STATIC GRADIENT ATTRIBUTION UNDERPERFORMS A DENSITY-MATCHED RANDOM MASK ON LOSS-BASED FORGETTING WITHIN LORA'S B-MATRIX

Anonymous authors
Paper under double-blind review

ABSTRACT

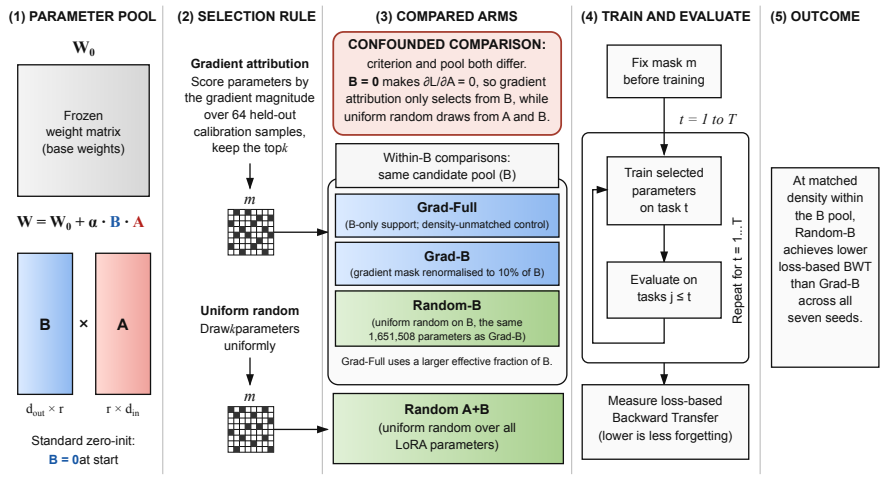
Parameter-efficient continual learning updates a restricted subset of adapter parameters to incorporate sequential tasks while limiting catastrophic forgetting. Gradient attribution is a common rule for selecting that subset, but it requires an additional backward pass over calibration data. We identify a parameter-pool confound in comparisons between initialization-time gradient masks and unrestricted random masks under standard zero-initialized LoRA ($B = 0$): the gradient with respect to A is identically zero ($\partial\mathcal{L}/\partial A = 0$), so gradient masks select strictly from the B matrix whereas unrestricted random masks draw from both A and B. Such a comparison changes both the selection rule and the candidate parameter pool. Isolating the selection criterion by restricting both approaches to the same parameter pool, we find a negative result: a uniform random mask consistently achieves lower loss-based backward transfer than gradient attribution across seven seeds on Qwen2.5-3B. The effect is concentrated on the medical task, which contributes 80% of the aggregate; code and math move in the same direction, and legal shows no reliable difference. This advantage is consistent with a stability–plasticity trade-off under the tested task sequence, in which random masking slightly reduces single-task fitting. On the two discrete-option tasks (Medical and Legal), the mean accuracy-BWT difference is not statistically distinguishable from zero. The 95% interval extends to 4.80 percentage points in the gradient mask's favour, so the pre-specified 4-point non-inferiority criterion is not met. On loss-based forgetting, Random-B consistently outperforms the density-matched gradient mask across seven seeds; on accuracy, the evidence remains inconclusive at $n = 7$. The attribution pass is therefore not justified by the loss-based forgetting evidence, while its accuracy consequences require a better-powered evaluation.

1 INTRODUCTION

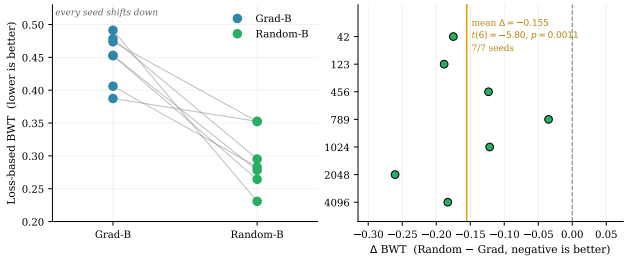
Parameter-efficient adapters such as LoRA (Hu et al., 2021; Dettmers et al., 2023) make it possible to incorporate a sequence of tasks while updating only a small fraction of a language model. Sequential updates can nevertheless overwrite capabilities acquired on earlier tasks, the classical problem of catastrophic forgetting (McCloskey & Cohen, 1989; French, 1999). Continual learning for LoRA adapters updates only a chosen subset of parameters, and gradient attribution is the common rule for choosing it: a backward pass over held-out calibration data scores each parameter, and the highest-scoring ones are updated. Choosing the subset uniformly at random costs nothing instead, so whether the attribution pass earns its cost has a direct engineering answer.

Standard zero-initialized LoRA introduces a potential confound. With $B = 0$ at initialization, $\partial\mathcal{L}/\partial A = 0$, so gradient attribution assigns zero importance to every `lora_A` parameter and a gradient mask selects only from `lora_B`, while an unrestricted random baseline draws from both factors together. Any such comparison mixes the selection criterion with the parameter pool. Attribution-Guided CL (Liu et al., 2026) is the closest prior use of gradient attribution in LLM continual learning, but it estimates importance from a separately trained model; because $B \neq 0$ at scoring time, the zero-initialization confound studied here does not apply to its attribution procedure.

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.



(a) Method and comparison.



(b) Key effect (D1, seven seeds).

Figure 1: The density-matched comparison. (a) Both arms draw 10% of the rank-16 `lora_B` pool, so they train exactly the same 1.6M parameters and differ only in how those coordinates are chosen; because $B = 0$ at initialization, the gradient score is zero for every `lora_A` coordinate, and any baseline that draws from both pools changes the pool as well as the criterion. (b) Under that matched density the random mask achieves lower loss-based BWT than the gradient mask in all seven seeds.

We separate the two by restricting both arms to the same pool and parameter count. On Qwen2.5-3B with rank-16 LoRA over a five-task sequence we compare a uniformly random mask on `lora_B` (Random-B) against a gradient mask renormalised to the same 10% of `lora_B` (Grad-B), so both train exactly the same parameters and differ only in how those parameters are chosen. Across seven seeds the random mask achieves the lower loss-based backward transfer, and the gap holds in every seed (Table 2, Figure 1). The aggregate gap is concentrated on the medical task, with code and math moving in the same direction and legal showing no reliable difference, a pattern consistent with a stability–plasticity trade-off: the random mask fits each task less well on its own diagonal yet retains earlier tasks better. Above all, this work makes two key contributions:

A gradient-attribution confound and density-matched evaluation.

We show that under standard zero-initialized LoRA, initialization-time gradient attribution assigns zero importance to `lora_A`, so an unrestricted gradient-versus-random comparison changes both the selection rule and the candidate parameter pool. We resolve this confound with a within-`lora_B`, density-matched design that holds the candidate pool and update budget fixed, leaving the selection criterion as the only varying factor.

Practical validation and scope. We quantify the cost of the attribution pass and evaluate the same comparison using accuracy-based retention (accuracy-BWT). The accuracy-BWT interval spans zero and does not establish the pre-specified 4-point non-inferiority criterion, so the engineering recommendation is scoped to the loss-based evidence. Removing attribution eliminates a one-time 64-sample calibration forward–backward pass (64 steps), equal to 12.8% of one 500-step task-training budget, together with the calibration-data requirement.

2 Background and Related Work

Continual learning and parameter importance. Continual learning aims to reduce catastrophic forgetting as models are updated on sequential tasks (McCloskey & Cohen, 1989; French, 1999). Importance-based methods such as EWC (Kirkpatrick et al., 2017) and SI (Zenke et al., 2017) protect parameters judged important to earlier tasks, while GPM (Saha et al., 2021), OGD (Farajtabar et al., 2020), and gradient-based selective-update methods (Qiao et al., 2025; Wang et al., 2025) constrain updates using gradient or parameter-importance structure. Pruning methods pose a related selection problem: Wanda (Sun et al., 2024) uses activation-aware saliency and SNIP (Lee et al., 2019) scores connections before training. These lines motivate our question: once the candidate pool and update budget are controlled, does the *selection rule* itself improve continual-learning retention.

Random subspaces and intrinsic dimensionality. A complementary view is that the update budget may matter more than the exact coordinates selected. Random subnetworks can remain effective (Frankle & Carbin, 2019; Ramanujan et al., 2020), fine-tuning often occupies a low-dimensional subspace (Aghajanyan et al., 2021), and large fractions of delta parameters can be removed (Yu et al., 2024). Related PEFT results point in the same direction: HAT (Serrà et al., 2018) learns task-specific masks, Xu & Zhang (2024) show that random masking can recover strong PEFT subnetworks, and Biderman et al. (2024) find that LoRA forgets less than full fine-tuning. Because LoRA already constrains updates to a low-rank subspace, masking imposes a second restriction within it; our question is whether gradient attribution improves over a random restriction of the same size.

LoRA-specific continual learning and parameterization. LoRA-FA (Zhang et al., 2023) freezes `lora_A` and trains only `lora_B`, whereas we freeze a random or gradient-selected subset of a standard LoRA. O-LoRA (Wang et al., 2023) and InfLoRA (Liang & Li, 2024) instead organize task updates into separated subspaces. PiSSA (Meng et al., 2024) is important to the scope of our analysis because it initializes $B \neq 0$, so the zero-gradient confound we identify is specific to standard zero-initialized LoRA.

Closest and concurrent work. Most directly related, Attribution-Guided CL (Liu et al., 2026) uses gradient attribution in LLM continual learning, but estimates importance from a separately trained model and gates gradients continuously. Because $B \neq 0$ at scoring time, our zero-initialization confound does not apply to that procedure; it nevertheless motivates the broader question of whether attribution improves retention relative to simpler controls. Concurrent results also point to substantial redundancy in adapter or sparse parameterizations: Dieudonné et al. (2026) study low-rank redundancy across task-specific LoRA adapters, while Adewuyi et al. (2026) find that random sparse subnetworks suffice for RLVR. Our contribution is complementary: we isolate the selection criterion itself by matching both the LoRA parameter pool and the update budget.

3 Methodology: Masking Mechanisms

3.1 Problem Formulation

We consider the standard sequential fine-tuning protocol for continual learning. A base model f_θ is adapted to task t on its training split $\mathcal{D}_t$ and then evaluated on all tasks $j \leq t$. Tasks arrive in a fixed order indexed by $t = 1, \ldots, T$, with $T = 5$ throughout this paper.

We use LoRA adapters, which add a low-rank update $\Delta W = BA$ to a frozen pretrained weight matrix W_0, where $B \in \mathbb{R}^{d_{\text{out}} \times r}$, $A \in \mathbb{R}^{r \times d_{\text{in}}}$, and $r \ll \min(d_{\text{out}}, d_{\text{in}})$:

$$W = W_0 + \alpha BA. \tag{1}$$

Table 1 summarizes the notation used throughout. A binary mask $m \in \{0,1\}^{|A|+|B|}$ restricts optimization to coordinates for which $m_i = 1$. The mask is constructed once from the calibration set before the first task and remains fixed throughout the task sequence (Algorithm 1).

We compare two coordinate-selection rules: *gradient attribution*, which ranks coordinates by their calibration-set gradient magnitude, and *uniform random selection*. For a fixed candidate parameter pool, both methods activate the same budget k; they therefore differ only in which coordinates are selected.

Table 1: Notation. Indices run over tasks $t = 1, \ldots, T$ with $T = 5$, and over prior tasks $j \in \mathcal{P} = \{1, \ldots, T-1\}$.

Symbol	Meaning
f_θ, W_0	base model and its frozen pretrained weight matrix
B, A	low-rank LoRA factors, $B \in \mathbb{R}^{d_{\text{out}} \times r}$, $A \in \mathbb{R}^{r \times d_{\text{in}}}$
r, α	LoRA rank ($r = 16$) and scaling ($\alpha = 32$)
W	adapted weight, $W = W_0 + \alpha BA$
t, T	task index and number of tasks ($T = 5$)
$\mathcal{D}_t$	training split of task t
$\mathcal{D}$	held-out calibration set (64 Alpaca samples)
m	binary mask over the trainable coordinates, $m \in \{0,1\}^{\lvert A\rvert+\lvert B\rvert}$
k	selection budget: number of coordinates with $m = 1$
g_i	gradient-attribution score of coordinate i (§3.3)
$\mathcal{L}_j^{(\tau)}$	evaluation loss on task j after training through task τ
$\mathcal{P}$	prior-task set $\{1, \ldots, T-1\}$
BWT	loss-based backward transfer, Eq. equation 2

We measure forgetting using loss-based backward transfer:

$$\text{BWT} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \left(\mathcal{L}_j^{(T)} - \mathcal{L}_j^{(j)} \right), \tag{2}$$

where $\mathcal{P} = \{1, \ldots, T-1\}$. This is the mean increase in evaluation loss on prior tasks after completing the full sequence; lower values indicate less forgetting.

3.2 The $B = 0$ Gradient-Attribution Confound

Standard LoRA initializes $B = 0$. For

$$W = W_0 + \alpha BA, \tag{3}$$

the gradient with respect to A is

$$\frac{\partial \mathcal{L}}{\partial A} = \alpha B^\top \frac{\partial \mathcal{L}}{\partial W}. \tag{4}$$

Hence, at initialization,

$$B = 0 \quad \Longrightarrow \quad \frac{\partial \mathcal{L}}{\partial A} = 0. \tag{5}$$

Gradient attribution therefore assigns zero importance to every `lora_A` coordinate and selects exclusively from `lora_B`. In contrast, an unrestricted random mask samples from the full LoRA parameter pool, `lora_A` ∪ `lora_B`. A direct comparison between these masks consequently changes two factors at once: the *selection criterion* and the *candidate parameter pool*.

To isolate the selection criterion, the candidate pool must therefore be matched. Restricting gradient attribution explicitly to `lora_B` is a null operation under zero initialization, whereas restricting the random baseline to `lora_B` removes the pool mismatch. Our primary comparison accordingly evaluates gradient and random selection within the same `lora_B` pool; Section 5.2 additionally matches their selection density.

This scope excludes Attribution-Guided CL (Liu et al., 2026), which computes attribution from a separately trained model. Because $B \neq 0$ at scoring time in that procedure, the zero-initialization confound identified here does not apply to it.

This confound is specific to standard zero-initialized LoRA. For example, PiSSA (Meng et al., 2024) initializes both A and B from principal singular values, so $B \neq 0$ and Equation 4 does not force the gradient with respect to A to vanish. LoRA+ (Hayou et al., 2024) addresses related asymmetric optimization dynamics between the two LoRA factors. Whether gradient attribution is beneficial under non-zero initialization remains an open question.

3.3 Density-Matched Mask Construction

Let $\mathcal{S}$ denote the candidate set of LoRA coordinates from which a mask is selected. For gradient attribution, each coordinate $i \in \mathcal{S}$ is scored by its mean absolute gradient over the held-out calibration set $\mathcal{D}$:

$$g_i = \mathbb{E}_{x \sim \mathcal{D}} \left[\left| \frac{\partial \mathcal{L}(x)}{\partial w_i} \right| \right]. \tag{6}$$

Given a selection budget k, the binary mask activates the k highest-scoring coordinates:

$$m_i = \mathbf{1}\left[i \in \mathrm{TopK}\left(\{g_j\}_{j \in \mathcal{S}},\, k \right) \right]. \tag{7}$$

For uniform random selection, we instead draw independent scores

$$u_i \sim \mathrm{Uniform}(0, 1), \qquad i \in \mathcal{S}, \tag{8}$$

and apply the same top-k operator in Equation 7 with u_i in place of g_i. Thus, once the candidate pool $\mathcal{S}$ and budget k are fixed, the two methods differ only in the rule used to rank coordinates.

Density-matched comparison. Following the pool confound established in §3.2, our primary comparison restricts both selection rules to the `lora_B` coordinate pool, denoted by $\mathcal{B}$. Both headline arms activate exactly 10% of this pool:

$$k_{\mathrm{D1}} = 1{,}651{,}508. \tag{9}$$

Grad-B applies Equation 6 within $\mathcal{B}$ and selects the top k_{D1} coordinates, whereas *Random-B* selects the same number of coordinates uniformly at random from $\mathcal{B}$. The two arms therefore have the same candidate pool and the same number of trainable coordinates; the selection criterion is the only varying factor.

Reference and control arms. For comparison with the conventional design, *Grad-Full* applies gradient attribution with the full LoRA coordinate set $\mathcal{A} \cup \mathcal{B}$ as its nominal candidate pool and uses a 10% full-pool budget of 2,993,357 coordinates. Under standard zero initialization, its selected support nevertheless lies entirely within $\mathcal{B}$ by §3.2; because its budget is larger than k_{D1}, Grad-Full is pool-matched to Random-B in effective support but not density-matched. *Random A+B* draws the same 10% full-pool budget uniformly from $\mathcal{A} \cup \mathcal{B}$ and therefore constitutes the conventional pool-unmatched random baseline.

We additionally use two auxiliary controls where relevant. *No mask* is standard LoRA with all adapter coordinates updated. *Uniform-B* uses the same `lora_B` budget as the matched comparison while distributing selections uniformly across LoRA modules; it is used only as a module-concentration control in §6.2.

Algorithm 1 summarizes mask construction and sequential training.

4 Experimental Setup

4.1 Models, Benchmarks & Training Details

Qwen2.5-3B (Team, 2024) in 4-bit NF4 (bitsandbytes), LoRA rank 16 (α=32), AdamW (lr 2×10^{-4}, warmup 0.03), 1 epoch, 500 train samples per task, batch size 1, max length 512; calibration uses 64 Alpaca samples. All methods operate on 29.9M trainable LoRA parameters (0.96% of 3.1B); the full-pool top-10% selects 2,993,357, falling entirely within `lora_B` ($B = 0$) and activating $\sim$20% of B, against 10% for Random-B. The comparison is matched on pool but not density; §5.2 equalizes it.

Five domains are used in a fixed order: Code (sahil2801/CodeAlpaca-20k), Medical (GBaker/MedQA-USMLE-4-options), Legal (lex_glue, casehold subset), Math (openai/gsm8k), Creative (ChaoticNeutrals/Creative_Writing-ShareGPT, 6.6K instruction-response pairs). Calibration: tatsu-lab/alpaca (64 samples); 100 eval samples per task.

Table 2: **Headline result (D1, density-matched).** Both arms select the same 1,651,508 parameters (10% of `lora_B`); only the criterion differs. Per-seed loss-based BWT and Δ = Random-B − Grad-B (negative = random-B forgets less), seven sessions, one seed each. The last column is the density-*unmatched* contrast from the same sessions, that is, Random-B against the unrestricted gradient mask (Δ = Random-B − Grad-Full) rather than against Grad-B.

Seed	Random-B BWT ↓	Gradient-B BWT ↓	Δ (matched)	Δ (unmatched)
42	**0.278**	0.453	−0.175	−0.220
123	**0.264**	0.453	−0.188	−0.119
456	**0.283**	0.406	−0.123	−0.131
789	**0.352**	0.387	−0.035	−0.074
1024	**0.352**	0.474	−0.121	−0.109
2048	**0.231**	0.491	−0.261	−0.187
4096	**0.295**	0.478	−0.183	−0.166
Mean	**0.294**	0.449	−0.155	−0.144

4.2 Evaluation Metrics

We use Loss-based BWT in preference to accuracy-based BWT because loss is continuous and more sensitive to the small differences between masking strategies. All reported forgetting metrics are loss-based, reported as mean ± std over seven seeds (42, 123, 456, 789, 1024, 2048, 4096); part of the runs used a second GPU. The accuracy side of the same comparison, together with the option log-likelihood scorer and its scope, is reported separately in §5.4.

5 Experiments and Main Results

5.1 Confounded and Density-Matched Comparisons

A first comparison follows the usual design, in which each method draws from its own pool. Under that design gradient attribution shows no advantage: its loss-based BWT is 0.437 ± 0.061 against 0.428 ± 0.019 for a shared random mask over five seeds, a seed-dependent difference that is not significant (Table 5, Appendix D). Both methods still reduce forgetting relative to an unmasked adapter, but the comparison is uninformative about the selection rule, for the reason set out in §3.2: gradient attribution selects only `lora_B` while the random baseline draws from both pools, so the design varies pool and criterion together. The result is reported for completeness and is distinct from the seven-seed comparison that follows, the two coming from different sessions whose absolute levels are not comparable (Appendix F).

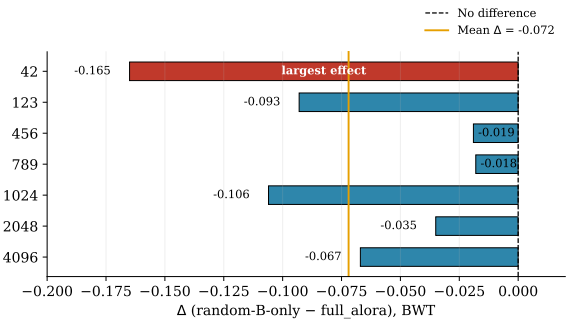


Figure 2: Per-seed within-`lora_B` differences (Δ = Random-B − Grad-Full, loss-based BWT, 7 seeds). All seven are negative (Random-B forgets less), mean $\Delta = -0.072$ (orange line); seed 42 is the largest effect (-0.165). Leave-one-out excluding seed 42 remains significant (paired $t(5)$, $p = 0.015$).

Holding the pool fixed changes the picture. We introduce a Random-B control drawn from `lora_B` alone, matching the gradient mask's pool, and compare it against a gradient mask restricted to the same pool. Each seed is an independent session, so absolute BWT values drift between runs (Appendix F) and only within-session paired observations are used. This comparison is matched on pool but not on density, since the gradient mask activates nearly twice as many `lora_B` coordinates as Random-B. Even so, the random mask achieves the lower loss-based BWT in all seven seeds, with a mean difference of -0.072 that survives dropping the largest-effect seed (two-sided paired $t(5) = -3.62$, $p = 0.015$; Figure 2, Appendix E). This value is the direct recomputation on the six retained paired contrasts and corrects the $p = 0.029$ value reported in an earlier draft. That the direction is unchanged when the pool is fixed is the first indication that the criterion, rather than the pool, is what the earlier comparison was measuring.

5.2 Density-Matched Control

Pool and criterion are separated cleanly only when density is matched as well. The comparison above leaves the gradient arm activating about 20% of the pool against 10% for Random-B, so the two arms still differ in how many coordinates they train. D1 removes that difference, with both arms drawing exactly 10% of `lora_B` and only the ranking rule differing. At matched density the random mask again achieves the lower loss-based BWT in every seed, by a larger margin than in the density-unmatched comparison (Table 2). Under a common [Grad-B $-$ Random-B] convention, the mean contrast is $+0.155$ on the V100 stack and $+0.152$ on the A800 stack. Thus the contrast reproduces at a similar magnitude in a second hardware/software environment; in a separate three-seed sensitivity check, reversing the execution order of the two arms does not reverse the Random-B–Grad-B contrast (Appendix F).

5.3 Task-Level Decomposition

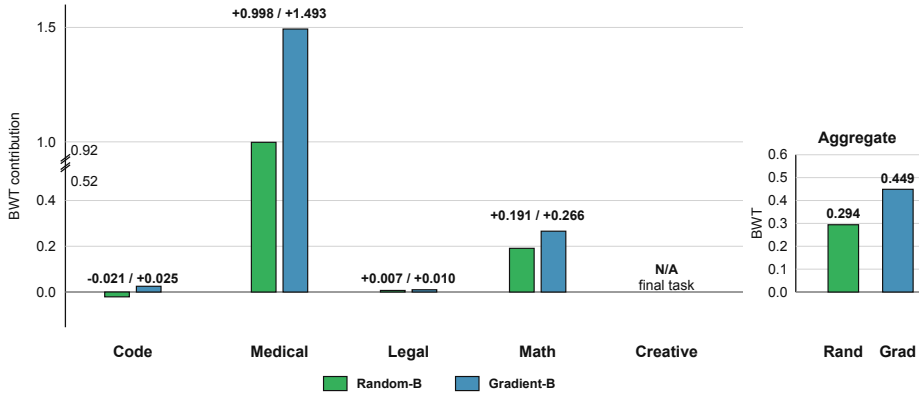


Figure 3: The density-matched D1 per-task decomposition, averaged over seven seeds. For each prior task, the bars show the loss-based BWT contribution of Random-B and Grad-B, with the corresponding values reported above each pair. Because the task-level contributions span roughly two orders of magnitude, the left panel uses a broken y-axis: the slashes mark the omitted interval from 0.52 to 0.92, and the labels above the Medical bars report their full values ($+0.998$ and $+1.493$) rather than truncated heights. Creative is the final task and therefore has no backward-transfer contribution. The right panel shows the aggregate loss-based BWT on a separate scale. Random-B exhibits less loss-based forgetting on Code, Medical, and Math; the Legal difference is negligible and changes sign across seeds. Most of the aggregate difference is concentrated on Medical. Per-seed results are reported in Appendix Q.

The aggregate gain is not spread evenly across the five tasks, and where it concentrates bears on how the effect should be read. Medical carries about four fifths of it, while code and math contribute the remainder in the same direction; on legal the arms are indistinguishable. Two explanations can be

assessed from the same data. First, the gradient arm's loss on each task's own diagonal is lower than the random arm's in every seed and task, yet it subsequently forgets more, so the two arms differ in retention rather than initial learning capacity. The same seven within-session pairs also show lower mean final held-out loss over the four prior tasks for Random-B: the Grad-B minus Random-B endpoint difference is $+0.0813$ (paired $t(6) = 3.03$, $p = 0.023$; 6/7 seeds). This endpoint result is less robust to leave-one-out deletion than the headline BWT contrast, but it shows that the effect is not merely a BWT normalization artifact (Appendix R). By contrast, restoring the gradient arm's full budget produces seed-dependent results: at seed 42 the density-matched gradient arm outperforms the unrestricted one, whereas the ordering reverses at the seven-seed mean. It therefore does not provide a stable explanation of the headline effect.

5.4 Accuracy-Based Validation

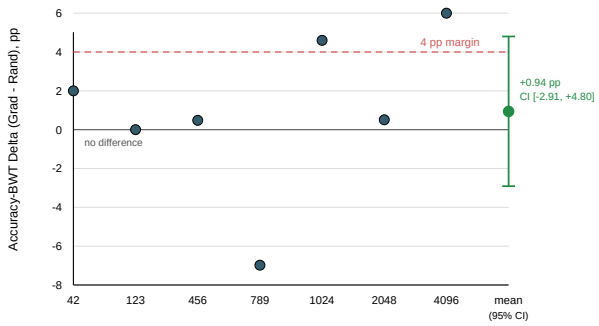


Figure 4: Accuracy-based validation. Blue points are the seven per-seed accuracy-BWT differences (Grad-B minus Random-B, percentage points) on the two discrete-option tasks, spanning -6.98 to $+6.00$ pp; green is the seven-seed mean with its 95% interval $[-2.91, +4.80]$ pp. The plotted quantity is a difference of accuracy-BWT (mean final-minus-learned accuracy over prior tasks), a retention measure, not a difference of final accuracies. The interval spans zero, so no statistically reliable accuracy difference is detected, and it reaches past the 4 pp non-inferiority margin, so that criterion is not met. Per-seed values are in Appendix N.

Loss-based BWT is a proxy for forgetting rather than the quantity of ultimate interest, so we evaluate the same masking comparison on task accuracy and ask whether the accuracy ordering agrees with the loss ordering. The evaluation reuses the primary protocol and the corrected task evaluators over the same seven seeds and 100 evaluation items per task, retraining from the base model on an A800 rather than re-scoring a checkpoint. Accuracy is scored only for the two tasks with discrete options (Medical, MedQA four options; Legal, CaseHOLD), so every quantity below concerns those two.

Across seven seeds, the accuracy-BWT difference averages $+0.94$ percentage points with a 95% interval of $[-2.91, +4.80]$ pp that spans zero. Here accuracy-BWT is the mean final-minus-learned accuracy over the prior discrete-option tasks, so it measures retention on the same footing as loss-based BWT and is not a difference of final accuracies. The individual seeds range from -6.98 to $+6.00$ pp, and it is that spread rather than a large mean that makes the interval wide at $n = 7$. A 4 pp non-inferiority margin was specified in advance, and the measured upper bound exceeds it, so non-inferiority is not established.

On the loss side, the same masks are separated: Grad-B forgets more in all seven seeds by a mean of $+0.152$ on the A800 loss runs. This agrees in mean to within 0.003 with the V100 estimate after aligning sign conventions ($+0.155$ under [Grad $-$ Random]). Loss therefore resolves a difference the accuracy measurement does not; the accuracy interval bounds that difference rather than contradicting it.

Per-seed values and the interval construction are reported in Appendix N. Math is excluded from the accuracy read-out because its generation-based score is confounded by truncation (Appendix O).

6 Analysis and Extensions

6.1 Sparsity, Scale, and Cross-Architecture Generalization

The headline comparison fixes the sparsity budget at 10%. Sweeping it from 5% to 30% at a single seed, no tested level shows gradient attribution outperforming a shared random mask (Figure 6, Appendix H). The sweep is preliminary and inherits the pool confound rather than isolating the budget, so we report it as a boundary check on the direction of the effect rather than as evidence for the sparsity-constraint view.

No single-seed probe reverses the conclusion, but a reversed task order shifts BWT by more than any method difference, which is why every conclusion in this paper is scoped to the tested sequence (Table 13, Appendices H–K). On Phi-2, an architecturally distinct family, we additionally run a clean density-matched D1 replication: both arms select exactly 10% of the same `lora_B` pool and therefore differ only in the ranking rule. Random-B achieves lower loss-based BWT in all four seeds, with mean Δ = Random-B − Grad-B = -0.1152 (SD 0.0173), compared with -0.1552 on Qwen2.5-3B. The task-level decomposition also reproduces the primary pattern: Medical contributes 67.2% of the Phi-2 aggregate contrast and is the largest contributor in every seed, versus 79.9% on Qwen2.5-3B. Because the replication has only four seeds, we treat its parametric test as supporting evidence and base the cross-architecture claim primarily on the matched design and 4/4 directional replication; full per-seed results and the sign-test caveat are in Appendix S.

6.2 Module Concentration and Calibration Robustness

The gradient mask does not select uniformly across module types: within `lora_B` it concentrates on output projections and to a lesser degree on `up_proj`, together accounting for a little over half of all selections, while down projections receive well below the uniform rate. The pattern is consistent across calibration sources and peaks at middle layers (Appendix J). That concentration invites the reading that it causes the gradient mask's higher BWT, and the `uniform_B` controller tests it by spreading the same budget across modules. The controller contradicts the reading: three of four seeds land on the gradient side rather than the random side. Because those offsets are taken against different session baselines (Appendix F), we treat the ablation as failing to establish the mechanism rather than as evidence against the concentration itself, and neither the within-`lora_B` finding nor its density-matched control depends on it (Figure 7). The result is also insensitive to the calibration domain: pooling in-domain rather than out-of-domain calibration changes 42% of the selected parameters, yet the loss-based BWT moves by less than a tenth of the between-seed spread (Appendix L).

7 Conclusion

At equal parameter count within the `lora_B` pool, the random mask achieves lower loss-based BWT in every seed, an advantage that reproduces in mean and direction in a second hardware/software environment and is not an artifact of density. The same paired runs also show lower mean final held-out loss on the prior tasks for Random-B (Grad-B minus Random-B = $+0.0813$, $p = 0.023$), although this endpoint result is less leave-one-out robust than the BWT contrast. On the loss-based evidence, the attribution pass is not justified: removing it eliminates a one-time 64-sample calibration forward–backward pass (64 steps), equal to 12.8% of one 500-step task-training budget, and removes the calibration-data requirement. Accuracy-BWT remains inconclusive at seven seeds, with the pre-specified non-inferiority criterion unmet. All conclusions are scoped to the tested task order and static masks; a per-task adaptive mask and a better-powered accuracy evaluation are the most informative next tests.

Limitations. The within-`lora_B` finding is limited to static masks, a single primary task order, and rank-16 LoRA under standard zero initialization. Our primary conclusions are based on loss-based BWT, with accuracy validation limited to the two discrete-option tasks. Absolute BWT also varies across sessions, so all statistical claims rely on within-session paired comparisons (Appendix F).

REPRODUCIBILITY STATEMENT

Experiments use Python 3.12, PyTorch 2.8.0+cu128 (4-bit NF4 with bf16 compute), Transformers 5.16.1, PEFT 0.20.0, bitsandbytes 0.50.2, datasets 5.0.1, and CUDA 12.8. The primary seven-seed within-`lora_B` comparison, scale runs, and ablations use a single NVIDIA A800 80GB PCIe GPU. The D1 density-matched control and the M2 uniform-B control ran in a separate session on a Tesla V100-PCIE-32GB under an earlier environment (Python 3.12.3, PyTorch 2.3.0+cu121, Transformers 4.46.3, PEFT 0.13.2, bitsandbytes 0.43.3); as both are within-run paired comparisons, the hardware difference does not affect their paired results. Primary results use Qwen2.5-3B (seven seeds); cross-architecture within-B results use Phi-2 2.78B (four seeds); the confounded Phi-2 comparison uses five seeds under a matched 100-eval-sample protocol (an additional seed-42 observation used 20 eval samples and is reported separately); scale observations use Qwen2.5-0.5B/1.5B/7B (single seed). LoRA is applied to all linear projection layers.

Model training uses AdamW (β_1=0.9, β_2=0.999, weight decay 0.01) at learning rate 2×10^{-4} with a linear warmup of 0.03 and cosine annealing over the per-task step budget; LoRA dropout 0.05, rank 16, α=32, batch size 1, max length 512, one epoch per task, and top-k 0.1 masking. Five sequential tasks are used: Code (code_alpaca_20k), Medical (medqa, 4-option MC), Legal (lexglue/case_hold), Math (gsm8k), and Creative Writing (Creative_Writing-ShareGPT). Each task uses 500 training and 100 evaluation samples; gradient masks use 64 out-of-domain Alpaca calibration samples. Seeds are 42, 123, 456, 789, 1024, 2048, and 4096; approximately 60 GPU-hours were used in total.

Boundary analyses (rank sweep, task order reversal, scale observations) are single-seed; a method-*order* check over three seeds is reported in Appendix F. An isolated-process protocol equivalence check confirms that running each method in a separate process reproduces the multi-method BWT to within 9.6×10^{-5} ($10\times$ margin below the pre-specified 10^{-3} threshold). All reported forgetting metrics are loss-based BWT. Code and configuration files are released as anonymized supplementary material (`code_anonymous.zip`): training and attribution scripts, five-task data preparation, mask generation, the analysis scripts computing BWT and the paired statistics, and pinned dependency manifests for both environments.

ETHICS STATEMENT

This work studies parameter masking for continual learning and does not involve human subjects or sensitive data. It does not introduce new capabilities or datasets that raise ethical concerns. On data sources and licensing: the Code task uses CodeAlpaca-20k (CC BY 4.0), the Math task uses GSM8K (MIT), and the Medical and Legal tasks use MedQA and CaseHOLD, which are distributed on Hugging Face with no affirmative license declaration, so we use them for research evaluation only. The Creative task uses a synthetic creative-writing set composed of model-generated outputs rather than user conversations. The Alpaca data used to calibrate the gradient masks is released under CC BY-NC 4.0, a non-commercial license; we therefore make no commercial-use claim for models trained with it. The finding that random selection outperforms gradient attribution within LoRA's B-matrix may reduce computational cost in continual-learning deployments by removing the need for gradient-based parameter attribution. We do not foresee negative societal consequences specific to this contribution.

AI USE DISCLOSURE

AI coding assistants were used for experiment-script development and debugging, and AI assistance aided literature search and manuscript drafting and revision. The research design, experimental methodology, statistical analysis, and scientific conclusions are the authors' own work. All reported results were produced by the described experimental pipeline without AI-generated synthetic data; all analyses and claims were verified by the authors.

REFERENCES

Israel Adewuyi, Solomon Okibe, and Vladmir Ivanov. The multiple ticket hypothesis: Random sparse subnetworks suffice for RLVR. *arXiv preprint arXiv:2602.01599*, 2026.

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021. URL https://arxiv.org/abs/2012.13255.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision*, pp. 139–154, 2018. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Rahaf_Aljundi_Memory_Aware_Synapses_ECCV_2018_paper.html.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2405.09673.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *arXiv preprint arXiv:2305.14314*, 2023. URL https://arxiv.org/abs/2305.14314.

Tanguy Dieudonné, Giulia Lanzillotta, Enis Simsar, Louis Barinka, and Thomas Hofmann. When one adapter speaks for many: Discovering low-rank redundancy in continual fine-tuning. *arXiv preprint arXiv:2606.28117*, 2026.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773, 2020. URL https://arxiv.org/abs/1910.07104.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1803.03635.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024. URL https://arxiv.org/abs/2402.12354.

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. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://arxiv.org/abs/1612.00796.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.02340.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23638–23647, 2024. URL https://arxiv.org/abs/2404.00228.

Yazheng Liu, Yuxuan Wan, Rui Xu, Xi Zhang, Sihong Xie, and Hui Xiong. Attribution-guided continual learning for large language models. *arXiv preprint arXiv:2605.05285*, 2026.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*. Academic Press, 1989.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 121038–121072, 2024. URL https://arxiv.org/abs/2404.02948.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Yanyun Qu, Wensheng Zhang, Zhi Han, and Yuan Xie. Gradient projection for continual parameter-efficient tuning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9316–9329, 2025.

Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Ramanujan_Whats_Hidden_in_a_Randomly_Weighted_Neural_Network_CVPR_2020_paper.html.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2103.09762.

Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4548–4557, 2018. URL https://arxiv.org/abs/1801.01423.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PxoFut3dWW.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. 2020. URL https://arxiv.org/abs/2002.07376.

Chenxu Wang, Yilin Lyu, Zicheng Sun, and Liping Jing. Continual gradient low-rank projection fine-tuning for LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025. URL https://aclanthology.org/2025.acl-long.721/.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. O-LoRA: Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671, 2023. URL https://arxiv.org/abs/2310.14152.

Jing Xu and Jingzhao Zhang. Random masking finds winning tickets for parameter efficient fine-tuning. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2405.02596.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are Super Mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2311.03099.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995, 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. LoRA-FA: Efficient and effective low rank representation fine-tuning. *arXiv preprint arXiv:2308.03303*, 2023. URL https://arxiv.org/abs/2308.03303.

Algorithm 1 Masked continual fine-tuning with a frozen selection mask

Require: base model f_θ with frozen W_0; task stream $\{(\mathcal{D}_t)\}_{t=1}^{T}$; calibration set $\mathcal{D}$; budget k; rule $\in$ {gradient, uniform}
Ensure: frozen mask m; adapted adapter $W = W_0 + \alpha BA$

$\mathcal{S} \leftarrow$ the (A, B) LoRA coordinates ▷ trainable coordinates
if rule $=$ gradient **then**
  for each coordinate $i \in \mathcal{S}$ **do**
    $g_i \leftarrow \mathbb{E}_{x\sim\mathcal{D}}\left[|\partial\mathcal{L}(x)/\partial w_i|\right]$ ▷ Eq. equation 6
  end for
else
  $g_i \leftarrow \mathrm{Uniform}(0, 1)$ for each $i \in \mathcal{S}$
end if
$m \leftarrow \mathbb{1}[\, g_i \geq \tau_k \,]$ ▷ top-k rule, Eq. equation 7
freeze m ▷ shared by all tasks; never updated
$W \leftarrow W_0$ ▷ standard zero-init LoRA, $B = 0$
for $t = 1$ to T **do**
  train the coordinates with $m_i = 1$ on $\mathcal{D}_t$, updating only those
  evaluate $\mathcal{L}_j^{(t)}$ for all $j \leq t$ ▷ every task seen so far
end for
return m, and BWT from Eq. equation 2 over $\mathcal{P} = \{1, \ldots, T-1\}$

A The End-to-End Procedure

The notation is collected in Table 1 in the Methodology. Figure 1(a) shows the same procedure as a diagram; Algorithm 1 states it step by step.

B Attribution Cost

Table 3: Attribution cost on Qwen2.5-3B (BF16). Both columns are from a single *full-parameter* measurement run (`--include-frozen`, `--full-precision`); the exact correspondence and its limits are described below. Peak GPU is `torch.cuda.max_memory_allocated` converted to decimal GB (profiling fields `*_peak_alloc_mb` = 13168 / 7102 MiB, i.e. 13.8 / 7.4 GB); Time is the per-task attribution wall-clock recorded by the run (584.9 s gradient / 12.1 s Wanda).

Method	Peak GPU (GB)	Time (s)	Backward?
Gradient mask	13.8[†]	584.9	Yes
Wanda proxy	7.4	12.1	No
Shared random / RRM	**0**[‡]	**0**	No
No mask	0[‡]	0	No

[†] *Full-parameter* attribution configuration (`--include-frozen`, scoring all 3,085,938,688 base parameters); no method in this paper uses this configuration. The attribution used in the continual-learning experiments records a ∼0.73 GB transient allocation above the resident base model during its separate calibration pass (4-bit NF4, seq 512, 64 calibration samples, gradient checkpointing enabled; `overhead_mb` = 699.43 MiB). This is not an increase over the ordinary training backward peak; the removable costs are the additional calibration pass, its wall-clock compute, and its data requirement.
[‡] Zero by construction because no separate attribution pass is run, not a measured end-to-end peak-memory saving.

Both measured columns of Table 3 come from a single full-parameter attribution profiling run (`--include-frozen`, `--full-precision`), so they are internally consistent; that run scores all 3,085,938,688 base-model parameters, a configuration no reported method uses. We describe the memory figure first, then the time figure.

Gradient attribution requires 13.8 GB peak GPU memory for Qwen2.5-3B (`max_memory_allocated`, 13168 MiB → 13.8 GB in decimal units). This figure is from the full-parameter attribution configuration, which scores all 3,085,938,688 parameters of the base model. The peak comprises four components (the resident base-model weights, the forward and

backward activation graph over the full model, the full-parameter gradient accumulator, and the CUDA context), but these were not separately instrumented in the surviving run record, so we report the total rather than a measured decomposition. The backward pass does traverse the base model: attribution computes $\partial\mathcal{L}/\partial w$ over calibration samples, so both the forward and backward graphs span the full parameter space rather than the 29.9M-parameter LoRA surface. This configuration peaks at 13.8 GB, of which $\sim$90% is resident base-model weights (6.23 GB in BF16) plus the full-model gradient buffer (6.23 GB); these are costs the CL configuration does not pay. It is therefore an upper bound on a configuration no method uses, not a measure of attribution's own overhead. The CL attribution pass records a $\sim$0.73 GB transient allocation above the resident base model, but ordinary training also performs backward passes; this is not an incremental end-to-end peak-memory cost. The random baseline removes the separate calibration pass and its time/data costs, not the training backward peak (Table 5).

The 584.9 s per-task figure is the wall-clock of that same full-parameter run (`full_alora_time_s` = 584.9 s). As a checkpointing-off contrast, an earlier LoRA-only attribution profiling run (4-bit, seq 1024, 256 calibration samples, gradient checkpointing disabled) reports 3,512 MiB of overhead, a different regime (checkpointing off, longer sequences) that should not be combined with the 0.73 GB figure above.

C Pruning Saliency Is Orthogonal to Gradient Importance

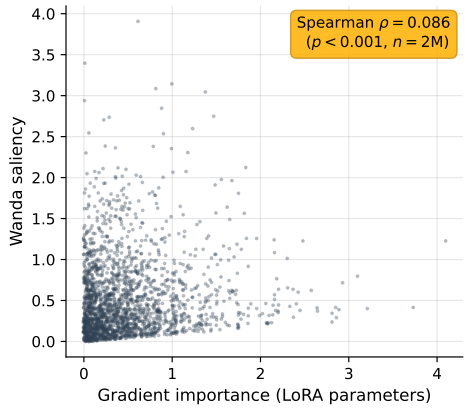


Figure 5: Spearman $\rho = 0.086$ between Wanda and gradient importance (LoRA parameters) on Qwen2.5-3B. These metrics rank parameters nearly independently.

Spearman correlation between Wanda ($|W| \times \text{RMS}(X)$) and gradient importance ($|\partial\mathcal{L}/\partial w|$) across 2M sampled parameters: (Figure 5; $\rho = 0.086$, 95% CI $\approx [0.084, 0.088]$, n=2M; correlation-only, measured outside a continual-learning context).

D Confounded Null-Result Statistics

Table 4: Per-seed BWT breakdown (Qwen2.5-3B, rank-16 LoRA, 5 tasks).

Seed	Gradient BWT ↓	Random BWT ↓	No Mask BWT ↓
42	0.491	0.435	0.507
123	0.463	0.417	**0.409**
456	**0.388**	0.414	0.539
789	**0.357**	0.417	0.484
1024	0.486	0.458	0.479
Mean	0.437	**0.428**	0.484
Std	0.061	0.019	0.047

Table 5: Confounded comparison (Qwen2.5-3B, rank-16 LoRA, 5 tasks, n=5 seeds). The last column records whether an additional calibration forward–backward pass is required; its time and data costs are described in Appendix B. Across all tables, bold marks the better value in its column.

Method	BWT ↓	Loss ↓	Calibration pass?
Shared random	$\mathbf{0.428} \pm 0.019$	$\mathbf{1.940} \pm 0.021$	No
Gradient mask	0.437 ± 0.061	1.966 ± 0.037	Yes
No mask	0.484 ± 0.047	1.970 ± 0.033	No

Table 6: Within-`lora_B` comparison (Qwen2.5-3B, rank-16 LoRA, 5 tasks, 7 seeds). All comparisons are within-run (same method ordering and RNG state). Δ = random_B_only − Grad-Full; negative Δ indicates random-B achieves lower (better) BWT.

Seed	random_B_only BWT ↓	full_alora BWT ↓	Δ
42	**0.326**	0.491	−0.165
123	**0.320**	0.413	−0.093
456	**0.337**	0.356	−0.019
789	**0.346**	0.364	−0.018
1024	**0.336**	0.442	−0.106
2048	**0.367**	0.402	−0.035
4096	**0.331**	0.398	−0.067
Mean	$\mathbf{0.338} \pm 0.015$	0.409 ± 0.046	−0.072

A paired t-test on per-seed BWT (Table 4; random vs. gradient: seeds 42/123/456/789/1024 yield differences $+0.056$, $+0.046$, -0.026, -0.059, $+0.028$) gives $t(4) = 0.41$, $p = 0.70$ (two-tailed), Cohen's $d_z = 0.18$ (small effect). The direction is seed-dependent, consistent with no systematic advantage for either method. Normalizing BWT per task by zero-shot loss ($\Delta L/L_0$) does not change the method ranking. Tables 5 and 6 report different `full_alora` BWT for the same seed because they come from independent runs; the absolute BWT level varies across sessions (§F).

E Within-lora_B Comparison (Density-Unmatched)

The within-`lora_B` comparison matches the parameter pool (both arms draw from `lora_B`) but *not* the density: the gradient mask activates approximately $2\times$ the B parameters of random-B-only. It is therefore an intermediate, density-unmatched step; the density-matched control (D1) is the headline comparison in the main text. All comparisons are within-run (same method ordering and RNG state); Δ = random_B_only − Grad-Full, negative Δ favors random-B. Absolute BWT values carry session-level variation and should not be compared across seeds (§F).

F Variation Across Runs and Method-Order Variation

Three quantities in this paper are sometimes read as competing estimates of one noise level. They are not: each measures a different thing at a different granularity, and the table below separates them.

(a) Same process, protocol split. Running each method in its own process reproduces the multi-method BWT to within 9.6×10^{-5}, an order of magnitude below the pre-specified 10^{-3} threshold. This is a check on the protocol implementation, not on run-to-run variation of the model: two code paths that should agree do agree. The paired t tests instead use the across-seed standard deviation of the within-session paired contrasts, $\mathrm{SD}(\Delta_s) = 0.071$ for D1.

(b) Same session, arm order swapped. Within one D1 session the two arms run back to back, and reversing their order is a genuine perturbation of the run. At seed 42 the `gradient_B` arm's BWT moves by 0.064 between the two orders; at seeds 789 and 2048 both arms stay within 1.4×10^{-2}. This is a within-session positional effect, and the within-run contrast survives it at all three tested

Table 7: Distinct quantities that are sometimes all called run-to-run *noise*. The paired tests use the across-seed SD of the within-session contrasts; the protocol-equivalence discrepancy is an implementation check and does not enter inference. Position and cross-session effects are reported as sensitivities rather than as interchangeable error estimates.

Level	What varies	Magnitude	What it measures
(a) protocol-equivalence check	execution path only	9.6×10^{-5}	numerical agreement between two code paths; not an inferential error term
(b) across-seed paired contrast	independent seed	$\mathrm{SD}(\Delta_s) = 0.071$	variability used by the paired significance test
(c) position (ordering) sensitivity	position of the two arms	0.064 (seed 42, `gradient_B`)	method-position interaction in a three-seed check
(d) cross-session variation	session, hardware, and method	up to 0.120 (`full_alora`, seed 42)	method-specific offset of unknown origin

Table 8: Method-ordering sensitivity (D1 density-matched, three seeds). Δ_{pos} is the BWT difference between running at position 1 and position 2. The last two columns express each arm's positional shift as a fraction of the main D1 contrast ($|\Delta| = 0.155$).

Seed	$\lvert\Delta_{\text{pos}}(\text{rB})\rvert$	$\lvert\Delta_{\text{pos}}(\text{gB})\rvert$	rB / main	gB / main
42	1.5×10^{-3}	6.4×10^{-2}	0.01	0.41
789	3.1×10^{-5}	6.2×10^{-3}	0.00	0.04
2048	8.4×10^{-6}	1.3×10^{-2}	0.00	0.08

seeds. Because this sensitivity analysis covers three rather than seven seeds, it does not establish the population distribution of the position effect.

(c) Cross-session, same seed. The absolute BWT of a fixed method and seed is not reproducible across sessions. For seed 42, `full_alora` records 0.491, 0.378, and 0.498 in three separate runs (spread 0.120), while `random_B_only` records 0.326, 0.324, and 0.278 (spread 0.048). The spread is method-dependent, so it does not act as a constant offset that cancels in a comparison.

Method position can shift the absolute BWT of Gradient-B, most notably at seed 42, but reversing the execution order does not reverse the Random-B–Grad-B contrast in any of the three tested seeds. (Table 8)

Table 9: Absolute loss-based BWT for the same seed 42 under identical method definitions, across sessions and hardware. Run C is the D1 density-matched session and Run D the uniform-B suppression control (the M2 controller) session, both on a separate GPU and software stack; `random_A+B` was not run in either. A Run-D entry exists only for methods Run D ran, and it did not train a plain `full_alora` arm, so that cell is `---` and its gradient-side arm is `gradient_B`, entered on its own row. Spread is $\max - \min$ across the runs in which the method appears.

Method	Run A	Run B	Run C	Run D / spread
Gradient mask (`full_alora`)	0.491	0.378	0.498	— (0.120)
Gradient-B@10% (`gradient_B`)	—	—	0.453	0.423 (0.030)
Random mask (`random_A+B`)	0.435	0.349	—	—
Random B-only (`random_B_only`)	0.326	0.324	0.278	0.278 (0.048)

Why this is not called NF4 non-determinism. Quantization is the same in every run of Table 9, so it cannot by itself explain a cross-session spread while leaving a same-process protocol split at 9.6×10^{-5}. The cross-session spread is method-specific rather than a common shift shared by all arms. We therefore report it as a *method-specific cross-session offset of unknown origin* and do not attribute it to a mechanism. Two consequences follow. First, only within-run comparisons are used for any claim in this paper. Second, the `random_B_only` value that enters Table 2 is the D1 session's own 0.278, taken from the same session as its partner arm, not an average across sessions.

Attribution of the level (c) spread. The level (c) spread has no single cause. Method position is not such a cause within a session: `set_seed` is called before each method's model load and no random numbers are consumed between iterations, so the seed state entering training does not depend on position. Level (b) shows that position can still matter, and level (c) shows that session identity can. Which of these dominates across a full re-run is not tested here.

Therefore, absolute BWT levels should not be compared across sessions; all statistical claims in the paper use within-session paired contrasts.

G RRM Boundary Analysis

We investigated whether assigning each task a different random mask (Rotating Random Masks, RRM) provides additional benefit beyond shared random masking. RRM generates a deterministic per-task mask via $m_i^{(t)} = \mathbf{1}[\text{hash}(t\|i) \bmod 10 < k]$, reducing expected pairwise inter-task parameter overlap from 100% (shared mask) to k^2 (1% at 10% sparsity) (Table 10).

Table 10: RRM boundary analysis (Qwen2.5-3B, single seed per condition except rank-16 reference).

Condition	RRM BWT	Random BWT	Δ	Direction
Rank 4	0.464	0.423	+0.041	Random better
Rank 8	0.427	0.373	+0.055	Random better
Rank 16 (ref.)	0.401	0.435	−0.034	RRM better
Rank 32	0.414	0.390	+0.024	Random better
Reverse order	0.115	0.109	+0.006	Random better

RRM shows a favorable paired difference at rank 16 only. The overlap-reduction hypothesis predicts a consistent RRM advantage; the data do not support this across conditions. RRM's effect appears condition-dependent rather than robust.

H Sparsity Sweep

We sweep the masking budget over 5%, 10%, 20%, and 30% (single seed) to test whether the within-pool ordering is specific to the 10% sparsity used in the main experiments. Across all four levels, shared random achieves numerically lower BWT than gradient attribution, with no cross-over; the largest margin is at 30%.

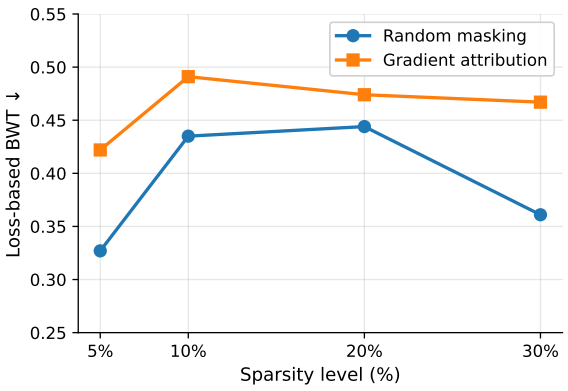


Figure 6: Sparsity sweep (Qwen2.5-3B, single seed). Shared random achieves numerically lower BWT than gradient attribution at all four tested levels.

Table 11: Sparsity sweep numerical values (Qwen2.5-3B, single seed). Random BWT $\leq$ Gradient BWT at all tested levels.

Sparsity	Random BWT ↓	Gradient BWT ↓
5%	0.327	0.422
10%	0.435	0.491
20%	0.444	0.474
30%	0.361	0.467

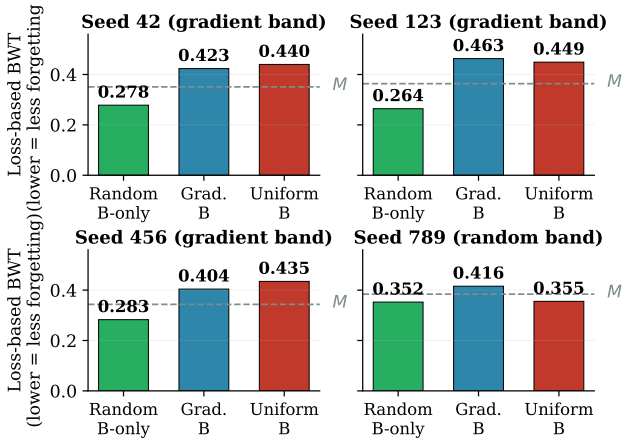


Figure 7: The suppression control's outcome is *seed-dependent*. All three arms train 1,651,508 parameters, which is 5.5% of the full LoRA pool (each arm draws 10% of the `lora_B` sub-pool, and B holds 55% of the pool's parameters), so this is the density-matched setting throughout. The uniform-B controller redistributes the gradient mask's budget uniformly across LoRA modules. At seed 42 it sits in the gradient band (0.440 above gradient-B's 0.423 and well above random-B-only's 0.278); seeds 123 and 456 also land on the gradient side; at seed 789 it falls to the random band (0.355 below gradient-B's 0.416, alongside random-B-only's 0.352). Three of four seeds fall on the gradient side. M2 is an independent session from D1, so the same-seed gradient-B values differ between them (e.g. seed 42: 0.423 here vs 0.453 in the D1 table; seed 789: 0.416 vs 0.387; §F). We therefore do not treat this ablation as establishing the `o_proj`-concentration mechanism (§6).

I Training Stability: Rank-8 No-Mask Divergence

The rank-8 no-mask ablation reveals a discrepancy between two endpoint-based forgetting summaries. Endpoint BWT is 0.176; average forgetting is 0.2837. Training produced a severe loss spike during task 4 (Math): evaluation losses on three prior tasks exceeded 14, compared to values below 4 for all masked conditions at the same rank. This spike subsequently recovered during task 5, yielding deceptively low endpoint BWT. We do not interpret the rank-8 no-mask endpoint BWT of 0.176 as evidence that no-mask outperforms masking.

The same signature recurs at rank 16 and is seed-dependent: at seed 123 the unmasked arm spiked during task 4 with evaluation losses of 16.2/16.0/14.7 on the three prior tasks (code/medical/legal), while every masked arm in that run stayed below 2.5; at seed 42 the unmasked arm did *not* spike (task-4 evaluations $\leq$ 1.2). The instability is therefore a property of the unmasked (all-parameter) configuration rather than of a particular rank, and it appears in a minority of seeds.

J Mask Concentration by Module

The o_proj suppression ablation and module-concentration figures, discussed in §6, are reproduced below together with the module-concentration table for reference.

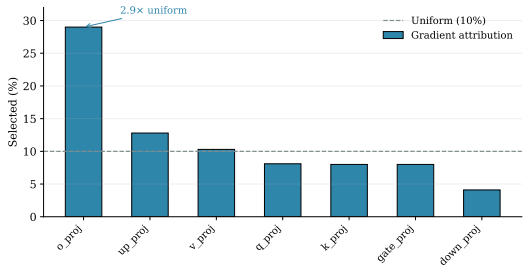


Figure 8: Gradient-mask concentration by LoRA module (Qwen2.5-3B). Gradient attribution selects 29% of `o_proj` parameters (2.9× the 10% uniform rate) and is under-represented in `down_proj` (4.1%); the dashed line marks uniform 10% selection.

Table 12: Gradient mask concentration by LoRA module (Qwen2.5-3B). Random masking distributes uniformly at 10% per module. Gradient attribution selects exclusively `lora_B` parameters (all `lora_A` receive zero importance due to $B = 0$ initialization).

Module	Gradient density	Random density	Ratio
o_proj	29.0%	10.0%	2.9×
up_proj	12.8%	10.0%	1.3×
v_proj	10.3%	10.0%	1.0×
q_proj	8.1%	10.0%	0.8×
k_proj	8.0%	10.0%	0.8×
gate_proj	8.0%	10.0%	0.8×
down_proj	4.1%	10.0%	0.4×

K Generalization and Boundary Checks

Scale Observations We additionally observe the within-pool comparison at three parameter scales (0.5B, 1.5B, 7B; single seed each). These are observational rather than multi-seed, and at 0.5B and 1.5B only the shared-random and no-mask conditions were run, so no gradient attribution baseline is available for a direct comparison there.

Table 13: BWT at additional scales (single seed, observational).

Scale	Random BWT ↓	Gradient BWT ↓	No Mask BWT ↓
0.5B	0.343	—	0.431
1.5B	0.309	—	0.356
7B	0.447	0.441	0.514

Task Ordering (single seed) A reversed-order experiment yields BWT $\approx$ 0.11 for all masking methods, compared to $\approx$ 0.42 under the standard order, a nearly 4× reduction that far exceeds the method differences measured within this experiment (within-session method contrasts of order 10^{-2}). This is a *single seed*, so the 4× magnitude is a qualitative pointer, not a quantitative estimate; it is reported to show that task ordering dominates the forgetting landscape, not as a measured effect size. The conclusions of this paper apply only to the tested Code→Medical→Legal→Math→Creative sequence; additional task orderings with multi-seed replication are needed before the null result can be generalized.

Cross-Architecture Comparison Both cross-architecture designs are tabulated in Table 14: the confounded design (gradient selecting B-only, random selecting A+B) and the within-`lora_B` design (both methods restricted to `lora_B`). The two designs are not comparable row-for-row because the Phi-2 within-B rows use a different LoRA target set; §6.1 discusses both.

Table 14: Cross-architecture comparison on Qwen2.5-3B and Phi-2 (2.78B, GPT-NeoX-style), with both arms within a row sharing one target set. Qwen2.5-3B targets `q/k/v/o/gate/up/down_proj` (pool 29,933,568), whereas Phi-2 uses GPT-NeoX-style `q,k,v,fc1,fc2,dense` projections (pool 23,592,960) in the confounded design. In the within-`lora_B` design, the Phi-2 rows omit `dense` (pool 20,971,520, 12.5% smaller), so the two Phi-2 designs are not comparable row-for-row. The within-`lora_B` rows are matched on pool but *not* density (the gradient mask activates approximately $2\times$ the B parameters of Random-B), as on Qwen2.5-3B.

Architecture	**Comparison**	**Grad. BWT ↓**	**Rand. BWT ↓**	Δ
Qwen2.5-3B (n=5)	Confounded	0.437 ± 0.061	0.428 ± 0.019	-0.009
Phi-2 (n=5)	Confounded	0.158 ± 0.019	0.183 ± 0.016	$+0.025$
Qwen2.5-3B (n=7)	Within-B	0.409 ± 0.046	0.338 ± 0.015	-0.072
Phi-2 (n=4)	Within-B	0.149 ± 0.036	0.065 ± 0.017	-0.084

L In-Domain Calibration Ablation

The primary experiments use out-of-domain Alpaca calibration data (64 samples). To test whether this choice drives the null result, we repeat the gradient-mask experiment with *pooled in-domain* calibration: 60 samples drawn equally from the five CL tasks.[1] All other parameters are identical (Qwen2.5-3B, seed 42, rank 16, 10% sparsity).

Loss-based BWT is 0.495 vs 0.491 with Alpaca calibration ($\Delta = 0.004$, 0.8% relative), well within the 5-seed standard deviation of 0.061. Random and no-mask controls are numerically identical across both runs, confirming infrastructure determinism. The *cross-calibration* overlap (Alpaca vs pooled) is Jaccard $= 0.585$: changing the calibration domain alters 42% of selected parameters. Despite this substantial mask difference, BWT changes by only 0.004. This robustness is of the *outcome*, not the mask: the mask does change, but *among gradient-derived masks* the specific 10% selected has little influence on forgetting, whereas swapping the selection criterion entirely (gradient for random) does, as the density-matched control shows (§5.1).

M B=0 Restriction Asymmetry

The within-`lora_B` comparison varies the selection criterion while holding the parameter pool fixed. That the pool restriction is the right control to hold follows from an asymmetry between the two arms at seed 42 (Table 15). Restricting the gradient mask to `lora_B` changes nothing, because under $B = 0$ every `lora_A` importance score is already zero and the restriction is a no-op. The same restriction applied to the random arm removes the `lora_A` dilution and improves BWT by 0.025. A restriction that does nothing to one arm and helps the other is evidence that the two arms were drawing from different pools before the restriction was applied, which is what makes the pool-matched comparison the informative one.

Table 15: Effect of restricting the mask to `lora_B` (seed 42). For gradient attribution, the restriction is a no-op by construction (B=0 collapses every `lora_A` importance score to zero); for random, restricting to B removes `lora_A` dilution and improves BWT by 0.025. This asymmetry demonstrates that the within-B comparison varies the selection criterion, not the parameter pool.

Method	**Unrestricted BWT**	**B-only BWT**	**Gap**
Gradient	0.378	0.378	$+0.000$
Random	0.349	0.324	-0.025

N Per-Seed Accuracy and Matched Loss Contrast

The accuracy evaluation is reported in the main text, §5.4, together with the option-scoring protocol; its per-seed values and the matched loss contrast are tabulated below. (Table 16)

[1] $\lfloor 64/5 \rfloor \times 5 = 60$; a 6.25% size difference from integer division.

Table 16: Per-seed results, both columns in the direction [Grad − Random] (positive = gradient forgets more). Col. 2 is the accuracy-BWT difference: the difference in mean final-minus-learned accuracy over the prior discrete-option tasks, not a difference of final accuracies. It is therefore a retention quantity on the same footing as the loss column and does not mix learning with retention. Col. 2 is reported in percentage points and is measured in the A800 accuracy runs; Col. 3 is the loss-based BWT difference from the A800 loss runs. Together they form a second environment relative to the original V100 loss runs. Col. 2 carries the 95% t-interval on the seven-seed mean. Col. 3 has mean $+0.152$ (SD 0.0749, $t = 5.37$, CI $[+0.083, +0.221]$, 7/7 same direction). The V100 loss contrast is $+0.155$ under the same [Grad − Random] convention. Across matched seeds, the A800-minus-V100 loss-contrast difference has mean -0.0033 and SD 0.0441, so individual seeds move between environments even though the average does not.

Seed	**Accuracy-BWT Δ [Grad − Rand], pp**	**Loss BWT Δ [Grad − Rand]**
42	$+2.00$	$+0.2320$
123	0.00	$+0.1647$
456	$+0.48$	$+0.1300$
789	-6.98	$+0.0786$
1024	$+4.60$	$+0.0514$
2048	$+0.51$	$+0.2572$
4096	$+6.00$	$+0.1492$
Summary (column 2 in percentage points, column 3 in BWT units)		
Mean	$+0.94$ pp	$+0.152$
SD	4.17 pp	0.0749
95% CI	$[-2.91, +4.80]$ pp	$[+0.083, +0.221]$

The interval is wide because the per-seed differences are wide, not because the mean is large. With the observed across-seed standard deviation of 4.17 pp, the 95% interval half-width is about 3.9 pp at $n = 7$ and would fall below 2.0 pp only at roughly twenty seeds.

O The Math Task: Truncation Audit

Loss on math decreases with training while accuracy on the same task collapses to zero for all arms: peak accuracy at math's own training phase is 0.04 and every arm ends at 0.00. The numeric-match function accepts a bare number, a number with surrounding text, and a complete solution, and the prompt prefixes are identical; nevertheless, the interaction between output format and the 256-token limit makes the generation-based score unreliable. A generation audit reports 56–95% of generations across all tasks and arms reaching the cap without an end-of-sequence token, and for math specifically 81–90% of generations reach the cap. The last number in a truncated solution is often an intermediate quantity, so the reference answer is absent from the value the scorer inspects. Every arm shares the same cap and scorer, so the bias is common rather than arm-specific. We do not treat the math column as a forgetting result: under option-based scoring math yields no score, and under generation scoring it is confounded with truncation.

P Positioning Against Gradient-Attribution Families

Table 17 places our contribution against the gradient-attribution families we are positioned against, on the two axes that drive the study.

Table 18 lists the selection rules this paper is positioned against and marks which we train under one protocol. Only the rules we train contribute empirical numbers here; the rest are compared by their published design, with none of their reported numbers presented as a measurement of ours.

Q D1 Per-Task and Per-Seed Decomposition

Table 19 gives the same quantity for each seed, so the task-level pattern can be checked against the seed-level one; the main text reports the same decomposition as a figure (Figure 3). Each entry is

Table 17: Positioning against gradient-attribution families. "B=0 pool" marks a method whose attribution is computed at a zero-initialized LoRA, where the gradient with respect to `lora_A` is zero and the mask is confined to `lora_B`; "within-pool grad-vs-random" marks a method compared against a matched random baseline drawn from the same pool. Attribution-Guided CL computes its attribution from a separately trained model ($B \neq 0$ at scoring time) and gates gradients continuously rather than selecting a binary mask, so it is outside the first axis by construction and unmatched on the second.

Method	B=0 pool	Within-pool grad-vs-random
EWC / SI / MAS	No	No
PiSSA ($B \neq 0$)	N/A by init	No
Attribution-Guided CL (post-training attribution)	N/A by design	No
Ours (within-`lora_B`)	**Yes**	**Yes**

Table 18: Selection rules this paper is positioned against. The last column marks the four rules we train and evaluate under one protocol (§3.3); the rest are compared by design only, and none of their published numbers is cited as a result of ours.

Rule	What it scores, and what for	Run
EWC (Kirkpatrick et al., 2017); MAS (Aljundi et al., 2018)	Fisher information, output sensitivity; a penalty, not a mask	no
SNIP (Lee et al., 2019); GraSP (Wang et al., 2020)	gradient at initialisation; one-shot pruning mask	no
Wanda (Sun et al., 2024)	weight×activation magnitude; one-shot pruning mask	no
LoRA-FA (Zhang et al., 2023)	none, structural; freezes `lora_A`, trains `lora_B`	no
Grad-Full, Grad-B (§3.3)	calibration gradient magnitude; static CL mask, full pool (Grad-Full) or `lora_B` at 10% (Grad-B)	**yes**
Random-B; Random A+B (Xu & Zhang, 2024)	none, uniform; static CL mask, `lora_B` at 10% (Random-B) or full pool (A+B)	**yes**

loss_matrix$[4][j]$ − loss_matrix$[j][j]$ for prior task j; the final column is the mean over the four tasks, which is the per-seed aggregate Δ of Table 2.

Table 19: Per-seed, per-task D1 decomposition (Δ = random-B − gradient-B for each prior task). Negative favours random-B. Legal is the only task whose sign is not consistent across seeds.

Seed	Code	Medical	Legal	Math	Aggregate
42	−0.040	−0.409	−0.002	−0.249	−0.175
123	−0.002	−0.698	−0.002	−0.052	−0.188
456	−0.074	−0.381	−0.004	−0.033	−0.123
789	−0.063	−0.054	+0.001	−0.023	−0.035
1024	−0.044	−0.410	−0.011	−0.021	−0.121
2048	−0.041	−0.950	+0.010	−0.062	−0.261
4096	−0.059	−0.568	−0.014	−0.091	−0.183
Mean	−0.046	−0.496	−0.003	−0.076	−0.155

R Ruling Out a Learning Deficit in the D1 Gap

The D1 density-matched comparison (both arms 10% of `lora_B`, 1,651,508 parameters) shows random-B-only achieving lower BWT than gradient-B. One rival reading is that gradient-B's top-10% is the more concentrated half of the original 20% mask, so gradient-B learns each task less

well and the gap is a learning deficit rather than a retention difference. The per-task diagonal losses (each task's loss immediately after it is learned) exclude this: gradient-B's diagonal loss is lower than random-B-only's in *all* 35 seed×task cells (7 seeds × 5 tasks), with no exceptions.

Table 20: Diagonal loss (mean ± std over the seven D1 seeds), by task. Lower is better. gradient-B is lower than random-B-only in every cell (Table 20); Grad-Full is the uncontrolled reference. Loss diagonals only; the D1 run records contain no accuracy matrix (accuracy is measured separately in the accuracy branch, §5.4).

Task	random_B_only	gradient_B	full_alora
code	0.952 ± 0.111	$\mathbf{0.901 \pm 0.100}$	0.898 ± 0.099
medical	1.190 ± 0.040	$\mathbf{1.152 \pm 0.040}$	1.144 ± 0.035
legal	0.388 ± 0.035	$\mathbf{0.368 \pm 0.038}$	0.365 ± 0.037
math	3.838 ± 0.103	$\mathbf{3.652 \pm 0.096}$	3.642 ± 0.100
creative	2.104 ± 0.053	$\mathbf{2.067 \pm 0.053}$	2.056 ± 0.054

A second rival reading is that BWT, being a difference of losses, is inflated for gradient-B simply because it starts from a lower per-task loss. The endpoint closes this from both sides. Because BWT is the mean over prior tasks of (final loss − learned loss) and the start is the mean learned loss over those same tasks, the endpoint is exactly the mean final held-out loss over the first four tasks, read directly from the final row of the stored loss matrix rather than reconstructed. gradient-B starts *lower* yet ends *higher* in 6 of 7 seeds, so its larger BWT reflects greater forgetting rather than a lower baseline (Table 21). Measured on the same seven within-session pairs, the endpoint difference averages $+0.0813$ (SD 0.0710; paired $t(6) = 3.03$, $p = 0.023$, 95% CI $[+0.016, +0.147]$, Cohen's $d_z = 1.15$). The second environment agrees in both mean and direction: $+0.078$ ($t(6) = 2.62$, $p = 0.039$, 6/7), a mean separation of 0.003 from the V100 estimate. Thus Random-B's advantage is not confined to a BWT normalization: it also achieves lower final held-out loss on the prior tasks.

The endpoint significance is less robust than the headline BWT contrast to single-seed deletion. Three of the seven leave-one-out fits exceed $p = 0.05$: dropping seed 42 or 123 gives $p = 0.058$, and dropping seed 4096 gives $p = 0.059$, all with a mean of about $+0.076$. The remaining four deletions give $p \leq 0.048$. We therefore treat the endpoint as a directional result supported by the paired design rather than as an effect that survives arbitrary single-seed removal; the density-matched BWT contrast on the same seven pairs is robust to every single-seed deletion ($p \leq 0.005$ throughout).

Table 21: Mean final held-out loss over the four prior tasks, per seed, read from the final row of the stored loss matrix. Start is the same quantity at each task's own training step; end is at the end of the sequence. gradient-B's endpoint is higher in 6 of 7 seeds (seed 789 is the exception); paired $t(6) = 3.03$, $p = 0.023$, mean $+0.081$ on the V100, and $+0.078$ ($p = 0.039$) on the A800. Three of the seven single-seed deletions fall above $p = 0.05$ (seeds 42, 123, 4096); see text.

Seed	rB start	rB end	gB start	gB end (diff)
42	1.543	1.821	1.478	1.931 (+0.110)
123	1.632	1.896	1.553	2.006 (+0.110)
456	1.575	1.858	1.494	1.900 (+0.042)
789	1.548	1.900	1.475	1.862 (−0.038)
1024	1.566	1.918	1.489	1.962 (+0.045)
2048	1.665	1.896	1.587	2.078 (+0.182)
4096	1.618	1.913	1.553	2.031 (+0.118)

Both checks are loss-based (accuracy is measured separately in the accuracy branch, §5.4), use a single task order, and cover Qwen2.5-3B only. The endpoint is read directly from the final row of the stored matrices, so its per-seed differences are load-bearing rather than derived; the caveat above concerns the $n = 7$ fragility of the full-set significance, not the definition of the quantity.

Table 22: Density-matched D1 replication on Phi-2, four seeds. Both arms select the same number of coordinates from the same `lora_B` pool; Δ = Random-B − Grad-B in loss-based BWT, so negative values mean the random mask forgets less.

Seed	Random-B BWT ↓	Grad-B BWT ↓	Δ
42	**0.0397**	0.1661	−0.1264
123	**0.0363**	0.1273	−0.0910
456	**0.0461**	0.1749	−0.1289
1024	**0.0480**	0.1624	−0.1144
Mean	—	—	−0.1152 (SD 0.0173)

Table 23: Density-matched D1 contrast on both architectures under the same sign convention, Δ = Random-B − Grad-B (negative means Random-B forgets less). Phi-2 is an architecturally distinct four-seed replication of the seven-seed Qwen2.5-3B result.

Architecture	n	Mean Δ	SD	t	p
Qwen2.5-3B	7	−0.1552	0.0707	−5.80	0.0011
Phi-2	4	−0.1152	0.0173	−13.32	0.0009

S Cross-Architecture Density-Matched Replication (Phi-2)

The density-matched contrast of §5.1 is also run on Phi-2 (2.78B, GPT-NeoX-style), an architecturally distinct family from Qwen2.5-3B. As in the primary D1 comparison, both arms select from `lora_B` alone and use the same density: each activates exactly 1,048,576 coordinates, which is 10% of the 10,485,760-coordinate `lora_B` sub-pool (and 5.0% of the 20,971,520 trainable $A{+}B$ pool). The candidate pool and update budget are therefore matched, leaving only the ranking rule different between Random-B and Grad-B.

The direction replicates in all four seeds (Table 22). The mean paired contrast is -0.1152 (SD 0.0173), with paired $t(3) = -13.32$, $p = 0.0009$, and 95% CI $[-0.143, -0.088]$. A two-sided sign test over four concordant seeds gives $p = 0.125$; because $n = 4$ is too small for a strong distributional claim and the normality assumption behind the paired t-test is weak, we treat the t statistic as a supporting parametric summary rather than resting the replication claim on it. The replication claim is instead based on the density-matched design and the 4/4 directional agreement.

Table 23 places the clean Phi-2 replication beside the primary Qwen2.5-3B result under the same sign convention. The task-level decomposition also resembles the primary model rather than merely sharing the aggregate sign. Averaged over the four Phi-2 seeds, Medical contributes 67.2% of the aggregate contrast, Math 24.1%, Code 5.8%, and Legal 2.9%; Medical is the largest contributor in every seed. On Qwen2.5-3B, Medical contributes 79.9% of the matched contrast. Thus the two architectures agree on where most of the effect is concentrated, although the concentration is somewhat less extreme on Phi-2.

The replication uses the same five task domains, LoRA rank, calibration set, and 10% `lora_B` budget as D1. It differs in model family, in the Phi-2 LoRA target set (which omits `dense`, giving a 20,971,520-parameter trainable pool), and in the number of seeds. The earlier Phi-2 within-B comparison in Appendix K remains a density-unmatched boundary check; the results here are the clean cross-architecture counterpart to the headline density-matched D1 comparison.