SIFT-LORA: CHOOSING WHAT A LAYER-SPARSE ADAPTER INHERITS

Anonymous authors
Paper under double-blind review

ABSTRACT

A trained LoRA adapter should be reusable across deployment storage budgets chosen after task adaptation. We introduce SIFT-LoRA, which compiles a same-task Full-LoRA parent into a layer-sparse adapter under an exact serialized-byte cap without further training or changing the retained weights. A layer's value can depend on which other updates are retained, so fixed importance scores may miss its contribution to the sparse child. Our selector, Conditional SIFT-KL, evaluates how much each candidate layer reduces parent–child predictive KL when added to the current child. It chooses the feasible addition with the largest positive reduction per added serialized byte, then recomputes these reductions for the remaining candidates. Across three models and three tasks at 12.5% and 25% of the full adapter's serialized size, Conditional SIFT-KL lowers predictive KL against each of the fixed-score and contiguous-layer baselines in 16 of 18 conditions. It also achieves the highest task utility among the three selectors in 14 of 18 conditions, including ties. On Qwen3.5-2B/HellaSwag at the 12.5% cap, it improves mean accuracy over fixed scoring by 3.0 percentage points across three independently trained parents at nearly identical adapter sizes. In a separate seven-method BoolQ comparison at the 12.5% cap under unmerged bf16 H100 inference, SIFT-LoRA lies on the observed utility–latency Pareto frontier for all three models. Broader deployment experiments cover five tasks and both byte budgets. These results support choosing layers by what they add to the retained set and separating task learning from deployment capacity.

1 INTRODUCTION

Foundation models make it possible to reuse one pretrained model across many tasks. For large language models (LLMs), however, fine-tuning and storing a separate model for each task is costly. Parameter-efficient fine-tuning addresses this problem by learning a small set of task-specific parameters while keeping the pretrained backbone fixed (Houlsby et al., 2019). LoRA represents these changes as low-rank weight updates that can be stored separately from the shared model (Hu et al., 2022). The same backbone can then support different tasks by loading different adapters. This makes adaptation more economical, but leaves a deployment question: which learned updates should be retained when the storage budget is chosen after task adaptation?

Existing work allocates rank or selects adapter placements during training (Zhang et al., 2023; Veprikov et al., 2026). We study post-training adapter compilation: a same-task Full-LoRA parent, with LoRA modules in every Transformer layer, is already trained, and the deployment budget is specified afterward. The goal is to select which learned layer updates a smaller child should inherit under an exact serialized-byte cap, without retraining or changing the retained factors. This separates task adaptation from the later choice of deployment capacity.

A layer's contribution can depend on the updates retained alongside it, so a fixed ranking may miss its value in the sparse child. Chen et al. (2025) select a lower-layer prefix by validation utility, LoRA-Drop uses local output energy (Zhou et al., 2025), and LoRA-Squeeze and PARA reduce rank after tuning (Vulić et al., 2026; Kumaravelu et al., 2026). We instead measure each whole-layer addition through the resulting child's predictive fidelity to the parent, allowing non-contiguous layers and reassessing their value within the retained set.

We introduce SIFT-LoRA, which selects layers under an exact serialized-byte cap without further training (Figure 1). Conditional SIFT-KL compares the child's next-token predictions with the

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.

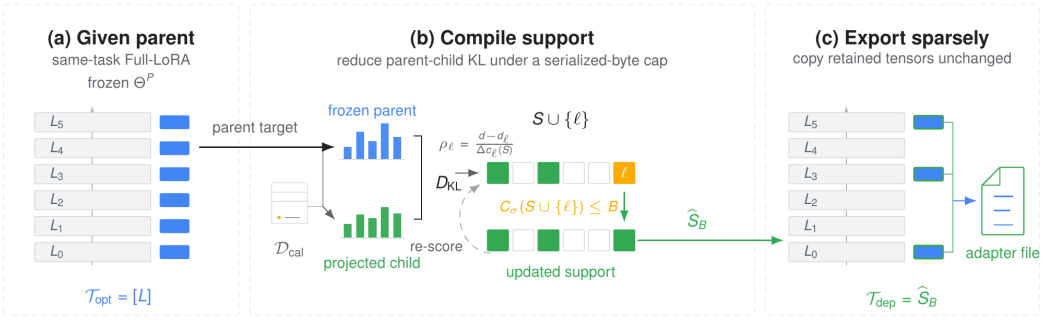


Figure 1: **SIFT-LoRA compiles a trained Full-LoRA parent into a smaller layer-sparse adapter without retraining.** Starting from a frozen same-task parent, Conditional SIFT-KL selects byte-feasible whole-layer updates by their marginal reduction in parent–child predictive KL, rescoring after each addition. The exporter then keeps the selected factors unchanged and removes the rest, producing a physical adapter that fits the requested serialized-byte budget.

parent's on fixed calibration contexts using KL divergence (Hinton et al., 2015). Starting with no retained layers, it repeatedly chooses the feasible addition with the largest positive marginal KL reduction per added serialized byte and rescores the remaining candidates. Export preserves selected factors, omits unselected tensors, and includes loading metadata; the shared backbone is excluded from the budget.

To test the value of rescoring, we compare Conditional SIFT-KL with harmonized Static-KL, which freezes its initial scores while sharing the parent, calibration data, byte cap, and evaluation procedure. Boundary-KL provides a separate control restricted to contiguous windows. The controlled selector evaluation covers Qwen3.5-2B, Llama-3.1-8B-Instruct, and Gemma-3-4B-Instruct on BoolQ, HellaSwag, and GSM8K at 12.5% and 25% of the full adapter's serialized size. Conditional SIFT-KL achieves lower predictive KL in 16 of 18 conditions against each control and the highest task utility among the three selectors in 14, including ties (§5.1).

Across three independently trained Qwen3.5-2B/HellaSwag parents at the 12.5% cap, conditional selection improves accuracy over fixed scoring for every parent, with a mean gain of 3.0 percentage points at nearly identical adapter sizes (Appendix C.3). This connects conditional rescoring to downstream benefit beyond the predictive-KL objective.

We also measure deployment trade-offs in a separate seven-method BoolQ/12.5% comparison under unmerged bf16 H100 inference (§5.2). SIFT-LoRA lies on the observed utility–median-inter-token-latency (ITL) Pareto frontier for all three models. Relative to the fastest budget-compliant alternative attaining at least its accuracy, it reduces median ITL by 27.5–42.2%, with 1.47–3.49 percentage points lower accuracy. We further extend the deployment evaluation across five tasks and both 12.5% and 25% budgets: SIFT-LoRA lies on the observed seven-method utility–ITL frontier in 24 of 30 model–task–budget panels, while fresh Conditional SIFT-KL matches or exceeds Static-KL utility in 28 of 30 cells. Together, these results show that the deployment trade-offs extend beyond the primary BoolQ setting across models, tasks, and serving budgets.

Our contributions are:

- **Train once, compile for later deployment budgets.** We formulate post-training adapter compilation from a frozen same-task Full-LoRA parent, allowing deployment capacity to be chosen after task adaptation without training a new sparse child for each budget. Compilation operates under the serialized size of the artifact that will actually be deployed.
- **Choose inherited layers in context.** Conditional SIFT-KL recomputes each candidate layer's marginal reduction in parent–child predictive KL per added serialized byte within the current child. Controlled comparisons isolate the benefit of conditional rescoring over fixed-score and contiguous-support selection.
- **Reuse learned state, not only sparse support.** Post-training compilation preserves the learned state on the selected support rather than reducing the problem to support discovery alone. Matched-support controls show that this inherited state provides a stronger initialization and higher early learning-curve utility, while the retrofit study quantifies the additional optimization required to recover the inherited starting point once the parent already exists.

2 Related Work

Efficient LoRA adaptation. Training-time methods allocate or prune LoRA rank (AdaLoRA, SoRA, NormAL LoRA) or learn adapter placements (IST, Flexora, FLoE, WeightLoRA) (Zhang et al., 2023; Ding et al., 2023; Aastik et al., 2025; Yao et al., 2024; Wei et al., 2025; Wang et al., 2025; Veprikov et al., 2026). SIFT-LoRA instead selects learned updates from an already trained Full-LoRA parent for deployment.

Post-hoc adapter reduction. LoRA-Squeeze's Post-Squeeze reconstructs lower-rank factors (Vulić et al., 2026); PARA allocates rank spectrally and can remove negligible modules (Kumaravelu et al., 2026). Chen et al. (2025) remove upper-layer LoRA modules by selecting a lower-layer boundary from validation performance, while SPLoRA uses safety-alignment weight-space distances for adapter removal (Ao et al., 2025). LoRA-Drop scores modules from their local output energy and shares low-importance modules; its no-share variant removes them, and its procedure includes preliminary updates and further adaptation (Zhou et al., 2025).

SIFT-LoRA selects learned updates for a budget specified after task adaptation, preserving the retained parent factors. Within this setting, Conditional SIFT-KL evaluates each complete child formed by a whole-layer addition and recomputes marginal parent-to-child predictive-KL reduction per added serialized byte as the support changes. This differs from the validation-selected lower prefix of Chen et al. (2025) and LoRA-Drop's local output-energy scoring; harmonized Static-KL is our matched fixed-score control. Selection and physical export use the same serialized-byte constraint, linking support selection to the size of the saved adapter (Section 3).

3 Method

Given a frozen same-task Full-LoRA parent Θ^P, calibration set $\mathcal{D}_{\text{cal}}$, budget B, and serializer σ, SIFT-LoRA selects whole-layer updates and physically exports retained weights unchanged. Parent training is outside compilation; no child weights are trained.

3.1 Learned State and Receiving Support

We separate learned values from the layers that receive them. Let W_0 be a frozen Transformer with layer set $[L]$, and let $\mathcal{M}_\ell$ contain the LoRA-adapted linear modules in layer ℓ; artifact records use zero-based indices. Module m has factors $A_{\ell m}$ and $B_{\ell m}$, rank $r_{\ell m}$, and scaling $\alpha_{\ell m}/r_{\ell m}$. Gate g_ℓ turns every prescribed LoRA branch in that layer on or off:

$$\Delta W_{\ell m}(\Theta, g) = g_\ell \frac{\alpha_{\ell m}}{r_{\ell m}} B_{\ell m} A_{\ell m}. \tag{1}$$

Support $S = \{\ell : g_\ell = 1\}$ retains every prescribed factor pair $\{A_{\ell m}, B_{\ell m} : m \in \mathcal{M}_\ell\}$ in each selected layer; selected layers need not be contiguous.

We write the checkpoint restricted to S as $\mathcal{P}^S(\Theta)$. Branches outside S are inactive; those inside retain their original values, ranks, and scaling. Thus $\mathcal{P}^S(\Theta) \in \mathcal{H}_S$, the sparse hypothesis class for S. This logical child is used during selection; Section 3.3 describes its physical export.

Compilation changes support from $\mathcal{T}_{\text{opt}} = [L]$ to $\mathcal{T}_{\text{dep}} = \hat{S}_B$, while leaving the dense backbone and all retained factors unchanged; none is compressed or re-optimized.

3.2 Conditional Fidelity under a Size Budget

Measuring adapter size. We measure serialized size, not layer or parameter counts. The canonical serializer σ fixes format, tensor dtype and names, required loading metadata, and alignment. It writes retained adapter tensors and support/configuration metadata, excluding masked checkpoints and the shared backbone:

$$C_\sigma(S) = \left|\text{Serialize}_\sigma\left(\mathcal{P}^S(\Theta^P)\right)\right|_{\text{bytes}}. \tag{2}$$

For fractional budget b, $B = \lfloor b\, C_\sigma([L]) \rfloor$, using the full-parent adapter saved with the same schema, dtype, and required metadata as the denominator. Different supports may leave part of the common cap unused.

Measuring agreement with the parent. We compare each complete child's next-token predictions with the parent's under the same teacher-forced calibration contexts. This evaluates retained updates together rather than a layer's local output alone, and full distributions retain information discarded by top-1 agreement (Hinton et al., 2015).

SIFT-KL uses the parent-to-child Kullback–Leibler divergence (Kullback & Leibler, 1951):

$$\mathcal{J}_{\mathrm{KL}}(S;\Theta^P) = \frac{\tau^2}{|\mathcal{D}_{\mathrm{cal}}|} \sum_{x\in\mathcal{D}_{\mathrm{cal}}} \frac{1}{|\mathcal{I}(x)|} \sum_{j\in\mathcal{I}(x)} D_{\mathrm{KL}}\Big(p^{\tau}_{\Theta^P}(\cdot \mid x,j) \,\|\, p^{\tau}_{\mathcal{P}^S(\Theta^P)}(\cdot \mid x,j)\Big). \tag{3}$$

Here, $p^{\tau}_{\Theta}(\cdot \mid x,j) = \mathrm{softmax}(z_{W_0,\Theta}(x,j)/\tau)$ is the next-token distribution at position j. The task template fixes $\mathcal{I}(x)$ before compilation. We average first over positions within each example and then over examples, so longer completions receive no greater weight. We use $\tau = 1$, cache the frozen parent distributions, and update no child weights.

Selection objective. We seek the support whose predictions best match the parent while keeping the saved adapter within the budget:

$$S^{\star}_B \in \arg\min_{S\subseteq[L]} \mathcal{J}_{\mathrm{KL}}(S;\Theta^P) \qquad \text{s.t.} \qquad C_\sigma(S) \le B. \tag{4}$$

Here, $S^{\star}_B$ denotes an optimizer of the constrained problem, whereas $\hat{S}_B$ denotes the support returned by our greedy search. Calibration KL measures fidelity to the parent; it is not task loss on the evaluation subset and does not guarantee downstream task utility. Parent–child top-1 agreement serves as a secondary diagnostic.

Choosing layers in context. Conditional SIFT-KL starts with empty support and evaluates each complete one-layer extension $S \cup \{\ell\}$. With added cost $\Delta c_\ell(S) = C_\sigma(S \cup \{\ell\}) - C_\sigma(S)$, it selects the feasible, positive-cost extension with the greatest calibration-KL reduction per byte. It then recomputes all remaining scores. Selection stops when no candidate is feasible or the best candidate does not reduce KL.

Algorithm 1 Conditional SIFT-KL: selecting whole layers under a size cap.

Input: frozen parent model (W_0, Θ^P), calibration set $\mathcal{D}_{\mathrm{cal}}$, byte cap B, temperature τ, and canonical serializer σ.
Output: selected whole-layer support $\hat{S}_B$.

1 Require $C_\sigma(\varnothing) \le B$; otherwise report an infeasible cap.
2 Cache $q_{x,j} \leftarrow p^{\tau}_{\Theta^P}(\cdot \mid x,j)$ for every $x \in \mathcal{D}_{\mathrm{cal}}$ and $j \in \mathcal{I}(x)$.
3 Initialize $S \leftarrow \varnothing$ and $d \leftarrow \mathcal{J}_{\mathrm{KL}}(S;\Theta^P)$.
4 Dry-serialize the one-layer extensions and form $\mathcal{F}(S) = \{\ell \in [L] \setminus S : C_\sigma(S \cup \{\ell\}) \le B,\ \Delta c_\ell(S) > 0\}$.
5 If $\mathcal{F}(S) = \varnothing$, go to line 8. Otherwise, for each $\ell \in \mathcal{F}(S)$, compute $d_\ell \leftarrow \mathcal{J}_{\mathrm{KL}}(S \cup \{\ell\};\Theta^P)$ and $\rho_\ell \leftarrow (d - d_\ell)/\Delta c_\ell(S)$ using the cached parent distributions.
6 Select $\ell^\star \leftarrow \arg\max_{\ell\in\mathcal{F}(S)} \rho_\ell$. Break ties by lower d_ℓ, fewer resulting bytes, then lower layer index.
7 If $\rho_{\ell^\star} \le 0$, go to line 8. Otherwise, set $S \leftarrow S \cup \{\ell^\star\}$ and $d \leftarrow d_{\ell^\star}$, then return to line 4.
8 Return $\hat{S}_B \leftarrow S$.

Using the same serializer for selection and export keeps every accepted support within B. Whole-layer granularity and the positive-gain rule can leave space unused or miss groups that require a non-improving intermediate addition. We assume neither additivity nor submodularity, so the classical cardinality-constrained greedy approximation guarantee for monotone submodular maximization does not apply (Nemhauser et al., 1978). The returned support is feasible but has no global-optimality or approximation-ratio guarantee. With K accepted layers, conditional rescoring requires $O(L(K+1))$ projected-child evaluations and $O(L^2)$ in the worst case, compared with a single $O(L)$ scoring pass for harmonized Static-KL. This is a one-time offline compilation cost rather than a serving-time cost; Appendix B gives the detailed count and failure modes.

Controls for rescoring and layer placement. Harmonized Static-KL evaluates each complete singleton child once using

$$\frac{\mathcal{J}_{\mathrm{KL}}(\varnothing;\Theta^P) - \mathcal{J}_{\mathrm{KL}}(\{\ell\};\Theta^P)}{C_\sigma(\{\ell\}) - C_\sigma(\varnothing)},$$

and fixes that ordering. Conditional SIFT-KL recomputes marginal scores after each selection. The comparison fixes the parent, calibration and calibration-disjoint evaluation examples, candidate layers, byte feasibility, admission/stopping and tie rules, and serialization/packing; only frozen versus conditional scoring is intended to differ (Section 4; Appendix C).

Boundary-KL instead considers only contiguous layer windows and selects the feasible window with the lowest calibration predictive KL. All three selectors share the hard cap and evaluation procedure, although final sizes can differ. Boundary-KL is not a reproduction of Chen et al. (2025), whose method selects a lower-layer prefix using validation task performance rather than predictive KL.

3.3 PHYSICAL EXPORT AND THE COMPILER CONTRACT

The exporter saves exactly the parent factors in $\hat{S}_B$ and the required support/configuration metadata. Unselected adapter tensors are absent rather than masked or zero-valued. For exported adapter $\mathcal{A}_{\hat{S}_B}$,

$$\mathrm{Tensors}(\mathcal{A}_{\hat{S}_B}) = \{A^P_{\ell m}, B^P_{\ell m} : \ell \in \hat{S}_B, m \in \mathcal{M}_\ell\}, \quad |\mathcal{A}_{\hat{S}_B}|_{\mathrm{bytes}} = C_\sigma(\hat{S}_B) \le B. \tag{5}$$

Because B enters only at compilation, the same frozen parent can be recompiled under different deployment budgets, with Eq. (5) realizing each resulting support as a physical adapter artifact. We accept the final bundle only if its measured size equals the selection-time count and fits the cap; the shared dense backbone is excluded.

Structural checks confirm byte count, cap compliance, selected keys, loading metadata, and retained-factor preservation. Reload checks compare the physical adapter with its logical projection on identical deterministic inputs. All 54 controlled exports pass the structural audit and, separately, all 54 pass reload at a prespecified 2×10^{-4} absolute/relative logit tolerance, with argmax agreement checked (Appendix B).

Physical omission lets a support-aware loader skip unselected LoRA branches, but does not guarantee lower latency because a runtime may create or execute placeholders. We evaluate sparse execution and workload-specific utility, latency, and throughput separately.

Compilation ends at export. Section 4 separately evaluates deployment and copied-versus-reset state under matched continuation.

4 EVALUATION DESIGN

We evaluate SIFT-LoRA along four questions organized around three aspects: support selection (RQ1), deployment behavior (RQ2), and state inheritance (RQ3–RQ4). The four studies use separate artifact populations and are analyzed separately.

RQ1: Does conditional rescoring improve support selection? We compare harmonized Static-KL, Conditional SIFT-KL, and Boundary-KL across three models—Qwen3.5-2B, Llama-3.1-8B-Instruct, and Gemma-3-4B-Instruct (Qwen Team, 2026a; Grattafiori et al., 2024; Gemma Team et al., 2025)—and three tasks: BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), and GSM8K (Cobbe et al., 2021). Each model–task pair is evaluated at 12.5% and 25% of the Full-LoRA adapter size, giving 18 conditions and 54 compiled adapters. Each condition uses one frozen same-task Full-LoRA parent, 128 selector-calibration examples, and a separate, ID-disjoint 300-example validation-derived evaluation set. Appendix C.1 records the recovered nine-family parent, checkpoint, data-lineage, and reference-evaluator definitions.

All selectors share the parent, data, cap, serializer, and evaluation procedure. Static-KL and Conditional SIFT-KL differ only in frozen versus recomputed marginal scores; Boundary-KL restricts support to contiguous windows. The controlled selector comparison is conditional on the supplied frozen parent checkpoint rather than on the best achievable parent for each model–task family. Because every selector within a condition receives the same parent, the comparison isolates support selection from parent-training quality; downstream utility nevertheless remains conditional on the learned parent state. We report selector-evaluation KL, utility, task loss, and realized size; calibration KL is used only for selection.

To measure how much of the parent's improvement is retained, we use

$$R_L(\Theta^P, S) = \frac{\mathcal{L}_{\text{ret}}(W_0) - \mathcal{L}_{\text{ret}}(\mathcal{P}^S(\Theta^P))}{\mathcal{L}_{\text{ret}}(W_0) - \mathcal{L}_{\text{ret}}(\Theta^P)}, \tag{6}$$

$$R_U(\Theta^P, S) = \frac{U_{\text{ret}}(\mathcal{P}^S(\Theta^P)) - U_{\text{ret}}(W_0)}{U_{\text{ret}}(\Theta^P) - U_{\text{ret}}(W_0)}. \tag{7}$$

We report these ratios only when the parent shows positive measured progress and its 95% paired-example bootstrap interval (Efron, 1979) excludes zero. Appendix C gives the full ledger, evaluator/statistics contract, and three-parent Qwen3.5-2B/HellaSwag comparisons at each cap. The budgets are analyzed separately because cross-budget evaluator/anchor parity is not established.

RQ2: What utility–runtime trade-offs do the compiled adapters provide? We evaluate deployment across five tasks—BoolQ, HellaSwag, GSM8K, MMLU (Hendrycks et al., 2021), and XSum (Narayan et al., 2018)—and two byte budgets. The primary BoolQ/12.5% study combines a historical seven-method campaign with a separate fresh comparison of current Conditional SIFT-KL, Static-KL, and Boundary-KL exports. We then extend the deployment evaluation across tasks and budgets: Appendix E.9 consolidates the seven-method operating points for all five tasks at 12.5% and 25%, while Appendix E.10 reports the fresh current-selector comparisons across the same five-task setting. Runtime uses the reported physical-unmerged bf16 execution paths; BoolQ uses median ITL, the auxiliary seven-method cohorts retain their recorded ITL statistic, XSum reports generation ITL, and the fresh MMLU comparison reports utility without generation ITL under teacher-forced multiple-choice scoring. Pareto analyses are computed within each model, task, budget, and candidate set.

RQ3: Does inherited state help when support is fixed? For one Qwen3.5-2B/HellaSwag parent, we compare copied parent factors with reset factors on the same Conditional SIFT-KL supports at 12.5% and 25%. Support, bytes, rank, optimizer, data order, and continuation horizon are fixed within each pair. We summarize early performance using normalized area under the learning curve,

$$\text{AULC}_H(r) = \frac{1}{H} \sum_{i=0}^{m-1} (h_{i+1} - h_i) \frac{V_r(h_i) + V_r(h_{i+1})}{2}, \tag{8}$$

for $H \in \{250, 500\}$. Appendix F records the continuation protocol, AULC, and endpoint evaluation.

RQ4: What is the incremental cost once a parent already exists? A separate HellaSwag pilot compares canonical exact-byte Conditional SIFT-KL compilation with fresh Tuned Direct-Sparse recovery on the identical support for Qwen3.5-2B and Llama-3.1-8B-Instruct at the 12.5% serialized-parent-adapter byte cap. The estimand begins after the Full-LoRA parent exists; parent training is excluded (Appendix H).

5 RESULTS

5.1 CONDITIONAL RESCORING SELECTS MORE FAITHFUL ADAPTERS

Across 18 model–task–budget conditions, Conditional SIFT-KL lowers selector-evaluation predictive KL in 16 against harmonized Static-KL and 16 against Boundary-KL (Figure 2), with mean differences of -0.0280 and -0.0265. Among the three selectors, it ranks first in KL in 15 conditions, task utility in 14/18 including ties, and retained loss progress R_L in 13/16 defined conditions (Table 1).

Among 15/18 conditions where Static-KL and Conditional SIFT-KL differ by at most 25 serialized bytes, Conditional lowers evaluation KL in 13 and improves utility in 12, with one utility tie. Thus the benefit persists at nearly identical adapter sizes.

Across three independently trained Qwen3.5-2B/HellaSwag parents at the 12.5% cap, Conditional improves mean accuracy over Static-KL by 3.0 percentage points at nearly identical adapter sizes. Accuracy, predictive KL, R_L, and R_U favor Conditional for all three parents, as do all three within-budget pairs in the separately recorded 25% comparison (Table 12; Appendix C.3).

Beyond the KL-based controls (Figure 4), Conditional lowers evaluation KL against OutputNorm in both recorded cohorts and against SIFT-Mag in all but one comparison. In the separate lower-prefix

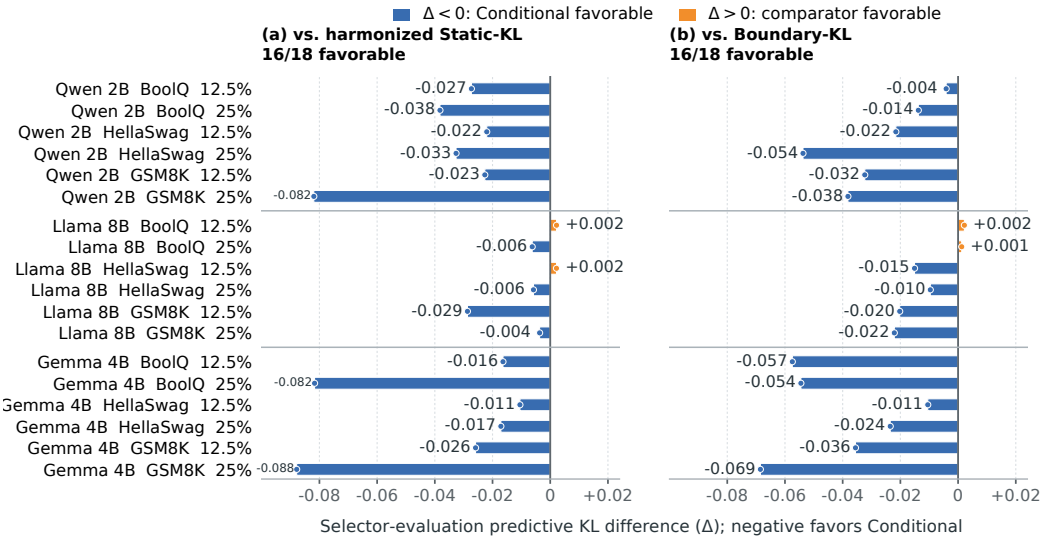


Figure 2: **Selector-evaluation fidelity under conditional, fixed-score, and contiguous-layer selection.** Values are Conditional-minus-comparator predictive KL; negative values favor Conditional SIFT-KL. Conditional SIFT-KL achieves lower KL in 16 of 18 conditions against harmonized Static-KL and separately in 16 against Boundary-KL. The grid covers three models, three tasks, and 12.5% and 25% size caps. Methods share the cap within each condition, while their final adapter sizes may differ.

Table 1: **Controlled selector comparisons across 18 conditions.** Differences are Conditional minus comparator. Lower KL, higher task utility, and higher R_L favor Conditional SIFT-KL; W/T/L denotes wins, ties, and losses. Means and medians weight conditions equally. R_L is defined in 16 conditions. Methods share a hard byte cap but may attain different adapter sizes.

Comparison	KL W/T/L	Mean ΔKL	Median ΔKL	Utility W/T/L	R_L W/T/L
Conditional vs. harmonized Static-KL	**16/0/2**	**−.0280**	**−.0223**	**14/1/3**	**13/0/3**
Conditional vs. Boundary-KL	**16/0/2**	**−.0265**	**−.0218**	**13/3/2**	**15/0/1**

replay of Chen et al. (2025), it lowers projection KL in all 18 conditions and improves utility in 10. Chen et al. (2025) achieve higher utility in all six GSM8K conditions, illustrating different fidelity and utility orderings (Appendix D).

Conditional rescoring costs more offline than harmonized Static-KL. Appendix C.4 reports condition-level wall-clock ratios; Appendix C.3 separately reports independent-parent timings. These costs are paid at compilation, not serving.

All 54 controlled exports pass the size, state-preservation, omission, and reload checks in §3.3 (Table 5). Deployment is evaluated on separate artifact populations.

5.2 LAYER-SPARSE ADAPTERS OFFER COMPETITIVE UTILITY–LATENCY TRADE-OFFS

In the historical seven-method BoolQ/12.5% campaign under physical-unmerged bf16 H100 inference, SIFT-LoRA lies on all three observed utility–median-ITL frontiers (Figure 3) and ranks first, second, and first in median ITL on Qwen, Llama, and Gemma. Relative to the fastest cap-constrained alternative with at least its utility, it reduces median ITL by 27.5–42.2% for 1.47–3.49 percentage points lower utility. Against LoRA-Drop (shared), ITL is 29.2–44.2% lower and throughput 37.4–81.7% higher, with lower utility. Separate Base references have lower latency but lower utility than SIFT-LoRA on all models.

Execution audits distinguish compact storage from sparse computation: SIFT-LoRA runs adapter branches in only 3/24 Qwen, 3/32 Llama, and 4/34 Gemma layers, versus every layer for LoRA-Drop (shared) (Table 23), despite nearly identical cap-accounted sizes on Llama/Gemma. Forward-hook checks across all 21 historical endpoints record no inactive-branch calls.

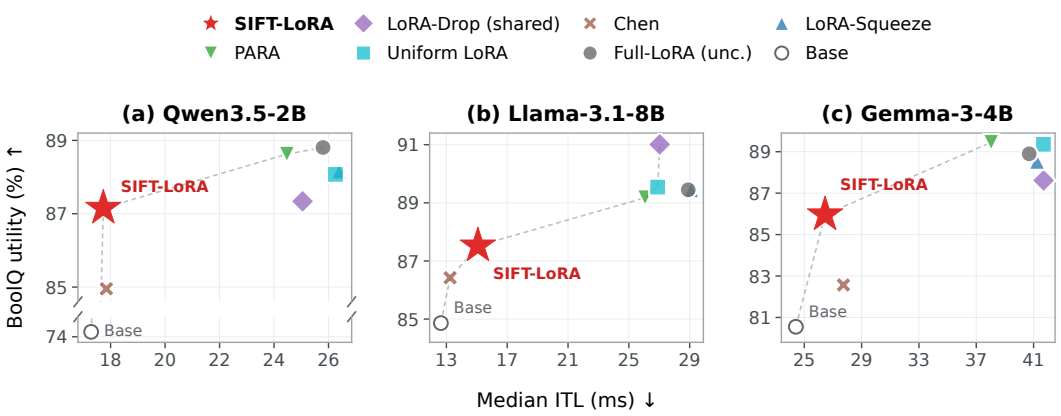


Figure 3: **Historical BoolQ/12.5% with separate no-adapter Base references.** Seven methods retain 21 historical endpoints; Base adds one separately measured, protocol-matched reference per model. Full-LoRA is uncapped. Guides show displayed point-estimate frontiers, not statistical dominance. Qwen's y-axis is broken.

Table 2: **Fresh matched current selectors on BoolQ/12.5%.** Accuracy uses 1,090 shared test rows; median ITL pools 2,850 intervals from 30 repetitions per label. Bold marks the best point estimate; Appendix E.8 gives matching, uncertainty, and audits.

	Conditional SIFT-KL		Static-KL		Boundary-KL	
Model	Acc. (%)	ITL (ms)	Acc. (%)	ITL (ms)	Acc. (%)	ITL (ms)
Qwen3.5-2B	**87.156**	**43.973**	86.789	44.141	86.881	44.314
Llama-3.1-8B†	**88.899**	**23.482**	88.807	23.538	88.807	23.540
Gemma-3-4B	**84.862**	33.269	82.294	32.034	80.734	**31.738**

†Llama controls share adapter/prediction identities but were timed separately. One declared session uses model-specific method order, not repeated-block counterbalancing; all nine labels pass the reported audits.

One frozen Qwen3.5-2B/HellaSwag checkpoint also yields feasible physical exports independently compiled at 6.25%, 12.5%, 25%, and 50% byte caps (Appendix G), illustrating post-training budget selection without retraining each child.

A separate fresh BoolQ/12.5% comparison evaluates the current Conditional SIFT-KL exports against Static-KL and Boundary-KL (Table 2). Conditional lies on the three-selector point-estimate frontier for every model. On Qwen and Llama, it has the highest utility and lowest median ITL, although the margins are small. On Gemma, it gains 2.569 and 4.128 accuracy points over Static-KL and Boundary-KL for 3.86% and 4.82% higher median ITL, respectively. Appendix E.8 reports the paired uncertainty and timing details; these exports remain separate from Figure 3's historical artifacts. The historical and fresh timing results come from separate measurement sessions, so we do not compare absolute latency values across the two campaigns; Pareto and selector conclusions are drawn only within their respective timing populations.

Across five tasks and both byte budgets (Appendices E.9–E.10), Conditional SIFT-KL improves utility over Static-KL in 26 of 30 fresh model–task–budget cells and ties in two; it improves over Boundary-KL in 21 and ties in three. In the separate seven-method evaluation, SIFT-LoRA lies on the observed utility–ITL frontier in 24 of 30 panels, including all 12 MMLU/XSum and 10 of 12 BoolQ/HellaSwag panels. On GSM8K, it is nondominated on Llama and Gemma at 12.5%; at 25%, Chen et al. (2025) achieve higher exact match and lower reported ITL.

5.3 Inherited Weights and Incremental Retrofit Cost

Table 3 compares copied factors with three reset children from one Qwen3.5-2B/HellaSwag parent, fixing support, rank, serialized bytes, and continuation. Relative to the reset mean, inheritance increases AULC@250 by .2517/.2683 at the 12.5%/25% caps and AULC@500 by .1269/.1317. The gain therefore comes from learned values, not additional capacity: inheritance improves initialization and early utility, while endpoint ordering remains mixed.

Table 3: **Matched state-control outcomes on SIFT-KL-selected supports.** Reset rows are means over three child seeds; inherited rows are deterministic trajectories from the same parent. Within each cap, support, serialized bytes, rank, optimizer, data order, and continuation horizon are fixed. G_0 is step-0 utility minus base utility on a 0–1 scale. Reset endpoint values show mean $\pm$ standard deviation.

Cap	State	AULC@250	AULC@500	G_0	Endpoint
12.5%	Reset	.6200	.7539	.0000	$.9130 \pm .0024$
12.5%	Inherit	**.8717**	**.8808**	**.5033**	.9148
25%	Reset	.6167	.7533	.0000	$\mathbf{.9247} \pm .0014$
25%	Inherit	**.8850**	**.8850**	**.5467**	.9205

In the post-parent HellaSwag retrofit pilot, Qwen/Llama compilation totals 394.5 s, versus 5,352.3 s for fresh Tuned Direct-Sparse adapters on identical supports to reach inherited step-0 validation utility: a 13.57$\times$ recovery-to-compilation time ratio (Appendix H). Both compiled artifacts satisfy the 12.5% cap. Parent training is excluded; each model has one recovery trajectory, measured at its first observed target-reaching anchor without interpolation.

6 LIMITATIONS

Conditional SIFT-KL uses greedy forward selection with conditional rescoring, so it does not guarantee the globally optimal support under the byte cap. Its repeated candidate evaluation also incurs greater offline compilation cost than harmonized Static-KL. Future work could combine SIFT-LoRA's conditional fidelity objective with faster and more effective search strategies to reduce compilation cost while finding stronger sparse supports.

Lower parent–child predictive KL does not always translate into higher downstream task utility. In the separate 18-cell lower-prefix replay of Chen et al. (2025) (Appendix D.3), the baseline achieves higher exact-match accuracy than Conditional SIFT-KL in all six GSM8K model–budget comparisons despite higher parent–child predictive KL. Two factors likely contribute. First, SIFT-LoRA is a post-training compiler: it selects which learned updates to inherit from an existing same-task parent rather than relearning the task, so parent quality remains an upstream constraint. In our primary selector grid, measured parent utility progress is nonpositive for Llama/GSM8K and Gemma/GSM8K. Second, teacher-forced token-level predictive fidelity is not identical to free-running multi-step generation utility, so lower KL need not preserve the sequence-level behaviors that determine exact-match accuracy. These results therefore bound the scope of predictive KL as a selection objective, particularly for autoregressive reasoning. Future work could combine parent-fidelity signals with lightweight task-aware or sequence-level objectives and stronger parent training.

Runtime evidence is specific to the reported physical-unmerged bf16 H100 execution paths. The observed utility–latency trade-offs may change with batching, concurrency, hardware, kernel implementations, or tighter serving constraints. Future work should evaluate SIFT-LoRA across a broader range of inference environments and hardware regimes, especially under deployment settings where memory, latency, or compute budgets are more restrictive.

7 CONCLUSION

In this paper, we propose SIFT-LoRA, which compiles a trained same-task Full-LoRA parent into a layer-sparse adapter under an exact serialized-byte budget without further training. Conditional SIFT-KL selects layers in context by repeatedly recomputing marginal predictive-KL reduction per added byte, while physical export preserves selected factors and removes unselected tensors. Across the controlled three-task, two-budget selector grid, SIFT-LoRA generally improves parent fidelity over static and contiguous controls. The broader deployment evaluation extends across five tasks and both byte budgets, showing competitive utility–latency trade-offs across models and task types, while matched-support controls show that inherited parent state improves early learning. More broadly, SIFT-LoRA separates task adaptation from deployment capacity, allowing one trained parent to be recompiled for different deployment budgets rather than retraining a new sparse adapter for each operating point.

AI USE STATEMENT

We used generative AI tools for feedback on methodology and experimental design, implementation and debugging, interpretation of preliminary results, and manuscript drafting and editing. The authors reviewed the AI-assisted work and retain full responsibility for the submission.

REPRODUCIBILITY STATEMENT

Section 3 and Algorithm 1 specify the SIFT-LoRA compiler, including conditional selection, exact-byte feasibility, and physical export. Section 4 and the Appendix document the evaluation protocols and supporting evidence for selector fidelity, deployment, state inheritance, multi-budget compilation, and retrofit cost, together with the corresponding export and execution audits. The supplementary code package provides the available result ledgers and aggregate summaries, together with plotting and analysis scripts used for the reported results; parent/checkpoint and evaluator definitions, record-level availability, and missing raw predictions or timing traces are documented with the corresponding experiments and in Appendix I.

ETHICS STATEMENT

This work studies adapter compilation using public language models and benchmarks, without human-subject data collection. We do not evaluate safety or bias after compilation; predictive fidelity to a parent model should not be interpreted as evidence of safe or unbiased behavior.

REFERENCES

Aastik, Meghana Topu, Chinmay Kulkarni, and Pragya Paramita Sahu. NormAL LoRA: What is the perfect size? In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 19716–19731. Association for Computational Linguistics, 2025. doi:10.18653/v1/2025.findings-emnlp.1074. URL https://aclanthology.org/2025.findings-emnlp.1074/.

Shuang Ao, Yi Dong, Jinwei Hu, and Sarvapali D. Ramchurn. Safe pruning LoRA: Robust distance-guided pruning for safety alignment in adaptation of LLMs. *Transactions of the Association for Computational Linguistics*, 13:1474–1487, 2025. doi:10.1162/tacl.a.44. URL https://aclanthology.org/2025.tacl-1.67/.

Guanhua Chen, Yutong Yao, Ci-Jun Gao, Lidia S. Chao, Feng Wan, and Derek F. Wong. Not all LoRA parameters are essential: Insights on inference necessity. *arXiv preprint arXiv:2503.23360*, 2025. doi:10.48550/arXiv.2503.23360. URL https://arxiv.org/abs/2503.23360.

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/.

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. *arXiv preprint arXiv:2110.14168*, 2021. doi:10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Hamed Damirchi, Cristian Rodriguez-Opazo, Ehsan Abbasnejad, Zhen Zhang, and Javen Shi. The quest for winning tickets in low-rank adapters. *arXiv preprint arXiv:2512.22495*, 2025. doi:10.48550/arXiv.2512.22495. URL https://arxiv.org/abs/2512.22495.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4133–4145. Association for Computational Linguistics, 2023. doi:10.18653/v1/2023.emnlp-main.252. URL https://aclanthology.org/2023.emnlp-main.252/.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. doi:10.1214/aos/1176344552. URL https://doi.org/10.1214/aos/1176344552.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. doi:10.48550/arXiv.2503.19786. URL https://arxiv.org/abs/2503.19786.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi:10.48550/arXiv.2407.21783. URL https://arxiv.org/abs/2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2009.03300.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. doi:10.48550/arXiv.1503.02531. URL https://arxiv.org/abs/1503.02531.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

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. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Hugging Face. Safetensors: Simple, safe tensor storage. Software, 2022. URL https://github.com/safetensors/safetensors.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. doi:10.1214/aoms/1177729694. URL https://doi.org/10.1214/aoms/1177729694.

Vishnuprasadh Kumaravelu, Sunil Gupta, and P. K. Srijith. Post-optimization adaptive rank allocation for LoRA. *arXiv preprint arXiv:2604.27796*, 2026. doi:10.48550/arXiv.2604.27796. URL https://arxiv.org/abs/2604.27796.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

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

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, Benjamin Bossan, and Marian Tietz. PEFT: State-of-the-art parameter-efficient fine-tuning methods. Software, 2022. URL https://github.com/huggingface/peft.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12:153–157, 1947. doi:10.1007/BF02295996. URL https://doi.org/10.1007/BF02295996.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807. Association for Computational Linguistics, 2018. doi:10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206/.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14:265–294, 1978. doi:10.1007/BF01588971. URL https://doi.org/10.1007/BF01588971.

Qwen Team. Qwen3.5-2B. Hugging Face model card, March 2026a. URL https://huggingface.co/Qwen/Qwen3.5-2B.

Qwen Team. Qwen3.5-9B. Hugging Face model card, March 2026b. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1gSj0NKvB.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7930–7946. Association for Computational Linguistics, 2021. doi:10.18653/v1/2021.emnlp-main.626. URL https://aclanthology.org/2021.emnlp-main.626/.

Andrey Veprikov, Vladimir Solodkin, Alexander Zyl, Andrey Savchenko, and Aleksandr Beznosikov. WeightLoRA: Keep only necessary adapters. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12419–12437. Association for Computational Linguistics, 2026. doi:10.18653/v1/2026.acl-long.566. URL https://aclanthology.org/2026.acl-long.566/.

Ivan Vulić, Adam Grycner, Quentin de Laroussilhe, and Jonas Pfeiffer. LoRA-squeeze: Simple and effective post-tuning and in-tuning compression of LoRA modules. *arXiv preprint arXiv:2602.10993*, 2026. doi:10.48550/arXiv.2602.10993. URL https://arxiv.org/abs/2602.10993.

Xinyi Wang, Lirong Gao, Haobo Wang, Yiming Zhang, and Junbo Zhao. FLoE: Fisher-based layer selection for efficient sparse adaptation of low-rank experts. *arXiv preprint arXiv:2506.00495*, 2025. doi:10.48550/arXiv.2506.00495. URL https://arxiv.org/abs/2506.00495.

Chenxing Wei, Yao Shu, Ying Tiffany He, and Fei Yu. Flexora: Flexible low-rank adaptation for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14643–14682. Association for Computational Linguistics, 2025. doi:10.18653/v1/2025.acl-long.713. URL https://aclanthology.org/2025.acl-long.713/.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45. Association for Computational Linguistics, 2020. doi:10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1977–1992. Association for Computational Linguistics, 2024. doi:10.18653/v1/2024.findings-emnlp.109. URL https://aclanthology.org/2024.findings-emnlp.109/.

Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard Baraniuk, Zhangyang Wang, and Yingyan Lin. Drawing early-bird tickets: Toward more efficient training of deep networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJxsrgStvr.

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, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, Tiejun Zhao, and Muyun Yang. LoRA-drop: Efficient LoRA parameter pruning based on output evaluation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5530–5543. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.371/.

A Evidence Map and Populations

The Appendix organizes evidence for support selection, deployment behavior, and inherited state. Table 4 identifies the experimental units and their locations; detailed protocols and interpretation limits appear with each result. The breadth summaries count cellwise outcomes across separately recorded cohorts; metric values, uncertainty, and audits remain specific to their original populations.

Table 4: **Contribution-to-evidence map.** Base points are separately measured no-adapter references, not additional adapter endpoints in the historical or broad seven-method campaigns.

Contribution	Evidence	Population or unit	Location
Support selection	Controlled selector grid and physical export	18 conditions $\times$ 3 selectors; nine parent families; 128 calibration and 300 validation-derived evaluation examples per condition	B; C.1–C.2
Parent sensitivity	Independent-parent comparisons	Three Qwen/HellaSwag parents; 12.5% and 25% panels analyzed separately	C.3
Practical selection	Heuristics, Chen replay, compiler cost, and random supports	Original 12-condition cohort and six-condition 25% extension; separate 18-cell Chen replay; 18×20 feasible random supports	C.4–C.5; D
Deployment	Historical BoolQ/12.5% campaign	21 seven-method endpoints; 1,090 utility examples and 30 timing repetitions each; three separate Base references	E.1–E.7; Fig. 3
Current selectors	Fresh matched BoolQ/12.5%	Nine labels/eight artifact identities; 1,090 utility examples and 30 repetitions per label	E.8; Table 2
Deployment breadth	Separate seven-method auxiliary cohorts	HellaSwag/GSM8K: 78 source records/84 displayed labels at two caps, plus six separate Base references. BoolQ/25%: 21 point estimates plus three reused Base references	E.9; Figs. 5–6
Current-selector breadth	Fresh matched cross-task deployment	HellaSwag/GSM8K at 12.5%: 18 selector endpoints. BoolQ/HellaSwag/GSM8K at 25%: 27 selector endpoints plus nine matched Base references; 30 repetitions per endpoint	E.10
State inheritance	Fixed-support copied versus reset factors	One 2B parent, two caps, three reset seeds/support; separate one-parent 9B control	F; Table 3
Later budget choice	Multi-budget physical compilation	One frozen checkpoint, four independently compiled byte caps	G
Post-parent cost	Exact-cap retrofit	Qwen and Llama, one recovery trajectory each on the compiled support	H
Deployment extension	MMLU/XSum seven-method cohorts	78 source records; 72 capped endpoints and six Full-LoRA references; 84 budget-specific method labels; three XSum Base points and three MMLU Base accuracies in captions	E.9
Current-selector extension	Additional MMLU/XSum selectors	12 conditions; 36 selector endpoints and six Base references; reported utility and XSum ITL; MMLU generation ITL N/A	E.10

Appendix I summarizes record availability; file-level inventory and verification details are provided in `RECORD_INVENTORY.md` and `VERIFICATION_DETAILS.md`.

B Compiler Contract and Physical Realization

Serialization and feasibility. Selection and export use the same serializer, dtype, tensor naming, and required loading metadata (Eqs. 2–5). The shared backbone is excluded. The final bundle is accepted only when its measured bytes equal $C_\sigma(\widehat{S}_B)$ and do not exceed B.

Structural and reload checks. Table 5 preserves two separate audits of the 54 primary controlled exports. Structural inspection checks bundle bytes, cap, support, omission, configuration, and retained-tensor identity to the ledger-declared parent; masked or zero-valued tensors do not count as omitted. Reload compares each logical support-restricted projection with its physically reloaded adapter on identical deterministic inputs (up to 300 examples, batch size 4), using prespecified 2×10^{-4} absolute and relative logit tolerances and an argmax-agreement check. All 54 pass each audit, with no failed or skipped records. Reload checks therefore confirm numerical agreement between each logical support projection and its physical export on the audited inputs; sparse execution is evaluated separately in Appendix E.

Table 5: **Physical-export correctness audit of the full 54-export controlled-selector population.** Structural counts come from direct filesystem/safetensors inspection (Hugging Face, 2022), not parameter-count proxies. Reload equivalence compares each logical support projection with the corresponding physically reloaded export on identical deterministic inputs.

Audited clause	Result	Direct evidence
Bundle bytes = ledger bytes	**54/54 PASS**	Filesystem bundle versus ledger, selection, and artifact manifests
Bundle bytes $\leq$ integer cap	**54/54 PASS**	Filesystem bundle versus the per-condition cap
Exact selected-support membership	**54/54 PASS**	Actual layer/module keys versus selected support and config
Omitted-layer LoRA-key absence	**54/54 PASS**	Direct enumeration; total omitted-layer key count = 0
Serializer/config consistency	**54/54 PASS**	Valid offsets/payloads, recorded dtype/configuration, A/B pairs, rank, adapted-module set, and metadata
Retained state = declared parent	**54/54 PASS**	Every retained observation matches in name, shape, dtype, and raw bytes
Structural audit	**54/54 PASS**	Every clause above passes for every controlled export
Logical projection $\approx$ physical reload	**54/54 PASS**	Identical deterministic inputs; absolute/relative logit tolerance 2×10^{-4}; argmax agreement checked

Complexity and failure modes. With K accepted layers, conditional greedy uses at most $2 + \sum_{j=0}^{K}(L-j)$ calibration evaluations: parent, empty child, and complete candidates, including a possible unsuccessful final round. This is $O(L(K+1))$ projected-child passes and $O(L^2)$ in the worst case. A cap below fixed overhead is infeasible; otherwise selection stops when no feasible positive-gain addition remains, without padding the artifact. A useful group may require a non-improving intermediate support. Without submodularity, feasibility provides no global-optimality or classical greedy approximation guarantee (Nemhauser et al., 1978).

Table 5 covers the primary controlled exports; deployment-specific audits are reported with the corresponding experiments in Appendix E.

C Controlled Selector Evidence

C.1 Protocol and Evaluators

Harmonized comparison. The grid covers Qwen3.5-2B, Llama-3.1-8B-Instruct, and Gemma-3-4B-Instruct on BoolQ, HellaSwag, and GSM8K at 12.5% and 25% byte caps: 18 conditions from nine same-task Full-LoRA parents. Within each condition, the three selectors share the frozen parent, 128 calibration examples, a 300-example evaluation subset, candidate layers, byte cap, serializer, and evaluator. Static-KL scores complete singleton children against the empty projection once and freezes the ordering. Conditional SIFT-KL uses the same feasibility, admission, stopping, and tie rules but recomputes marginal scores after each accepted layer. Boundary-KL instead minimizes calibration KL over feasible contiguous supports. The statistical unit is a model–task–cap condition under its frozen parent; calibration KL is the selection objective, not independent evaluation evidence.

Parents and recipes. Directly inspected configurations report seed 0. Table 6 summarizes the supplied checkpoints and the recipe information available for each parent family.

Table 6: **Primary-parent checkpoints and inspected recipes.** All nine families supply checkpoint 375 at both caps. The supplied checkpoint is the fixed selector input and need not coincide with the trainer-reported best checkpoint. Q/L/G denote Qwen/Llama/Gemma. Recipe codes refer to the fields below; a dash indicates no row-linked recipe or trainer record.

Family	Supplied step	Recipe	Trainer record
Q / BoolQ	375	A	best 375; final 1500
Q / HellaSwag	375	A	best 1500
Q / GSM8K	375	C	best 500; final 1500
L / BoolQ	375	B	best 250; final 375
L / HellaSwag	375	A	best 1500
L / GSM8K	375	B	best=final 375
G / BoolQ	375	–	—
G / HellaSwag	375	–	—
G / GSM8K	375	–	—

Recipes A/B share the inspected core fields: rank and alpha 16, dropout 0, learning rate 2×10^{-4}, AdamW (Loshchilov & Hutter, 2019) ($\beta_1 = .9$, $\beta_2 = .999$, $\epsilon = 10^{-8}$ where recorded), bf16, maximum length 1,024, batch 20, gradient accumulation 1, warmup .03, and validation `eval_loss`. For A, Qwen/BoolQ uses 1,500 steps and 375-step save/evaluation anchors 375/750/1125/1500; cadence is run-specific for the HellaSwag runs. B retains the 1,500-step schedule but stops at 375; Llama/BoolQ saves/evaluates every 250 steps with anchor 375. C is the separate 1,500-step Qwen/GSM8K replication with 250-step save/evaluation cadence. These settings summarize the directly inspected configurations; model-specific recipe information is retained as recorded.

Evaluation lineage. Each condition uses a 300-example evaluation set from the task validation data. For BoolQ and HellaSwag, 300 evaluation examples are sampled deterministically with seed 0; GSM8K uses the full 300-example validation file. The 128 selector-calibration examples are selected separately, and calibration and evaluation examples are disjoint by ID in every condition. For GSM8K specifically, the evaluation set is the full validation file, whereas selector calibration uses a separate, non-overlapping 128-example pool from prepared/gsm8k/calibration.jsonl. Because the validation data also support parent evaluation/checkpoint selection, we treat these as calibration-disjoint validation-derived evaluations rather than independent test sets (Table 7).

Table 7: **Primary selector-evaluation populations.** Each condition uses 128 selector-calibration examples and a separate evaluation set; calibration and evaluation examples are disjoint by example ID. "Evaluation construction" describes only how the evaluation set is formed. For GSM8K, evaluation uses the full 300-example validation file, while selector calibration uses a separate, non-overlapping 128-example pool from prepared/gsm8k/calibration.jsonl. The same task-level evaluation files are reused across models and byte caps. These are validation-derived evaluations rather than independent test sets.

Task	Validation n	Evaluation n	Calibration n	Evaluation construction	Calibration source
BoolQ	1,090	300	128	Seed-0 validation subset	prepared/boolq/ calibration.jsonl
HellaSwag	3,348	300	128	Seed-0 validation subset	prepared/hellaswag/ calibration.jsonl
GSM8K	300	300	128	Full validation file	prepared/gsm8k/ calibration.jsonl

The available BoolQ and HellaSwag calibration files each contain 1,024 examples; each selector uses 128 of them. GSM8K's separate calibration file contains 128 examples, all used for selection.

Evaluator and statistics contract. Equation 3 measures parent-to-child KL at $\tau = 1$, averaging scored positions within each example and then examples. For BoolQ and HellaSwag, the recovered reference evaluator scores option-token causal log-likelihood only: prompt tokens are not scored, prompt tokenization adds model special tokens, option tokenization does not, summed likelihood is the default and token-length normalization is an explicit alternative, and ties choose the first option.

Batching uses the tokenizer pad token with EOS fallback, and the recovered selector configurations use maximum length 1,024. Utility is accuracy and evaluator loss is the negative option score; no free generation is used.

For GSM8K selector fidelity, the recovered reference evaluator compares parent and child distributions by teacher-forced KL at target positions. It tokenizes the prompt with special tokens and the target without them, masks all but target rationale/answer causal positions, and uses maximum length 1,024 with documented target and prompt truncation. Target-token NLL is a separate task-loss operation. The generation evaluator, which supplies task utility, uses deterministic greedy decoding (no sampling, one beam), EOS and end-of-turn stopping when available, and pad-to-EOS fallback. Gold answers use the final #### numeric marker; predictions prefer that marker and otherwise use the last numeric candidate before normalized exact matching. These evaluator definitions distinguish predictive KL, target-token NLL, and generated-answer exact match in the reported analyses.

The ratio audit uses 10,000 paired-example bootstrap resamples with seed 20260904. Equations 6–7 require positive measured parent progress and a 95% denominator interval excluding zero. KL-derived ranks use one plus the number of strictly lower KL values within the condition; equal values share a rank. Other metric ranks retain their recorded definitions.

C.2 Canonical Replay, Anchors, and Full Ledger

Canonical GSM8K evaluation. All six GSM8K conditions use the frozen teacher-forced target-token KL replay at maximum length 1,024, with the same 300 examples and scoring contract for all three selectors within a condition. Only evaluation KL and KL-derived summaries change; parents, supports, bytes, calibration scores, task utilities, task losses, and R_L/R_U are unchanged. BoolQ/HellaSwag retain their prior evaluations. Target-token counts are 37,057/30,003/36,911 for Qwen/Llama/Gemma, with selector/budget parity within each model.

Condition-level utility anchors. Qwen/HellaSwag uses one condition-level base/parent utility pair across both caps. A performance-blind rule selects the latest complete capture matching model, task, parent, split, evaluator, and evaluation-file identity. Only R_U is recomputed from this shared anchor; child utility, loss anchors, and R_L remain unchanged.

Attained bytes and aggregate scope. Table 1 gives the main condition-level summaries. Among the 15/18 Static–Conditional pairs within 25 serialized bytes, Conditional has lower KL in 13/15, utility W/T/L 12/1/2, and higher defined R_L in 11/13. In the six Llama/Gemma 25% cells, its KL comparison is favorable in 6/6 against Static and 5/6 against Boundary. The near-byte subset compares selectors at nearly identical storage sizes, while the six 25% extension cells provide the corresponding same-cap comparison.

Ratio availability. R_L is undefined only for Llama/BoolQ at both byte caps because measured parent loss progress is nonpositive. We report R_U for all three selectors in the 10 conditions where its denominator criterion is satisfied (Table 8). For R_U, the denominator interval crosses zero in both Llama/BoolQ and both Gemma/BoolQ conditions, while measured parent utility progress is nonpositive in Llama/GSM8K and Gemma/GSM8K at both caps. Ratios are not clipped.

Table 8: **Recomputed R_U values in the 10 conditions where the ratio is defined.** Columns are harmonized Static (S), Conditional (C), and Boundary (B). Qwen/HellaSwag uses one condition-level base/parent utility pair at both caps.

Cell	S	C	B
Qwen / BoolQ / 12.5%	1.166667	1.066667	.966667
Qwen / BoolQ / 25%	1.100000	1.000000	1.133333
Qwen / HellaSwag / 12.5%	.866667	.915152	.884848
Qwen / HellaSwag / 25%	.963636	.993939	.963636
Qwen / GSM8K / 12.5%	1.539683	1.634921	1.587302
Qwen / GSM8K / 25%	.698413	1.380952	1.380952
Llama / HellaSwag / 12.5%	1.039216	1.078431	1.058824
Llama / HellaSwag / 25%	1.117647	1.098039	1.098039
Gemma / HellaSwag / 12.5%	.777778	.808081	.777778
Gemma / HellaSwag / 25%	.929293	.949495	.909091

Full numeric ledger. Tables 9–11 retain all 54 controlled rows. S/C/B denote harmonized Static-KL, Conditional SIFT-KL, and Boundary-KL. "Bytes/cap" reports exact serialized bytes and cap with utilization rounded to 0.01 percentage points. KL is evaluation parent–child predictive KL; "Task" is child utility. A dash is an undefined R_L, not zero. The full-precision data and superseded records remain in the source package.

Table 9: **Qwen3.5-2B controlled selector ledger (six conditions, 18 rows).**

Task	Cap	Sel.	Support	Bytes / cap (util.)	KL ↓	Task ↑	R_L
BoolQ	12.5%	S	[4, 5, 8]	4,722,557 / 5,460,181 (86.49%)	.133123	.900000	1.186374
		C	[1, 4, 10]	4,722,568 / 5,460,181 (86.49%)	.105795	.890000	1.176584
		B	[3, 4]	4,133,121 / 5,460,181 (75.70%)	.109931	.880000	1.128524
BoolQ	25%	S	[4, 5, 6, 8, 9, 10]	9,444,224 / 10,920,362 (86.48%)	.097103	.893333	1.278022
		C	[1, 2, 4, 10, 11, 14]	10,428,694 / 10,920,362 (95.50%)	.058882	.883333	1.251565
		B	[3, 4, 5, 6, 7]	9,839,213 / 10,920,362 (90.10%)	.072664	.896667	1.240666
HellaSwag	12.5%	S	[6, 10, 12]	4,722,579 / 5,460,181 (86.49%)	.188294	.826667	.891671
		C	[1, 9, 12]	4,722,568 / 5,460,181 (86.49%)	.166289	.853333	.924620
		B	[11, 12]	4,133,147 / 5,460,181 (75.70%)	.187864	.836667	.892022
HellaSwag	25%	S	[1, 4, 6, 8, 10, 12]	9,444,227 / 10,920,362 (86.48%)	.094329	.880000	.970318
		C	[1, 4, 8, 9, 11, 12]	10,428,691 / 10,920,362 (95.50%)	.061686	.896667	.997360
		B	[7, 8, 9, 10, 11]	9,839,239 / 10,920,362 (90.10%)	.115437	.880000	.952789
GSM8K	12.5%	S	[2, 4, 12]	4,722,568 / 5,460,181 (86.49%)	.218939	.620000	.527278
		C	[2, 12, 20]	4,722,579 / 5,460,181 (86.49%)	.196264	.640000	.590657
		B	[12, 13, 14]	4,722,590 / 5,460,181 (86.49%)	.228607	.630000	.501808
GSM8K	25%	S	[0, 1, 2, 4, 8, 12]	9,444,224 / 10,920,362 (86.48%)	.193306	.443333	.578920
		C	[2, 8, 12, 13, 20, 22]	9,444,249 / 10,920,362 (86.48%)	.111370	.586667	.792435
		B	[9, 10, 11, 12, 13, 14]	10,428,716 / 10,920,362 (95.50%)	.149551	.586667	.693479

Table 10: **Llama-3.1-8B-Instruct controlled selector ledger (six conditions, 18 rows).**

Task	Cap	Sel.	Support	Bytes / cap (util.)	KL ↓	Task ↑	R_L
BoolQ	12.5%	S	[8, 9, 10]	15,735,979 / 20,980,270 (75.00%)	.062132	.896667	—
		C	[2, 8, 9]	15,735,956 / 20,980,270 (75.00%)	.064313	.896667	—
		B	[8, 9, 10]	15,735,979 / 20,980,270 (75.00%)	.062132	.896667	—
BoolQ	25%	S	[4, 8, 9, 10, 12, 19, 20]	36,716,068 / 41,960,540 (87.50%)	.040281	.890000	—
		C	[2, 7, 8, 9, 10, 12, 20]	36,716,045 / 41,960,540 (87.50%)	.034032	.896667	—
		B	[6, 7, 8, 9, 10, 11, 12]	36,716,045 / 41,960,540 (87.50%)	.032771	.893333	—
HellaSwag	12.5%	S	[11, 12, 15]	15,736,025 / 20,980,270 (75.00%)	.100317	.943333	.949543
		C	[10, 11, 15]	15,736,025 / 20,980,270 (75.00%)	.102482	.950000	.961706
		B	[10, 11, 12]	15,736,025 / 20,980,270 (75.00%)	.117573	.946667	.921849
HellaSwag	25%	S	[8, 9, 10, 11, 12, 14, 15]	36,716,091 / 41,960,540 (87.50%)	.036003	.956667	1.045160
		C	[3, 7, 10, 11, 12, 14, 15]	36,716,091 / 41,960,540 (87.50%)	.030187	.953333	1.062824
		B	[10, 11, 12, 13, 14, 15, 16]	36,716,129 / 41,960,540 (87.50%)	.039814	.953333	1.051675
GSM8K	12.5%	S	[4, 5, 6]	15,735,956 / 20,980,270 (75.00%)	.134211	.863333	.758216
		C	[5, 6, 17]	15,735,979 / 20,980,270 (75.00%)	.105515	.886667	.823632
		B	[7, 8, 9]	15,735,956 / 20,980,270 (75.00%)	.125744	.856667	.774238
GSM8K	25%	S	[3, 4, 5, 6, 9, 15, 17]	36,716,022 / 41,960,540 (87.50%)	.059627	.850000	.919326
		C	[0, 1, 5, 6, 14, 17, 18]	36,716,045 / 41,960,540 (87.50%)	.055945	.893333	.931655
		B	[11, 12, 13, 14, 15, 16, 17]	36,716,129 / 41,960,540 (87.50%)	.078047	.836667	.873242

Table 11: **Gemma-3-4B-Instruct controlled selector ledger (six conditions, 18 rows).**

Task	Cap	Sel.	Support	Bytes / cap (util.)	KL ↓	Task ↑	R_L
BoolQ	12.5%	S	[17, 25, 28, 29]	14,035,509 / 14,911,902 (94.12%)	.189974	.836667	.856718
		C	[4, 11, 17, 29]	14,035,486 / 14,911,902 (94.12%)	.173594	.856667	.909211
		B	[27, 28, 29, 30]	14,035,509 / 14,911,902 (94.12%)	.231046	.826667	.776453

Table 11: Gemma-3-4B-Instruct controlled selector ledger (continued).

Task	Cap	Sel.	Support	Bytes / cap (util.)	KL ↓	Task ↑	R_L
BoolQ	25%	S	[17, 22, 24, 25, 26, 27, 28, 29]	28,070,113 / 29,823,804 (94.12%)	.169777	.843333	.880536
		C	[4, 11, 13, 16, 17, 28, 29, 33]	28,070,098 / 29,823,804 (94.12%)	.088098	.876667	1.008571
		B	[11, 12, 13, 14, 15, 16, 17, 18]	28,070,113 / 29,823,804 (94.12%)	.142581	.870000	.958872
HellaSwag	12.5%	S	[13, 14, 15, 16]	14,035,509 / 14,911,902 (94.12%)	.181087	.870000	.907322
		C	[11, 13, 14, 15]	14,035,509 / 14,911,902 (94.12%)	.170580	.880000	.905513
		B	[13, 14, 15, 16]	14,035,509 / 14,911,902 (94.12%)	.181087	.870000	.907322
HellaSwag	25%	S	[11, 12, 13, 14, 15, 16, 17, 22]	28,070,113 / 29,823,804 (94.12%)	.083582	.920000	.960120
		C	[5, 9, 11, 13, 14, 15, 18, 19]	28,070,075 / 29,823,804 (94.12%)	.066472	.926667	.968680
		B	[13, 14, 15, 16, 17, 18, 19, 20]	28,070,113 / 29,823,804 (94.12%)	.089988	.913333	.958868
GSM8K	12.5%	S	[15, 17, 18, 20]	14,035,509 / 14,911,902 (94.12%)	.330180	.813333	.573874
		C	[14, 17, 20, 28]	14,035,509 / 14,911,902 (94.12%)	.304296	.820000	.612316
		B	[17, 18, 19, 20]	14,035,509 / 14,911,902 (94.12%)	.339855	.833333	.559873
GSM8K	25%	S	[11, 13, 15, 16, 17, 18, 19, 20]	28,070,113 / 29,823,804 (94.12%)	.227792	.746667	.710209
		C	[14, 17, 18, 19, 20, 28, 29, 32]	28,070,113 / 29,823,804 (94.12%)	.139920	.816667	.836628
		B	[15, 16, 17, 18, 19, 20, 21, 22]	28,070,113 / 29,823,804 (94.12%)	.208440	.810000	.736800

C.3 Independent-Parent Selector Comparisons

The 12.5% repeat recompiles Static-KL and Conditional SIFT-KL from three independently trained Qwen3.5-2B/HellaSwag parents. The separately recorded 25% panel uses the same three parent model-file identities, without parent or child retraining; each within-budget pair shares its parent, cap and 300-example evaluation file. The 12.5% and 25% panels are reported separately using their recorded within-budget anchors. Together they test selector behavior across three independently trained Qwen3.5-2B/HellaSwag parents at each cap.

Conditional has lower KL and higher R_L, R_U, and accuracy for all three parents within each panel. At 12.5%, mean KL falls from .1964 to .1785 (9.1% relative), R_L rises from 87.5% to 90.2% ($+2.8$ pp), R_U rises from 85.9% to 91.4% ($+5.4$ pp), and accuracy rises from 82.3% to 85.3% ($+3.0$ pp). At 25%, mean Conditional-minus-Static differences are $-.0328$ KL, $+.0258$ R_L, $+.0177$ R_U, and $+.0100$ accuracy. Supports vary by parent. All 12.5% pairs retain three layers, use 86.49% of the cap, and differ by at most 22 bytes; 25% byte differences are $+984{,}464$, -11, and 0 for seeds 0/1/2. Thus two of the three 25% pairs are exact/near-byte matched, while seed 0 uses more of the common cap under Conditional.

Mean Static/Conditional compiler times are 120.8/262.8 s at 12.5% and 96.4/486.9 s at 25%; these auxiliary means are separate from the primary cost cohorts below. Ratio intervals use 10,000 paired-example resamples (seed 20260904) within each artifact. All six 25% exports pass the recorded structural checks. Compiler timings and interval details are reported separately from the primary-grid aggregates.

Table 12: **Independent-parent selector outcomes in the recorded 12.5% repeat and separately recorded 25% extension.** Each row pair shares one parent artifact and exact cap within its panel. Bold marks the better metric value within a parent and panel. Percentages are reported for R_L, R_U, and task accuracy; displayed values are rounded from the source records. The panels are not pooled across budgets. The separately recorded 25% panel retains its own evaluation records and is therefore not numerically interchangeable with corresponding primary-grid rows in Table 9, even when support and serialized bytes coincide.

Seed	Selector	Support	Bytes	KL ↓	R_L ↑	R_U ↑	Acc. ↑
(a) 12.5% cap: 5,460,181 bytes							
0	Static-KL	[6, 10, 12]	4,722,579	.1883	89.17	86.67	82.67
0	**Conditional**	[1, 9, 12]	4,722,568	**.1663**	**92.46**	**91.52**	**85.33**
1	Static-KL	[10, 12, 13]	4,722,590	.1883	86.62	83.53	82.33
1	**Conditional**	[0, 9, 12]	4,722,568	**.1665**	**89.94**	**90.00**	**86.00**
2	Static-KL	[6, 10, 12]	4,722,579	.2127	86.61	87.58	82.00
2	**Conditional**	[0, 10, 12]	4,722,579	**.2028**	**88.33**	**92.55**	**84.67**
(b) Separately recorded 25% cap: 10,920,362 bytes							
0	Static-KL	[1, 4, 6, 8, 10, 12]	9,444,227	.1547	89.61	94.19	88.33
0	**Conditional**	[1, 4, 8, 9, 11, 12]	10,428,691	**.1175**	**92.25**	**96.51**	**89.67**
1	Static-KL	[5, 6, 9, 10, 12, 13]	9,444,238	.1015	94.91	92.44	87.33
1	**Conditional**	[0, 5, 8, 9, 10, 12]	9,444,227	**.0809**	**96.70**	**94.19**	**88.33**
2	Static-KL	[4, 5, 6, 8, 10, 12]	9,444,227	.1261	94.16	98.16	87.67
2	**Conditional**	[0, 4, 6, 9, 10, 12]	9,444,227	**.0854**	**97.48**	**99.39**	**88.33**

C.4 Offline Compiler Cost

Table 13 reports two separately recorded offline compiler-timing cohorts. Because their timing protocols were recorded in different experiment cohorts, we summarize each cohort separately rather than pool them into a single 18-condition wall-clock statistic. These are one-time compilation costs, not serving costs.

Table 13: **Controlled-selector compiler cost by timing cohort.** Legacy 12 comprises the retained timing conditions; New 6 contains the Llama/Gemma 25% extension. Entries are within-cohort wall-time medians; independent-parent timings in Appendix C.3 are separate.

Cohort	Selector	Median wall time (s)
Legacy 12	Harmonized Static-KL	96.574
Legacy 12	Conditional SIFT-KL	295.300
Legacy 12	Boundary-KL	279.458
New 6	Harmonized Static-KL	55.271
New 6	Conditional SIFT-KL	310.337
New 6	Boundary-KL	308.864

The median of condition-level Conditional/Static wall-clock ratios is $2.990\times$ for Legacy 12 and $5.864\times$ for New 6; these are not ratios of the selector-wise medians.

C.5 Balanced-18 Random-Support Reference

The random-support study provides a finite-sample reference of 20 deterministic byte-feasible supports per condition (`random00`–`random19`, seeds 0–19), totaling 360 supports across the balanced 18-cell grid: 240 from the original cohort and 120 from the 25% extension. It is reported separately from the three-selector controlled ablation. The packaged cell ledger preserves the finite-sample KL ranks/percentiles. Its GSM8K KL and Qwen/HellaSwag/25% R_U anchor retain their own recorded definitions, not the primary canonical replay and normalized anchor.

Conditional's mean/median utility is $.855741/.885000$ over 18 entries versus $.809620/.850000$ over 360 random supports. For defined ratios, R_L is $.922122/.928138$ over 16 Conditional entries versus $.682593/.708117$ over 320 random entries; R_U is $1.093174/1.033333$ over 10 Conditional entries versus $.789017/.806645$ over 200 random entries. Undefined ratios are omitted, not zero-filled.

D Baseline Definitions and Practical Breadth

D.1 Structural Taxonomy and Evaluated Implementations

Table 14 distinguishes the source methods of Vulić et al. (2026); Chen et al. (2025); Zhou et al. (2025); Ao et al. (2025); it is a structural taxonomy, not an empirical ranking or a complete recipe for the evaluated adaptations. Training-time FLoE and bottleneck AdapterDrop are complementary, structurally different approaches (Wang et al., 2025; Rücklé et al., 2021).

Table 14: **Structural differences among nearby compact-adapter methods.** Post-hoc means that structure reduction follows broad LoRA training. Whole-layer omission reduces adapter work only on a support-aware path.

Method	Selection unit	Resulting adapter structure	Relation to SIFT-LoRA
LoRA-Squeeze (Post-Squeeze)	Rank component / target rank	Lower-rank factors at broad placements	Post-tuning rank compression; execution can remain depth-dense
Chen et al.	Boundary	Contiguous lower-layer LoRA prefix	Post-hoc lower-prefix alternative; no further training; our contiguous-window control differs
LoRA-Drop	Projection-level module	Important branches plus shared LoRA	Output-energy principle; shared branches still execute
LoRA-Drop (no share)	Projection-level module	Low-importance branches removed	Direct removal ablation; training procedure differs from ours
SPLoRA	LoRA layer (source term)	Safety-alignment weight-distance-guided subset	Training-free post-hoc removal; safety criterion
SIFT-LoRA	Transformer layer	Possibly non-contiguous whole-layer support; retained tensors unchanged	Conditional complete-child predictive KL under realized serialized cost

Chen et al. (2025) select a lower-layer prefix by validation task utility and copy the retained parent factors without child optimization. Boundary-KL instead searches byte-feasible contiguous windows by predictive KL. OutputNorm is a local-output-energy diagnostic, not LoRA-Drop's full preliminary-update, threshold, sharing and further-training procedure; SIFT-Mag is the magnitude heuristic. The evaluated external labels follow the recorded adaptations of Chen et al. (2025), LoRA-Drop (shared), LoRA-Squeeze, and PARA (Kumaravelu et al., 2026). Table 14 focuses on the structural differences relevant to SIFT-LoRA; LoRA-Drop's compact shared storage and depth-dense runtime representation are distinguished in Appendix E.7.

The current per-byte scoring rule reproduces all 18 recorded final supports in a calibration-only compatibility replay; in the six Qwen cells that distinguish the two scoring variants, per-byte scoring matches 6/6 supports while raw marginal-KL scoring matches 0/6. The preserved historical v0.4.0 snapshot uses raw marginal KL rather than the current per-byte rule. Source-level availability for historical records is summarized in Appendix I.

D.2 Practical Selector Cohorts

The original 12-condition breadth cohort compares Conditional, the recorded Static-KL variant, OutputNorm, and SIFT-Mag. The six-condition Llama/Gemma 25% extension uses a separate five-way set including harmonized Static-KL and Boundary-KL. These cohorts use their recorded selector procedures and are analyzed separately from the harmonized Static–Conditional ablation. Random supports are reported separately in Appendix C.5. GSM8K metrics retain their own paired records rather than substituting either side from the primary canonical grid.

D.3 Heuristic Comparisons and Separate Chen Replay

Figure 4 and Table 15 retain the cohort-specific pairwise results. The separate 18-cell replay of Chen et al. (2025) matches parent, model, task, cap and 300-example evaluation identity; its six GSM8K KL pairs use a common teacher-forced projection-KL replay. It neither retrains nor reselects the supports and does not alter primary utility anchors or ranks.

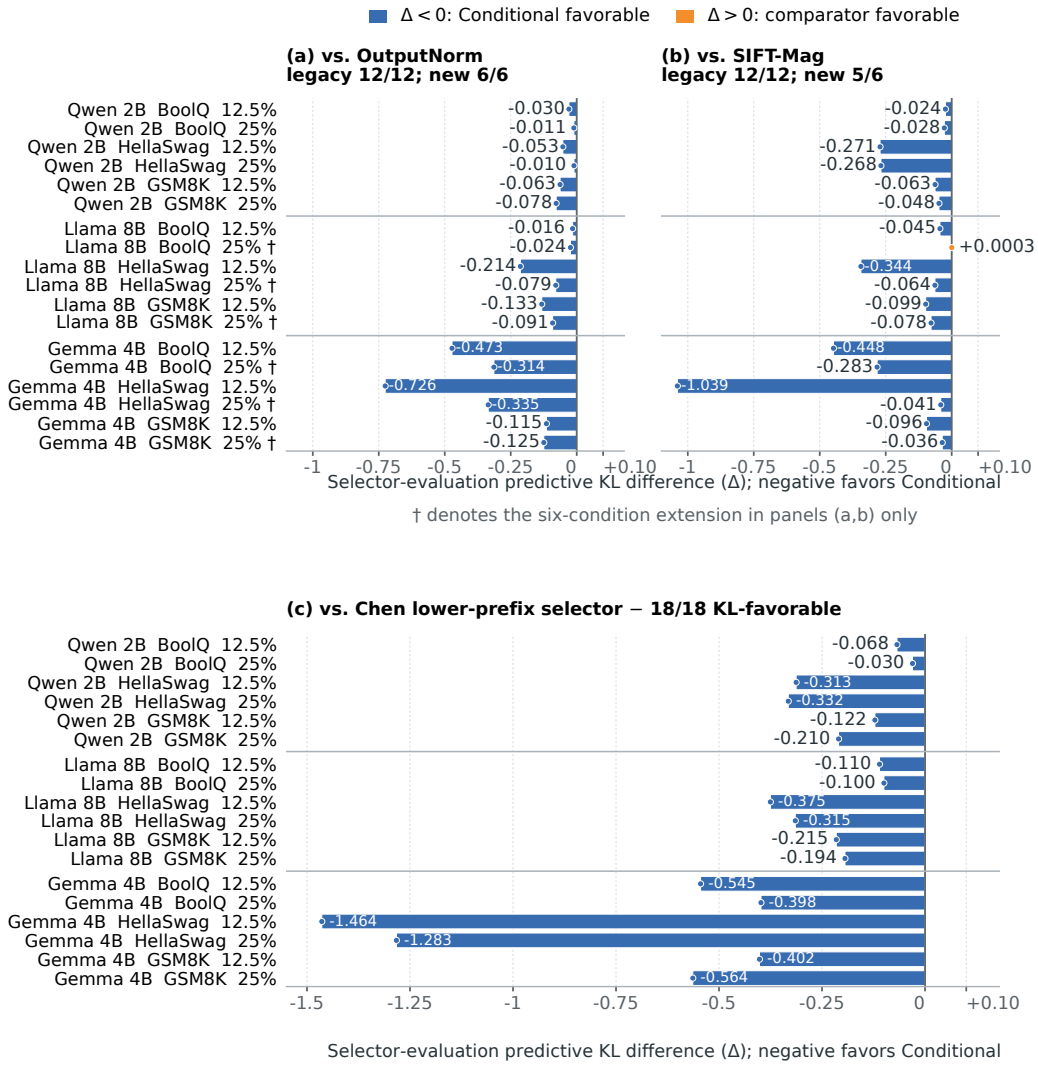


Figure 4: **Practical selector breadth across the balanced 18-condition task–budget layout.** Values are Conditional-minus-comparator selector-evaluation predictive KL; negative values favor Conditional. Panels (a,b) retain the earlier OutputNorm and SIFT-Mag breadth records: Conditional is favorable in 12/12 legacy and 6/6 new conditions against OutputNorm and in 12/12 and 5/6, respectively, against SIFT-Mag. Daggers apply only to the six-condition extension in panels (a,b). Panel (c) reports the separately recorded selector-evaluation projection-KL comparison with the Chen validation-selected lower-prefix baseline; Conditional has lower selector-evaluation projection-KL in 18/18 cells (mean/median difference $-.3912/-.3139$). Panels (a,b) and (c) come from separately recorded comparison cohorts; panel (c) uses a wider horizontal range. Table 16 reports the corresponding task-level utility patterns.

Table 15: **Practical-selector pairwise summaries by cohort.** Differences are Conditional minus comparator; W/T/L counts lower/equal/higher KL. Means and medians weight conditions equally within each cohort.

Cohort	Comparator	W/T/L	Mean Δ	Median Δ
Legacy 12	OutputNorm	**12/0/0**	−.1599	−.0705
	SIFT-Mag	**12/0/0**	−.2311	−.0970
New 6	OutputNorm	**6/0/0**	−.1614	−.1080
	SIFT-Mag	**5/0/1**	−.0836	−.0525
Chen replay 18	Chen lower-prefix	**18/0/0**	−.3912	−.3139

Table 16: **Separately recorded Conditional–Chen KL comparison and utility across 18 model–task–budget cells.** $\Delta\mathrm{KL}$ and ΔU are Conditional minus Chen; lower KL and higher utility favor Conditional. Utility is percentage accuracy/exact match. Bold negative ΔU entries are utility reversals in which Chen is higher despite Conditional's lower predictive KL.

Model	Task	Cap	C KL	Chen KL	ΔKL	C U	Chen U	ΔU (pp)
Qwen	BoolQ	12.5%	.1058	.1739	−.0681	89.00	88.00	+1.00
Qwen	BoolQ	25%	.0589	.0893	−.0304	88.33	88.67	**−.33**
Qwen	HellaSwag	12.5%	.1663	.4792	−.3129	85.33	76.67	+8.67
Qwen	HellaSwag	25%	.0617	.3936	−.3319	89.67	78.67	+11.00
Qwen	GSM8K	12.5%	.199841	.321525	−.121684	64.00	67.00	**−3.00**
Qwen	GSM8K	25%	.111662	.321525	−.209864	58.67	67.00	**−8.33**
Llama	BoolQ	12.5%	.0643	.1748	−.1105	89.67	88.67	+1.00
Llama	BoolQ	25%	.0340	.1339	−.0998	89.67	90.00	**−.33**
Llama	HellaSwag	12.5%	.1025	.4779	−.3754	95.00	79.67	+15.33
Llama	HellaSwag	25%	.0302	.3450	−.3149	95.33	87.67	+7.67
Llama	GSM8K	12.5%	.105644	.321008	−.215365	88.67	93.00	**−4.33**
Llama	GSM8K	25%	.055974	.250258	−.194284	89.33	91.00	**−1.67**
Gemma	BoolQ	12.5%	.1736	.7184	−.5448	85.67	83.33	+2.33
Gemma	BoolQ	25%	.0881	.4863	−.3982	87.67	86.00	+1.67
Gemma	HellaSwag	12.5%	.1706	1.6346	−1.4641	88.00	63.67	+24.33
Gemma	HellaSwag	25%	.0665	1.3495	−1.2830	92.67	67.67	+25.00
Gemma	GSM8K	12.5%	.304395	.706231	−.401836	82.00	84.00	**−2.00**
Gemma	GSM8K	25%	.140122	.703795	−.563673	81.67	83.33	**−1.67**

Task-level utility patterns. Conditional has lower projection KL in all 18 comparisons with Chen et al. (2025). Utility favors Conditional in 10/18 cells overall, including all six HellaSwag cells, while Chen et al. (2025) achieve higher utility in all six GSM8K cells; BoolQ is 4/0/2 in Conditional W/T/L. The task split illustrates that parent fidelity and downstream utility can favor different supports. Table 16 reports all cell-level values.

E Deployment Evidence

E.1 Historical Original Seven-Method Campaign

The historical BoolQ/12.5% campaign contains 21 endpoints: three models × six newly built capped adapters plus three uncapped frozen Full-LoRA references. All 21 have utility results, 30 timing repetitions each (630 total), and passing execution audits; all 18 capped artifacts meet their byte caps. Figure 3 additionally displays three separately measured Base references, not historical endpoints. These historical artifacts and measurements remain distinct from the fresh current-selector population in Table 2.

E.2 Workload and Cap-Accounting Contract

Historical utility uses the same option-log-likelihood evaluator and 1,090 shared BoolQ test examples. Runtime uses physical-unmerged bf16 H100 Transformers/PEFT execution (Wolf et al., 2020; Mangrulkar et al., 2022), 512-token prompts, 96 generated tokens, concurrency one, five warmups and 30 timed repetitions per endpoint. Each repetition contributes 95 ITL intervals; TTFT and throughput are summarized from the same recorded runs, with throughput computed as generated tokens divided by total wall time. Utility is evaluated separately from the timed generations. Method order is randomized within each model. The following workload contract applies to this historical population.

The cap counts 12.5% of the serialized full-parent adapter, including required metadata but excluding the backbone; it is neither resident bf16 storage nor peak memory. For LoRA-Drop (shared), the capped object is the compact shared bundle, not its larger alias-preserving per-layer runtime representation. The SIFT-LoRA/LD-sh pair is not near-byte matched on Qwen; Llama and Gemma cap-accounted sizes differ by only 0.052% and 0.085%, respectively.

E.3 Historical Utility–ITL and Byte Ledger

Table 17 retains every historical endpoint. "L/B" denotes adapter-executing layers/total layers, then executed branches; LD-sh/LSq/Full abbreviate LoRA-Drop (shared)/LoRA-Squeeze/Full-LoRA.

Table 17: **Historical 21-endpoint utility–ITL ledger.** Utility is percent, latency is median ITL in ms, throughput is tokens/s, and byte values are decimal MB. All 18 capped artifacts meet the exact cap in the recorded cap-accounting representation; Full-LoRA is uncapped.

Model	Method	Utility	ITL	TPS	Bytes/cap	L/B
Qwen	SIFT-LoRA	87.156	17.732	50.77	4.723/5.460	3/24;9
	LD-sh	87.339	25.048	36.94	5.122/5.460	24/24;96
	Chen	84.954	17.844	49.40	3.149/5.460	2/24;6
	LSq	88.165	26.391	35.96	2.771/5.460	24/24;96
	PARA	88.624	24.471	38.29	5.434/5.460	24/24;80
	Uniform	88.073	26.255	36.32	2.762/5.460	24/24;96
	Full	88.807	25.799	36.50	43.681/–	24/24;96
Llama	SIFT-LoRA	87.523	15.073	65.39	15.736/20.980	3/32;21
	LD-sh	91.009	27.030	35.99	15.744/20.980	32/32;224
	Chen	86.422	13.243	74.83	5.246/20.980	1/32;7
	LSq	89.450	29.100	33.64	10.570/20.980	32/32;224
	PARA	89.174	26.067	38.09	20.953/20.980	32/32;179
	Uniform	89.541	26.893	36.85	10.554/20.980	32/32;224
	Full	89.450	28.905	33.82	167.842/–	32/32;224
Gemma	SIFT-LoRA	85.963	26.449	37.31	14.035/14.912	4/34;28
	LD-sh	87.615	41.700	23.64	14.047/14.912	34/34;238
	Chen	82.569	27.733	35.48	14.035/14.912	4/34;28
	LSq	88.532	41.240	24.09	7.558/14.912	34/34;238
	PARA	89.450	38.041	26.16	14.869/14.912	34/34;191
	Uniform	89.358	41.686	23.59	7.534/14.912	34/34;238
	Full	88.899	40.695	23.66	119.295/–	34/34;238

Compared with LD-sh, historical SIFT-LoRA has 29.2%/44.2%/36.6% lower median ITL, 37.4%/81.7%/57.8% higher mean throughput, and $-.183/-3.486/-1.651$ pp utility differences on Qwen/Llama/Gemma. It lies on the original utility–ITL frontier for all three models; Qwen/Gemma have the lowest ITL and Llama the second-lowest.

Separate no-adapter references. The separately measured no-adapter Base references use the same 1,090 BoolQ example IDs/order and provide a protocol-matched latency context (Table 18). Utility is option-likelihood accuracy without length normalization. Historical SIFT-LoRA improves utility over Base by 12.94/2.66/5.41 pp on Qwen/Llama/Gemma at the cost of adapter execution and higher median ITL. Base uses no adapter state and is a contextual reference, not a capped selector. Figure 6 reuses these cap-independent references for the separate BoolQ/25% display.

Table 18: **Separately measured no-adapter Base references displayed in Figures 3 and 5.** Utility uses raw predictions on the same 1,090 BoolQ examples as the historical deployment population. Runtime is measured separately. Base has no adapter state and is not one of the historical capped methods.

Model	Utility (%)	Correct/1,090	ITL (ms)	Bytes	Layers/branches
Qwen3.5-2B	74.220	809/1,090	17.285	0	0/0
Llama-3.1-8B	84.862	925/1,090	12.639	0	0/0
Gemma-3-4B	80.550	878/1,090	24.410	0	0/0

E.4 HISTORICAL RUNTIME DISTRIBUTIONS

Tables 19 and 20 retain the historical distributions. TTFT and throughput use 30 repetitions per endpoint; ITL pools 2,850 measured token intervals. Repetitions and token intervals are not independent parent-training seeds.

E.5 PAIRED UTILITY UNCERTAINTY

Historical utility uncertainty pairs the same 1,090 examples, with 10,000 bootstrap draws using recorded row-specific seeds and exact two-sided McNemar tests (McNemar, 1947). In Table 21, b counts SIFT-LoRA wrong/comparator right, and c the reverse. The p-values are unadjusted. Pareto membership is computed from the reported utility and median-ITL point estimates; Table 21 separately reports paired utility uncertainty.

Table 19: **Original seven-method latency distributions.** TTFT uses 30 repetition-level observations per endpoint; ITL pools 2,850 intervals. Latency values are in ms.

Model	Method	TTFT (ms)			ITL (ms)		
		Median	IQR	95th	Median	IQR	95th
Qwen	SIFT-LoRA	171.75	8.13	189.92	17.73	0.72	18.56
	LD-sh	184.99	8.87	196.20	25.05	0.95	26.33
	Chen	172.97	12.76	318.03	17.84	1.42	19.31
	LSq	184.88	7.54	195.59	26.39	1.58	27.08
	PARA	181.63	8.40	192.73	24.47	0.91	25.05
	Uniform	184.67	11.03	200.58	26.26	1.94	27.54
	Full	187.02	17.83	282.26	25.80	1.65	27.12
Llama	SIFT-LoRA	20.96	0.07	21.32	15.07	0.49	15.81
	LD-sh	37.65	0.09	37.91	27.03	0.45	29.88
	Chen	20.18	0.05	20.25	13.24	0.06	13.34
	LSq	33.12	0.13	37.41	29.10	0.34	31.53
	PARA	33.44	0.09	33.57	26.07	0.24	26.55
	Uniform	33.02	0.11	33.13	26.89	0.21	27.79
	Full	37.29	0.07	37.75	28.91	0.40	31.96
Gemma	SIFT-LoRA	28.84	0.47	29.37	26.45	0.31	27.30
	LD-sh	47.42	0.59	49.14	41.70	0.63	46.13
	Chen	30.13	1.51	37.09	27.73	1.23	29.60
	LSq	43.56	0.52	44.21	41.24	0.37	42.98
	PARA	41.72	0.30	42.21	38.04	0.36	38.99
	Uniform	44.20	1.70	46.39	41.69	1.69	44.70
	Full	46.69	3.88	60.79	40.69	3.41	45.61

Table 20: **Original seven-method throughput and memory distributions.** Throughput reports mean $\pm$ sample SD and median over 30 repetitions in tokens/s. Peak GPU memory is GiB; resident bf16 adapter storage is MiB. Both memory columns retain the maximum recorded over repetitions.

Model	Method	Throughput (tokens/s)		Peak	Resident
		Mean $\pm$ SD	Median	(GiB)	(MiB)
Qwen	SIFT-LoRA	50.77 ± 1.23	51.51	4.42	4.50
	LD-sh	36.94 ± 0.99	37.51	4.42	4.88
	Chen	49.40 ± 2.02	49.80	4.42	3.00
	LSq	35.96 ± 1.02	35.55	4.42	2.60
	PARA	38.29 ± 0.95	37.78	4.42	5.15
	Uniform	36.32 ± 1.18	36.30	4.42	2.60
	Full	36.50 ± 0.97	36.09	4.46	41.62
Llama	SIFT-LoRA	65.39 ± 1.68	64.96	15.19	15.00
	LD-sh	35.99 ± 0.74	36.18	15.19	15.00
	Chen	74.83 ± 0.11	74.84	15.18	5.00
	LSq	33.64 ± 0.23	33.66	15.19	10.00
	PARA	38.09 ± 0.19	38.17	15.20	19.92
	Uniform	36.85 ± 0.28	36.97	15.19	10.00
	Full	33.82 ± 0.35	33.86	15.33	160.00
Gemma	SIFT-LoRA	37.31 ± 0.67	37.59	8.37	13.38
	LD-sh	23.64 ± 0.53	23.51	8.37	13.38
	Chen	35.48 ± 1.13	35.22	8.37	13.38
	LSq	24.09 ± 0.16	24.14	8.37	7.11
	PARA	26.16 ± 0.12	26.16	8.37	14.10
	Uniform	23.59 ± 0.63	23.93	8.37	7.11
	Full	23.66 ± 0.66	23.87	8.47	113.69

Table 21: **Historical seven-method paired utility uncertainty.** Δ and the bootstrap 95% interval are percentage points for SIFT-LoRA minus the named baseline.

Model	Baseline	Δ	95% interval	$b/c;\ p$
Qwen	LD-sh	-0.183	$[-1.284, +0.826]$	$18/16; .8642$
	Chen	$+2.202$	$[+0.550, +3.853]$	$31/55; .0127$
	LSq	-1.009	$[-2.385, +0.367]$	$35/24; .1925$
	PARA	-1.468	$[-2.844, -0.092]$	$38/22; .0519$
	Uniform	-0.917	$[-2.569, +0.734]$	$47/37; .3261$
	Full	-1.651	$[-3.303, +0.000]$	$53/35; .0693$
Llama	LD-sh	-3.486	$[-5.229, -1.835]$	$64/26; 7.66\times10^{-5}$
	Chen	$+1.101$	$[-0.459, +2.661]$	$34/46; .2185$
	LSq	-1.927	$[-3.394, -0.459]$	$44/23; .0139$
	PARA	-1.651	$[-3.211, -0.183]$	$44/26; .0414$
	Uniform	-2.018	$[-3.853, -0.275]$	$61/39; .0352$
	Full	-1.927	$[-3.578, -0.367]$	$50/29; .0238$
Gemma	LD-sh	-1.651	$[-3.486, +0.183]$	$62/44; .0982$
	Chen	$+3.394$	$[+1.835, +4.954]$	$21/58; 3.76\times10^{-5}$
	LSq	-2.569	$[-4.128, -1.101]$	$50/22; .0013$
	PARA	-3.486	$[-5.046, -2.018]$	$56/18; 1.11\times10^{-5}$
	Uniform	-3.394	$[-5.138, -1.560]$	$70/33; .0003$
	Full	-2.936	$[-4.587, -1.284]$	$59/27; .0007$

E.6 HISTORICAL PARETO SUMMARY

A point dominates another only when weakly better on every stated objective and strictly better on at least one. Table 22 maximizes utility and minimizes median ITL within the original seven-method set, including uncapped Full-LoRA; it is not a capped-only frontier. The full four-objective membership ledger is retained in the source package.

Table 22: **Utility–median-ITL frontier members within the original seven-method campaign.**

Model	Utility–median-ITL frontier members
Qwen	SIFT-LoRA, PARA, Full-LoRA
Llama	Chen, SIFT-LoRA, PARA, Uniform LoRA, LoRA-Drop (shared)
Gemma	SIFT-LoRA, PARA

Adding the separate Base references to Figure 3's displayed points leaves both Base and historical SIFT-LoRA nondominated for all models, without changing the original campaign membership above.

E.7 EXECUTION TOPOLOGY AND SHARED STORAGE

Forward-hook audits match physical and runtime branch paths at all 21 historical endpoints with zero inactive-branch calls. Table 23 focuses on sparse/shared methods; "Stored" counts modules in the cap-accounting representation. These are execution checks, separate from the 54-export structural/reload audit.

Table 23: **Original seven-method execution topology.** Layers are executed/total. The final column is the number of observed inactive-branch calls.

Model	Method	Layers	Branches	Stored	Inactive calls
Qwen	SIFT-LoRA	3/24	9	9	0
	LD-sh	24/24	96	14	0
	Chen	2/24	6	6	0
Llama	SIFT-LoRA	3/32	21	21	0
	LD-sh	32/32	224	21	0
	Chen	1/32	7	7	0
Gemma	SIFT-LoRA	4/34	28	28	0
	LD-sh	34/34	238	28	0
	Chen	4/34	28	28	0

The LD-sh alias audit passes on all models, with 23/30/31 shared layers for Qwen/Llama/Gemma and one unique A/B pair per shared module family, while computation executes in every layer. Compact bundles occupy 5.122/15.744/14.047 MB; alias-preserving per-layer runtime representations serialize

to 43.681/167.842/119.295 MB and are not the capped objects. The three separate Base references execute zero adapter layers/branches with zero inactive calls. These audits confirm that SIFT-LoRA's physically omitted branches are also absent from adapter execution, while LoRA-Drop (shared) retains depth-dense adapter execution despite compact shared storage. Utility and latency are measured separately in Appendices E.3–E.4.

E.8 FRESH MATCHED CURRENT-SELECTOR DEPLOYMENT

Fresh BoolQ/12.5% contains three current-selector labels for each model, giving nine labels and eight physical-export identities. The Llama Static-KL and Boundary-KL labels share one physical export and prediction identity but retain separately measured timing series. Within each model, all selectors share the parent, 128 calibration examples, byte cap, export contract, and one declared timing session with model-specific method order. Utility is option-only causal log-likelihood accuracy on 1,090 shared test examples, with no length normalization or free generation. Runtime uses physical-unmerged bf16, 512/96 prompt/generated tokens, concurrency one, five warmups, and 30 repetitions per label (270 total). ITL pools 2,850 intervals per label; TTFT uses repetitions, and throughput is the mean of per-repetition tokens/wall-time.

Table 24: **Fresh matched timing, export, and topology details.** ITL IQR and 95th percentile, TTFT median, and throughput are computed from the supplied fresh records; latency is in ms and throughput in tokens/s. "Bytes/cap" uses decimal MB from exact integer serialized-byte records. Layers/branches are the audited physical execution counts. Accuracy and median ITL are in Table 2.

Model	Selector	Support	Bytes/cap	ITL IQR	ITL 95th	TTFT (ms)	TPS (tokens/s)	L/B
Qwen	Conditional	1,4,10	4.723/5.460	2.897	47.627	294.666	21.943	3/9
	Static-KL	4,5,8	4.723/5.460	2.702	47.728	294.520	21.489	3/9
	Boundary-KL	3,4	4.133/5.460	2.827	47.963	299.338	21.431	2/10
Llama	Conditional	2,8,9	15.736/20.980	0.547	24.970	40.317	42.219	3/21
	Static-KL†	8,9,10	15.736/20.980	0.645	24.938	41.072	42.132	3/21
	Boundary-KL†	8,9,10	15.736/20.980	0.319	24.504	40.890	42.195	3/21
Gemma	Conditional	4,11,17,29	14.035/14.912	2.708	36.895	81.535	29.493	4/28
	Static-KL	17,25,28,29	14.036/14.912	2.704	35.524	80.776	30.533	4/28
	Boundary-KL	27,28,29,30	14.036/14.912	2.984	35.495	80.400	30.925	4/28

†Same recorded adapter and prediction identities; the two method labels have separately measured timing series.

Table 25: **Fresh matched BoolQ paired utility uncertainty.** Differences and 10,000-draw paired-bootstrap 95% intervals are accuracy percentage points for Conditional SIFT-KL minus the comparator. Counts b/c denote Conditional wrong/comparator right and the reverse; exact two-sided McNemar p-values are unadjusted.

Model	Comparator	Δ	95% interval	b/c; p
Qwen	Static-KL	$+0.367$	$[-0.917, +1.560]$	$22/26$; .6655
	Boundary-KL	$+0.275$	$[-0.917, +1.468]$	$22/25$; .7709
Llama	Static-KL†	$+0.092$	$[-0.917, +1.101]$	$14/15$; 1.000
	Boundary-KL†	$+0.092$	$[-0.826, +1.101]$	$14/15$; 1.000
Gemma	Static-KL	$+2.569$	$[+1.009, +4.128]$	$23/51$; .001516
	Boundary-KL	$+4.128$	$[+2.477, +5.872]$	$21/66$; 1.4101×10^{-6}

†Identical adapter/prediction identity; timing was measured separately for the two labels.

Conditional lies on the three-label frontier for all models (Table 2). Qwen/Llama show the best utility and median-ITL point estimates under Conditional, while Gemma gains 2.569/4.128 pp utility for 3.86%/4.82% higher median ITL versus Static/Boundary. Table 25 reports the corresponding paired utility uncertainty. All nine labels pass recorded structural, execution and logical-to-physical reload checks, with support/cap/bytes linkage and zero inactive calls. Reload uses 128 calibration examples and absolute/relative tolerance 0.02; all nine report agreement 1.0 and zero maximum absolute/relative logit error. This is distinct from the primary audit's 2×10^{-4} tolerance. Execution counts are measured, not inferred from layers: Qwen Boundary has 2 layers/10 branches versus 3/9 for Conditional and Static.

E.9 SEVEN-METHOD DEPLOYMENT TRADE-OFFS ACROSS FIVE TASKS

Figures 5 and 6 consolidate the seven-method deployment evaluation across five tasks—BoolQ, HellaSwag, GSM8K, MMLU (Hendrycks et al., 2021), and XSum (Narayan et al., 2018)—three model families, and two byte budgets. This gives a common 30-panel view of six cap-constrained methods and uncapped Full-LoRA, with separately measured no-adapter Base references shown for context. Utility is accuracy for BoolQ, HellaSwag, and MMLU, exact match for GSM8K, and ROUGE-L (Lin, 2004) for XSum. BoolQ reports median ITL; the auxiliary cohorts retain their recorded ITL statistic.

Across the seven-method panels, SIFT-LoRA lies on the observed utility–ITL frontier in 24 of 30 conditions: all 12 MMLU/XSum panels, 10 of the 12 BoolQ/HellaSwag panels, and two of the six GSM8K panels. It also has higher exact match and lower reported ITL than LoRA-Drop (shared) in all six GSM8K model–budget cells. Together, the five-task view shows that these deployment trade-offs extend beyond BoolQ and a single model family.

On BoolQ, SIFT-LoRA lies on all three 12.5% frontiers and on the Llama/Gemma frontiers at 25%. At 25%, it records 24.0–42.1% lower median ITL than LoRA-Drop (shared) for only 0.73–0.83 percentage points lower utility. On HellaSwag, SIFT-LoRA is on all three frontiers at 12.5% and on the Llama/Gemma frontiers at 25%. Its 12.5% ITL is 22.4–40.1% lower than LoRA-Drop (shared), and it achieves higher utility than Chen et al. (2025) in all six HellaSwag cells.

On GSM8K at the 12.5% cap, SIFT-LoRA lies on the seven-method frontier for Llama and Gemma. It achieves 83.472% and 81.350% exact match, respectively, exceeding Chen et al. (2025) by 0.910 and 0.758 percentage points at 14.432 and 27.584 ms reported ITL. Chen et al. (2025) have lower ITL on both models, so these are accuracy–latency trade-offs. At 25%, SIFT-LoRA reaches 82.411% and 81.425% exact match on Llama and Gemma, respectively, below Chen et al. (2025) by 0.152 and 0.076 points, while the method of Chen et al. (2025) remains faster. Chen et al. (2025) also achieve higher exact match and lower reported ITL on Qwen at both caps; SIFT-LoRA is therefore off the observed frontier in all three 25% panels.

MMLU and XSum provide the broadest additional evidence. SIFT-LoRA lies on the seven-method frontier in all six model–budget cells for each task. On MMLU/Llama at 25%, SIFT-LoRA reaches 69.36% accuracy at 14.91 ms, compared with PARA at 69.50% and 22.31 ms, giving 33.2% lower ITL for a 0.14-point utility difference. On XSum, Qwen/12.5% reaches 15.37 ROUGE-L at 18.31 ms, while Gemma/25% reaches 22.62 at 30.43 ms and exceeds LoRA-Squeeze in both utility and reported ITL. These results extend SIFT-LoRA's deployment trade-offs from classification and reasoning to broad knowledge evaluation and generation.

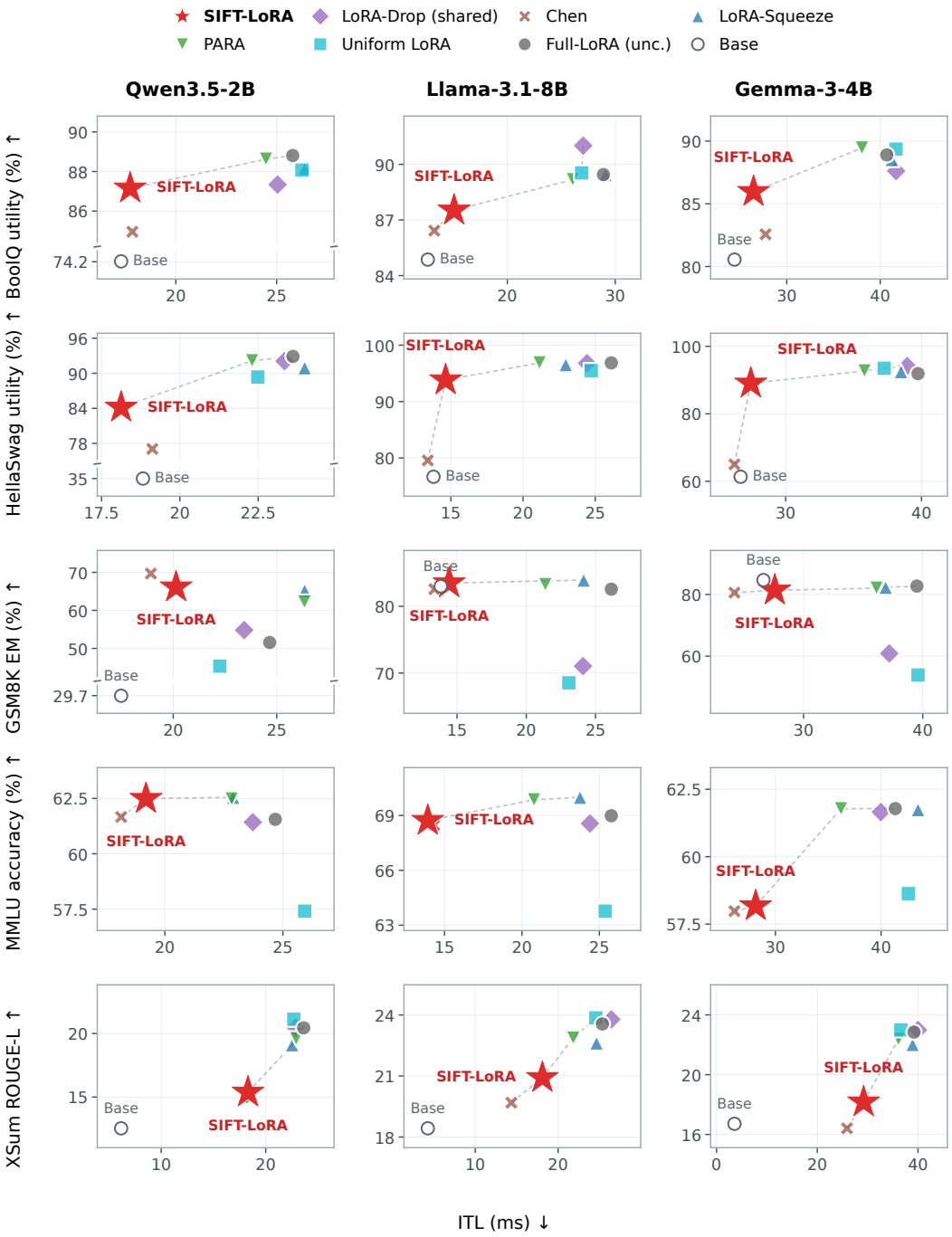


Figure 5: **Seven-method deployment trade-offs at the 12.5% byte cap.** Rows denote tasks and columns denote models. Marker colors, shapes, and sizes follow Figure 3. Guides connect the observed frontier within each seven-method cohort; uncapped Full-LoRA is included and the separate hollow-circle Base references are excluded. Utility is scaled by 100. BoolQ records median ITL; the auxiliary adapter cohorts report ITL. MMLU Base accuracy is 57.724%/68.291%/56.782% for Qwen/Llama/Gemma; generation ITL is N/A under teacher-forced multiple-choice scoring, so no Base latency point is drawn. XSum Base uses the supplied 6.165/4.269/3.560 ms coordinates. Qwen's first three y-axes are broken.

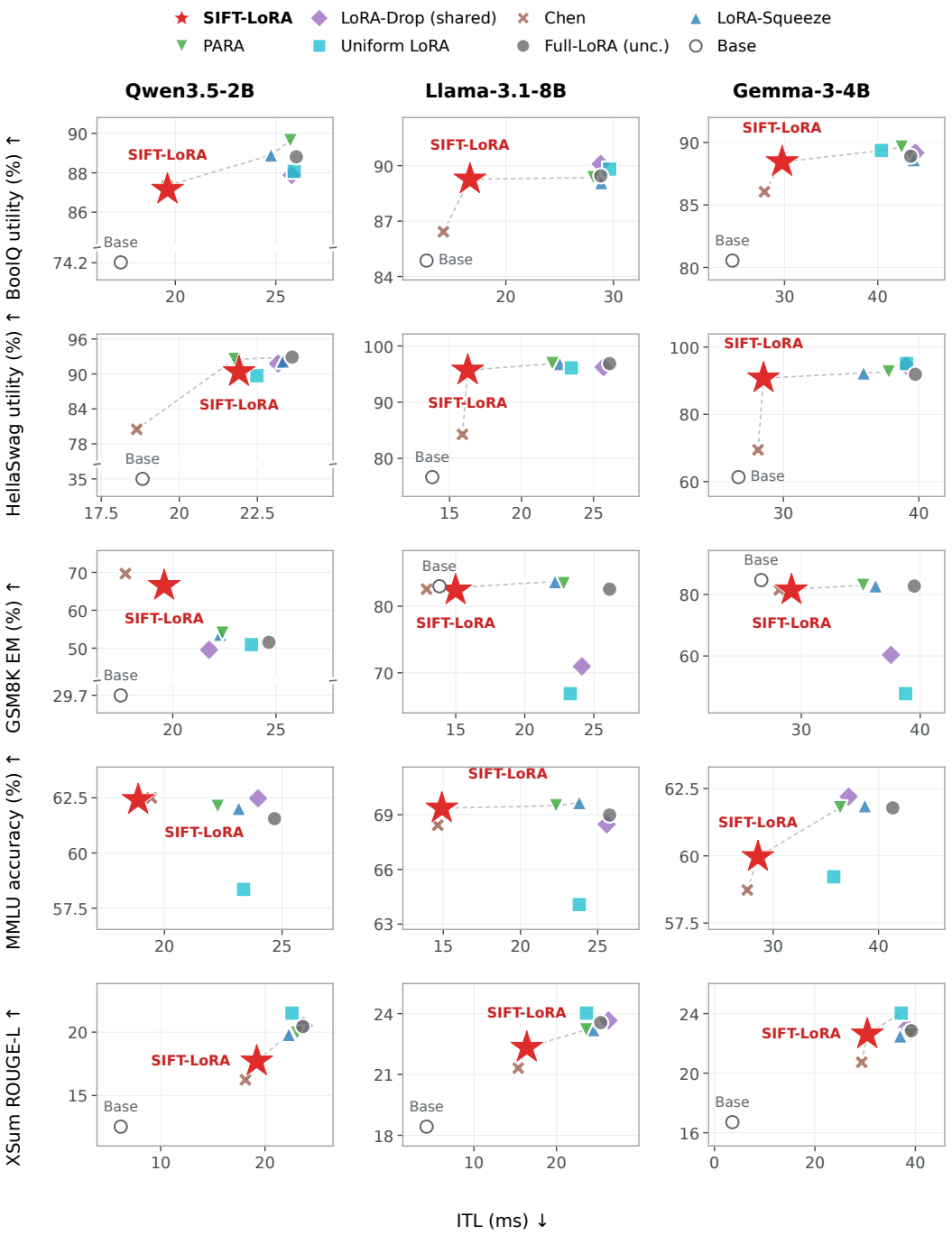


Figure 6: **Seven-method deployment trade-offs at the 25% byte cap.** The layout, axes, and marker styles match Figure 5. Frontiers are computed independently within each seven-method cohort, excluding Base. Base and Full-LoRA references are reused only where their recorded identity and coordinates agree; BoolQ retains distinct Full-LoRA measurements at each cap. MMLU Base accuracy is 57.724%/68.291%/56.782% for Qwen/Llama/Gemma, with generation ITL N/A under teacher-forced multiple-choice scoring. The hollow-circle XSum Base coordinates remain 6.165/4.269/3.560 ms. Utility is scaled by 100; BoolQ records median ITL and auxiliary cohorts report ITL.

E.10 FRESH CURRENT-SELECTOR COMPARISONS ACROSS FIVE TASKS AND TWO BUDGETS

Together with the BoolQ/12.5% results in Table 2, the fresh current-selector evaluation spans five tasks, three model families, and two byte budgets, giving 30 model–task–budget cells. Across this full grid, Conditional SIFT-KL improves utility over Static-KL in 26/30 cells and ties in two more; against Boundary-KL, it improves utility in 21/30 cells and ties in three. Thus conditional rescoring matches or exceeds frozen-score selection in 28/30 cells and contiguous-support selection in 24/30, while preserving the same post-training compilation setting. This grid uses its own parent checkpoints and evaluation records, separate from the seven-method study in Appendix E.9.

The 12.5% HellaSwag/GSM8K comparisons reinforce this pattern. Conditional achieves higher utility than Static-KL in all six cells and than Boundary-KL in five, while remaining on the three-selector utility–latency frontier in all six. The corresponding matched deployment records also pass the reported structural, linkage, runtime-parity, and execution checks with zero inactive-branch calls.

At 25%, the matched BoolQ/HellaSwag/GSM8K cohort shows the same utility advantage across a wider budget. Conditional improves over Static-KL in 8/9 cells and matches or exceeds Boundary-KL in 7/9. The largest gains over Static-KL occur on GSM8K: +14.33 percentage points on Qwen, +4.33 on Llama, and +6.67 on Gemma. Latency ordering varies by task and model, giving multiple useful operating points rather than a single latency–utility ordering.

The additional MMLU and XSum results extend the comparison to knowledge-intensive evaluation and summarization. Across these 12 cells, Conditional improves utility over Static-KL in nine and ties in two; it improves over Boundary-KL in nine. On MMLU, Conditional is highest or tied highest on both Llama and Gemma at both budgets. On XSum, it improves ROUGE-L over Static-KL in all six cells and over Boundary-KL in five. At Qwen/25%, Conditional reaches 18.676 ROUGE-L versus 17.490 for Static-KL, while Gemma/25% gains 2.525 and 2.622 ROUGE-L points over Static-KL and Boundary-KL, respectively.

Taken together, the fresh five-task grid shows a consistent advantage for conditional rescoring across models, budgets, and task types. The strongest effect is on utility: Conditional retains more of the parent's task behavior in the large majority of comparisons, while the resulting supports expose different latency–quality trade-offs at deployment. MMLU reports utility without generation ITL under teacher-forced multiple-choice scoring, and XSum reports generation ITL in Table 26.

Table 26: **Fresh current-selector utility and latency across tasks and budgets.** Utility is accuracy for BoolQ, HellaSwag and MMLU, exact match for GSM8K, and macro ROUGE-L for XSum, all scaled by 100. ITL is the recorded median in ms. Bold separately marks the highest utility and lowest ITL among the three selectors, including utility ties at the supplied precision. MMLU ITL is N/A under teacher-forced multiple-choice scoring. BoolQ/12.5% remains in Table 2.

			Conditional SIFT-KL		Static-KL		Boundary-KL	
Task	Budget	Model	Utility	ITL (ms)	Utility	ITL (ms)	Utility	ITL (ms)
BoolQ	25%	Qwen3.5-2B	87.706	20.067	87.523	**18.848**	**87.798**	20.263
		Llama-3.1-8B	**90.000**	17.301	89.908	16.141	**90.000**	**16.060**
		Gemma-3-4B	86.055	30.811	82.110	28.560	**87.431**	**28.189**
HellaSwag	12.5%	Qwen3.5-2B	**85.333**	**18.640**	82.667	18.954	83.667	18.979
		Llama-3.1-8B	**95.000**	15.200	94.333	**15.055**	94.667	15.224
		Gemma-3-4B	**88.000**	28.547	87.000	**27.589**	87.000	28.677
	25%	Qwen3.5-2B	**89.667**	19.503	88.333	**18.938**	88.000	19.177
		Llama-3.1-8B	95.333	15.995	**95.667**	**15.805**	95.333	16.306
		Gemma-3-4B	**92.667**	30.907	92.000	**28.141**	91.333	28.283
GSM8K	12.5%	Qwen3.5-2B	**64.000**	**19.277**	62.000	19.324	63.000	19.321
		Llama-3.1-8B	**88.667**	15.258	86.333	**14.980**	85.667	15.289
		Gemma-3-4B	82.333	28.193	81.333	**27.519**	**83.333**	28.377
	25%	Qwen3.5-2B	**58.667**	20.279	44.333	**19.105**	**58.667**	20.524
		Llama-3.1-8B	**89.333**	16.127	85.000	17.393	83.667	**16.060**
		Gemma-3-4B	**81.333**	**28.474**	74.667	30.622	81.000	30.482
MMLU	12.5%	Qwen3.5-2B	62.776	N/A	62.776	N/A	**62.783**	N/A
		Llama-3.1-8B	**69.725**	N/A	**69.725**	N/A	68.912	N/A
		Gemma-3-4B	**59.672**	N/A	58.530	N/A	59.358	N/A
	25%	Qwen3.5-2B	62.369	N/A	**62.533**	N/A	62.419	N/A
		Llama-3.1-8B	**70.104**	N/A	69.704	N/A	69.361	N/A
		Gemma-3-4B	**60.442**	N/A	59.180	N/A	60.307	N/A
XSum	12.5%	Qwen3.5-2B	**16.707**	5.741	16.124	**1.725**	15.470	1.729
		Llama-3.1-8B	**21.753**	4.672	21.473	**4.626**	20.258	4.682
		Gemma-3-4B	**18.555**	4.273	18.310	**4.114**	18.481	4.151
	25%	Qwen3.5-2B	**18.676**	1.799	17.490	**1.772**	17.327	1.774
		Llama-3.1-8B	23.066	5.296	22.782	**5.138**	**23.118**	5.162
		Gemma-3-4B	**22.091**	4.660	19.566	4.166	19.469	**4.094**

F FIXED-SUPPORT STATE CONTROLS

Unlike work on finding trainable sparse subnetworks (You et al., 2020; Frankle & Carbin, 2019; Damirchi et al., 2025), this study fixes support and compares copied with reset LoRA state, related to pruning-and-rewinding controls (Renda et al., 2020). The 2B and 9B records are separate studies, not a scaling comparison.

F.1 QWEN3.5-2B MATCHED SUPPORT–STATE CONTROL

One Qwen3.5-2B/HellaSwag parent at checkpoint 375 (parent seed 0) supplies the two SIFT-KL-selected supports. Table 3 reports all outcomes: reset means use child seeds 0/1/2; each inherited trajectory is deterministic. Within each cap, support, rank, bytes, optimizer, data order and horizon are fixed. The trajectory split has 300 examples; the disjoint endpoint split has 3,347. Continuation uses rank/alpha 16, dropout 0, AdamW (Loshchilov & Hutter, 2019), learning rate 2×10^{-4}, batch 20, maximum length 1,024, bf16, cosine decay, warmup .03 and 1,500 steps. Equation 8 uses logged anchors $0, 250, 500, 750, 1000, 1250, 1500$ and $H \in \{250, 500\}$; G_0 is step-0 utility minus base utility on a 0–1 scale. Reset endpoints include standard deviations.

The 12.5% pair shares export size 4,722,769 bytes under cap 5,460,181; the 25% pair shares 10,428,892 bytes under 10,920,362. These are separate state-control exports, not the primary ledger's artifacts. Inheritance improves early AULC at both caps while endpoint ordering remains mixed, isolating its main effect as a stronger initialization and higher early utility on the same sparse structure.

F.2 QWEN3.5-9B ONE-PARENT STATE CONTROL

The separate rank-16 Qwen3.5-9B (Qwen Team, 2026b) HellaSwag parent (checkpoint 375) is compiled at 12.5%; support $\{14, 15, 17\}$ uses 11,408,696/14,554,729 bytes (78.38%). One inherited and three reset DIRECT-SPARSE children continue for 1,500 steps, using 300 trajectory examples and

a disjoint 3,347-example endpoint. Table 27 retains the positive early gap and $+0.24$ pp endpoint difference. This auxiliary 9B control provides a consistent larger-model check of the inherited-state effect.

Table 27: **Qwen3.5-9B one-parent state control.** AULC entries are inherited-minus-mean-reset gaps on a 0–100 utility scale. Reset seeds measure child-continuation variability conditional on one parent.

Metric	Qwen3.5-9B
ΔAULC@250 / @500 (pp)	$+1.32$ / $+0.69$
Entry gap $V_I(0) - \bar{V}_D(0)$ (pp)	$+1.67$
Parent–child KL / agreement	.0692 / 94.33%
Support / export bytes / cap bytes	$\{14, 15, 17\}$ / 11,408,696 / 14,554,729
Endpoint, inherited / mean reset ($n_D = 3$)	97.22% / 96.98%
Dense-parent endpoint	97.70%

G Multi-Budget Compilation Diagnostic

One frozen Qwen3.5-2B/HellaSwag checkpoint is independently compiled at 6.25%, 12.5%, 25%, and 50% byte caps with calibration data, serializer, and dtype fixed. Table 28 reports the resulting physical exports; all four satisfy their requested caps.

Table 28: **Four exact-cap exports from one frozen checkpoint.** All recorded artifacts fit. Supports are independently compiled and need not be nested; these are not primary-grid artifacts or a utility–runtime frontier.

Cap (%)	Cap bytes	Export bytes	Cap Utilization (%)	Active layers
6.25	2,730,090	2,559,434	93.75	1/24
12.5	5,460,181	4,722,758	86.49	3/24
25	10,920,362	10,428,869	95.50	6/24
50	21,840,724	21,446,174	98.19	13/24

The supports are compiled independently and need not be nested. Together, the four exports demonstrate that a single frozen parent can be compiled across an $8\times$ range of deployment byte caps after training.

H Existing-Parent Exact-Cap Retrofit Pilot

This HellaSwag pilot measures incremental work *after* a same-task Full-LoRA parent exists. Qwen3.5-2B and Llama-3.1-8B-Instruct are compiled by exact-byte Conditional SIFT-KL at 12.5%; then fresh zero-initialized Tuned Direct-Sparse adapters train on the identical supports. Parent acquisition and training are excluded. Recovery is the first observed evaluation anchor reaching the inherited artifact's step-0 validation utility, without interpolation.

Table 29: **Exact-cap HellaSwag retrofit timing and physical size.** Both artifacts meet their caps. First hit is the observed recovery step; parent training is excluded. Times are rounded, while ratios use unrounded records.

Model	Cap bytes	Export bytes	Compile (s)	First hit (steps)	Recovery (s)	Ratio
Qwen	5,460,181	4,722,568	296.8	250	4,658.0	$15.69\times$
Llama	20,980,270	15,736,025	97.7	500	694.3	$7.10\times$

Qwen support $[1, 9, 12]$, cap utilization 86.49%; Llama support $[10, 11, 15]$, utilization 75.00%. Ratio = recovery/compilation time.

Compilation totals 394.5 s across Qwen and Llama, compared with 5,352.3 s for fresh recovery to the inherited starting utility; the ratio of unrounded totals is $13.57\times$. This is a post-parent comparison on the recorded supports, with one recovery trajectory per model. Gemma passes the cap audit but

is kept separate because its recovery run uses the canonical layer-wise-sparse recipe rather than the tuned recipe used for Qwen/Llama.

I Record Availability and Access Boundaries

The released source package provides the available result ledgers and aggregate summaries, plotting/analysis scripts, and supplied configuration and lineage information. `docs/RECORD_INVENTORY.md` lists package-relative entry points and recomputation boundaries; `docs/release/VERIFICATION_DETAILS.md` preserves file-level audits and access boundaries; `docs/release/HISTORICAL_DIAGNOSTICS.md` retains retired diagnostics.

The canonical controlled selector population is supplied as a full-precision 54-row ledger, together with the recorded structural/reload audit ledger. These records are sufficient to recompute the reported controlled-grid summaries, but per-example predictions and the historical adapter binaries needed to rerun the 54 physical reload checks are not packaged. The fresh BoolQ current-selector snapshot contains nine endpoint labels with supplied utility/runtime summaries; raw predictions and per-repetition timing traces are not included. The complete 21-endpoint historical BoolQ campaign is not released as a complete record-level ledger in this package.

The packaged MMLU and XSum current-selector files contain aggregate utility cells and available selector-level runtime ranges, with three Base utility references per task. Raw predictions and timing repetitions are not included. MMLU uses teacher-forced multiple-choice scoring and therefore has no generation ITL. XSum retains the recorded batch-size caveat; macro ROUGE-L cannot be recomputed from per-example predictions because those predictions are not packaged. The package does not contain a complete raw record set for the full 30-cell fresh current-selector grid in Appendix E.10.

Availability of parent/adapter binaries, historical environments, intermediate trajectories, and population-specific records is documented in the inventory; scientific scope remains defined in Appendices C–H. Missing raw records are not reconstructed from aggregates. Breadth summaries in the manuscript count reported cellwise outcomes from their original cohorts and should not be interpreted as pooled metric estimates or as claims that every underlying raw record is released.