(x_0) - \phi(x_T) &\leq \left|\int_0^1 \langle r(t) - Q(r(t)), r'(t)\rangle\, dt\right| \\ &\leq \int_0^1 \|r(t) - Q(r(t))\| \cdot \|r'(t)\|\, dt \leq \max_{x \in [x_0, x_T]}\|x - Q(x)\| \cdot \|x_0 - x_T\|. \end{aligned}$$

Table 2: Hyperparameters for Llama-style pretraining runs (per model size). Token budgets follow a 100 tokens-per-parameter (TPP) rule.

Model (N)	Layers	Heads	d_{model}	Base LR	Tokens (D)
30M	6	5	640	1.2×10^{-3}	3B
50M	7	6	768	1.2×10^{-3}	5B
100M	8	8	1024	6.0×10^{-4}	10B
200M	10	10	1280	3.0×10^{-4}	20B
430M	13	13	1664	1.5×10^{-4}	43B
800M	16	16	2048	7.5×10^{-5}	80B

B Hyperparameters and Reproducibility

Shared settings. Sequence length L=512; batch size 64 with gradient accumulation 8 (effective tokens/step fixed across sizes). Optimizer AdamW with β_1=0.9, β_2=0.95, ε=10^{-8}, weight decay 0.1, gradient clip 1.0. Cosine LR schedule with warmup 10% of total steps. FP32 master weights and optimizer states; bfloat16 compute. Hardware: 8× H100 (NCCL, `compile` enabled).

Quantization. QuEST row-wise Hadamard quantizer for weights *and* activations; bit-widths $b \in \{2, 3, 4\}$ (symmetric, MSE-optimal Gaussian clipping). Trust-masked STE in transform domain.

CAGE settings. Decoupled variant. silence ratio s=0.9. λ=2.0