UaRA: Uncertainty-aware Rank Allocation for Parameter-Efficient Fine-Tuning of Diffusion Large Language Models

Anonymous authors
Paper under double-blind review

Abstract

Diffusion large language models (dLLMs) have recently attracted increasing attention for their iterative denoising paradigm and potential for flexible parallel generation. Unlike autoregressive generation, diffusion decoding operates over continuously changing noise states, leading to varying adaptation demands throughout the generation process. However, existing parameter-efficient fine-tuning (PEFT) methods such as LoRA are primarily designed for autoregressive models and do not account for the dynamic adaptation demands of diffusion decoding. In particular, standard LoRA adopts a fixed rank, allocating the same adaptation capacity to all tokens and noise conditions despite their varying prediction difficulty. Existing adaptive-rank methods such as AdaLoRA are also not directly suited to dLLMs, because their gradient-based importance estimates can be distorted by noise-dependent loss scaling rather than reflecting intrinsic parameter importance. To address this issue, we propose UaRA: Uncertainty-aware Rank Allocation for PEFT of dLLMs, which dynamically allocates low-rank capacity according to the evolving adaptation demand across noise levels. UaRA first calibrates gradient-weight sensitivity using the known loss-scaling coefficient, removing noise-induced scale bias from parameter-importance estimation. It then models the importance of each rank component as a function of the noise level using second-order Legendre bases fitted by online recursive least squares, enabling the active-rank distribution to adapt continuously throughout diffusion training. Extensive experiments demonstrate that UaRA consistently outperforms strong PEFT baselines, validating the effectiveness of uncertainty-aware and noise-conditioned rank allocation for dLLMs.

1 Introduction

Large language models (LLMs) have achieved remarkable progress across a wide range of language understanding and generation tasks (OpenAI, 2022; Touvron et al., 2023; Grattafiori et al., 2024; DeepSeek-AI et al., 2024; Qwen et al., 2025). Most existing LLMs follow the autoregressive paradigm, generating tokens sequentially from left to right. Recently, diffusion large language models (dLLMs) (Nie et al., 2025; Ye et al., 2025; Zhu et al., 2025; Bie et al., 2025) have emerged as a promising alternative by formulating text generation as an iterative denoising process. Instead of predicting the next token under a fixed causal order, dLLMs progressively recover masked tokens from partially observed contexts, enabling flexible and partially parallel generation. This generation paradigm, however, also introduces a distinct adaptation problem. During denoising, the model continuously moves across noise states with different amounts of available context and prediction uncertainty, suggesting that the adaptation capacity required by a dLLM should vary throughout the generation process (Wang et al., 2026).

Existing parameter-efficient fine-tuning (PEFT) methods do not fully account for such noise-dependent adaptation demands. Standard LoRA (Hu et al., 2022) uses a fixed-rank update throughout fine-tuning, assigning the same adaptation capacity across diffusion states. Recent dLLM-specific methods such as NaRA (Wang et al., 2026) condition low-rank updates on the noise level, demonstrating the importance of adapting PEFT to diffusion dynamics, but still retain a fixed rank across noise states. However, the adaptation demand of dLLMs is not uniform across diffusion

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.

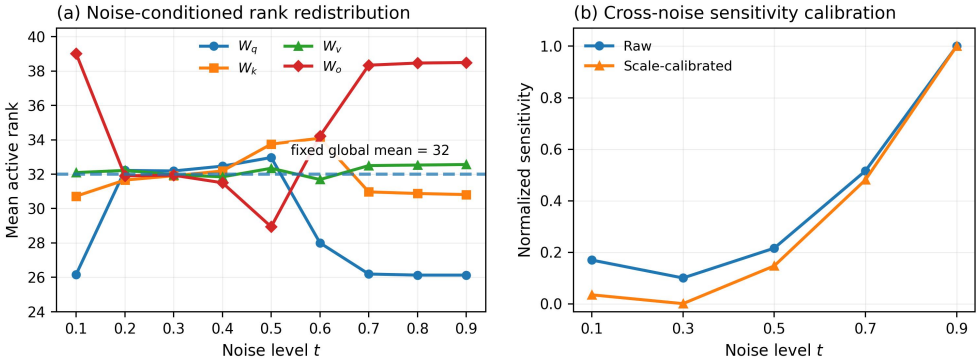

Figure 1: Rank allocation and sensitivity vary across diffusion states. (a) Even under a fixed global rank budget, different projection modules receive different active ranks at different noise levels. (b) Scale calibration reduces the noise-dependent bias in raw sensitivity and produces a more comparable importance signal across diffusion states. Together, these observations support noise-conditioned rank allocation for dLLMs.

states. Different noise states induce distinct corrupted inputs and denoising contexts, which can change the relative importance of low-rank components and the capacity required by different modules. As illustrated in Figure 1(a), the preferred rank allocation varies substantially across noise levels even under a fixed global rank budget. These observations motivate *noise-conditioned rank allocation*, where the effective rank adapts to the current diffusion state rather than remaining globally fixed.

A natural solution is to apply adaptive-rank PEFT methods such as AdaLoRA (Zhang et al., 2023) and IGU-LoRA (Cui et al., 2026), which allocate rank budgets based on component importance. However, these methods are not directly suited to diffusion training. Their gradient-based sensitivity scores approximate the loss change induced by removing a component (Molchanov et al., 2019), but diffusion objectives scale the loss with the sampled noise level t, causing sensitivity magnitudes to vary systematically across noise states. As shown in Figure 1(b), scale calibration substantially reduces this noise-dependent bias, yielding a more comparable sensitivity signal across diffusion states. Without such calibration, importance estimates can be dominated by noise-dependent scaling rather than intrinsic component importance, leading to poorly calibrated rank allocation. Moreover, existing methods do not explicitly model how component importance and adaptation requirements vary throughout denoising, making their adaptive rank allocation insensitive to the noise-dependent requirements of dLLMs.

To address these limitations, we propose **UaRA**: **U**ncertainty-**a**ware **R**ank **A**llocation for Parameter-Efficient Fine-Tuning of Diffusion Large Language Models, a noise-conditioned adaptive-rank framework that models component importance across diffusion states. UaRA first calibrates gradient-weight sensitivity using the known loss-scaling coefficient to remove noise-induced scale bias. It then models each rank component's importance as a continuous function of t using second-order Legendre basis functions, whose coefficients are updated online with recursive least squares (RLS) and a forgetting factor. At each forward pass, the predicted importance at the current noise level serves as the rank-selection score, allowing the most relevant components to be activated under a fixed budget. This enables the effective rank distribution to adapt continuously to the evolving prediction uncertainty throughout denoising.

This design turns rank from a fixed hyperparameter into a noise-conditioned adaptation resource. Unlike fixed-rank LoRA and existing adaptive-rank methods, UaRA explicitly models component importance as a function of the diffusion state, allowing adaptation capacity to vary across noise levels under a fixed budget. We evaluate UaRA on LLaDA across a broad set of commonsense and mathematical reasoning benchmarks. UaRA achieves the best overall performance among the compared PEFT methods, while further analyses reveal clear noise-dependent rank allocation patterns.

Our contributions are summarized as follows:

1. **We identify a structural mismatch between existing adaptive-rank PEFT methods and diffusion language models.** We show that noise-dependent loss scaling can distort gradient-based

sensitivity estimates across diffusion states, while noise-agnostic rank allocation fails to capture the heterogeneous adaptation demands induced by iterative denoising.

2. **We propose UaRA, an uncertainty-aware and noise-conditioned adaptive-rank framework tailored to dLLMs.** UaRA calibrates gradient-weight sensitivity to remove noise-induced scale bias and learns a continuous importance function over noise levels for each rank component using Legendre basis functions and online RLS, enabling the active-rank distribution to adapt dynamically to the current diffusion state.

3. **We demonstrate the effectiveness of UaRA across diverse reasoning benchmarks.** Experiments on LLaDA over commonsense and mathematical reasoning tasks show that UaRA achieves the best overall performance among the compared PEFT methods. Further analyses reveal clear noise-dependent rank allocation patterns, validating the benefit of adapting low-rank capacity across diffusion states.

2 Related Work

2.1 Diffusion Large Language Models

Diffusion large language models (dLLMs) provide an alternative to the dominant autoregressive generation paradigm (Ye et al., 2025; Bie et al., 2025; Zhu et al., 2025; Bie et al., 2026). Early studies explored continuous diffusion for text generation and discrete diffusion over token spaces, including Diffusion-LM (Li et al., 2022), D3PM (Austin et al., 2021), and DiffusionBERT (He et al., 2023). Subsequent methods such as SEDD (Lou et al., 2024) and MDLM (Sahoo et al., 2024) substantially improved discrete diffusion language modeling. More recently, LLaDA (Nie et al., 2025) demonstrated that masked diffusion can scale to billion-parameter language models with competitive language understanding and generation capabilities. Despite rapid progress in model scaling and decoding, parameter-efficient adaptation of dLLMs remains relatively underexplored. Most PEFT methods were originally developed without explicitly modeling the evolving noise conditions of diffusion generation. A recent exception is NaRA (Wang et al., 2026), which conditions low-rank updates on the diffusion noise level and allows the adaptation parameters to vary along the denoising trajectory. However, NaRA does not explicitly model how the required *rank capacity* changes across noise states, leaving the rank dimension fixed while adapting the low-rank update itself. In contrast, our UaRA focuses on noise-conditioned rank allocation, dynamically adjusting the effective low-rank capacity according to the adaptation requirements induced by different diffusion states.

2.2 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) adapts large pretrained models by optimizing only a small number of additional parameters while keeping most backbone parameters frozen. LoRA (Hu et al., 2022) has become one of the most widely used PEFT methods by representing weight updates with two trainable low-rank matrices. A series of variants subsequently improve LoRA from different perspectives. Delta-LoRA (Zi et al., 2024) propagates the change of the low-rank product back to pretrained weights without explicitly optimizing their gradients; LoRA+ (Hayou et al., 2024) assigns different learning rates to the two low-rank matrices to improve optimization; and DoRA (Liu et al., 2024) decomposes pretrained weights into magnitude and direction and applies low-rank adaptation to directional updates. Another line of work investigates adaptive rank allocation. AdaLoRA (Zhang et al., 2023) parameterizes updates using an SVD-like decomposition and dynamically allocates the parameter budget by pruning less important singular components. More recently, IGU-LoRA (Cui et al., 2026) improves adaptive rank allocation with integrated-gradient-based importance estimation and uncertainty-aware scoring. These adaptive-rank methods improve rank allocation but are not designed for the noise-dependent dynamics of dLLMs. UaRA addresses this gap by modeling rank-component importance across noise levels, enabling noise-conditioned rank allocation throughout denoising.

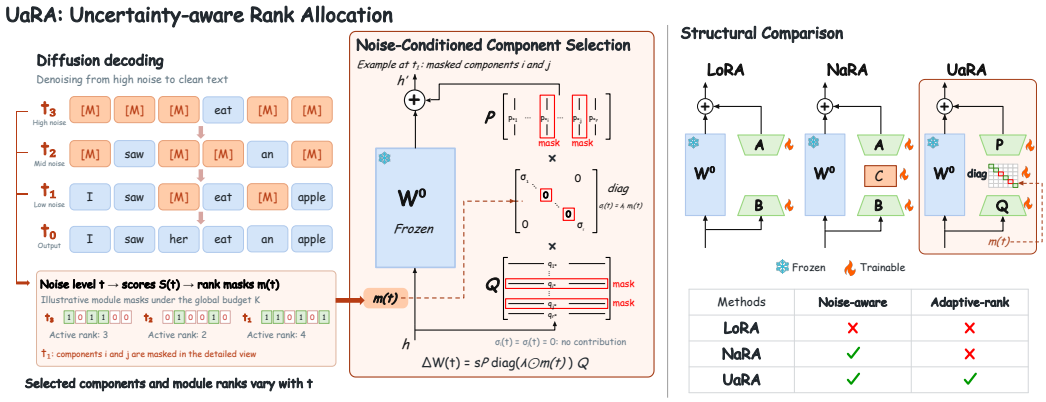

Figure 2: Overview of UaRA. UaRA performs noise-conditioned component selection during diffusion decoding, dynamically adjusting active components and module ranks according to the current noise level. Compared with LoRA and NaRA, UaRA is both noise-aware and adaptive-rank.

3 METHODOLOGY

We propose **UaRA**, an uncertainty-aware rank allocation method for parameter-efficient fine-tuning of dLLMs. As shown in Figure 2, UaRA calibrates gradient-based sensitivity samples, models their dependence on the noise level, and combines the predicted importance with recent prediction residuals to select rank components under a fixed global budget. Algorithm 1 in the appendix summarizes training.

3.1 SCALE-CALIBRATED COMPONENT SENSITIVITY

In masked diffusion, $t \in (0, 1]$ specifies the probability of masking a target token. Let $n \in \mathbb{N}$ index forward–backward microsteps and Ψ denote the trainable adapter parameters. At microstep n, the sampled training example and its corruption at noise level t_n define the backpropagated loss

$$L_n(\Psi) = c_n \, \ell_n(\Psi; t_n), \qquad c_n > 0, \tag{1}$$

where ℓ_n is the masked-token cross-entropy under the chosen base reduction, and c_n collects known multiplicative factors independent of Ψ. For example, LLaDA uses a $1/t$ weight on the masked-token loss (Nie et al., 2025). We use the coefficient supplied by the actual loss computation, including any accumulation scaling. Equation (1) applies to a single example or a microbatch with a common noise level and scaling factor; Appendix A.2 treats heterogeneous batches.

Gradient-weight sensitivity approximates the effect of deleting a parameter (Zhang et al., 2023). Holding the sample, noise, and active mask fixed, the deletion of a scalar parameter w satisfies

$$\ell_n(w \leftarrow 0) - \ell_n = -w \, \partial_w \ell_n + \mathcal{E}_w, \qquad |\mathcal{E}_w| \le \tfrac{1}{2} H_w w^2, \tag{2}$$

where H_w bounds the absolute second derivative along the deletion path. This finite-step error depends on curvature along the entire path, whereas the raw sensitivity also contains the factor c_n: $|w \partial_w L_n| = c_n |w \partial_w \ell_n|$. Consequently, pooling sensitivity samples across noise levels weights their contributions by c_n in addition to their normalized sensitivity. Appendices A.1–A.2 derive the error bound and a cross-noise ranking reversal. These motivate separate treatment of sensitivity scaling and noise dependence. If an additional regularizer $\eta \mathcal{R}$ enters the scoring gradient, division by c_n leaves $\nabla \ell_n + (\eta / c_n) \nabla \mathcal{R}$. UaRA therefore computes sensitivity samples from the denoising objective in Eq. (1) and optimizes the adapter without an orthogonality penalty. Appendix A.3 analyzes this gradient contribution and distinguishes it from the geometry of orthogonal bases.

Rank-one parameterization. We freeze the pretrained weights and adapt M linear projections through additive low-rank updates (Hu et al., 2022). For module ℓ, we directly learn $P_\ell \in \mathbb{R}^{d_\ell^{\text{out}} \times R}$, $Q_\ell \in \mathbb{R}^{R \times d_\ell^{\text{in}}}$, and $\boldsymbol{\lambda}_\ell \in \mathbb{R}^R$, following the three-factor parameterization of AdaLoRA (Zhang et al.,

2023). A noise-conditioned mask $\boldsymbol{m}_\ell(t) \in \{0,1\}^R$ selects components:

$$W_\ell(t) = W_\ell^0 + \Delta W_\ell(t),$$
$$\Delta W_\ell(t) = \frac{\alpha}{\bar{r}} P_\ell \operatorname{diag}(\boldsymbol{\lambda}_\ell \odot \boldsymbol{m}_\ell(t)) Q_\ell = \frac{\alpha}{\bar{r}} \sum_{j=1}^{R} m_{\ell j}(t) \lambda_{\ell j} \boldsymbol{p}_{\ell j} \boldsymbol{q}_{\ell j}^\top. \tag{3}$$

Here $\boldsymbol{p}_{\ell j}$ and $\boldsymbol{q}_{\ell j}^\top$ are a column of P_ℓ and a row of Q_ℓ, respectively; α is the adapter scale and $\bar{r} = K/M$ for a fixed active-component budget $K > 0$. Each triplet is one selectable rank-one component. Its coefficient $\lambda_{\ell j}$ is trainable. The active rank is $r_\ell(t) = \sum_j m_{\ell j}(t)$ and $\operatorname{rank}(\Delta W_\ell(t)) \le r_\ell(t)$.

Calibrated sensitivity targets. Let $g_n(v) = \partial_v L_n$ be the gradient contribution captured before accumulation across microsteps and before gradient clipping. We aggregate coordinate sensitivities within each triplet:

$$I_{\ell j,n}^{\text{raw}} = |\lambda_{\ell j} g_n(\lambda_{\ell j})| + \frac{\|\boldsymbol{p}_{\ell j} \odot g_n(\boldsymbol{p}_{\ell j})\|_1}{d_\ell^{\text{out}}} + \frac{\|\boldsymbol{q}_{\ell j} \odot g_n(\boldsymbol{q}_{\ell j})\|_1}{d_\ell^{\text{in}}},$$
$$y_{\ell j,n} = I_{\ell j,n}^{\text{raw}} / c_n. \tag{4}$$

Positive homogeneity makes $y_{\ell j,n}$ exactly the same aggregation computed from $\nabla \ell_n$ under Eq. (1). The pair $(t_n, y_{\ell j,n})$ is a regression sample: t_n is the input noise level and $y_{\ell j,n}$ the calibrated sensitivity target. We accept it only for active components with finite targets and valid scaling coefficients. The weighted training loss and its optimizer gradients remain those in Eq. (1).

3.2 Noise-Conditioned Importance Estimation

For each component, we fit importance as a function of noise using

$$\boldsymbol{\phi}(t) = [1,\ 2t-1,\ 6t^2 - 6t + 1]^\top. \tag{5}$$

These shifted Legendre polynomials are orthogonal under the uniform measure on $[0,1]$. Each component maintains coefficients $\boldsymbol{a} \in \mathbb{R}^3$ and inverse information matrix $C \in \mathbb{R}^{3\times 3}$. For each valid sample (t_n, y_n), recursive least squares with forgetting factor $\zeta \in (0,1]$ updates the regressor as follows, with component indices omitted for clarity (Goel et al., 2020):

$$\begin{aligned} \boldsymbol{k}_n &= \frac{C^- \boldsymbol{\phi}_n}{\zeta + \boldsymbol{\phi}_n^\top C^- \boldsymbol{\phi}_n}, \qquad \boldsymbol{\phi}_n = \boldsymbol{\phi}(t_n), \\ \boldsymbol{a}^+ &= \boldsymbol{a}^- + \boldsymbol{k}_n (y_n - \boldsymbol{\phi}_n^\top \boldsymbol{a}^-), \\ C^+ &= \zeta^{-1}(C^- - \boldsymbol{k}_n \boldsymbol{\phi}_n^\top C^-). \end{aligned} \tag{6}$$

Superscripts $-$ and $+$ denote states before and after the current update. We initialize $\boldsymbol{a} = 0$ and $C = \delta I_3$ with $\delta > 0$. The recursion tracks recent sensitivity samples as the adapter evolves; its least-squares objective is derived in Appendix A.4.

We share information across projections of the same type $g(\ell) \in \{q,k,v,o\}$. Let $\mathcal{V}_{\ell j}$ contain the microstep indices of valid samples for component (ℓ, j), so its sample count after microstep n is $N_{\ell j} = |\mathcal{V}_{\ell j} \cap \{1,\dots,n\}|$. Let $\overline{\boldsymbol{a}}_{g(\ell)}$ be the sample-count-weighted mean of local coefficients of that type, set to zero before any valid sample. The coefficients used for prediction are

$$\widetilde{\boldsymbol{a}}_{\ell j} = \frac{N_{\ell j} \boldsymbol{a}_{\ell j} + \kappa \overline{\boldsymbol{a}}_{g(\ell)}}{N_{\ell j} + \kappa}, \qquad \kappa > 0. \tag{7}$$

Inspired by sensitivity–deviation weighting (Zhang et al., 2023), we measure uncertainty through the residual of the noise-conditioned prediction. Before updating the regressor at microstep n, we compute

$$e_{\ell j,n} = y_{\ell j,n} - \boldsymbol{\phi}(t_n)^\top \widetilde{\boldsymbol{a}}_{\ell j}^-, \qquad u_{\ell j}^+ = \beta u_{\ell j}^- + (1-\beta)|e_{\ell j,n}|, \tag{8}$$

where $u_{\ell j}$ starts at zero and $\beta \in [0,1)$. The resulting score is

$$\widehat{I}_{\ell j}(t) = [\boldsymbol{\phi}(t)^\top \widetilde{\boldsymbol{a}}_{\ell j}]_+, \qquad S_{\ell j}(t) = \widehat{I}_{\ell j}(t) u_{\ell j}, \tag{9}$$

with $[z]_+ = \max(z, 0)$. The residual EMA is an operational uncertainty proxy for unexplained sensitivity variation. Multiplication gives greater priority to components with both high predicted sensitivity and substantial recent residuals. Appendix A.5 separates predictable noise dependence from residual variation.

3.3 BUDGETED RANK ALLOCATION

For $Mr_{\min} \le K \le MR$, the feasible masks satisfy $\sum_{\ell,j} m_{\ell j} = K$ and $\sum_j m_{\ell j} \ge r_{\min}$ for every module. At each forward pass, UaRA maximizes the sum of selected scores:

$$\boldsymbol{m}^\star(t) \in \arg\max_{\boldsymbol{m}\in\mathcal{F}} \sum_{\ell=1}^{M}\sum_{j=1}^{R} S_{\ell j}(t) m_{\ell j}, \tag{10}$$

where $\mathcal{F}$ is the feasible set above. We first reserve the $r_{\min}$ highest-scoring components in each module, forming $\mathcal{R}(t)$, then select the best $K - Mr_{\min}$ remaining components globally. With candidate set $\mathcal{C}$, the active set is

$$\mathcal{A}(t) = \mathcal{R}(t) \cup \mathrm{Top}_{K-Mr_{\min}}(\mathcal{C} \setminus \mathcal{R}(t); S(t)). \tag{11}$$

Ties are resolved by a fixed ordering. This procedure solves Eq. (10) exactly (Appendix A.6). It changes both module-wise active ranks and component identities while maintaining the total active rank K. Scores can be evaluated at any noise level; the selected set switches when their ordering changes.

Training and inference. We initialize the factors with $\Delta W_\ell = 0$ and activate all R candidates per module during warmup. After at least $T_{\min}$ optimizer steps, warmup ends when noise-coverage and score-stability criteria are met, or when its maximum duration is reached. Subsequent forward passes use Eq. (11). The mask is fixed throughout each forward–backward pair, and sensitivity samples update the regression states for later passes. With G accumulation microsteps per optimizer step s, the h-th microstep has index $n = (s-1)G + h$. Adapter parameters minimize $\mathcal{L}_{\mathrm{train}}(\Psi) = \mathbb{E}[L_n(\Psi)]$, where the expectation is over training examples, noise levels, and masking. At inference, adapter parameters, regression coefficients, and residual statistics are frozen. Each denoising step selects components using its current t and the same budget rule. The checkpoint retains all R candidates per module, so components can be reactivated across noise levels.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models and datasets. We use LLaDA-8B-Base and LLaDA-8B-Instruct (Nie et al., 2025), representing pretrained and instruction-tuned backbones, respectively. For commonsense reasoning, we fine-tune on Commonsense170k (Hu et al., 2023) and evaluate on eight benchmarks: ARC-Challenge and ARC-Easy (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), and WinoGrande (Sakaguchi et al., 2020). For mathematical reasoning, we fine-tune on Math14k (Hu et al., 2023) and evaluate on GSM8K (Cobbe et al., 2021), AddSub (Hosseini et al., 2014), AQuA (Ling et al., 2017), and MultiArith (Roy & Roth, 2015). Details of the baselines, evaluation protocol, and implementation settings are provided in Appendix B.1.

Rank configurations and parameter budgets. We evaluate LoRA, HiRA, and NaRA at configured ranks $r \in \{32, 48\}$, with reported trainable-parameter ratios of 0.420% and 0.624%, respectively. UaRA maintains a maximum candidate rank of 48 per adapted module, imposes a minimum active rank of 24, and allocates ranks under a global budget corresponding to an average active rank of 32. Its trainable-parameter ratio is 0.624%, reflecting the full candidate capacity. Thus, the rank-32 baselines provide a comparison at the same nominal rank budget as our average active rank, whereas the rank-48 baselines provide a comparison at the same reported trainable-parameter ratio.

4.2 MAIN RESULTS

Commonsense reasoning. Table 1 reports results on the eight commonsense benchmarks. On LLaDA-8B-Base, UaRA achieves the highest average accuracy among the evaluated methods at 87.51%, exceeding LoRA with rank 32 by 0.33 points and the strongest baseline, NaRA with rank 48, by 0.26 points. The improvements are not uniform across tasks: UaRA attains the best scores on ARC-Easy, BoolQ, OpenBookQA, SIQA, and WinoGrande, while LoRA with rank 32 remains

Table 1: Commonsense reasoning accuracy (%). Avg. is the unweighted mean over eight benchmarks. Best and second-best results within each backbone are bolded and underlined, respectively. Rank denotes the configured adapter rank for the baselines; † denotes our average active rank of 32, with maximum rank 48 and minimum rank 24. Param. reports the percentage of trainable parameters relative to the backbone, including the full candidate rank capacity.

Method	Rank	Param. (%)	ARC-C	ARC-E	BoolQ	Hella	OBQA	PIQA	SIQA	Wino	Avg.
LLaDA-8B-Base											
Zero-shot	–	–	82.34	93.10	64.62	80.98	78.00	76.88	71.03	52.72	74.96
LoRA	32	0.420	88.31	96.04	69.30	**95.40**	90.00	**88.08**	81.88	88.40	87.18
	48	0.624	89.16	96.30	69.17	95.23	89.00	87.43	82.34	88.71	87.17
NaRA	32	0.420	88.57	96.30	69.11	95.12	89.80	86.94	83.11	88.19	87.14
	48	0.624	89.16	95.62	69.69	95.23	90.00	87.38	82.96	87.92	87.25
HiRA	32	0.420	84.81	94.23	66.61	90.16	83.60	83.95	75.84	77.82	82.13
	48	0.624	**94.53**	85.92	66.42	90.59	84.00	84.22	77.07	78.61	82.67
AdaLoRA	32†	0.624	85.24	94.74	65.99	90.96	85.00	84.44	78.71	81.14	83.28
UaRA (Ours)	32†	0.624	89.59	**96.38**	**69.76**	94.65	**90.60**	87.16	**83.16**	**88.79**	**87.51**
LLaDA-8B-Instruct											
Zero-shot	–	–	86.35	95.33	63.88	79.91	84.00	82.26	76.77	75.53	80.50
LoRA	32	0.420	87.80	95.54	68.41	93.74	**89.80**	86.56	81.88	86.11	86.23
	48	0.624	**88.31**	95.88	67.98	94.53	88.20	87.76	81.68	87.53	86.48
NaRA	32	0.420	87.20	95.79	69.66	94.06	88.60	87.60	81.73	86.74	86.42
	48	0.624	87.71	96.17	69.63	94.65	88.60	87.87	**82.40**	88.63	86.96
HiRA	32	0.420	86.52	95.58	67.71	87.53	84.80	84.33	78.30	78.30	82.88
	48	0.624	86.52	95.50	67.95	88.09	85.40	84.06	78.76	78.61	83.11
AdaLoRA	32†	0.624	86.24	95.10	67.62	91.85	87.63	86.51	80.28	84.92	85.02
UaRA (Ours)	32†	0.624	88.05	**96.30**	**70.09**	**95.04**	88.60	**88.25**	82.24	**89.11**	**87.21**

Table 2: Mathematical reasoning accuracy (%). Avg. is the unweighted mean over four benchmarks. Best and second-best results within each backbone are bolded and underlined, respectively. Rank, Param., and † follow the definitions in Table 1.

Method	Rank	Param. (%)	GSM8K	AddSub	AQuA	MultiArith	Avg.
LLaDA-8B-Base							
Zero-shot	–	–	67.12	32.91	35.43	83.67	54.78
LoRA	32	0.420	69.22	**84.05**	51.57	94.50	74.84
	48	0.624	68.23	79.75	51.18	96.00	73.79
NaRA	32	0.420	69.52	81.77	53.15	94.83	74.82
	48	0.624	68.84	80.76	**54.72**	93.33	74.41
HiRA	32	0.420	41.24	45.82	43.70	56.83	46.90
	48	0.624	55.80	78.99	50.39	81.50	66.67
AdaLoRA	32†	0.624	49.89	67.85	49.30	65.50	58.21
UaRA(ours)	32†	0.624	**75.58**	83.54	49.61	**98.33**	**76.77**
LLaDA-8B-Instruct							
Zero-shot	–	–	70.16	83.29	57.87	94.67	76.50
LoRA	32	0.420	**72.55**	81.52	62.20	89.50	76.44
	48	0.624	72.10	83.54	**63.78**	93.17	78.15
NaRA	32	0.420	71.34	78.48	58.27	90.83	74.73
	48	0.624	70.81	77.97	61.42	88.33	74.63
HiRA	32	0.420	70.74	85.57	52.76	93.67	75.69
	48	0.624	71.72	**86.33**	60.63	95.33	78.50
AdaLoRA	32†	0.624	68.16	80.23	55.69	89.73	73.45
UaRA(ours)	32†	0.624	72.40	83.54	61.42	**96.83**	**78.55**

stronger on HellaSwag and PIQA. On LLaDA-8B-Instruct, UaRA achieves an average accuracy of 87.21%, improving over LoRA with rank 32 by 0.98 points and NaRA with rank 48 by 0.25 points. It obtains the highest accuracy on five of the eight benchmarks, including BoolQ (70.09%), HellaSwag (95.04%), and WinoGrande (89.11%). These results show modest improvements in average commonsense accuracy on both backbone variants.

Mathematical reasoning. Table 2 presents results on the four mathematical reasoning benchmarks. On LLaDA-8B-Base, UaRA with $r_{\min} = 24$ reaches the highest average accuracy of 76.77%, surpassing LoRA with rank 32 by 1.93 points and NaRA with rank 48 by 2.36 points. It leads on GSM8K (75.58%) and MultiArith (98.33%), improving over LoRA with rank 32 by 6.36 and 3.83 points, respectively. AddSub accuracy is 83.54%, the second-best result, while AQuA accuracy is 49.61%. On LLaDA-8B-Instruct, UaRA achieves an average accuracy of 78.55%, improving over LoRA with ranks 32 and 48 by 2.11 and 0.40 points, respectively. Its average is close to the strongest baseline, HiRA with rank 48 (78.50%), with a numerical difference of 0.05 points. UaRA attains the best MultiArith accuracy of 96.83%, but does not achieve the highest score on the other three benchmarks. The benefit therefore depends on both the backbone and the reasoning task.

4.3 Ablation Study and Analysis

Effect of Residual Weighting. To isolate the contribution of residual weighting, we replace $S_{\ell j}(t) = \widehat{I}_{\ell j}(t)u_{\ell j}$ in Eq. (9) with $S_{\ell j}(t) = \widehat{I}_{\ell j}(t)$, while continuing to update the residual EMA. Both variants use LLaDA-8B-Instruct with identical training and evaluation settings and a post-warmup active-rank budget of $K = 4096$ (32 per module on average).

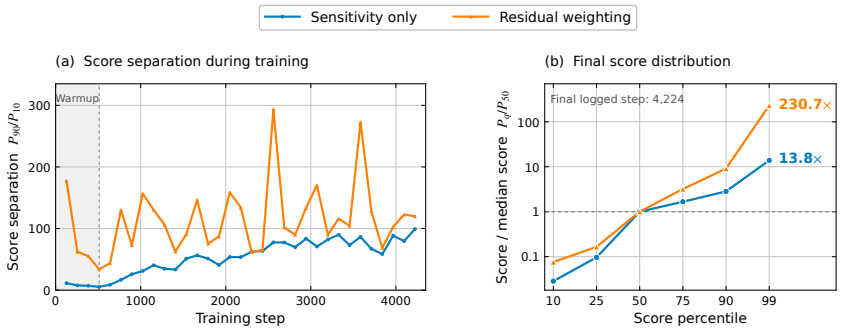

Figure 3: Effect of residual weighting on MultiArith with LLaDA-8B-Instruct. (a) P_{90}/P_{10} across training checkpoints. (b) Final score quantiles normalized by the median on a logarithmic scale.

Figure 3(a) tracks the ratio between the 90th and 10th score percentiles across the 6,144 candidate components. The median of this ratio over post-warmup checkpoints rises from 60.4 without residual weighting to 105.4 with it. The final score distribution shows a particularly strong change in the upper tail: after normalizing each variant by its own median score, P_{90} increases from 2.83 to 9.08, and P_{99} from 13.83 to 230.69 (Figure 3(b)). Thus, residual weighting reshapes the relative score distribution, amplifying the high-score tail. Under the same rank budget, mean MultiArith accuracy over three runs improves from 89.00% to 92.26% (**+3.26 percentage points**), supporting residuals as an additional priority signal for rank selection.

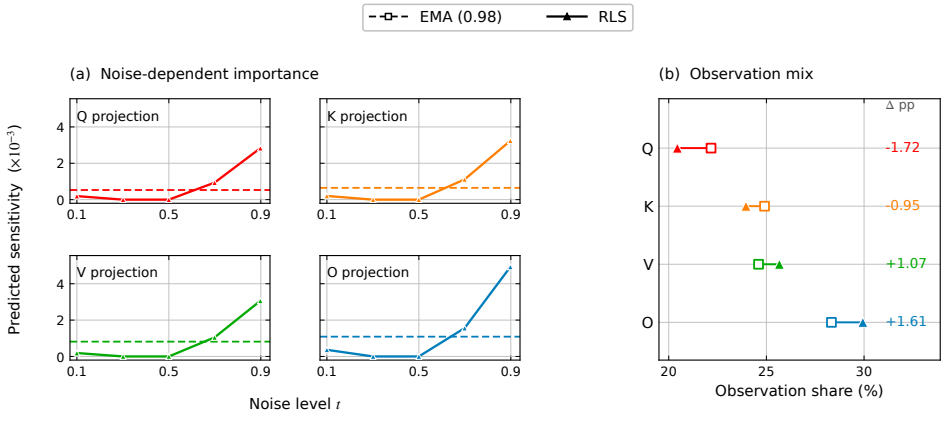

Figure 4: RLS versus EMA importance estimation. (a) Module-wise sensitivity. (b) Post-warmup observation shares, with labels indicating RLS–EMA differences.

Effect of different importance estimation RLS assigns markedly different predicted sensitivities across noise levels (Figure 4(a)). Its nonnegative predictions are zero at the logged points near $t = 0.3$ and $t = 0.5$, but reach 3.75–5.34 times the EMA estimates near $t = 0.9$ across the four projection types. EMA instead supplies a single estimate across noise levels. The estimator choice also changes the distribution of component observations: over the common post-warmup interval, V/O projections account for 55.60% of logged observations with RLS, compared with 52.93% with EMA (Figure 4(b)). Table 3 shows improvements on all four benchmarks, raising average accuracy from 72.25% to 76.77% (**+4.52 percentage points**), with the largest gains on GSM8K ($+8.72$) and AddSub ($+5.57$). These results support noise-conditioned importance estimation beyond temporal smoothing alone. More details in Appendix C.5

Table 3: Performance comparison between RLS and EMA importance estimation on math tasks. Both variants are fine-tuned on LLaDA-8B-Base.

Estimator	GSM8K	AddSub	AQuA	MultiArith	Avg.
EMA (0.98)	66.87	77.97	48.82	95.33	72.25
RLS	**75.59**	**83.54**	**49.61**	**98.33**	**76.77**

Table 4: Performance comparison with and without scale calibration on math tasks. Both variants are fine-tuned on LLaDA-8B-Base.

Observation	GSM8K	AddSub	AQuA	MultiArith	Avg.
Raw	70.66	82.53	49.21	97.16	74.89
Scale-calibrated	**75.59**	**83.54**	**49.61**	**98.33**	**76.77**

Effect of Importance Scale Calibration. We compare the calibrated observation $y_{\ell j,n} = I^{\text{raw}}_{\ell j,n}/c_n$ with directly using $I^{\text{raw}}_{\ell j,n}$ in the importance estimator. Both variants retain the same valid-observation filtering rules and leave the training loss and optimizer gradients unchanged. As Eq. (4) shows, calibration removes the known loss multiplier from the sensitivity observation supplied to the estimator. Table 4 shows the corresponding accuracy. The calibrated variant achieves higher accuracy on all four benchmarks, with average accuracy increasing from 74.89% to 76.77% (**+1.88 percentage points**). The largest gain is on GSM8K ($+4.93$ points).

Analysis of Noise-Conditioned Rank Allocation. Figure 5 shows that the active-rank allocation varies substantially across both noise levels and network depth, indicating that the adaptation demand is strongly noise- and layer-dependent. At moderate noise levels ($t = 0.2$–0.4), the allocated ranks are relatively uniform across layers. As the noise level increases, the allocation becomes more structured, with higher ranks concentrated in specific middle and late layers. In particular, for $t = 0.7$–0.9, Layers 17–24 consistently receive large ranks, while the final layers are assigned much smaller ranks. These results suggest that a fixed-rank design cannot fully capture the dynamic adaptation requirements of diffusion language models, motivating noise-aware rank allocation.

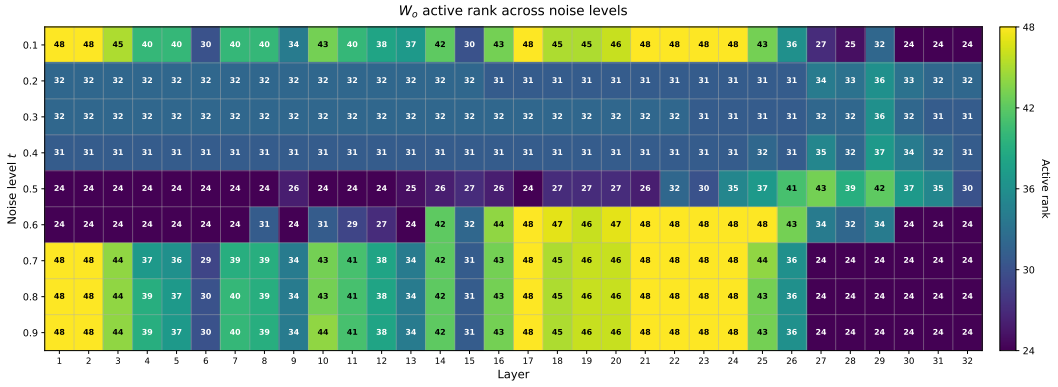

Figure 5: Noise-conditioned rank allocation on LLaDA-8B-Base. Rows denote noise levels and columns denote Transformer layers; colors and values indicate the active rank of W_o.

5 Conclusion

In this work, we introduced UaRA, an uncertainty-aware adaptive-rank framework for parameter-efficient fine-tuning of diffusion large language models. UaRA models rank allocation as a diffusion-state-dependent problem, enabling low-rank capacity to adapt across noise levels under a fixed budget. Experiments demonstrate consistent improvements over strong baselines and reveal clear noise-dependent rank allocation patterns, highlighting the importance of noise-conditioned parameter-efficient adaptation for dLLMs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing, draft portions of the manuscript, mathematical derivations related to recursive least squares (RLS) and the analysis and presentation of experimental results. We have not used generative AI tools for the research idea, methodological design, organization of the paper, and experimental procedures were developed and determined by the authors. All AI-assisted content was reviewed and verified by the authors. AI-assisted mathematical derivations were manually checked by one author against the relevant RLS literature and independently verified for mathematical correctness. AI-assisted code used for visualization and implementation was manually inspected and tested by one author, and identified implementation issues were corrected before the reported experiments were finalized. Analyses suggested by generative AI were checked against the actual experimental results before being incorporated into the manuscript. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, Jianfeng Tan, Lanning Wei, Ji-Rong Wen, Yipeng Xing, Xiaolu Zhang, Junbo Zhao, Da Zheng, Jun Zhou, Junlin Zhou, Zhanchao Zhou, Liwang Zhu, and Yihong Zhuang. Llada2.0: Scaling up diffusion language models to 100b, 2025. URL https://arxiv.org/abs/2512.15745.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, Zhuocheng Gong, Yanmei Gu, Jian Guan, Kaiyuan Guan, Hongliang He, Zenan Huang, Juyong Jiang, Zhonghui Jiang, Zhenzhong Lan, Chengxi Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Yuan Lu, Yuxin Ma, Xingyu Mou, Zhenxuan Pan, Kaida Qiu, Yuji Ren, Jianfeng Tan, Yiding Tian, Zian Wang, Lanning Wei, Tao Wu, Yipeng Xing, Wentao Ye, Liangyu Zha, Tianze Zhang, Xiaolu Zhang, Junbo Zhao, Da Zheng, Hao Zhong, Wanli Zhong, Jun Zhou, Junlin Zhou, Liwang Zhu, Muzhi Zhu, and Yihong Zhuang. Llada2.1: Speeding up text diffusion via token editing, 2026. URL https://arxiv.org/abs/2602.08676.

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, 2020. doi: 10.1609/aaai.v34i05.6239. URL https://ojs.aaai.org/index.php/AAAI/article/view/6239.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *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. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Xuan Cui, HuiYue Li, Run Zeng, Yunfei Zhao, Jinrui Qian, Wei Duan, Bo Liu, and Zhanpeng Zhou. Igu-lora: Adaptive rank allocation via integrated gradients and uncertainty-aware scoring. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 38129–38154, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/407106f4b56040b2e8dcad75a6e461e5-Paper-Conference.pdf.

DeepSeek-AI, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, et al. Deepseek llm: Scaling open-source language models with longtermism, 2024. URL https://arxiv.org/abs/2401.02954.

Ankit Goel, Adam L. Bruce, and Dennis S. Bernstein. Recursive least squares with variable-direction forgetting: Compensating for the loss of persistency [lecture notes]. *IEEE Control Systems Magazine*, 40(4):80–102, 2020. doi: 10.1109/MCS.2020.2990516. URL https://arxiv.org/abs/2003.03523.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17783–17806. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/hayou24a.html.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4521–4534, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.248. URL https://aclanthology.org/2023.acl-long.248/.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 523–533. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1058. URL https://aclanthology.org/D14-1058/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5254–5276. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.319. URL https://aclanthology.org/2023.emnlp-main.319/.

Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TwJrTz9cRS.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM improves controllable text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=3s9IrEsjLyk.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 158–167. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1015. URL https://aclanthology.org/P17-1015/.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bn.html.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

OpenAI. Introducing ChatGPT, November 2022. URL https://openai.com/index/chatgpt/.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, et al. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1743–1752. Association for Computational Linguistics, 2015. doi: 10.18653/v1/D15-1202. URL https://aclanthology.org/D15-1202/.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *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. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, et al. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Shuaidi Wang, Zhan Zhuang, HUANG Ruping, and Yu Zhang. NaRA: Noise-aware loRA for parameter-efficient fine-tuning of diffusion LLMs. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=s8UGIRF38a.

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.

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. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Fengqi Zhu, Zebin You, Yipeng Xing, Zenan Huang, Lin Liu, Yihong Zhuang, Guoshan Lu, Kangyu Wang, Xudong Wang, Lanning Wei, Hongrui Guo, Jiaqi Hu, Wentao Ye, Tieyuan Chen, Chenchen Li, Chengfu Tang, Haibo Feng, Jun Hu, Jun Zhou, Xiaolu Zhang, Zhenzhong Lan, Junbo Zhao, Da Zheng, Chongxuan Li, Jianguo Li, and Ji-Rong Wen. Llada-moe: A sparse moe diffusion language model, 2025. URL https://arxiv.org/abs/2509.24389.

Bojia Zi, Xianbiao Qi, Lingzhi Wang, Jianan Wang, Kam-Fai Wong, and Lei Zhang. Delta-loRA: Fine-tuning high-rank parameters with the delta of low-rank matrices, 2024. URL https://openreview.net/forum?id=FAO4VS9QRV.

A MATHEMATICAL ANALYSIS OF UARA

The analysis separates finite-deletion approximation error, multiplicative loss scaling, and regularization gradients. All differentiation holds the corruption sample, noise level, and discrete selection mask fixed.

A.1 FINITE DELETION AND FIRST-ORDER SENSITIVITY

Let θ collect the trainable parameters, e_i be coordinate i's basis vector, and $D_i = \ell(\theta - \theta_i e_i) - \ell(\theta)$ be the signed loss change after deleting that coordinate.

Proposition A.1 (Exact remainder for coordinate deletion)**.** If ℓ is twice continuously differentiable on a neighborhood of the deletion segment, then

$$D_i = -\theta_i \partial_i \ell(\theta) + \theta_i^2 \int_0^1 (1-s)\, \partial_{ii}^2 \ell(\theta - s\theta_i e_i)\, ds. \tag{12}$$

If $|\partial_{ii}^2 \ell| \le H_i$ on this segment, the score $I_i = |\theta_i \partial_i \ell|$ obeys

$$\big||D_i| - I_i\big| \le \tfrac{1}{2} H_i \theta_i^2. \tag{13}$$

Proof. Apply Taylor's theorem with integral remainder to $f(s) = \ell(\theta - s\theta_i e_i)$ on $[0,1]$. The derivatives are $f'(0) = -\theta_i \partial_i \ell(\theta)$ and $f''(s) = \theta_i^2 \partial_{ii}^2 \ell(\theta - s\theta_i e_i)$. Integrating $(1-s)f''(s)$ gives Eq. (12); the reverse triangle inequality and $\int_0^1 (1-s)ds = 1/2$ give the bound. □

The bound is controlled by curvature over a finite path and the squared deletion magnitude. A large bound permits a large error; it does not force one. For two coordinates, their ordering by I_i agrees with ordering by $|D_i|$ whenever the score gap exceeds the sum of their two error bounds.

A cross-entropy counterexample. For a binary positive-label example with logit $w > 0$, let

$$\ell(w) = \log(1 + e^{-w}), \qquad I(w) = \frac{w}{1 + e^{w}} \longrightarrow 0, \qquad D(w) = \log 2 - \log(1 + e^{-w}) \longrightarrow \log 2. \tag{14}$$

Thus the sensitivity can approach zero while the deletion cost remains positive. Nevertheless, $0 \le \ell''(w) = \sigma(w)(1 - \sigma(w)) \le 1/4$. The failure comes from using an endpoint derivative for a finite deletion, even with bounded curvature.

For multiclass cross-entropy $\ell(z, y) = -z_y + \log \sum_k e^{z_k}$ and $p = \mathrm{softmax}(z)$,

$$\nabla_z^2 \ell = \mathrm{diag}(p) - pp^\top, \qquad 0 \preceq \nabla_z^2 \ell \preceq \tfrac{1}{2} I. \tag{15}$$

Positive semidefiniteness follows because this is a categorical covariance matrix. Row k has absolute row sum $2p_k(1 - p_k) \le 1/2$, proving the spectral bound. With logits $z(\theta)$ and Jacobian J, the parameter Hessian is

$$\nabla_\theta^2 \ell = J^\top (\mathrm{diag}(p) - pp^\top) J + \sum_k (p_k - \mathbf{1}[k = y]) \nabla_\theta^2 z_k. \tag{16}$$

Parameter-space curvature also depends on the network. These identities apply to both autoregressive and diffusion models and do not establish that diffusion has larger curvature.

Coordinate scores versus component deletion. For one rank-one update D_j, introduce a gate γ and define $F(\gamma) = \ell(W_{-j}+\gamma D_j)$, where W_{-j} contains all other updates. Its deletion has first-order term $-F'(1) = -\langle \nabla_W \ell, D_j \rangle_F$ and integral remainder $\int_0^1 (1-s)F''(1-s)ds$. The sum of absolute coordinate scores in Eq. (4) is an aggregation proxy, not this gate derivative. For scalar $D = p\lambda q$, each of $p\partial_p \ell$, $\lambda \partial_\lambda \ell$, and $q\partial_q \ell$ equals $D\partial_D \ell$; their absolute sum is $3|D\partial_D \ell|$. This distinction prevents a coordinate Taylor bound from being misidentified as an exact triplet-deletion bound.

A.2 Exact Scale Calibration and Its Scope

Proposition A.2 (Positive scalar loss scaling)**.** For $L_n = c_n \ell_n$ with $c_n > 0$ independent of parameters, any importance aggregation $\mathcal{T}_j$ positively homogeneous in the gradient satisfies

$$\mathcal{T}_j(\nabla L_n) = c_n \mathcal{T}_j(\nabla \ell_n), \qquad \frac{\mathcal{T}_j(\nabla L_n)}{c_n} = \mathcal{T}_j(\nabla \ell_n). \tag{17}$$

Proof. The chain rule gives $\nabla L_n = c_n \nabla \ell_n$. Substitution into the positive homogeneity identity proves the result. Each absolute value and norm in Eq. (4) is positively homogeneous. □

For the coordinate-deletion target of Proposition A.1,

$$\left| \frac{|\theta_i \partial_i L_n|}{c_n} - |D_i| \right| \le \tfrac{1}{2} H_i \theta_i^2. \tag{18}$$

Calibration removes the multiplicative scale; the finite-deletion remainder is unchanged. At a fixed observation, a common positive c_n preserves the ordering of all raw scores. The effect on ordering arises when differently scaled observations are pooled across noise levels or training steps.

A cross-noise reversal. Fix the adapter state and a noise distribution ρ. Let $Y_j(t)$ be the expected normalized sensitivity of component j at noise t. Unconditional raw averaging targets

$$\mathbb{E}_\rho[c(t)Y_j(t)] = \mathbb{E}_\rho[c(t)]\mathbb{E}_\rho[Y_j(t)] + \mathrm{Cov}_\rho(c(t), Y_j(t)). \tag{19}$$

The covariance term can differ across components. With two equally likely noise levels $t = 1$ and $t = 1/4$, let $c(t) = 1/t$, and assign normalized sensitivities $(Y_a(1), Y_a(1/4)) = (2, 0)$ and $(Y_b(1), Y_b(1/4)) = (0, 1)$. Then

$$\mathbb{E}[Y_a] = 1 > \tfrac{1}{2} = \mathbb{E}[Y_b], \qquad \mathbb{E}[cY_a] = 1 < 2 = \mathbb{E}[cY_b]. \tag{20}$$

This construction demonstrates a possible reversal relative to the normalized-sensitivity target, not a measured property of a checkpoint. The weight $c(t)$ is part of the intended training objective; UaRA removes it only from the statistic used for allocation. If the target were importance for the weighted loss itself, multiplication by $c(t)$ would not be a bias relative to that target.

Heterogeneous batches and gradient accumulation. For $L = \sum_b c_b \ell_b$ with sample-specific coefficients, $\nabla L = \sum_b c_b \nabla \ell_b$. Dividing this aggregate by $\bar{c}$ yields $\sum_b (c_b/\bar{c}) \nabla \ell_b$, which generally differs from $\sum_b \nabla \ell_b$. For example, scalar gradients $(1, -1)$ and coefficients $(1, 2)$ give a weighted sum -1 but an unweighted sum 0. No finite scalar division recovers the latter. Exact calibration therefore requires a common factor per observation or access to contributions before heterogeneous weighting. Averaging sample-specific scales for a mixed microbatch is an approximation. Likewise, the absolute value of an accumulated gradient generally differs from the sum of absolute microstep contributions; UaRA captures each contribution before accumulation. Unobserved or zero-loss batches supply no valid regression target. A common coefficient must also be independent of the parameters to apply Proposition A.2.

A.3 Orthogonality, Subspace Variation, and Gradient Mixing

AdaLoRA uses the penalty (Zhang et al., 2023)

$$\mathcal{R}(P,Q) = \|P^\top P - I_R\|_F^2 + \|QQ^\top - I_R\|_F^2. \tag{21}$$

Proposition A.3 (Orthogonality does not fix a subspace)**.** For $R \le \min(d_{\text{out}}, d_{\text{in}})$, every matrix D of rank at most R admits $D = P\Lambda Q$ with $\mathcal{R}(P,Q) = 0$. When R is smaller than an ambient dimension, zero penalty also allows the corresponding column or row subspace to change over training.

Proof. Take a compact SVD $D = U\Sigma V^\top$ and, if needed, extend the columns of U and V to R orthonormal columns and pad Σ with zeros. Setting $P = U$, $Q = V^\top$ yields zero penalty. For any orthogonal matrices $O_L(s), O_R(s)$, the paths $P(s) = O_L(s)P$ and $Q(s) = QO_R(s)^\top$ remain orthonormal. When R is smaller than the ambient dimension, these paths can rotate the column and row subspaces. The penalty contains no comparison between different values of s. □

Noise-dependent active subspaces with an orthogonal bank. Let $P = Q = I_2$, $\boldsymbol{\lambda} = (1,1)$, and

$$\boldsymbol{m}(t_1) = (1,0), \qquad \boldsymbol{m}(t_2) = (0,1). \tag{22}$$

Ignoring the common adapter scale, the two updates are $D_1 = e_1 e_1^\top$ and $D_2 = e_2 e_2^\top$. They have different rank-one subspaces and zero orthogonality penalty. This disproves a necessary conflict between orthogonality and noise-dependent component selection. For two equally weighted quadratic target losses, every noise-independent update D obeys

$$\tfrac{1}{2}\|D - D_1\|_F^2 + \tfrac{1}{2}\|D - D_2\|_F^2 = \left\|D - \tfrac{D_1+D_2}{2}\right\|_F^2 + \tfrac{1}{4}\|D_1 - D_2\|_F^2 \ge \tfrac{1}{2}. \tag{23}$$

Conditional selection attains zero in this constructed example at active rank one, using a bank of two stored components. The gap concerns conditioning, not orthogonality or equal stored parameter counts.

Proposition A.4 (Regularization in a calibrated scoring gradient)**.** Suppose the captured gradient is that of $L = c\ell + \eta\mathcal{R}$, where $c > 0$ is parameter-independent and $\eta \ge 0$. For a coordinate w, define $I^{\text{data}} = |w\partial_w \ell|$ and $I^{\text{mix}} = |w\partial_w L|/c$. Then

$$I^{\text{mix}} = \left|w\partial_w \ell + \frac{\eta}{c} w\partial_w \mathcal{R}\right|, \qquad |I^{\text{mix}} - I^{\text{data}}| \le \frac{\eta}{c}|w\partial_w \mathcal{R}|. \tag{24}$$

For the penalty in Eq. (21),

$$\nabla_P \mathcal{R} = 4P(P^\top P - I_R), \qquad \nabla_Q \mathcal{R} = 4(QQ^\top - I_R)Q. \tag{25}$$

Proof. Substitute $\partial_w L = c\partial_w \ell + \eta\partial_w \mathcal{R}$ and use the reverse triangle inequality. The matrix gradients follow by differentiating the squared Frobenius norms. □

When $c = c(t)$ varies, the relative regularizer contribution is $\eta/c(t)$. Combining Propositions A.1 and A.4 gives the coordinate bound

$$\left|I_i^{\text{mix}} - |D_i|\right| \le \tfrac{1}{2}H_i\theta_i^2 + \frac{\eta}{c(t)}|\theta_i \partial_i \mathcal{R}|. \tag{26}$$

The regularizer can contribute independently of denoising sensitivity if its gradient is included in the captured score. It contributes nothing to this bound when $\eta = 0$ or when scoring uses only data-loss gradients; at exact orthogonality its gradient is also zero. The bound identifies a conditional mechanism, not a universal explanation of transfer degradation. UaRA sets $\eta = 0$; the benefit of that choice is an empirical question.

A.4 The Regression Objective and the Legendre Basis

Fix a component (ℓ, j) and enumerate its valid microsteps as $n_1 < \cdots < n_k$. Using the local update index r, write $(\boldsymbol{\phi}_r, y_r) = (\boldsymbol{\phi}(t_{n_r}), y_{\ell j, n_r})$. The states $\boldsymbol{a}_r$ and C_r are indexed by these valid updates; forgetting is applied once per valid update.

Proposition A.5 (Exponentially weighted least squares)**.** With $\boldsymbol{a}_0 = 0$, $C_0 = \delta I_3$, $\delta > 0$, and $\zeta \in (0, 1]$, the unshrunk RLS coefficients in Eq. (6) minimize

$$J_k(\boldsymbol{a}) = \sum_{r=1}^{k} \zeta^{k-r}(y_r - \boldsymbol{\phi}_r^\top \boldsymbol{a})^2 + \zeta^k \delta^{-1} \|\boldsymbol{a}\|_2^2. \tag{27}$$

Proof. Define

$$H_k = \zeta^k \delta^{-1} I_3 + \sum_{r=1}^{k} \zeta^{k-r} \boldsymbol{\phi}_r \boldsymbol{\phi}_r^\top, \qquad b_k = \sum_{r=1}^{k} \zeta^{k-r} \boldsymbol{\phi}_r y_r. \tag{28}$$

The positive-definite matrix H_k yields the unique solution $\boldsymbol{a}_k = H_k^{-1} b_k$. Since $H_k = \zeta H_{k-1} + \boldsymbol{\phi}_k \boldsymbol{\phi}_k^\top$ and $b_k = \zeta b_{k-1} + \boldsymbol{\phi}_k y_k$, the matrix inversion lemma gives the $C_k = H_k^{-1}$ update in Eq. (6). Substitution into $\boldsymbol{a}_k = C_k b_k$ gives its coefficient update. □

The result concerns the observed records. Selective activation limits which noise–component pairs are observed, and gradients also depend on the other active components. Warmup increases coverage; the recursion alone does not guarantee accurate extrapolation into unobserved regions.

Direct integration of the three basis polynomials gives

$$\int_0^1 \boldsymbol{\phi}(t) \boldsymbol{\phi}(t)^\top dt = \operatorname{diag}(1, 1/3, 1/5). \tag{29}$$

The continuous Gram matrix is diagonal for uniformly sampled noise. Finite samples, forgetting, and selective observation can change the empirical Gram matrix; the identity is not a guarantee of persistent excitation. Shared shrinkage in Eq. (7) is applied after fitting each local least-squares model.

A.5 What the Residual Measures

At a fixed adapter state and observation distribution, let Y be the calibrated sensitivity, $\mu(t) = \mathbb{E}[Y \mid t]$, and $\varepsilon = Y - \mu(t)$. For finite second moments,

$$\begin{aligned} \operatorname{Var}(Y) &= \mathbb{E}[\operatorname{Var}(Y \mid t)] + \operatorname{Var}(\mu(t)), \\ \mathbb{E}[(Y - f(t))^2] &= \mathbb{E}[\operatorname{Var}(Y \mid t)] + \mathbb{E}[(\mu(t) - f(t))^2]. \end{aligned} \tag{30}$$

Proof. Write $Y - f(t) = \varepsilon + \mu(t) - f(t)$, expand the square, and use $\mathbb{E}[\varepsilon \mid t] = 0$ to eliminate the cross term. Setting $f(t) = \mathbb{E}[Y]$ gives the first identity. □

A noise-conditioned predictor can explain the $\operatorname{Var}(\mu(t))$ term that a constant predictor leaves in its residual. For the absolute residual used by UaRA, the reverse triangle inequality gives

$$|\mathbb{E}[|Y - f(t)| \mid t] - \mathbb{E}[|\varepsilon| \mid t]| \le |\mu(t) - f(t)|. \tag{31}$$

Thus a good predictor reduces the contribution of systematic noise dependence to its residual. UaRA's $u_{\ell j}$ is an EMA over the component's observed noise levels, not a separate uncertainty estimate for every t. It mixes stochastic variation, model mismatch, and training drift. It is neither a calibrated posterior variance nor the Taylor remainder in Eq. (12). The product $\widehat{I}(t)u$ is a

prioritization rule: larger residuals increase priority at a fixed positive predicted importance. No confidence-bound or deletion-error guarantee follows from this product alone.

A.6 Allocation Optimality and Sensitivity to Score Error

Proposition A.6 (Exact reserve-then-global selection)**.** With equal unit cost per component, fixed scores, and the constraints in Eq. (10), Eq. (11) returns an optimal feasible mask.

Proof. Start from an optimal feasible set. If a module is missing one of its $r_{\min}$ highest-scoring components, it contains a selected component outside that local top set, since its selected count is at least $r_{\min}$. Swapping them preserves the budget and all module constraints and cannot reduce the objective. Repeating produces an optimal set that contains every reserved component. The remaining problem has only a cardinality constraint, so selecting the globally largest remaining $K - Mr_{\min}$ scores is optimal. A fixed tie ordering makes the construction deterministic. □

Proposition A.7 (Conditional selection and score perturbations)**.** Let $V_t(m) = \sum_j s_j(t) m_j$ on the finite feasible set $\mathcal{F}$, where j now indexes all module–component pairs and every feasible mask selects K components. For integrable scores,

$$\mathbb{E}_t\left[\max_{m\in\mathcal{F}} V_t(m)\right] \geq \max_{m\in\mathcal{F}} \mathbb{E}_t[V_t(m)]. \tag{32}$$

Moreover, at a fixed t, if $\max_j |\widehat{s}_j - s_j| \leq \epsilon$ and $m^\star$, $\widehat{m}$ maximize the true and estimated score sums, respectively, then

$$0 \leq V_t(m^\star) - V_t(\widehat{m}) \leq 2K\epsilon. \tag{33}$$

Proof. For every fixed m, $\max_{m'} V_t(m') \geq V_t(m)$ pointwise; taking expectations and maximizing the right side proves Eq. (32). Equality holds exactly when some fixed feasible mask is optimal almost surely. For the second result, write $\widehat{V}(m) = \sum_j \widehat{s}_j m_j$. Optimality of $\widehat{m}$ and its cardinality yield

$$\begin{aligned} V_t(m^\star) - V_t(\widehat{m}) &= [V_t(m^\star) - \widehat{V}(m^\star)] + [\widehat{V}(m^\star) - \widehat{V}(\widehat{m})] \\ &\quad + [\widehat{V}(\widehat{m}) - V_t(\widehat{m})] \leq K\epsilon + 0 + K\epsilon. \end{aligned} \tag{34}$$

□

These statements concern the specified additive score objective. Interacting components and the finite-deletion remainder prevent direct identification with downstream accuracy. Conditional allocation offers no strict gain when one fixed set is optimal at all noise levels. For frozen regression states, Eq. (9) is continuous and piecewise quadratic in t, whereas the binary masks are piecewise constant, changing at score crossings rather than varying continuously.

The constraint is an active-component budget. The stored adapter parameter count is $\sum_\ell R(d_\ell^{\text{out}} + d_\ell^{\text{in}} + 1)$, plus regression statistics. Masking a dense implementation does not by itself reduce its matrix-multiplication cost. Parameter counts and measured runtime must therefore accompany comparisons at matched active rank.

B Implementation and Training Details

B.1 Experimental Details

Baselines and evaluation. We include the zero-shot backbones and compare UaRA with LoRA (Hu et al., 2022), HiRA (Huang et al., 2025), and NaRA (Wang et al., 2026). All fine-tuning methods use the same backbone variants, task-specific training datasets, and shared training schedule. We report accuracy in percentage and compute the unweighted average across the benchmarks in each domain. Improvements are reported in percentage points.

Implementation details. We fine-tune for one epoch on Commonsense170k and ten epochs on Math14k, using a learning rate of 10^{-4}. Every 64 training steps, we evaluate the validation loss on 128 held-out examples to monitor overfitting. The pretrained backbone remains frozen while the adaptation parameters are optimized. Training follows the masked diffusion objective: the prompt remains visible, response tokens are randomly masked, and the denoising loss is computed at the masked response positions.

B.2 Training Hyperparameters

Table 5 summarizes the UaRA configuration for fine-tuning LLaDA-8B-Base on Math14k. Commonsense170k training uses one epoch, while Math14k training uses ten; both use a learning rate of 10^{-4} and validation every 64 steps on 128 held-out examples. The backbone remains frozen, and the denoising loss is computed only at masked response positions, with the prompt kept visible.

Table 5: Training hyperparameters for UaRA on LLaDA-8B-Base / Math14k.

Hyperparameter	**Value**
Optimizer / learning-rate schedule	AdamW / cosine
Learning rate	10^{-4}
Weight decay / gradient clipping norm	0 / 1.0
Training epochs / sequence length	10 / 1024
Batch size per GPU / accumulation steps	1 / 16
Effective batch size	32 (2 GPUs)
Learning-rate warmup	220 steps
Validation	Every 64 steps; 128 examples
Target projections	W_q, W_k, W_v, W_o
Adapter scaling α / dropout	64 / 0.05
Candidate / minimum / average active rank	48 / 24 / 32
Global active-rank budget K	4096
RLS forgetting factor ζ	0.99
EMA coefficients (β_1, β_2)	$(0.85, 0.85)$

B.3 Training and Inference Procedure

Algorithm 1 summarizes UaRA training, including online utility estimation and budget-constrained rank allocation, as well as noise-conditioned component selection during inference.

C Additional Experiments and Analysis

C.1 Effect of scale calibration on predicted sensitivity

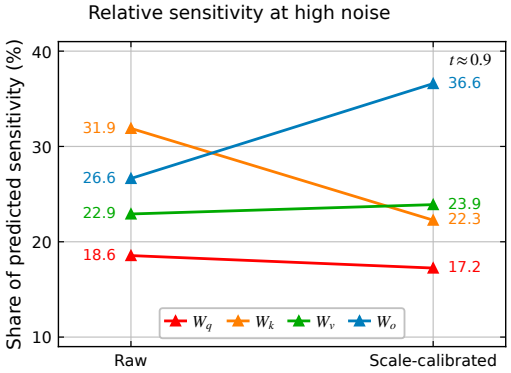

Figure 6: Effect of scale calibration on predicted sensitivity. Each projection type's share of their summed predictions at $t = 0.898$.

Algorithm 1 UaRA training and noise-conditioned inference

Input: Data $\mathcal{D}$; frozen weights W^0; optimizer steps T; accumulation steps G; $R, K, r_{\min}$; $\alpha, \zeta, \delta, \kappa, \beta$; warmup stopping rule

Initialize Ψ with zero weight updates; set warmup to true
Set $\boldsymbol{a}_{\ell j} = 0$, $C_{\ell j} = \delta I_3$, $u_{\ell j} = N_{\ell j} = 0$ for all components
for $s = 1, \ldots, T$ **do**
    Clear accumulated optimizer gradients
    for $h = 1, \ldots, G$ **do**
        Set $n = (s-1)G + h$; sample an example and noise t_n; obtain scale c_n
        if warmup **then**
            Activate all candidate components
        else
            Compute $S(t_n)$ by Eq. (9)
            Reserve module minima and allocate the remaining budget by Eq. (11)
        end if
        Compute $L_n = c_n \ell_n$ with the selected components and backpropagate
        Capture g_n before accumulation; accumulate it for the optimizer
        for each active component (ℓ, j) with a valid sensitivity sample **do**
            Compute $y_{\ell j, n}$ by Eq. (4)
            Compute the residual with the pre-update predictor; update $u_{\ell j}$ by Eq. (8)
            Update $\boldsymbol{a}_{\ell j}, C_{\ell j}$ by Eq. (6); increment $N_{\ell j}$
        end for
        Refresh shared coefficients and the shrunk predictors in Eq. (7)
    end for
    Update Ψ using accumulated training gradients; evaluate the warmup stopping rule
end for

Output: Ψ, frozen predictors $\widetilde{\boldsymbol{a}}$ and residuals u, and $K, r_{\min}$
Inference: at each denoising step, evaluate $S(t)$, select $\mathcal{A}(t)$, and apply Eq. (3)

Figure 6 shows that scale calibration substantially changes the relative sensitivity across projection types. At $t = 0.898$, $W_k > W_o > W_v > W_q$ becomes $W_o > W_v > W_k > W_q$. The share of W_o increases from 26.6% to 36.6%, while that of W_k decreases from 31.9% to 22.3%. These results indicate that calibration corrects the relative importance assigned to different projection types, rather than merely rescaling their sensitivity values.

C.2 Effect of the Minimum Active Rank

We examine post-warmup rank allocation on LLaDA-8B-Base with a candidate rank of $R = 48$. For minimum ranks $r_{\min} \in \{0, 24, 28, 32\}$, the global active-rank budget is $K = 4096$, or 32 per module on average. Figure 7 also includes the fixed-rank variant and the rank-48 configuration.

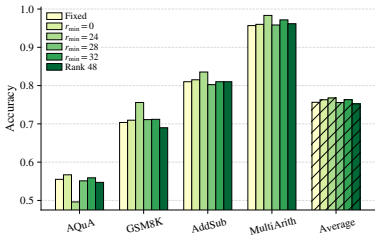

Figure 7: Post-warmup rank allocation on LLaDA-8B-Base. Hatched bars show average accuracy across four benchmarks; $r_{\min} = 24$ gives the best average.

A minimum rank of 24 gives the best average performance. It achieves 76.77% average accuracy, compared with 75.64% for the fixed-rank variant and 76.29% without a minimum-rank constraint. Increasing the minimum to 28 or 32 yields 75.58% and 76.32%, respectively; the rank-48 configuration reaches 75.22%. The $r_{\min} = 24$ setting leads on GSM8K (75.58%), AddSub (83.54%), and MultiArith (98.33%), while $r_{\min} = 0$ performs best on AQuA (56.69%).

The minimum rank controls how much capacity is reserved within each module. With $M = 128$ modules, $r_{\min} = 24$ reserves 3072 components and leaves 1024—25% of the active budget—for global allocation. Raising the minimum to 28 leaves 512 components for redistribution; at 32, every module receives exactly 32 components. We therefore adopt $r_{\min} = 24$: it combines the highest average accuracy with a per-module capacity floor and room for global rank redistribution.

Sensitivity to Cross-Noise Aggregation Weights. We compare three cross-noise weighting schemes on eight commonsense benchmarks. The grid is $\mathcal{T} \approx \{0.1, 0.3, 0.5, 0.7, 0.9\}$. With residual weighting enabled, the normalized weights define the auxiliary pruning score

$$\overline{S}_{\ell j}(\boldsymbol{w}) = u_{\ell j} \sum_{m=1}^{5} \frac{w_m}{\sum_{k=1}^{5} w_k} \widehat{I}_{\ell j}(t_m). \tag{35}$$

This score is stored for pruning and used in warmup diagnostics. All three runs enable noise-conditioned selection: each forward pass uses $S_{\ell j}(t)$ from Eq. (9) at the current noise level, with a post-warmup budget of $K = 4096$.

Figure 8 isolates the aggregation effect by applying all three weight vectors to the same logged sensitivity profiles from the uniform run. At the final checkpoint, the two high-noise points ($t = 0.699, 0.898$) contribute 39.9%, 63.1%, and 81.6% of the aggregated mean sensitivity under low-noise, uniform, and high-noise weighting, respectively. The contribution at $t = 0.1$ remains zero under all three choices because its predicted sensitivity is zero.

Table 6 reports average accuracies of 87.19%, 87.22%, and 87.37% for uniform, low-noise, and high-noise weighting, respectively: a range of only 0.18 percentage points. High-noise weighting has the highest average, while uniform weighting performs best on BoolQ and PIQA.

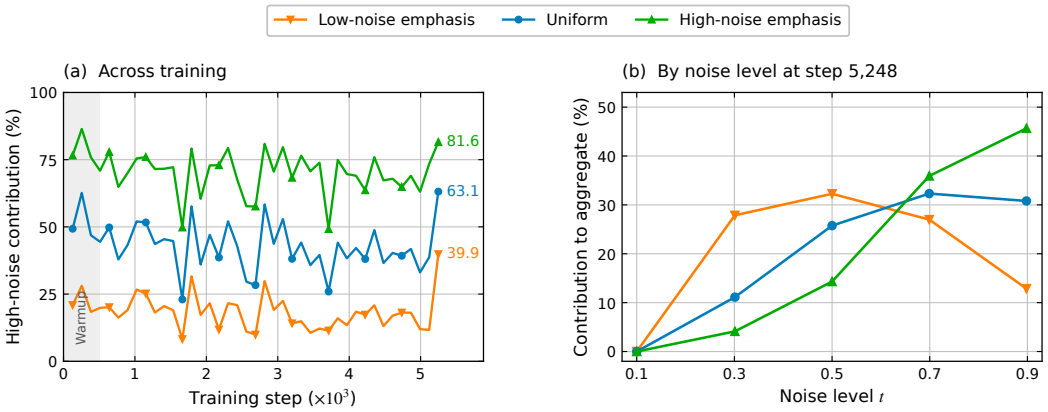

Figure 8: Reweighting the uniform run's logged predictions. Let $\mu(t_m)$ be the mean sensitivity over the four projection types; the contribution of a noise point is $w_m\mu(t_m)/\sum_k w_k\mu(t_k)$, before residual multiplication. (a) Combined contribution of $t = 0.699$ and 0.898 across training; shading marks warmup. (b) Contributions at step 5248. Each panel applies all three weight vectors to the same predictions.

Table 6: Accuracy(%) under cross-noise aggregation weights, ordered by increasing t: Uniform $(0.20, 0.20, 0.20, 0.20, 0.20)$, Low $(0.40, 0.30, 0.15, 0.10, 0.05)$, and High $(0.05, 0.10, 0.15, 0.30, 0.40)$. Avg. is the unweighted mean over eight benchmarks.

Weights	ARC-C	ARC-E	BoolQ	Hella	OBQA	PIQA	SIQA	Wino	Avg.
Uniform	89.08	96.13	**69.54**	95.17	89.60	**87.76**	82.29	87.92	87.19
Low	**89.33**	96.04	68.90	94.97	**91.40**	87.38	82.14	87.61	87.22
High	89.08	**96.21**	69.23	**95.26**	90.60	87.27	**82.50**	**88.79**	**87.37**

C.3 Sensitivity to the RLS Forgetting Factor

We vary the RLS forgetting factor $\zeta \in \{0.97, 0.98, 0.99\}$ to examine the effect of retaining past observations. Table 7 shows that $\zeta = 0.98$ achieves the highest average accuracy (75.17%), followed by 0.97 (74.90%) and 0.99 (73.79%). It also performs best on AddSub, AQuA, and MultiArith, while 0.99 yields a small advantage on GSM8K. These results favor $\zeta = 0.98$ among the tested settings.

Table 7: Sensitivity to the RLS forgetting factor. Accuracy is reported in percent; Avg. is the unweighted mean over the four benchmarks.

Forgetting factor	GSM8K	AddSub	AQuA	MultiArith	Avg.
0.99	**69.52**	79.75	50.39	95.50	73.79
0.98	69.44	**81.27**	**53.15**	**96.83**	**75.17**
0.97	69.32	81.02	52.75	96.50	74.90

C.4 Analysis of Noise-Conditioned Rank Allocation

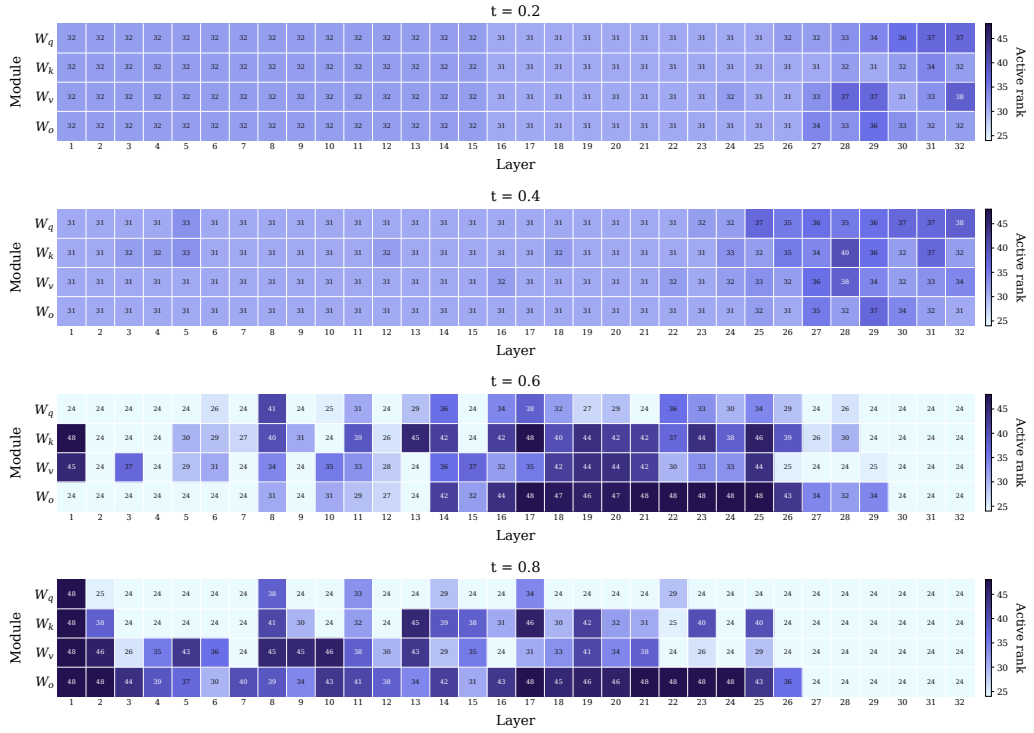

Figure 9: Noise-conditioned rank allocation on LLaDA-8B-Base after mathematical reasoning fine-tuning. Each panel corresponds to a different diffusion state, and each cell gives the active rank of one attention projection in one layer. Colors and values jointly indicate the allocated rank.

Noise level induces distinct layer- and projection-wise rank allocation patterns. As shown in Figure 9, the allocation is relatively uniform at low noise levels ($t = 0.2$ and 0.4), with most modules receiving ranks close to the global average. As the noise level increases, the distribution becomes substantially more heterogeneous. At $t = 0.6$, higher ranks concentrate in several middle layers, especially for W_k and W_o, while many other modules remain near the minimum rank. At $t = 0.8$, this specialization becomes more pronounced: W_o maintains high ranks across many early and middle layers, whereas W_q is largely compressed outside a few selected layers. These results indicate that adaptation demand varies not only across diffusion states but also across layers and projection types, supporting noise-conditioned and fine-grained rank allocation rather than a globally fixed rank.

C.5 Role of RLS Importance Estimation.

We replace the noise-conditioned RLS predictor with an EMA of calibrated importance (decay 0.98) on LLaDA-8B-Base. Both variants retain the prediction–residual product for scoring, compute residuals from the pre-update prediction, and refresh scores every 128 steps. RLS and EMA leave warmup at steps 640 and 768, respectively; both then use an active-rank budget of $K = 4096$ with a minimum rank of 24 per module.