CALIBUDGET: CALIBRATION-GUIDED SOURCE ALLOCATION FOR FIXED-BUDGET MIXED-REASONING ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Fixed-budget adaptation from heterogeneous data sources requires deciding not only how much data to use, but how much exposure each source receives. Size-proportional rules can crowd out small sources, whereas difficulty-only rules can chase noisy estimates or allocate residual budget to nearly saturated pools. We introduce CALIBUDGET, a floor-protected, reliability-aware integer allocator that treats source exposure as an explicit adaptation variable. From small train-internal calibration splits, it combines model need, post-floor availability, and bootstrap stability, then produces exact capacity-respecting quotas without changing the model, objective, or total budget. In a controlled setting combining mathematical and commonsense data, CALIBUDGET improves CommonAvg, FragileAvg, and MacroAvg over validation-error-with-floor, the strongest matched comparator, in all three paired LLaMA-2-7B LoRA+ runs. The respective mean gains are 0.56, 0.46, and 0.41 percentage points (pp). Overall increases by 0.18 pp, whereas MathAvg decreases by 0.20 pp, exposing a coverage–retention boundary rather than a uniform gain. CALIBUDGET changes only 1.14–1.42% of the source budget but improves performance in 15 of 24 comparisons across commonsense tasks and seeds. These results suggest that small changes in source quotas can matter; example-level selection can then determine which examples fill each quota. The code is available at `https://anonymous.4open.science/r/calibudget-anonymous-code-7470/`

1 INTRODUCTION

Modern language-model adaptation relies on heterogeneous collections whose sources differ in size, supervision style, answer format, difficulty, and relevance to the capabilities being acquired or retained (Wei et al., 2021; Sanh et al., 2021; Ouyang et al., 2022; Chung et al., 2024; Wang et al., 2022). In many adaptation rounds, the eligible pool exceeds what can be processed because of annotation, licensing, privacy, access, or compute constraints. Under such a cap, total data volume is fixed; the unresolved decision is how that budget should be distributed across sources. This exposure decision can matter even when the model, loss, and optimizer are unchanged, because aggregate accuracy can conceal underexposure of small, difficult, or format-distinct groups.

We study this setting as *fixed-budget multi-source adaptation*. A heterogeneous candidate pool is partitioned into predefined sources, a companion training component is held constant, and the remaining budget must be converted into exact integer source quotas. Standard policies expose complementary failure modes. Pooled or size-proportional sampling favors large sources. Equal allocation protects coverage but ignores differences in model need and usable capacity. Difficulty-only allocation responds to model error, yet can overreact to noisy source-level estimates or prioritize sources with little residual capacity. A practical allocator must therefore reconcile *need*, *availability*, and *estimation reliability* while satisfying hard budget, capacity, and minimum-exposure constraints.

Existing work optimizes data composition through continuous mixture weights, online sampling, proxy objectives, or example-level ranking (Xie et al., 2023a;b; Xia et al., 2024; Liu et al., 2024b; Lin et al., 2024). These mechanisms are valuable, but they do not by themselves specify an exact feasible quota vector under source capacities and coverage requirements. We isolate this preceding decision:

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.

given predefined sources and an exact total budget, how many examples should be drawn from each source? This source-level allocation layer is orthogonal to within-source quality or diversity selection, and isolating it makes the effect of exposure directly measurable and auditable.

We introduce **CALIBUDGET**, a calibration-guided integer allocator for fixed-budget multi-source adaptation. Figure 1 summarizes the design. CALIBUDGET reserves a small train-internal calibration split from each source and estimates three explicit signals: current model need, residual candidate availability, and the bootstrap stability of the need estimate. A floor first guarantees feasible minimum exposure; a utility-weighted residual stage then refines the allocation, followed by capacity-aware rounding that satisfies the budget exactly. The method neither encodes math- or commonsense-specific semantics nor changes the training objective. Its output is an inspectable quota vector determined by source statistics and feasibility constraints.

We evaluate this allocation layer in a controlled mixed-reasoning instantiation. A mathematical reasoning training subset is held fixed, while a capped budget is allocated across heterogeneous commonsense sources spanning yes/no, physical, social, adversarial-completion, and science reasoning. All paired comparisons use the same backbone, LoRA+ configuration, optimizer, schedule, parser, and random seed (Yu et al., 2023; Hayou et al., 2024). Across three LLaMA-2-7B seeds, CALIBUDGET improves CommonAvg, the prespecified fragile-task aggregate FragileAvg, and MacroAvg over the strongest matched validation-error-with-floor baseline in every seed, with mean gains of 0.56, 0.46, and 0.41 pp. Overall increases by 0.18 pp, whereas MathAvg decreases by 0.20 pp; we therefore characterize the result as a coverage-oriented trade-off, not a uniform improvement. The allocation changes only 1.14–1.42% of the 10,000-example source budget yet improves 15 of 24 comparisons across commonsense tasks and seeds, showing that small, structured quota corrections can affect downstream task outcomes under this controlled setting. Our contributions are threefold:

- We formulate capped heterogeneous adaptation as an exact source-quota problem, separating the choice of total data volume from the distribution of a fixed budget and distinguishing quota construction from continuous mixture weighting and example ranking.
- We develop CALIBUDGET, an auditable allocator that decomposes source exposure into a policy-controlled floor and a reliability-aware residual correction, while enforcing source capacities and exact integer feasibility.
- Under a strictly matched mixed-reasoning protocol, we show a consistent allocation-sensitivity pattern on source-balanced metrics and characterize its retention trade-off, mechanism diagnostics, operating-point sensitivity, and cross-backbone boundary.

2 METHOD

2.1 PROBLEM FORMULATION

CALIBUDGET addresses fixed-budget multi-source adaptation. Let $D_s = \bigcup_{k=1}^{K} D_k$ be a heterogeneous candidate pool partitioned into K predefined sources by dataset identity or metadata; each source is one allocation group. Each source provides a disjoint train-internal calibration split C_k. Let $\widetilde{T}$ denote a fixed companion training set included unchanged across methods. Given a pretrained model p_{θ_0}, the allocator must select exactly B examples from D_s before learning an update $\Delta\theta$.

The selected multi-source subset is $Q = \bigcup_{k=1}^{K} Q_k$, where $Q_k \subseteq D_k$ and $|Q| = B$. The final adaptation set is

$$\widetilde{D} = \widetilde{T} \cup Q. \tag{1}$$

Across paired comparisons, the model, adaptation backend, optimizer, schedule, evaluation pipeline, and random seed are fixed. The methodological object is therefore the quota vector $(q_1, \ldots, q_K)$ with $q_k = |Q_k|$: methods differ only in how they distribute the source budget B.

We instantiate this formulation in mixed-reasoning adaptation: $\widetilde{T}$ is a fixed mathematical training subset, and D_s is the union of commonsense sources. The allocator itself uses only source-level calibration statistics and capacity constraints, not math- or commonsense-specific semantics.

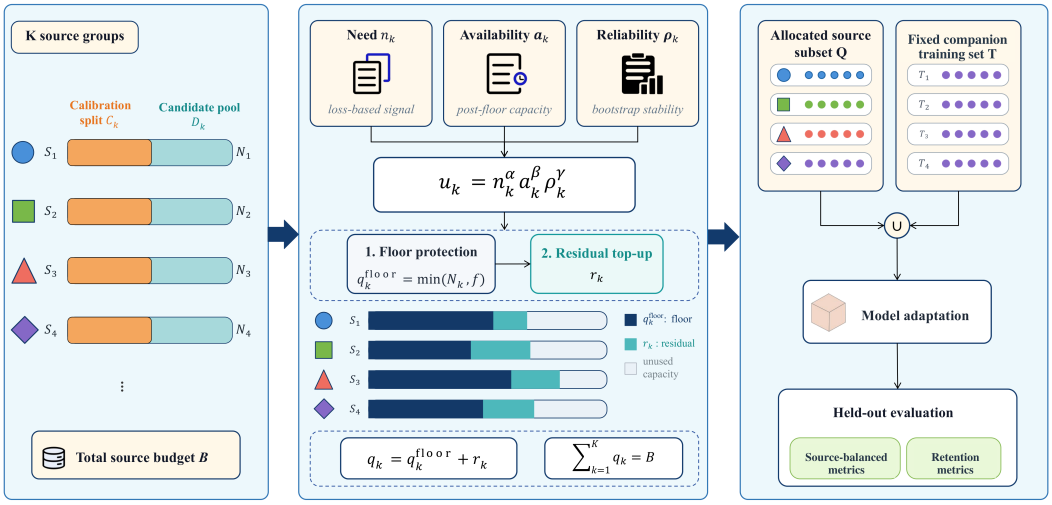


Figure 1: Overview of CALIBUDGET. A heterogeneous multi-source pool is split into train-internal calibration data and candidate pools. Floor protection guarantees minimum exposure; post-floor availability, source need, and estimate reliability define the utility used for residual top-up, producing capacity-respecting integer quotas under budget B. The allocated source subset is combined with a fixed companion training set and adapted under an otherwise unchanged training protocol.

2.2 Calibration-Aware Source Utility

CALIBUDGET weights residual allocation using three source-level factors: current model need, post-floor availability, and the stability of the need estimate. Using the disjoint calibration split C_k, we define

$$u_k = n_k^\alpha a_k^\beta \rho_k^\gamma, \tag{2}$$

where n_k is calibration need, a_k is a transformed residual-availability signal, and ρ_k is estimate reliability.

For calibration example $i \in C_k$, let

$$z_i = \ell_i + \mathbf{1}\{\widehat{y}_i \neq y_i\}, \tag{3}$$

where ℓ_i is the mean per-token teacher-forced response NLL and $\widehat{y}_i$ is the greedily decoded answer under the fixed task parser. The indicator adds a unit penalty for a parser-level answer error, so z_i combines graded model fit with discrete answer correctness. Let $m_k = |C_k|^{-1} \sum_{i \in C_k} z_i$, and let σ_k be the standard deviation of 200 bootstrap means from C_k (Efron, 1992). With a small numerical constant ϵ, the frozen implementation uses

$$n_k = \frac{\max(m_k, \epsilon)}{K^{-1} \sum_j \max(m_j, \epsilon)}, \quad \rho_k = (1 + \sigma_k)^{-1}, \quad a_k = \sqrt{\max(N_k - \min(N_k, f), \epsilon)}. \tag{4}$$

Each calibration split contains 100 held-out training examples, and $(\alpha, \beta, \gamma) = (1, 0.5, 1)$ is frozen before confirmatory runs. Because a_k already applies a square-root transform, raw residual capacity enters the utility with exponent $1/4$. This deliberately tempers pool-size dominance while reducing preference for nearly exhausted sources.

The three factors address different failure modes. Proportional allocation reflects source size but not model need; validation-error allocation captures need but treats finite calibration estimates as exact and ignores residual capacity; equal allocation protects exposure but suppresses informative source differences. CALIBUDGET combines these signals only after a shared coverage floor has been assigned. Here, "calibration" refers to estimating allocation statistics on held-out training data, not to post-hoc probability calibration.

The mechanism is intentionally source-level rather than example-level. Ranking and influence methods answer which examples are preferable within a candidate set and usually require proxy or gradient computation. CALIBUDGET instead asks how many examples each source should contribute.

Uniform within-source sampling isolates that decision; quality, diversity, or influence selectors can be composed within the resulting quotas.

2.3 FLOOR-PROTECTED RESIDUAL ALLOCATION

Let $N_k = |D_k|$ be the candidate size of source k. CALIBUDGET first assigns every feasible source a floor quota

$$q_k^{\text{floor}} = \min(N_k, f), \tag{5}$$

where f is chosen such that $\sum_k q_k^{\text{floor}} \le B$. The floor encodes a minimum-coverage requirement and prevents small or format-distinct sources from being crowded out. The residual budget is

$$R = B - \sum_{k=1}^{K} q_k^{\text{floor}}. \tag{6}$$

Let $c_k = \max(N_k - q_k^{\text{floor}}, 0)$ denote post-floor residual capacity. CALIBUDGET distributes R over active sources according to normalized utility:

$$\widetilde{r}_k = R \frac{u_k}{\sum_{j:c_j>0} u_j}. \tag{7}$$

These residual shares are real-valued. Capacity-aware largest-remainder rounding converts them to integer allocations, enforces $0 \le r_k \le c_k$, and iteratively redistributes overflow to sources with remaining capacity. The final source quota is

$$q_k = q_k^{\text{floor}} + r_k, \qquad \sum_{k=1}^{K} q_k = B. \tag{8}$$

After quotas are fixed, we sample q_k examples uniformly without replacement from each source and set Q_k to the selected subset.

Allocation guarantee. If $\sum_k \min(N_k, f) \le B \le \sum_k N_k$, capped largest-remainder redistribution returns integer quotas satisfying

$$\min(N_k, f) \le q_k \le N_k, \qquad \sum_k q_k = B. \tag{9}$$

Thus every feasible source receives its prescribed minimum exposure and the source budget is met exactly. The guarantee concerns feasibility and coverage only; it does not imply that downstream performance is monotone in quota or that the resulting vector is globally optimal.

This floor–residual decomposition separates *policy* from *preference*. The floor determines the admissible coverage prior, whereas the calibrated utility refines only the remaining budget. If the floor is too small, high-error sources can dominate and small sources can be under-covered; if it is too large, the procedure approaches equal allocation and leaves little scope for model-dependent refinement. In our experimental instantiation, $B = 10{,}000$ and $f = 1150$, so the floor assigns 9,069 examples and leaves $R = 931$ for utility-based correction. This conservative operating point tests whether calibration adds value beyond a dominant shared coverage prior. A prespecified $f = 950$ audit evaluates sensitivity without retuning.

2.4 PARAMETER-EFFICIENT ADAPTATION

After constructing Q, we train on $\widetilde{D} = \widetilde{T} \cup Q$ with LoRA+. For a pretrained weight matrix W, LoRA learns

$$W' = W + \Delta W, \qquad \Delta W = \frac{s}{r} B_L A_L, \tag{10}$$

where A_L and B_L are rank-r factors and s is the scaling coefficient. LoRA+ assigns different learning rates to the two factors. We update attention and MLP projections and optimize the standard autoregressive objective:

$$\mathcal{L}(\Delta\theta) = -\mathbb{E}_{(x,y)\sim\widetilde{D}} \sum_{t=1}^{|y|} \log p_{\theta_0+\Delta\theta}(y_t \mid x, y_{<t}). \tag{11}$$

Algorithm 1 CALIBUDGET

Require: Candidate sources $\{D_k\}_{k=1}^K$, calibration splits $\{C_k\}_{k=1}^K$, budget B, floor f

for $k = 1, \ldots, K$ **do**
  $q_k^{\text{floor}} \leftarrow \min(|D_k|, f)$; $c_k \leftarrow |D_k| - q_k^{\text{floor}}$
  Estimate n_k and ρ_k from C_k; set $a_k \leftarrow \sqrt{\max(c_k, \epsilon)}$
  $u_k \leftarrow n_k^\alpha a_k^\beta \rho_k^\gamma$
end for
$R \leftarrow B - \sum_k q_k^{\text{floor}}$
Set $\widetilde{r}_k = R u_k / \sum_{j:c_j>0} u_j$ for active sources
Apply capacity-aware largest-remainder rounding, $0 \le r_k \le c_k$
Sample $q_k = q_k^{\text{floor}} + r_k$ examples from each D_k
return $Q = \bigcup_k Q_k$ with $\sum_k q_k = B$

		Retention	Source-balanced		Aggregate	
Method	Allocation signal	Math (%)	Common (%)	Fragile (%)	Macro (%)	Overall (%)
Pooled uniform	pooled random	30.43	69.98	67.57	62.07	50.21
Proportional	source size	30.72	69.48	66.84	61.72	50.10
Equal	equal source quota	30.51	71.09	71.86	62.97	50.80
Floor-Sqrt	floor + $\sqrt{\text{residual size}}$	30.47	71.03	71.19	62.92	50.75
Val-error+floor	floor + calibration error	**30.91**	<u>71.31</u>	<u>72.15</u>	<u>63.23</u>	<u>51.11</u>
CALIBUDGET	floor + $na^{1/2}\rho$	30.72	**71.87**	**72.62**	**63.64**	**51.29**

Table 1: Three-seed mean performance (%) on LLaMA-2-7B under matched data budgets, adaptation backend, and evaluation. Bold denotes the best descriptive mean; underlining marks the strongest matched comparator when it is runner-up.

No source-specific loss reweighting is introduced: once examples are selected, all are trained identically and source exposure is controlled only by the quotas. LoRA+ provides the controlled adaptation backend in our experiments, but CALIBUDGET does not use its internal parameterization; compatibility with other training regimes is conceptually direct but remains to be validated empirically.

3 Experiments

We organize the evaluation around one confirmatory paired comparison, followed by mechanism diagnostics and transfer audits. The secondary analyses probe mechanism and boundary conditions; none is used to select or revise the frozen method.

3.1 Experimental Protocol

Training and budgets. The confirmatory protocol uses LLaMA-2-7B (Touvron et al., 2023) with LoRA+ (Hayou et al., 2024) for three epochs: rank $r = 128$, scaling $s = 128$, zero adapter dropout, geometric-mean factor learning rate 2×10^{-5} with ratio 8, effective batch size 32, bf16, 1,024-token context, cosine decay, 3% warmup, and zero weight decay. Adapters cover attention and MLP projections. We evaluate the final checkpoint, with no test-based checkpoint or allocation selection. The fixed companion set contains 100,000 MetaMath examples (Yu et al., 2023); the eight-source commonsense pool receives exactly $B = 10{,}000$ examples. Only source quotas and the corresponding uniformly sampled examples vary across methods, isolating allocation from changes in model capacity, parameterization, optimization, or total supervision.

Evaluation and baselines. We evaluate MATH and GSM8K (Hendrycks et al., 2021; Cobbe et al., 2021) together with BoolQ, PIQA, Social IQa, HellaSwag, WinoGrande, ARC-Easy, ARC-Challenge, and OpenBookQA (Clark et al., 2019; Bisk et al., 2020; Sap et al., 2019; Zellers et al., 2019; Sakaguchi et al., 2020; Clark et al., 2018; Mihaylov et al., 2018). MathAvg averages the two

mathematical tasks, CommonAvg averages the eight commonsense tasks, and MacroAvg averages all ten tasks; Overall gives equal weight to MathAvg and CommonAvg. FragileAvg is a prespecified coverage-sensitive diagnostic over Social IQa, ARC-Easy, ARC-Challenge, and OpenBookQA. Unweighted task means prevent large evaluation sets from silently dominating the aggregate and align evaluation with the source-coverage motivation. All methods share fixed task parsers and scorers. Baselines are pooled-uniform, proportional, equal, Floor-Sqrt, and validation-error-with-floor allocation, spanning pooled, size-driven, coverage-driven, and difficulty-driven quota rules. The last baseline shares CALIBUDGET's calibration split and floor but allocates the residual budget by calibration error alone, making it the closest matched comparator. DoReMi, LESS, and RHO-1 (Xie et al., 2023a; Xia et al., 2024; Lin et al., 2024) require proxy, gradient, or token-level decisions and are not direct integer-quota substitutions; we do not claim to outperform them, and they can be composed with source quotas.

Controls. The CALIBUDGET configuration was frozen before the confirmatory comparison. Calibration examples come only from the training pool and are excluded from both adaptation and evaluation. Paired runs match the model, LoRA+ backend, fixed companion subset, total source budget, parser, and random seed; primary results average seeds 42–44. Exploratory variants and cross-backbone extensions are reported only as diagnostics and do not revise the frozen method.

3.2 Main Paired Comparison

Table 1 reports the confirmatory LLaMA-2-7B comparison. CALIBUDGET achieves the highest CommonAvg, FragileAvg, MacroAvg, and Overall, while validation-error-with-floor retains the highest MathAvg. Relative to this strongest matched comparator, CALIBUDGET gains $+0.56$, $+0.46$, $+0.41$, and $+0.18$ pp on the four former metrics and loses 0.20 pp on MathAvg. The central result is therefore not a uniform Pareto improvement: CALIBUDGET consistently improves CommonAvg, FragileAvg, and MacroAvg across seeds, with a mean MathAvg trade-off.

Table 2 resolves the mean result by seed. CommonAvg, FragileAvg, and MacroAvg improve in all three paired runs. Overall declines only on seed 42, where the MathAvg reduction is also largest; seeds 43–44 improve on both MathAvg and Overall. With only three seeds, these patterns are descriptive rather than large-sample statistical evidence, but their paired consistency on these metrics supports the primary allocation-sensitivity claim.

Seed	Math	Common	Fragile	Macro	Overall
42	-1.12	$+0.49$	$+0.59$	$+0.16$	-0.32
43	$+0.27$	$+0.38$	$+0.32$	$+0.36$	$+0.33$
44	$+0.26$	$+0.81$	$+0.48$	$+0.70$	$+0.53$
Mean	-0.20	$+0.56$	$+0.46$	$+0.41$	$+0.18$
SD	0.80	0.22	0.14	0.27	0.44

Table 2: Paired performance differences (CALIBUDGET minus Val-error+floor, pp). Differences are computed before display rounding; SD is across seeds.

3.3 Mechanism and Selection Diagnostics

Reliability and the matched comparator. Removing ρ_k reduces CommonAvg, FragileAvg, MacroAvg, and Overall by 0.52, 0.95, 0.44, and 0.33 pp, respectively; the full rule is better on the first three metrics in every seed. This pattern is consistent with the intended role of ρ_k: discounting source-need estimates that fluctuate under bootstrap resampling instead of treating finite calibration means as equally trustworthy. Because the utility terms interact through normalization and quota rounding, the ablation is evidence for the reliability-aware rule rather than an isolated causal estimate of ρ_k. The validation-error-with-floor baseline remains stringent: its higher MathAvg but lower CommonAvg, FragileAvg, and MacroAvg illustrate the central coverage–retention boundary.

Guarding against seed-level selection. Table 3 records exploratory normalization variants that improve seed-42 FragileAvg, but none was promoted. Variant B remains below validation-error-with-floor on MathAvg, the recovery phase does not remove the trade-off, and reusing the same selected indices does not reproduce the gain at seed 43. Reporting these negative checks prevents a favorable single-seed variant from replacing the prespecified rule post hoc.

3.4 Allocation Geometry and Task-Level Effects

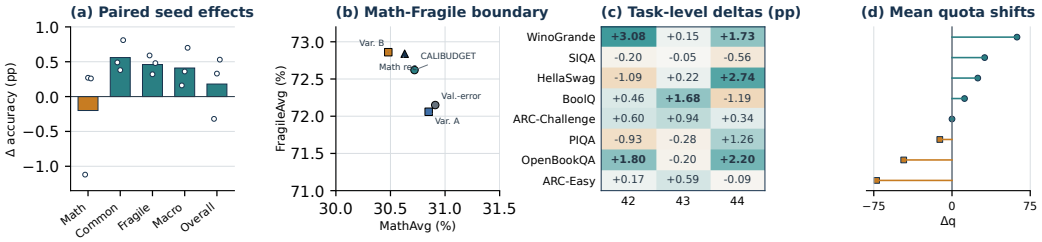


Figure 2: Diagnostics under the frozen comparison: (a) paired seed effects relative to validation-error-with-floor; (b) the observed MathAvg–FragileAvg boundary, including exploratory variants; (c) task-level accuracy differences (pp); and (d) mean source-quota shifts under the 10,000-example budget. No diagnostic panel selects a method or hyperparameter.

Model	Method	MathAvg (%)	CommonAvg (%)	FragileAvg (%)	MacroAvg (%)	Overall (%)
Qwen2.5-7B	Val-error+floor	$65.77_{\pm 0.30}$	$84.95_{\pm 0.29}$	$88.42_{\pm 0.37}$	$81.12_{\pm 0.29}$	$75.36_{\pm 0.30}$
	CALIBUDGET	$66.13_{\pm 0.31}$	$85.23_{\pm 0.21}$	$88.77_{\pm 0.32}$	$81.41_{\pm 0.18}$	$75.68_{\pm 0.19}$
LLaMA-3.1-8B	Floor-Sqrt	$60.77_{\pm 0.37}$	$82.99_{\pm 0.17}$	$84.60_{\pm 0.42}$	$78.54_{\pm 0.07}$	$71.88_{\pm 0.11}$
	CALIBUDGET	$60.72_{\pm 0.55}$	$82.97_{\pm 0.19}$	$84.57_{\pm 0.26}$	$78.52_{\pm 0.26}$	$71.84_{\pm 0.37}$

Table 4: Cross-backbone audits. Qwen2.5-7B (Qwen et al., 2025) averages three independently constructed adaptive allocations; LLaMA-3.1-8B (Grattafiori et al., 2024) averages three training seeds while holding each method's seed-42 allocation fixed. Comparators are matched within each audit. Main values are mean accuracy (%); subscripts report SD (pp).

Diagnostic	Math	Common	Fragile	Macro	Overall
Frozen references, seed 42					
CALIBUDGET	30.20	71.29	71.79	63.07	50.74
Val-error+floor	31.32	70.80	71.20	62.91	51.06
Exploratory normalization variants, seed 42					
Group-norm A	30.85	71.46	72.06	63.34	51.16
Group-norm B	30.48	71.84	72.86	63.57	51.16
Group-norm C	30.81	71.22	71.49	63.14	51.02
B + recovery	30.63	71.49	72.84	63.32	51.06
Fixed-index stability check					
B indices, seed 43	30.53	70.92	72.40	62.84	50.73

Table 3: Exploratory seed-42 normalization variants and a fixed-index seed-43 check (%). Internal variants A–C alter source normalization; recovery adds a short retention-oriented phase. None revises the frozen CALIBUDGET configuration.

Figure 2 connects aggregate behavior to the underlying intervention. Relative to validation-error-with-floor, CALIBUDGET reallocates only 142, 135, and 114 of 10,000 source examples across seeds 42–44 (1.42%, 1.35%, and 1.14%); no source changes by more than 76 examples. The performance differences therefore arise from a localized redistribution above a largely shared floor rather than from wholesale corpus replacement. Across the eight commonsense tasks, CALIBUDGET improves 15 of 24 task–seed comparisons, with recurrent gains on WinoGrande, OpenBookQA, ARC-Challenge, and HellaSwag. Quota and accuracy changes are nevertheless non-monotonic (Pearson $r = -0.005$; Spearman $\rho = -0.098$): some tasks improve after receiving fewer or unchanged examples. This behavior is plausible in multi-source instruction tuning because examples can transfer across skills and interact through shared parameters and the fixed mathematical component. The utility should therefore be interpreted as a joint allocation signal, not as a per-task treatment-effect estimator. This source-level role distinguishes CALIBUDGET from quality-, diversity-, and influence-based instance selectors (Liu et al., 2024b; Bukharin et al., 2024; Xia et al., 2024), which can operate within allocated quotas.

3.5 Robustness and Transfer Audits

The matched LLaMA-2 study supports the primary claim. The following frozen-rule audits test whether the observed allocation effects transfer and are interpreted as boundary evidence rather than as opportunities to select a new method.

Adaptive Qwen2.5 audit. The Qwen2.5 descriptive means favor CALIBUDGET on all five aggregates (Table 4). A paired bootstrap over the ten task-level scores yields deltas of $+0.35$ pp

on MathAvg (95% CI $[-0.38, +1.02]$), $+0.28$ pp on CommonAvg ($[-0.16, +0.69]$), $+0.29$ pp on MacroAvg ($[-0.17, +0.69]$), and $+0.32$ pp on Overall ($[-0.21, +0.76]$). Every interval crosses zero, and seed-level Overall differences are -0.23, $+0.61$, and $+0.57$ pp. Qwen2.5 therefore provides directionally consistent but seed-variable evidence, not backbone-invariant confirmation.

Fixed-allocation LLaMA-3.1 audit. The LLaMA-3.1 rows are effectively at parity. Unlike the adaptive Qwen2.5 audit, this experiment transfers each method's seed-42 allocation across three training seeds instead of recalibrating quotas for the new backbone. This near-parity result narrows the supported scope: because need is model-dependent, a quota vector estimated for one backbone should not be assumed to transfer without recalibration.

Modern-task and stability audits. On a frozen 320-prompt panel spanning BBH, GSM-Plus, HumanEval, IFEval, MATH-500, and MBPP (Suzgun et al., 2023; Li et al., 2024; Chen et al., 2021; Zhou et al., 2023; Lightman et al., 2024; Austin et al., 2021), strict ModernMacro deltas are $+0.61$ pp for Qwen2.5 (95% CI $[-1.22, +2.34]$) and $+0.17$ pp for LLaMA-2 ($[-0.69, +1.04]$); all intervals cross zero; CodeMacro differences are zero. The panel broadens the evaluation but does not establish modern-task adaptation gains. By contrast, the allocation is stable across three independently constructed calibration splits: CALIBUDGET preserves quota ordering (Spearman $\rho = 1.000$; Pearson $r \geq 0.9996$), with mean quota SD 1.47 and maximum range four examples, whereas validation-error-with-floor is less stable (mean SD 5.97, range 23, minimum $\rho = 0.790$).

Floor sensitivity. Lowering f from 1,150 to 950 expands the residual budget from 931 to 2,400 examples. At seed 42, CALIBUDGET retains relative gains over its matched comparator on CommonAvg ($+0.58$ pp), FragileAvg ($+0.57$ pp), MacroAvg ($+0.44$ pp), and Overall ($+0.23$ pp), but absolute performance falls below the prespecified continuation criterion. We stop before seeds 43–44. The relative ordering persists, but the lower-floor operating point is not sufficiently robust to promote, indicating that the floor is a consequential design constraint rather than an incidental constant.

3.6 Interpretation and Limitations

The matched LLaMA-2 LoRA+ experiments show that performance can depend on how a fixed source budget is allocated. Small, auditable quota changes improve CommonAvg, FragileAvg, and MacroAvg in all three seeds, while validation-error-with-floor retains slightly higher mean MathAvg. The reliability ablation supports the complete allocation rule, while the calibration-split audits show stable quotas. Transfer evidence is mixed: Qwen2.5 gains vary across seeds, LLaMA-3.1 performs similarly to its comparator under fixed allocations, and confidence intervals on the modern-task panel include zero. Lowering the floor preserves relative gains, but absolute performance falls below the same prespecified continuation criterion. Table 6 in Appendix G summarizes the supported claims and their limits.

Source capacities, the common floor, and the exact budget define feasible integer quotas; calibrated utility distributes the remaining budget. This design is useful when source groups are meaningful, candidate data exceed the training budget, minimum exposure is desirable, and train-internal calibration examples are available. It is less appropriate when all high-quality data can be used, source labels are arbitrary, or variation within sources dominates differences between them.

The mean MathAvg decrease accompanies gains in CommonAvg, FragileAvg, and MacroAvg, illustrating a trade-off between coverage and retention. Whether this trade-off is acceptable should be decided before evaluation. Source partitions and floors should be specified before test evaluation, rather than adjusted to improve downstream scores. Each quota can be traced to the floor, calibrated source statistics, remaining capacity, and deterministic rounding. Saving these quantities with selected indices and quota hashes allows exposure changes to be inspected. Capacity checks reject infeasible requests before training, and quality or diversity selection can be applied within each source.

The main study uses one setting combining mathematical and commonsense data, three seeds, predefined sources, and LoRA+; broader training domains and full-parameter tuning may yield different relationships between allocation and performance. CALIBUDGET does not model quality, diversity, contamination, or redundancy within sources, and the mathematical component remains fixed.

The allocation guarantee covers exact integer feasibility and minimum coverage, not optimality or worst-group generalization. The need statistic uses teacher-forced NLL and parser-based correctness, while the bootstrap term measures sampling stability rather than full epistemic uncertainty. All primary allocations use only training-pool metadata and train-internal calibration examples; test results are not used to select quotas, hyperparameters, or checkpoints. We also retain configurations, seeds, parser versions, negative audit results, and per-task outputs to distinguish confirmatory comparisons from exploration. Future work could incorporate calibration uncertainty into quotas and jointly budget the mathematical and heterogeneous source components.

4 Related Work

Data mixtures and source-level budgeting. Data composition is a central design variable in pretraining and instruction tuning (Longpre et al., 2023; Renduchintala et al., 2024; Li et al., 2025; Shin et al., 2026). DoReMi learns continuous domain-mixture weights through proxy training; RegMix and Data Mixing Laws predict promising mixtures from small-scale runs; and importance resampling estimates source or instance relevance to a target distribution (Xie et al., 2023a; Liu et al., 2025; Ye et al., 2024; Xie et al., 2023b). Recent instruction-tuning work further combines mixture design with submodular selection, scaling-law prediction, or dynamically updated sampling probabilities (Renduchintala et al., 2024; Li et al., 2025; Shin et al., 2026). At finer granularity, DEITA filters by complexity, quality, and diversity; QDIT studies quality–diversity trade-offs; LESS estimates target-specific gradient influence; and RHO-1 selects tokens online (Liu et al., 2024b; Bukharin et al., 2024; Xia et al., 2024; Lin et al., 2024). CALIBUDGET addresses an orthogonal discrete decision: before adaptation, it maps a finite budget to capacity-respecting integer quotas over predefined sources. Its floor guarantees feasible exposure and its rounding step satisfies the budget exactly; any example-level selector can subsequently rank candidates within each quota. Uniform within-source sampling is therefore a deliberate control that isolates source allocation.

Coverage, uncertainty, and adaptation backends. Group-robust learning improves minority-group performance by changing training weights or worst-group objectives (Sagawa et al., 2020); CALIBUDGET instead acts once before optimization, so its formal guarantee concerns integer feasibility and minimum exposure rather than worst-group risk. Its reliability term is a bootstrap stability discount on a source-level need statistic, not predictive probability calibration or a full uncertainty estimate under distribution shift (Efron, 1992; Guo et al., 2017; Ovadia et al., 2019). Parameter-efficient methods update adapters, prefixes, or low-rank factors (Houlsby et al., 2019; Li & Liang, 2021; Hu et al., 2022; Dettmers et al., 2023); AdaLoRA reallocates rank, LoRA+ changes factor-wise learning rates, and DoRA decomposes weight magnitude and direction (Zhang et al., 2023; Hayou et al., 2024; Liu et al., 2024a). These methods allocate parameter or optimization capacity, whereas CALIBUDGET allocates training examples and adds no PEFT module. We use LoRA+ as a controlled backend for mathematical and commonsense sources (Cobbe et al., 2021; Hendrycks et al., 2021; Yu et al., 2023; Clark et al., 2019; Bisk et al., 2020; Sap et al., 2019; Zellers et al., 2019; Sakaguchi et al., 2020; Clark et al., 2018; Mihaylov et al., 2018); generalization beyond the tested models, protocol, and source partition remains an empirical question.

5 Conclusion

CALIBUDGET makes source exposure a first-class decision in fixed-budget adaptation. A minimum-coverage floor and reliability-aware residual correction yield exact, auditable quotas. Under matched LLaMA-2 LoRA+ controls, reallocating only 1.14–1.42% of the source budget consistently improves CommonAvg, FragileAvg, and MacroAvg, while a small MathAvg decrease exposes the coverage–retention boundary. Transfer audits indicate that calibrated quotas remain model- and operating-point-dependent, positioning exact source budgeting as an adaptation control.

Reproducibility Statement

Section 2 specifies the allocator and feasibility guarantee, and Section 3.1 reports the matched training and evaluation protocol. The appendix documents the released allocation and audit code, deterministic integrity checks, local configuration boundary, and claim-specific audits. The anonymous code link in the abstract provides the accompanying implementation. Primary allocations use only training-pool metadata and train-internal calibration examples; test results are not used to select quotas, hyperparameters, or checkpoints.

AI Use Disclosure

Generative AI tools were used for manuscript drafting and language polishing, literature retrieval and reference-related tasks, minor figure-label editing, and feedback on experimental design and result interpretation. All AI-assisted outputs were reviewed by the authors and checked against the underlying sources, code, and experimental records where applicable. The authors take responsibility for the final text, citations, claims, and reported results. Generative AI was not used to generate synthetic datasets or to prove mathematical claims.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Alexander Bukharin, Shiyang Li, Zhengyang Wang, Jingfeng Yang, Bing Yin, Xian Li, Chao Zhang, Tuo Zhao, and Haoming Jiang. Data diversity matters for robust instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 3411–3425, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

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, 2019.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics: Methodology and distribution*, pp. 569–593. Springer, 1992.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17783–17806. PMLR, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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 *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

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.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. Gsm-plus: A comprehensive benchmark for evaluating the robustness of llms as mathematical problem solvers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2961–2984, 2024.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, 2021.

Yuan Li, Zhengzhong Liu, and Eric Xing. Data mixing optimization for supervised fine-tuning of large language models. *arXiv preprint arXiv:2508.11953*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, Weizhu Chen, et al. Not all tokens are what you need for pretraining. *Advances in Neural Information Processing Systems*, 37:29029–29063, 2024.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. In *International Conference on Learning Representations*, volume 2025, pp. 38305–38339, 2025.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024a.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *International Conference on Learning Representations*, volume 2024, pp. 22353–22373, 2024b.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *International conference on machine learning*, pp. 22631–22648. PMLR, 2023.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2381–2391, 2018.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, pp. 13991–14002, 2019.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

HSVNS Kowndinya Renduchintala, Sumit Bhatia, and Ganesh Ramakrishnan. Smart: Submodular data mixture strategy for instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12916–12934, 2024.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pp. 8346–8356. PMLR, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 8732–8740, 2020.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Haebin Shin, Lei Ji, Xiao Liu, Zhiwei Yu, Hyunwoo Yoo, Qi Chen, and Yeyun Gong. Dynamixsft: Dynamic mixture optimization of instruction tuning collections. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 39590–39603, 2026.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 5085–5109, 2022.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023a.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. Data selection for language models via importance resampling. *Advances in Neural Information Processing Systems*, 36: 34201–34227, 2023b.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*, 2024.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

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

Qingru Zhang, Minshuo Chen, Alexander Bukharin, 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.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Supplementary Package Overview

This appendix documents the anonymous supplementary code package accompanying CALIBUDGET. It specifies the package boundary, deterministic allocation and integrity contracts, CPU-checkable validation procedure, and the relation between released code components and the evidence reported in the main paper. It also consolidates claim boundaries and records separately completed audits without pooling them into the confirmatory three-seed results. The appendix introduces no new method, experimental setting, or headline claim.

B Scope and Relationship to the Main Paper

The supplied archive is designed to make the allocation and audit layer inspectable without requiring model weights, datasets, network access, or GPU execution. It contains two complementary forms of material. First, the compact package under `src/calibudget/` implements dependency-light allocation, record-validation, tokenization-contract, sketching, and file-binding primitives used by synthetic tests. Second, `source/` contains sanitized source snapshots of the training, allocation, proxy-contract, and external-evaluation implementations used in the study. These snapshots expose the relevant control flow and contracts, but deliberately omit private artifact roots, launch infrastructure, checkpoints, generated outputs, and machine-specific configuration.

The package therefore supports two levels of inspection. The default checks verify deterministic behavior, fail-closed validation, and file integrity on a CPU-only machine. Full experiment reproduction additionally requires the public model and dataset dependencies, the registered training environment, and locally supplied paths. This distinction prevents the lightweight verification package from being mistaken for a self-contained distribution of model weights or experimental outputs.

Package component	Purpose	Verification boundary
`src/calibudget/allocation.py`	Deterministic floor–residual allocation, capacity-aware largest-remainder rounding, and selected-index hashing.	Model- and dataset-independent synthetic inputs.
`src/calibudget/proxy.py`	Canonical subgroup-score record validation and stable four-shard aggregation.	Validates structure and ordering without exposing experiment-specific score values.
`src/calibudget/data.py`	Record normalization, normalized prompt hashing, and pairwise-disjoint role checks.	Dataset-agnostic integrity contracts.
`src/calibudget/modeling.py`	Response-only tokenization-length contract with caller-supplied tokenizer.	No model download or model identity is required.
`src/calibudget/sketch.py`	Deterministic CountSketch for finite one-dimensional vectors.	Synthetic numeric arrays only.
`src/calibudget/audit.py`	Strict JSON-object loading, SHA-256 calculation, and regular-file binding checks.	Local file integrity and provenance checks.
`source/`	Sanitized snapshots of training, allocation, proxy-contract, and modern-task evaluation code.	Source inspection and compilation; generated outputs and launch infrastructure are excluded.
`tests/`	Seven synthetic tests covering allocation, proxy records, role separation, sketch determinism, and file binding.	Runs without network access, model loading, or datasets.

Table 5: Map of the anonymous supplementary package. The compact modules expose the contracts exercised by the default tests, while the sanitized snapshots preserve additional implementation context.

C Deterministic Allocation Contract

C.1 Floor–Residual Decomposition

For source capacities N_k, total source budget B, and floor f, the compact implementation first assigns

$$q_k^{\text{floor}} = \min(N_k, f), \qquad R = B - \sum_{k=1}^{K} q_k^{\text{floor}}. \tag{12}$$

It rejects negative budgets or capacities, mismatched source keys, an infeasible floor, and any budget exceeding total capacity. For active sources with residual capacity $c_k = N_k - q_k^{\text{floor}} > 0$, nonnegative utilities are normalized over the active set. Real-valued residual shares are floored, capped by c_k, and completed by largest-remainder redistribution until the residual sum is exactly R. The returned quotas satisfy

$$\min(N_k, f) \le q_k \le N_k, \qquad \sum_{k=1}^{K} q_k = B. \tag{13}$$

The production allocation snapshot uses the frozen defaults $B = 10{,}000$, $f = 1{,}150$, $(\alpha, \beta, \gamma) = (1, 0.5, 1)$, and seed 42 unless overridden. It computes reliability as $(1 + \sigma_k)^{-1}$ and post-floor availability from the square root of remaining capacity. The released compact allocator intentionally accepts already constructed utility values; this separation permits the quota mechanism to be tested independently of model-based score estimation.

C.2 Stable Tie Breaking and Selection Identity

Every label traversal is lexically sorted, and equal fractional remainders are resolved by the source label. Consequently, the same capacities, utilities, budget, and floor produce the same quota dictionary across Python processes and platforms. The synthetic allocation test uses capacities $(3, 10, 10)$, utilities $(1, 2, 3)$, $B = 12$, and $f = 2$, and requires the exact output $(3, 4, 5)$.

Selected-example identity is represented separately from quota identity. The helper `canonical_selected_indices_sha256` rejects duplicate indices and hashes the ordered sequence. Order sensitivity is deliberate: two runs selecting the same set in a different stored order do not silently share an identifier. The production snapshot additionally verifies that selected examples do not overlap calibration indices and that the selected count equals the requested source budget.

D Data, Proxy, and Audit Contracts

D.1 Record and Role Integrity

The record normalizer accepts common field aliases such as instruction/prompt/question and response/output/answer, collapses instruction whitespace, strips the response, and rejects missing instruction–response pairs. When no explicit record ID is present, a SHA-256 ID is derived from the normalized instruction and response. Prompt hashing is case-insensitive after whitespace normalization.

Role indices are required to be nonempty, unique within each role, pairwise disjoint across roles, and named by lowercase identifiers. This contract is used to prevent accidental crossing among candidate, calibration, validation, or other registered roles. It is a structural safeguard rather than a claim that the archive contains the original datasets.

D.2 Proxy-Score and Shard Validation

The proxy contract freezes the eight-source order as ARC-Challenge, ARC-Easy, BoolQ, HellaSwag, OpenBookQA, PIQA, Social IQa, and WinoGrande. Each score record must include a canonical position, record ID, all eight finite subgroup means, and a finite aggregate score equal to the recomputed maximum subgroup mean. Aggregation requires exactly four canonical shards. A record at position i must appear in shard $i \bmod 4$; missing, duplicated, out-of-range, or cross-shard positions are rejected. After validation, records are ordered by descending score and then by candidate ID, yielding a stable selected-ID sequence.

These checks are intentionally fail-closed. They do not publish the experiment-specific proxy values, but they make malformed score records, subgroup-order drift, incomplete unions, and unstable tie breaking detectable before selection.

D.3 Tokenization, Sketching, and File Binding

The response-only tokenization helper constructs a fixed instruction/response template, applies truncation with a caller-supplied tokenizer, and reports both total length and the number of response tokens left unmasked. It rejects nonpositive maximum lengths and examples whose response has no surviving token after truncation.

The released CountSketch uses a deterministic BLAKE2b-derived bucket and sign for each vector position, processes long vectors in blocks, and rejects nonfinite values. Audit helpers require JSON roots to be objects and bind files by regular-file status, exact byte count, and SHA-256 digest; symbolic links are rejected for bound artifacts.

E CPU-Checkable Verification Procedure

The archive includes a PowerShell entry point and can be checked equivalently on a POSIX shell. From the archive root, the following commands compile the compact modules, sanitized snapshots, and tests, then execute the synthetic suite:

```
PYTHONPATH=src python -m compileall -q src source tests
PYTHONPATH=src python -m pytest -q tests
```

The distributed suite contains seven tests. They verify: (i) exact budget and capacity safety of floor–residual allocation; (ii) order-sensitive selected-index hashes; (iii) canonical four-shard membership and stable top-B selection; (iv) rejection of subgroup-order drift; (v) record normalization and role

disjointness; (vi) deterministic sketching and rejection of nonfinite vectors; and (vii) JSON/file-binding integrity. The archive also provides `SHA256SUMS.tsv`, which binds 26 distributed files by relative path, byte count, and SHA-256 digest. During appendix preparation, all 26 bindings matched and all seven synthetic tests passed in a clean CPU-only check.

F Reproduction Boundary and Local Configuration

The local path template exposes placeholders for model, data, and artifact roots and records the experimental defaults needed by the allocation layer: seed 42, commonsense budget 10,000, mathematical companion budget 100,000, and floor 1,150. Machine-specific paths are intentionally absent. The archive also excludes model weights, datasets, credentials, network launchers, SSH helpers, checkpoints, logs, and generated experiment outputs.

The compact verification modules require NumPy, PyTorch, and pytest as listed in `requirements.txt`; the default tests do not import optional model-training dependencies. Executing the sanitized training and evaluation snapshots requires the additional scientific and model-training environment used for the registered experiments. The code package therefore supports source inspection and contract validation directly, while end-to-end reruns depend on separately obtained public resources and local compute.

G Claim Boundaries and Additional Audits

Table 6 pairs each evidence layer with the strongest conclusion supported by the current results and with a stronger conclusion that remains unestablished. This separation is important because the confirmatory study uses three paired seeds, whereas transfer and diagnostic analyses are intended to delimit scope rather than select a new method.

Evidence layer	What the evidence supports	What it does not establish
Main paired study	Consistent gains on CommonAvg, FragileAvg, and MacroAvg under matched LLaMA-2 budgets and LoRA+ controls.	Uniform improvement, large-sample significance, or absence of a MathAvg trade-off.
Reliability ablation	The complete utility is consistently stronger than removing ρ_k on source-balanced metrics in the three observed seeds.	Independent causal effects among interacting utility terms.
Quota geometry	Small, structured source reallocations can accompany measurable aggregate changes.	A monotonic quota–accuracy response for individual tasks.
Qwen2.5 audit	Directionally similar descriptive gains under an independently recalibrated second backbone.	Backbone-invariant superiority; all reported bootstrap intervals cross zero.
LLaMA-3.1 and modern panel	Transferred LLaMA-3.1 allocations are near parity, and the broader modern-task panel shows no reliable gain.	Reliable gains under transferred quotas, on code tasks, or with full-parameter tuning.
Floor and split audits	Quota rankings are stable across calibration splits, while the floor operating point remains consequential.	A universal floor value or robustness to arbitrary source partitions.

Table 6: Claim boundary. Each evidence layer is paired with the conclusion supported by the current results and with a stronger conclusion that those results do not establish.

G.1 Separately Completed Qwen2.5 Reconstruction

Table 7 records a separately completed Qwen2.5-7B seed-42 reconstruction. It was run under matched source budget, training, and evaluation controls. Because it is a single-seed consistency audit, it is not pooled with the three-seed Qwen2.5 summary in the main paper.

Method	Math	Common	Fragile	Macro	Overall
Val-error+floor	65.38	84.55	88.04	80.71	74.96
CALIBUDGET	65.11	84.93	88.54	80.97	75.02
Δ (pp)	-0.26	$+0.39$	$+0.50$	$+0.26$	$+0.06$

Table 7: Separately audited Qwen2.5-7B seed-42 reconstruction (accuracy, %). Differences are CALIBUDGET minus Val-error+floor and are computed before display rounding.

G.2 Completed Component Check

A separately registered LLaMA-2 component audit had completed seeds 42–43 at the evidence cutoff. Need+reliability exceeded need+availability by $+0.20$ pp on CommonAvg, $+0.68$ pp on FragileAvg, $+0.15$ pp on MacroAvg, and $+0.08$ pp on Overall, with a -0.03 pp MathAvg difference. The direction is consistent with the reliability interpretation in the main paper, but seed 44 and the complete matched component grid were unfinished. We therefore treat this result as provisional convergent evidence, not as a final ablation or a basis for revising the frozen method.

Practical audit checklist. An independent rerun should retain source definitions and counts, calibration-role indices, need and bootstrap-stability statistics, floor, exponents, total budget, residual shares, final quotas, selected-index hash, random seed, parser/scorer versions, and per-task outputs. Execution should fail on role overlap, file-size or hash mismatch, subgroup-order drift, incomplete shards, capacity violation, or an incorrect final count. These records make the allocation decision auditable independently of downstream model training.