# TEMPOQUANT: TIMESTEP-AWARE CALIBRATION FOR POST-TRAINING QUANTIZATION OF MASKED DIFFUSION LANGUAGE MODELS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Post-training quantization (PTQ) of masked diffusion language models (MDLMs) is challenging because their activation states evolve throughout the denoising trajectory. We find that temporal activation variation alone does not reliably indicate quantization difficulty; instead, target-bit quantization vulnerability is localized across denoising states, channels, and operations. Based on this observation, we propose TempoQuant, a trajectory-aware PTQ framework that directly measures channel-wise computation distortion under joint weight–activation quantization. TempoQuant aggregates distortion across denoising states into trajectory-level vulnerability and uses it to guide quantizer adaptation, allocating greater adaptation freedom to vulnerable channels. To account for accumulated quantization errors, Transformer blocks are progressively calibrated on quantized-prefix inputs. All trajectory-dependent information is consolidated into static quantization parameters, requiring no timestep- or state-dependent quantizers at inference. Experiments on LLaDA-8B-Instruct demonstrate strong performance under both W8A8 and W4A4 quantization, with particularly substantial gains in the challenging W4A4 setting. The code is available at `https://anonymous.4open.science/r/TempoQuant-1DC7/`

## 1 INTRODUCTION

Masked diffusion language models (MDLMs) (Nie et al., 2025; Ye et al.) have emerged as a competitive alternative to autoregressive generation, achieving strong performance on open-ended text generation and reasoning benchmarks. Unlike autoregressive models that generate tokens left-to-right, MDLMs iteratively refine a masked sequence through a learned denoising process, processing the entire context bidirectionally at each step. This iterative structure, however, introduces a significant deployment challenge: generating a single response requires repeated forward passes throughout the denoising trajectory. Post-training quantization (PTQ) therefore provides a practical approach to reducing the computational and memory costs of MDLM inference without retraining.

Existing PTQ methods have been developed primarily for autoregressive transformers. Canonical approaches such as SmoothQuant (Xiao et al., 2023), GPTQ (Frantar et al., 2022), and AWQ (Lin et al., 2024) achieve effective low-bit compression by calibrating quantization parameters from representative model inputs. MDLMs introduce an additional challenge because the model repeatedly encounters different activation states as the masking ratio and token states evolve throughout denoising. In particular, a dedicated MASK token occupies a large fraction of the input early in the trajectory, while nearly complete sequences are processed near the end. Consequently, a quantized MDLM must remain robust across the entire denoising trajectory rather than a single input regime.

Recent concurrent efforts have begun addressing PTQ specifically for MDLMs (Yan et al., 2026; Xu & Yang, 2025; Zhang et al., 2026), with particular attention to the temporal and token-state variation induced by iterative denoising. However, whether temporal activation variation itself provides a sufficient measure of quantization difficulty remains unclear. Our analysis on LLaDA-8B-Instruct shows that temporal variation is indeed informative of quantization sensitivity, but remains an incomplete proxy for actual target-bit vulnerability. Channels exhibiting similar temporal variation can



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

experience substantially different computation distortion after quantization, while directly measured target-bit distortion provides a stronger predictor of deployment-time vulnerability.

A closer examination further reveals that target-bit vulnerability is highly heterogeneous across channels, layers, and operations (Yuan et al., 2023; Yao et al., 2022; Li et al., 2021). Consistent with prior observations that quantization difficulty is often concentrated in a small subset of channels or weights (Xiao et al., 2023; Lin et al., 2024; Ashkboos et al., 2024a), we find that this heterogeneity extends across Transformer blocks and linear projections in MDLMs. This suggests that treating all channels identically can unnecessarily constrain robust channels while failing to provide sufficient adaptation freedom to vulnerable ones. Moreover, quantization of upstream blocks progressively changes the representations encountered by downstream blocks, creating a mismatch between conventional calibration on full-precision inputs and the inputs observed by the deployed quantized model (Wei et al., 2022).

These observations motivate TempoQuant, a trajectory-aware PTQ framework for MDLMs that characterizes quantization vulnerability using target-bit joint weight–activation distortion rather than activation variation alone. TempoQuant measures channel-wise distortion across denoising states and aggregates it into trajectory-level vulnerability, allowing greater quantizer adaptation for vulnerable channels while anchoring stable ones near initialization. Calibration operates in a fixed quantization-friendly rotated representation, and all trajectory-dependent information is used only during calibration, yielding static transformation and clipping parameters at inference.

TempoQuant further addresses upstream quantization-induced representation shifts through progressive quantized-prefix calibration. Each block is calibrated on inputs propagated through already-quantized preceding blocks, using the corresponding full-precision representation as the reconstruction target. This exposes downstream blocks to inference-time quantization shifts and enables compensation for accumulated distortion. Together, these components produce a single static quantized model tailored to MDLM denoising dynamics without state-dependent inference overhead.

We summarize our contributions as follows:

- We show that temporal activation variation, while informative, is insufficient to fully characterize target-bit quantization vulnerability in MDLMs. Directly measured target-bit computation distortion provides a more predictive characterization of deployment-time quantization sensitivity.
- We identify that target-bit vulnerability is highly heterogeneous across channels, layers, and operations, motivating trajectory-level vulnerability estimation and vulnerability-guided channel adaptation.
- We propose TempoQuant, which combines trajectory-level target-bit vulnerability estimation with vulnerability-guided quantizer adaptation and progressive quantized-prefix calibration to account for upstream quantization-induced representation shifts.
- Experiments on LLaDA-8B-Instruct demonstrate strong performance under both W8A8 and W4A4 quantization. TempoQuant achieves $63.82\%$ on ARC-Challenge under W8A8 and an average accuracy of $50.88\%$ under W4A4, outperforming the strongest W4A4 baseline by $5.75$ percentage points.

## 2 What Makes MDLMs Vulnerable to Quantization?

Unlike autoregressive LLMs, masked diffusion language models (MDLMs) repeatedly process partially masked sequences throughout denoising, resulting in activation states that evolve along the trajectory. To understand how this affects quantization, we analyze LLaDA-8B-Instruct from three perspectives: temporal activation variation, the distribution of target-bit vulnerability, and error propagation across quantized blocks. These analyses motivate the design of TempoQuant.

### 2.1 Temporal Variation Does Not Imply Quantization Vulnerability

Activation statistics vary substantially throughout the denoising trajectory (Fig. 1(a)). Moreover, quantization error measured at a single calibration state can substantially underestimate the error encountered across denoising states (Table 1), indicating that MDLM quantization should account

| Bits | Local Error | Mean Deploy. | Max Deploy. |
|---|---|---|---|
| W8A8 | $4.70\times10^{-5}$ | $5.16\times10^{-4}$ | $3.56\times10^{-3}$ |
| W4A4 | $1.14\times10^{-2}$ | $1.06\times10^{-1}$ | $7.97\times10^{-1}$ |

Table 1: Single-state calibration underestimates trajectory-wide quantization error. Local error is measured at $t = 0.5$ and deployment error across denoising states.

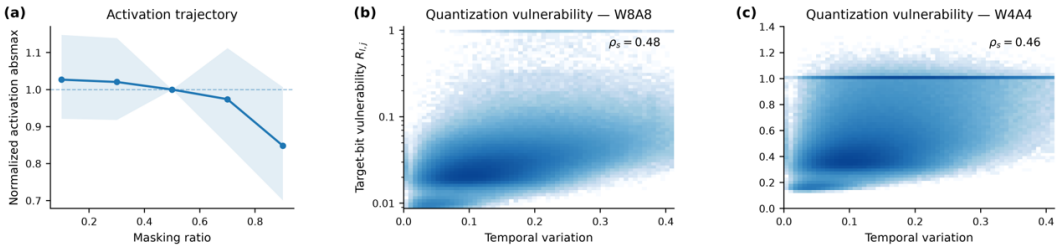



Figure 1: Temporal variation is informative but insufficient for characterizing quantization vulnerability. Channels with similar temporal variation can exhibit substantially different target-bit computation distortion.

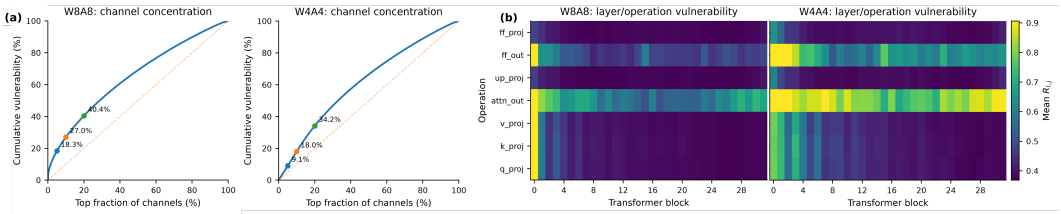



Figure 2: Quantization vulnerability is localized. Target-bit computation distortion is concentrated in a subset of denoising states, channels, and operations rather than uniformly distributed across the trajectory.

for the denoising trajectory. However, temporal variation alone does not fully characterize target-bit vulnerability. As shown in Fig. 1(b–c), temporal variation and quantization vulnerability are positively correlated, yet channels with similar variation can exhibit substantially different levels of target-bit computation distortion. Thus, temporal variation captures useful information about quantization sensitivity but remains an incomplete proxy for actual vulnerability.

**Observation 1.** *Temporal dynamics are informative of quantization sensitivity, but temporal variation alone is insufficient to characterize target-bit vulnerability.*

### 2.2 Quantization Vulnerability Is Structurally Localized

Target-bit quantization vulnerability is not uniformly distributed across the model. As shown in Fig. 2(a), a relatively small fraction of channels accounts for a disproportionate share of the cumulative vulnerability under both W8A8 and W4A4. This heterogeneity also extends across layers and operations: Fig. 2(b) shows distinct vulnerability patterns across Transformer blocks and linear projections, which further vary with the target precision. These results indicate that quantization sensitivity is structurally localized and should be measured under the target quantization setting rather than inferred from layer or operation identity alone.

**Observation 2.** *Target-bit quantization vulnerability is concentrated in a subset of channels and varies substantially across layers and operations.* This motivates TempoQuant to allocate channel-wise adaptation freedom according to directly measured vulnerability.

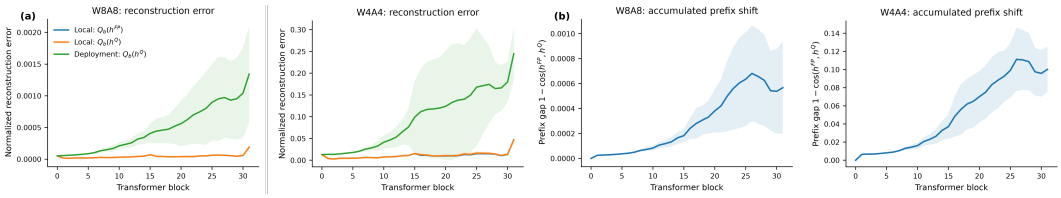



Figure 3: Effect of upstream quantization. Quantized-prefix inputs increasingly deviate from full-precision calibration inputs with depth, motivating progressive block-wise calibration on the inputs encountered by the quantized model.

### 2.3 Upstream Quantization Alters Downstream Calibration

Block-wise calibration is commonly performed by minimizing local reconstruction error at each block. However, a block deployed in the quantized model receives inputs that have already passed through quantized upstream blocks. Figure 3(a) shows that reconstruction error remains small under local calibration inputs, while the error under quantized-prefix inputs increases substantially with depth. This discrepancy is particularly pronounced under W4A4, indicating that locally well-reconstructed blocks can still encounter substantially different behavior at deployment.

Figure 3(b) further shows that the discrepancy between full-precision and quantized-prefix representations accumulates across Transformer blocks. The effect is mild under W8A8 but becomes substantially larger under W4A4, consistent with stronger error propagation at lower precision. Thus, calibration on full-precision inputs alone does not capture the input distribution encountered by deeper blocks after quantization.

**Observation 3.** *Upstream quantization progressively shifts the inputs encountered by downstream blocks, creating a gap between local calibration and quantized deployment.* This motivates TempoQuant to calibrate blocks sequentially on quantized-prefix inputs, so that each block is optimized under the representations it encounters after upstream quantization.

## 3 Method

We propose TempoQuant, a trajectory-aware PTQ framework for masked diffusion language models (MDLMs), as illustrated in Figure 4. TempoQuant measures channel-wise target-bit computation distortion across denoising states and aggregates it into a *trajectory-level quantization vulnerability*, which guides quantizer adaptation. Calibration is performed progressively on quantized-prefix inputs within a fixed rotated coordinate system, accounting for accumulated quantization errors while retaining state-independent quantization parameters at inference.

### 3.1 Quantization-Friendly Representation

As illustrated in Figure 4(a), aggressive low-bit quantization can be dominated by highly anisotropic weight and activation distributions, obscuring the trajectory-dependent vulnerability that TempoQuant aims to characterize. We therefore perform calibration in a fixed, quantization-friendly coordinate system obtained through an orthogonal rotation. For a linear layer with input $X_l$ and weight $W_l$, let $R_l$ be an orthogonal matrix satisfying $R_l R_l^\top = I$. We rewrite the linear transformation as

$$X_l W_l^\top = (X_l R_l)(W_l R_l)^\top. \tag{1}$$

Since the transformation is function-preserving in full precision, it changes the coordinate representation without altering the underlying linear operation.

In practice, we use a fixed randomized block-Hadamard transformation for computational efficiency. The same transformation is applied to the full-precision reference and the model being quantized, and remains fixed throughout calibration. Importantly, the rotation itself does not exploit timestep, masking ratio, or token-state information. It therefore serves only as the underlying low-bit quantization backbone; the MDLM-specific calibration is performed by TempoQuant as described below.

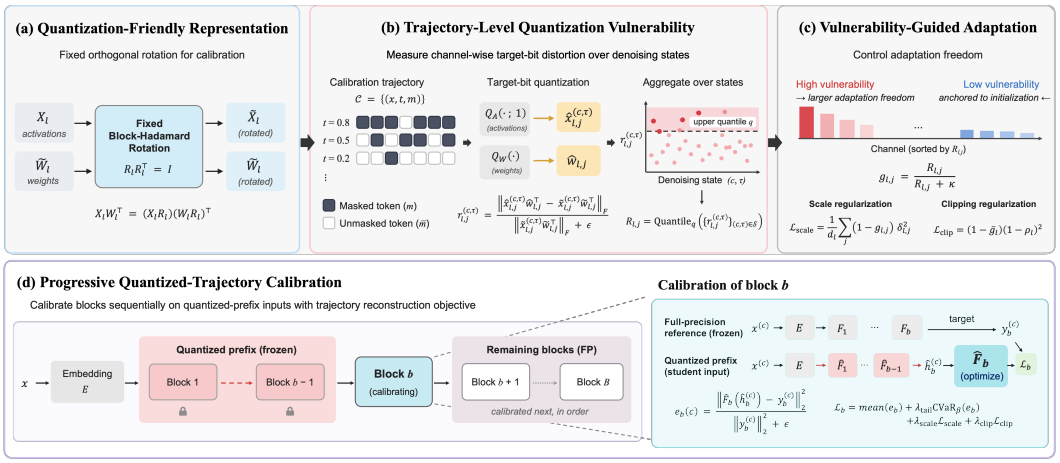



Figure 4: Overview of TempoQuant. Starting from a quantization-friendly rotated representation, TempoQuant measures channel-wise target-bit computation distortion across denoising states and aggregates it into trajectory-level vulnerability, which guides selective quantizer adaptation for vulnerable and stable channels. Transformer blocks are then progressively calibrated using quantized-prefix inputs to account for upstream quantization-induced representation shifts.

### 3.2 Trajectory-Level Quantization Vulnerability

**Calibration trajectory.** We construct a set of calibration cases $\mathcal{C} = \{c_1, \ldots, c_K\}$, where each case is defined as $c = (x, t, m)$, with $x$ denoting a calibration sequence, $t$ the masking ratio, and $m$ the masked token positions. For TempoQuant, we sample four source sequences from a 128-sequence calibration pool drawn from the WinoGrande `winogrande_xl` training split and truncate them to a maximum length of 96 tokens. Each sequence is independently masked once at each ratio in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ by uniformly sampling token positions without replacement, yielding five mask realizations per sequence and 20 calibration cases in total. These calibration states are constructed directly from masked text rather than obtained from an iterative generation rollout. To distinguish the two token states encountered during denoising, we partition the tokens of each case into $\tau \in \{m, \bar{m}\}$ and denote the resulting collection of trajectory states by $\mathcal{S}$.

**Channel-wise computation distortion.** Consider a linear layer $l$ with weight $W_l \in \mathbb{R}^{d_{\text{out}} \times d_l}$ and input $X_l \in \mathbb{R}^{n \times d_l}$, where both are expressed in the rotated coordinate system introduced above. Let $Q_W(\cdot)$ and $Q_A(\cdot; \rho_l)$ denote the target-bit weight and activation quantizers, respectively, where $\rho_l \in (0, 1]$ controls the activation clipping range.

We further introduce a channel-wise equivalent transformation

$$X_l W_l^\top = (X_l S_l^{-1})(W_l S_l)^\top = \widetilde{X}_l \widetilde{W}_l^\top, \tag{2}$$

where $S_l = \mathrm{diag}(s_{l,1}, \ldots, s_{l,d_l})$. We initialize $S_l$ using a SmoothQuant-style equivalent scale (Xiao et al., 2023) and optimize its residual during calibration.

For channel $j$, let $\widetilde{x}_{l,j}^{(c,\tau)}$ denote the $j$-th input channel restricted to token group $\tau$, and let $\widetilde{w}_{l,j}$ denote the $j$-th column of $\widetilde{W}_l$. Their target-bit quantized counterparts are

$$\widehat{x}_{l,j}^{(c,\tau)} = \left[ Q_A(\widetilde{X}_l^{(c)}; 1) \right]_{\tau,j}, \qquad \widehat{w}_{l,j} = \left[ Q_W(\widetilde{W}_l) \right]_{:,j}. \tag{3}$$

We use $\rho_l = 1$ when estimating vulnerability so that the measurement reflects the intrinsic target-bit distortion before optimizing the clipping range.

Since $\widetilde{x}_{l,j}^{(c,\tau)}\widetilde{w}_{l,j}^{\top}$ is the contribution of channel $j$ to the layer output, we define its state-wise vulnerability as

$$r_{l,j}^{(c,\tau)} = \frac{\left\|\widehat{x}_{l,j}^{(c,\tau)}\widehat{w}_{l,j}^{\top} - \widetilde{x}_{l,j}^{(c,\tau)}\widetilde{w}_{l,j}^{\top}\right\|_F}{\left\|\widetilde{x}_{l,j}^{(c,\tau)}\widetilde{w}_{l,j}^{\top}\right\|_F + \epsilon}. \tag{4}$$

Unlike activation magnitude or temporal distributional variation, $r_{l,j}^{(c,\tau)}$ directly measures the computation distortion induced by jointly quantizing the activation and weight operands at the target precision. We aggregate the state-wise distortion across the denoising trajectory using an upper quantile:

$$R_{l,j} = \mathrm{Quantile}_q\left(\left\{r_{l,j}^{(c,\tau)}\right\}_{(c,\tau)\in\mathcal{S}}\right). \tag{5}$$

As illustrated in Figure 4(b), $R_{l,j}$ summarizes state-wise target-bit distortion into a channel-level vulnerability estimate over the denoising trajectory. It becomes large when a channel exhibits severe computation distortion over a non-negligible portion of the trajectory, rather than when its activations merely vary across states. The vulnerability is estimated before optimizing each block and remains fixed during that block's calibration.

### 3.3 Vulnerability-Guided Quantizer Adaptation

Let $\mathcal{L}_b$ denote the set of linear layers in Transformer block $b$. For each layer, we parameterize the channel-wise equivalent scale as $s_{l,j} = s_{l,j}^{(0)}\exp(\delta_{l,j})$, where $s_{l,j}^{(0)}$ is the initial equivalent scale and $\delta_{l,j}$ is optimized during calibration. We additionally learn a state-independent activation clipping ratio $\rho_l$ for each layer while keeping the full-precision reference weights frozen.

Rather than directly rescaling channels according to their vulnerability, TempoQuant uses $R_{l,j}$ to determine their *adaptation freedom*. We map the measured vulnerability to

$$g_{l,j} = \frac{R_{l,j}}{R_{l,j} + \kappa}, \qquad g_{l,j} \in [0, 1), \tag{6}$$

where $\kappa > 0$ controls the transition between stable and vulnerable channels. The scale residual is regularized as

$$\mathcal{L}_{\mathrm{scale}} = \frac{1}{|\mathcal{L}_b|}\sum_{l\in\mathcal{L}_b}\frac{1}{d_l}\sum_{j=1}^{d_l}(1 - g_{l,j})\delta_{l,j}^2. \tag{7}$$

Consequently, low-vulnerability channels are anchored near their initial transformation, whereas vulnerable channels are allowed greater freedom to adapt.

Similarly, using the layer-average vulnerability $\bar{g}_l = d_l^{-1}\sum_j g_{l,j}$, we regularize the activation clipping ratio as

$$\mathcal{L}_{\mathrm{clip}} = \frac{1}{|\mathcal{L}_b|}\sum_{l\in\mathcal{L}_b}(1 - \bar{g}_l)(1 - \rho_l)^2. \tag{8}$$

This discourages unnecessary clipping in layers that are already robust to target-bit quantization. As illustrated in Figure 4(c), these objectives translate trajectory-level vulnerability into a continuous degree of adaptation freedom rather than hard channel selection, anchoring stable channels near their initialization while allowing vulnerable channels to adapt more freely.

### 3.4 Progressive Quantized-Trajectory Calibration

Block-wise calibration performed solely on full-precision activations ignores the distribution shift induced by upstream quantization. To address this mismatch, TempoQuant performs progressive quantized-prefix calibration, as illustrated in Figure 4(d), where each Transformer block is calibrated using inputs generated by the already-quantized preceding blocks.

Let $F_b$ and $\widehat{F}_b$ denote the full-precision and quantized versions of block $b$, respectively, and $E$ the embedding layer. For calibration case $c$, block $b$ receives the quantized-prefix input $\widehat{h}_b^{(c)} =$

$\widehat{F}_{b-1} \circ \cdots \circ \widehat{F}_1(E(x^{(c)}))$, while the corresponding full-precision prefix provides the reconstruction target $y_b^{(c)} = F_b \circ \cdots \circ F_1(E(x^{(c)}))$. We then minimize the normalized reconstruction error

$$e_b(c) = \frac{\|\widehat{F}_b(\widehat{h}_b^{(c)}) - y_b^{(c)}\|_2^2}{\|y_b^{(c)}\|_2^2 + \epsilon}. \tag{9}$$

This asymmetric reconstruction setup is deliberate: the student block is exposed to the distorted activations it will receive after deployment, while the target remains anchored to the corresponding full-precision representation.

For a mini-batch $\mathcal{B} \subset \mathcal{C}$, we optimize

$$\mathcal{L}_b = \underbrace{\frac{1}{|\mathcal{B}|}\sum_{c\in\mathcal{B}} e_b(c)}_{\mathcal{L}_{\text{mean}}} + \lambda_{\text{tail}} \underbrace{\text{CVaR}_\beta\left(\{e_b(c)\}_{c\in\mathcal{B}}\right)}_{\mathcal{L}_{\text{CVaR}}} + \lambda_{\text{scale}}\mathcal{L}_{\text{scale}} + \lambda_{\text{clip}}\mathcal{L}_{\text{clip}}. \tag{10}$$

Here, $\text{CVaR}_\beta$ averages the largest $\lceil \beta|\mathcal{B}| \rceil$ reconstruction errors for a tail fraction $\beta \in (0, 1]$. The mean term preserves reconstruction quality across the trajectory, whereas the CVaR term prevents calibration from sacrificing high-error denoising states.

During optimization, we periodically evaluate the reconstruction objective over the full calibration trajectory and retain the parameter state with the lowest calibration objective. This best-state restoration prevents later optimization steps from degrading the trajectory-level reconstruction obtained earlier in calibration.

Once block $b$ is calibrated, its quantization parameters are frozen and the resulting quantized block becomes part of the prefix used to calibrate block $b+1$. This procedure progressively exposes deeper blocks to the accumulated quantization distortion encountered at deployment.

### 3.5 Static Consolidation for Inference

After calibration, all trajectory-dependent statistics are discarded and the learned quantization parameters are fixed. For each layer, the transformed weight $W_l S_l$ is quantized once, while the inverse transformation $S_l^{-1}$ and optimized activation clipping ratio $\rho_l$ remain fixed for activation quantization; the orthogonal rotation is likewise fixed and shared across all denoising states. Consequently, neither $S_l$, $\rho_l$, nor the rotation depends on the masking ratio, denoising step, or masked/unmasked token state. TempoQuant therefore exploits trajectory information only during offline calibration and requires no timestep-conditioned parameter selection or mask-dependent quantization branches at inference.

## 4 Experiments

### 4.1 Experimental Settings

**Models and benchmarks.** We evaluate on LLaDA-8B-Instruct (Nie et al., 2025), using the BF16 checkpoint as the full-precision (FP) reference. Following prior MDLM quantization work (Xu & Yang, 2025; Yan et al., 2026), we report zero-shot benchmark scores on TruthfulQA-MC2 (Lin et al., 2022), ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), PIQA (Bisk et al., 2020), MMLU (Hendrycks et al., 2020), and C-EVAL (Huang et al., 2023). Each candidate answer is masked and scored by the conditional likelihood of the mask predictor (Nie et al., 2025), and all methods share the same prompts and evaluation code.

**Baselines.** We compare against general-purpose PTQ methods developed for autoregressive LLMs, including vanilla RTN, AWQ (Lin et al., 2024), and QuaRot (Ashkboos et al., 2024b). We further include DLLMQuant (Xu & Yang, 2025) and STaR-Quant (Yan et al., 2026), two recent PTQ methods specifically tailored to MDLMs. For DLLMQuant, we report its AWQ- and QuaRot-based variants (DLLMQuant$^+$ and DLLMQuant$^{++}$), and QuaRot uses GPTQ for weight quantization as in its official implementation. Method-specific hyperparameters follow the original papers and official implementations.

| Method | Truth. | ARC | Hella. | Wino. | PIQA | MMLU | C-EVAL | Avg. |
|---|---|---|---|---|---|---|---|---|
| **W8A8** | | | | | | | | |
| FP | 41.98 | 63.14 | 37.79 | 61.56 | 60.34 | 44.18 | 64.04 | 53.29 |
| RTN | 40.64 | 59.90 | 37.29 | 59.35 | 60.88 | 42.33 | 58.77 | 51.31 |
| AWQ | 41.86 | 57.17 | 36.90 | 58.48 | 60.61 | 41.40 | 56.02 | 50.35 |
| DLLMQuant+ | 42.23 | 61.09 | 37.71 | 59.12 | **61.43** | 42.83 | 59.96 | 52.05 |
| QuaRot | **42.35** | 62.71 | **37.96** | 61.25 | 60.50 | 43.97 | 63.52 | 53.18 |
| DLLMQuant++ | 41.74 | 62.71 | **37.96** | **61.64** | 60.94 | 44.08 | 63.89 | 53.28 |
| STaR-Quant | 40.27 | 55.80 | 35.87 | 58.72 | 57.62 | 40.96 | 54.53 | 49.11 |
| **TempoQuant** | 41.49 | **63.82** | 37.92 | 61.56 | 60.45 | **44.30** | **64.41** | **53.42** |
| **W4A4** | | | | | | | | |
| RTN | 48.35 | 23.12 | 25.19 | 50.67 | 51.47 | 24.34 | 26.00 | 35.59 |
| AWQ | **54.10** | 23.89 | 25.55 | 50.67 | 49.13 | 24.92 | 22.73 | 35.86 |
| DLLMQuant+ | 49.33 | 25.60 | 25.37 | 49.33 | 52.07 | 24.73 | 24.89 | 35.90 |
| QuaRot | 48.10 | 25.34 | 28.38 | 49.64 | 54.08 | 27.54 | 23.33 | 36.63 |
| DLLMQuant++ | 40.88 | 55.46 | 32.27 | 51.93 | 56.58 | 41.77 | 37.00 | 45.13 |
| STaR-Quant | 36.84 | 43.60 | 29.46 | 51.62 | 54.19 | 34.85 | 34.70 | 40.75 |
| **TempoQuant** | 41.74 | **58.79** | **35.03** | **59.10** | **61.86** | **42.59** | **57.06** | **50.88** |

Table 2: Zero-shot downstream performance of LLaDA-8B-Instruct under W8A8 and W4A4 quantization. The best quantized result for each task is highlighted in bold. FP denotes the full-precision model. Avg. is the unweighted mean of the seven reported benchmark scores, including the TruthfulQA-MC2 score.

**Quantization and calibration.** We evaluate W8A8 and W4A4, where W$x$A$y$ denotes $x$-bit weights and $y$-bit activations, using symmetric per-channel weight quantization and symmetric per-token activation quantization. Simulated quantization is applied to the weights and input activations of all linear layers, including the language-model head. The embedding layer, attention matrix multiplications, and all remaining operations use BF16. The quantized values are dequantized before computation, and the underlying matrix multiplications are executed using BF16 kernels. The calibration-based baselines use all 128 sequences in the WinoGrande calibration pool, whereas TempoQuant uses four source sequences to construct 20 trajectory-aware calibration cases across five masking ratios.

**Implementation details.** TempoQuant initializes the equivalent scales with SmoothQuant ($\alpha = 0.5$) (Xiao et al., 2023) and uses $q = 0.95$ and $\kappa = 0.15$ for vulnerability estimation. Each block is optimized for 80 steps with AdamW (batch size 4; learning rates $10^{-3}$ for scales and $2 \times 10^{-3}$ for clipping ratios), with $\beta = 0.2$, $\lambda_{\text{tail}} = 0.5$, and $\lambda_{\text{scale}} = \lambda_{\text{clip}} = 10^{-2}$. The same hyperparameters are used for all bit-widths. All experiments are run on a single NVIDIA A100 80GB GPU.

### 4.2 Main Results

Table 2 reports zero-shot benchmark scores for LLaDA-8B-Instruct under W8A8 and W4A4 quantization. Under W8A8, TempoQuant obtains an average score of 53.42%, compared with 53.28% for DLLMQuant++, and achieves the highest quantized scores on ARC-Challenge, MMLU, and C-EVAL. The average improvement is modest at 0.14 percentage points.

Under W4A4, TempoQuant achieves an average benchmark score of 50.88%, compared with 45.13% for DLLMQuant++, an improvement of 5.75 percentage points. TempoQuant obtains the best quantized result on six of the seven benchmarks. These results demonstrate the effectiveness of the complete TempoQuant framework under W4A4 quantization.

Notably, TempoQuant outperforms STaR-Quant under both settings despite using state-independent quantization parameters at inference. While STaR-Quant conditions its activation transformation on token state, TempoQuant uses state-dependent information only during calibration to estimate vulnerability and guide quantizer adaptation. The resulting transformation and clipping parameters remain fixed at inference, showing that MDLM trajectory information can be effectively exploited without state-dependent quantization during deployment.

| Vulnerability Indicator | W8A8 $\rho_s \uparrow$ | W4A4 $\rho_s \uparrow$ |
|---|---|---|
| Activation magnitude | $-0.5258$ | $-0.5196$ |
| Temporal variation | $0.6335$ | $0.6152$ |
| Weight magnitude | $-0.0519$ | $-0.0511$ |
| **TempoQuant** $R_{l,j}$ | $\mathbf{0.7774}$ | $\mathbf{0.7506}$ |

Table 3: Predictiveness of quantization vulnerability. Spearman correlation ($\rho_s$) between calibration-time indicators and deployment-time quantization distortion on held-out denoising trajectories.

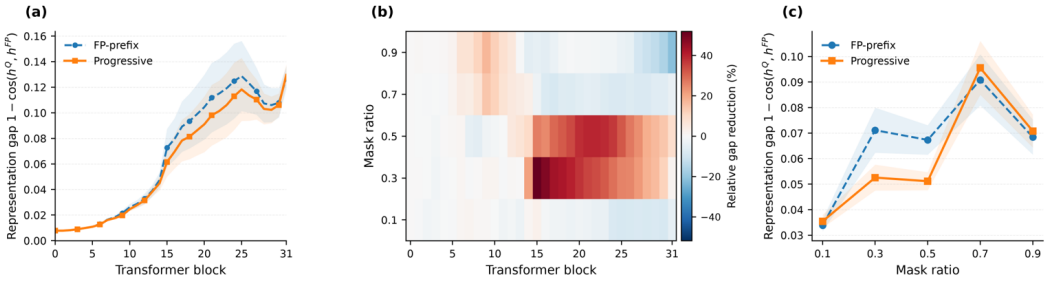



Figure 5: Progressive quantized-prefix calibration mitigates representation drift. (a) Representation gap across Transformer depth, averaged over denoising states. (b) Relative gap reduction across block–state pairs, with positive values indicating improvement. (c) Representation gap across denoising states, averaged over Transformer blocks.

### 4.3 Analysis of Quantization Vulnerability and Calibration

Table 3 compares calibration-time vulnerability indicators with quantization distortion measured on held-out denoising trajectories. Temporal variation is positively correlated with deployment-time distortion, confirming that temporal dynamics contain useful information about quantization sensitivity. Nevertheless, TempoQuant's $R_{l,j}$ achieves the strongest correlation under both W8A8 ($\rho_s = 0.7774$) and W4A4 ($\rho_s = 0.7506$), consistently exceeding temporal variation. Activation and weight magnitudes show substantially weaker associations. These results indicate that temporal variation is informative but incomplete, while target-bit computation distortion provides a more predictive measure of deployment-time vulnerability.

Figure 5 analyzes the effect of progressive quantized-prefix calibration across network depth and denoising states. When averaged over states, progressive calibration reduces the representation gap particularly in middle-to-late blocks, where upstream quantization errors have accumulated (Fig. 5(a)). The block–state analysis further reveals that this benefit is not uniform across the trajectory, but is concentrated at intermediate mask ratios and deeper blocks (Fig. 5(b)). After marginalizing over depth, the representation gap is reduced by $26.1\%$ and $24.0\%$ at mask ratios $0.3$ and $0.5$, respectively, while the two strategies remain comparable near the trajectory boundaries (Fig. 5(c)). These results suggest that quantized-prefix calibration is particularly effective in regions where trajectory-dependent representation drift is pronounced.

## 5 Conclusion

We presented TempoQuant, a trajectory-aware PTQ framework for masked diffusion language models. Motivated by the finding that temporal activation variation alone is insufficient to characterize quantization vulnerability, TempoQuant directly estimates trajectory-level target-bit distortion to guide quantizer adaptation and progressively calibrates blocks on quantized-prefix inputs. Experiments on LLaDA-8B-Instruct demonstrate effective performance preservation under both W8A8 and W4A4, highlighting the importance of accounting for trajectory-level quantization behavior in MDLM compression.

### AI USE STATEMENT

In this work, we used generative AI tools for none of the tasks with required disclosure. We have not used generative AI tools for research ideation, core methodology design, data collection, or experimental analysis, and code generation or synthetic data tasks are not applicable to this work. Additionally, we used generative AI tools for grammar correction and language polishing. We have reviewed all AI-assisted work. Specifically, all drafts were initially authored by the researchers, and AI-suggested grammatical modifications were selectively adopted only after human verification to maintain technical clarity and accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. Quik: Towards end-to-end 4-bit inference on generative large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3355–3371, 2024a.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024b.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, April 2020. doi: 10.1609/aaai.v34i05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 62991–63010. Curran Associates, Inc., 2023. doi: 10.52202/075280-2749. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/c6ec1844bec96d6d32ae95ae694e23d8-Paper-Datasets_and_Benchmarks.pdf.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 3214–3252, 2022.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pp.

50608–50646. Curran Associates, Inc., 2025. doi: 10.52202/085713-1689. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/48b383b24230e0e6e649d9c98dae4d8c-Paper-Conference.pdf.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*, 2022.

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

Chen Xu and Dawei Yang. Dllmquant: Quantizing diffusion-based large language models. *arXiv preprint arXiv:2508.14090*, 2025.

Xin Yan, Aqiang Wang, Zhenglin Wan, Xingrui Yu, and Ivor Tsang. Star-quant: State-time consistent post-training quantization for diffusion large language models. *arXiv preprint arXiv:2606.04945*, 2026.

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

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025. *URL https://arxiv. org/abs/2508.15487*.

Zhihang Yuan, Lin Niu, Jiawei Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. Rptq: Reorder-based post-training quantization for large language models. *arXiv preprint arXiv:2304.01089*, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Tianao Zhang, Zhiteng Li, Xianglong Yan, Haotong Qin, Yong Guo, and Yulun Zhang. Quant-dllm: Post-training extreme low-bit quantization for diffusion large language models. In *International Conference on Learning Representations*, volume 2026, pp. 79492–79503, 2026.