Model Knows What to Learn Now: Learner-Centric SFT Data Selection via Self-Rectify

Anonymous ACL submission

Abstract

LLM supervised fine-tuning (SFT) data have become abundant, mainly driven by synthetic generation. However, sheer scale does not guarantee effective SFT: large datasets often exhibit pathologies such as redundancy, imbalance, and poor learnability, making data selection critical for improving SFT data efficiency and efficacy. Existing selection methods typically rely on static or handcrafted criteria, which can be subjective, biased, and lack transferability. To address this, we propose LERS (**L**earner Self-**R**ectify **S**election), a learner-centric data selection framework. For learner families of models with similar capability levels, LERS uses a compact proxy learner to identify learning-worthy samples via learning feedback, and to dynamically rectify the selected subset throughout segment-wise selection. By adapting selection to the learner's evolving needs rather than fixed criteria, LERS surfaces informative, suitably challenging, and diverse samples that are otherwise overlooked. With only 10% of the data, LERS matches the SFT performance of 5× random sampling and consistently outperforms full-dataset training in multi-source settings across a wide range of learners. Our findings reveal that prioritizing learner-aligned, high-utility data is key to achieving "more with less" in SFT. [1]

1 Introduction

Large language models (LLMs), pretrained and fine-tuned on large-scale corpora (Xu et al., 2024), have achieved strong performance across natural language understanding, question answering, and reasoning tasks (OpenAI et al., 2024; Shao et al., 2024). In supervised fine-tuning, LLM-generated data (Wang et al., 2023; Taori et al., 2023) have become a main source as model capabilities scale, enabling compact models to distill knowledge from state-of-the-art generators and achieve competitive performance (Radosavovic et al., 2018; Yang et al., 2024b; DeepSeek-AI et al., 2025).

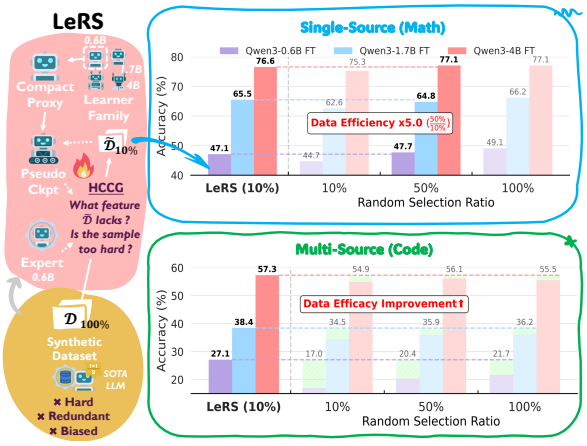


Figure 1: LERS, a learner-centric SFT data selection framework. By leveraging the evolving state of a proxy learner relative to an expert counterpart anchor, LERS rectifies data distribution while down-weighting overly difficult samples, yielding higher (1) data efficiency: reaching target performance with fewer samples; (2) data efficacy: enhancing the effectiveness of available data, where 10% LERS-selected data surpasses full-dataset SFT performance across the learner family.

However, challenges persist in ensuring efficient and effective utilization of synthetic SFT data, which often contains (i) redundancy that inflates training costs; (ii) corrupt samples or imbalanced feature distributions that harm generalization (Yang et al., 2023; Chen et al., 2024). Additionally, a pronounced capability gap between SOTA LLMs and small-scale models creates learning obstacles (Cai et al., 2025). The lack of guarantees on data efficiency and efficacy leads to excessive training overhead and degraded performance, especially burdening individual trainers who rely on paid third-party training services. This highlights the need to identify and select learning-worthy data.

Prior work on data selection largely focused on designing hand-crafted semantic criteria (Penedo et al., 2023; Weber et al., 2024) or assigning static quality scores (Chen et al., 2023; Liu et al., 2024a;

[1] The code and data will be released as open-source.

Li et al., 2024b; Wettig et al., 2024). However, such data-centric criteria encode implicit human assumptions about what benefits learning (Li et al., 2025), while overlooking how learner capacity shapes data utility (Harada et al., 2025). For example, smaller models often fail to learn effectively from complex, multi-step reasoning data that is well suited to SOTA LLMs (Cai et al., 2025). Consequently, these methods introduce inductive biases that generalize poorly across models and datasets. Although model-aware approaches based on perplexity or gradients have been proposed (Killamsetty et al., 2021; Xia et al., 2024a; Zhang et al., 2025b; Morris et al., 2025), they are typically computed only from the learner's initial state, limiting adaptation to its evolving data needs during training.

To address these challenges, we adopt a learner-centric perspective with proxy-based estimation of data utility. As shown in Figure 1, we use a compact learner proxy to approximately assess how a family of target learners would perform when trained on the current subset, thereby capturing intrinsic subset properties (e.g., difficulty mismatch, redundancy, and under-represented useful features). The estimated gap between the proxy and its expert counterpart is used to diagnose missing characteristics, and identify complementary data at suitable difficulty levels for the subset. The compact learner serves as a proxy for data valuation, efficiently predicting the utility data that would benefit homologous learners.

Further, we propose LERS (**L**earner **S**elf-**R**ectify **S**election), a learner-centric data selection framework. LERS uses a learner-aligned utility metric in data selection: the **H**omologous **C**orrected **C**ognitive **G**ap (HCCG), which quantifies the dynamic informational demand of the proxy learner by referencing the expert as a knowledge boundary. LERS partitions the selection iterations into segments and interpolates HCCG to identify additional data features the current subset still lacks as its volume increases. Our theoretical and empirical analyses reveal a self-rectification working mechanism behind LERS: when the selected subset deviates from the desired profile, the trained proxy learner deviates accordingly, and data that effectively reduce this deviation receive higher HCCG scores while overly hard data are penalized. This steers the selection toward a better-aligned data distribution. Across math and code domains, LERS-selected data yield superior fine-tuning performance.

Our contributions are summarized as follows:

- We introduce a learner-centric data selection workflow that leverages a compact learner as an efficient proxy to identify learning-worthy samples for an entire learner family.
- We build LERS, a learner-centric SFT data selection framework that rectifies the selected data distribution by tracking the utility metric HCCG on a compact proxy learner. HCCG identifies underrepresented data features and down-weights overly difficult samples.
- Extensive experiments show that LERS surpasses all data selection baselines, achieving $5\times$ data efficiency on single-source datasets and improving data efficacy on multi-source datasets by up to $+10.1\%$ over full-data fine-tuning, showing that learner-aligned data utility outweighs sheer data volume in SFT.

2 Related Work

2.1 Synthetic Data Generation

Synthetic data generation creates training samples automatically rather than through manual collection (Nadăș et al., 2025). LLM-synthesized corpora are now widely used for instruction tuning, reasoning, dialogue, and code tasks (Liu et al., 2022; Honovich et al., 2023; Taori et al., 2023; Bian et al., 2023; Zeng et al., 2024; Zhang et al., 2024a; Zheng et al., 2025; Sun et al., 2025). DeepSeek-R1 distillation further demonstrates the effectiveness of synthetic data and has enabled open reasoning datasets such as OpenThoughts-114k and OpenR1-Math-220k (DeepSeek-AI et al., 2025; Guha et al., 2025; Hugging Face, 2025).

Although preliminary filtering has been applied to synthetic datasets in terms of correctness and style, we show improving data utility and learnability is more essential for small models.

2.2 SFT Data Selection

Improving data quality substantially benefits SFT (Penedo et al., 2023; Zhou et al., 2023; Ye et al., 2025), and data selection seeks high-utility subsets for better efficiency and performance. Existing methods mainly fall into three groups: (i) static quality or diversity ranking with instruction-complexity metrics (Liu et al., 2024a; Zhao et al., 2024; Chen et al., 2023; Li et al., 2024a,b); (ii) distribution coverage via clustering, embedding distance, or compression ratio (Ge et al., 2024; Liu et al., 2024b; Yin et al., 2024); and (iii) gradient

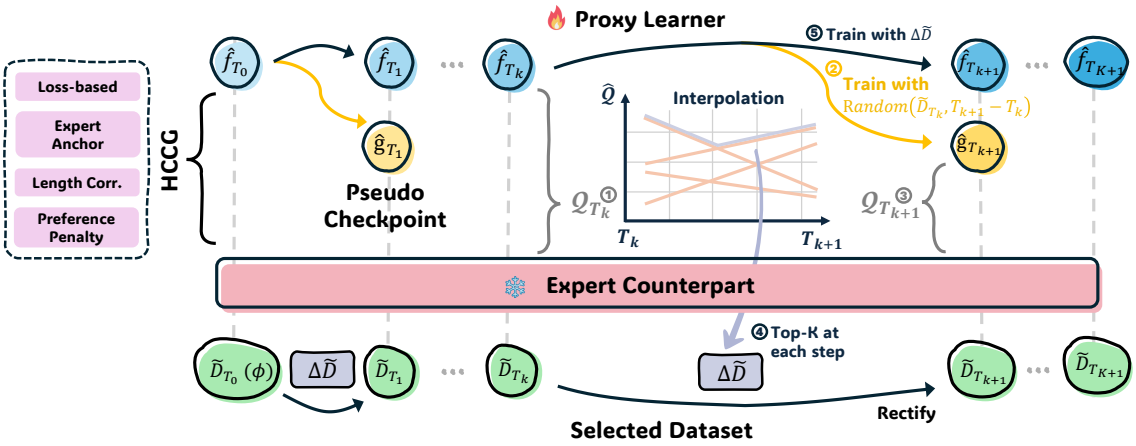


Figure 2: Overview of the proposed Learner Self-Rectify Selection, illustrated for the segment from T_k to T_{k+1}. It incrementally adds complementary samples $\Delta\tilde{D}$ for subset rectification and outputs the selected dataset $\tilde{D}_{T_{K+1}}$.

and trajectory alignment, such as matching validation gradients or expert trajectories (Xia et al., 2024b; Morris et al., 2025).

However, data-centric rules (such as expertise) often misalign with learner utility and introduce unlearnable patterns (Section 4.4). Gradient-trajectory signals are also noisy for LLMs, and loss criteria can over-reward intrinsically hard or noisy samples. As a result, many methods either fail to capture evolving learner needs or scale poorly.

Dynamic selection strategies, e.g., active data selection and curriculum learning (Zhang et al., 2024b, 2025a; Chen et al., 2025), adapt data using rules or model signals but are usually learner-specific. In contrast, LERS jointly rectifies subset distribution and difficulty with an efficient compact learner proxy, enabling one-shot selection that transfers across an entire learner family.

3 Method

LERS is a learner-centric data selection framework with segment-wise subset rectification. It progressively expands the selected subset across iteration segments. At each segment, we train the proxy learner on the current subset and compare its state with that of the expert counterpart to identify missing data characteristics. It prioritizes samples that are both learnable and most helpful for rectifying distribution bias. By repeatedly selecting such complementary samples, LERS refines the subset into a more balanced and effective training curriculum.

3.1 Finding Complementary Samples via Subset-Trained Learner

Assuming the currently selected subset is biased, when rectifying it by adding complementary samples, we investigate: *which remaining samples are optimal for reducing distributional bias and enriching features the learner needs?* We show that a sample's loss on the subset-trained learner quantifies how much the selected subset is biased with respect to that sample. Therefore, higher-loss samples are generally more informative as complement.

Definitions. Let the full dataset be denoted by $\mathcal{D}$, the loss function by $\mathcal{L}$, the input by $X \in \mathbb{R}^n$, and the corresponding ground-truth output by $Y \in \mathbb{R}^m$. The model defines a mapping $\mathbb{R}^n \to \mathbb{R}^m$.

Consider an intermediate state in which a subset $\tilde{\mathcal{D}} \subset \mathcal{D}$ has already been selected. We assume that the remaining data can be partitioned into k disjoint, feature-consistent categories, such that

$$\mathcal{D} \setminus \tilde{\mathcal{D}} = \bigcup_{i=1}^{k} \mathcal{D}_i, \quad \text{s.t.} \quad \mathcal{D}_i \cap \mathcal{D}_j = \varnothing, \forall i \neq j.$$

After training on $\tilde{\mathcal{D}}$, the resulting subset-trained learner $\hat{f}$ is an empirical approximation to the ideal predictor f. f is defined as the Bayes-optimal model under the data distribution $P_{\tilde{\mathcal{D}}}$:

$$f = \arg\min_{\hat{f}} \mathbb{E}_{(X,Y)\sim P_{\tilde{\mathcal{D}}}} \mathcal{L}(\hat{f}(X), Y). \quad (1)$$

For analytical tractability, we derive the formulation under Mean Squared Error (MSE). An analogous bias–variance decomposition also holds for cross-entropy (see Appendix A.1). Since the loss is coordinate-wise decomposable, we focus on the scalar case $m = 1$.

Modeling Predictions Bias. For any $(x, y) \notin \tilde{\mathcal{D}}$, a learner trained on the biased subset $\tilde{\mathcal{D}}$ incurs an additive prediction-bias term $\varepsilon_{\text{bias}} \sim \mathcal{N}(\mu, \sigma_b^2)$, while the ideal predictor $f(x)$ is still subject to

real-world noise $\varepsilon_{\text{noise}} \sim \mathcal{N}(0, \sigma_n^2)$. Thus,

$$y = f(x) + \varepsilon_{\text{noise}} + \varepsilon_{\text{bias}} = f(x) + \varepsilon, \quad (2)$$

where $\varepsilon \sim \mathcal{N}(\mu, \sigma_b^2 + \sigma_n^2) = \mathcal{N}(\mu, \sigma^2)$.

Loss Decomposition. We decompose the loss to investigate what it fundamentally represents:

$$\begin{aligned} \mathcal{L}_{\hat{f}} &= \mathbb{E}[(\hat{f} - y)^2] = \mathbb{E}[(\hat{f} - f - \varepsilon)^2] \\ &= \mathbb{E}[(\hat{f} - f)^2] + \sigma^2 + \mu^2 - 2\,\mathbb{E}[\varepsilon(\hat{f} - f)] \end{aligned} \quad (3)$$

The first term represents the empirical deviation between model $\hat{f}$ and f, which typically vanishes as training converges. To approximate the last correlation term, we follow Ghojogh and Crowley (2023) and invoke *Stein's lemma* (Stein, 1981).

Lemma 3.1 (Stein, 1981) *If $z \sim \mathcal{N}(\mu, \sigma^2)$ and $g : \mathbb{R} \to \mathbb{R}$ are differentiable, then*

$$\mathbb{E}[(z - \mu)g(z)] = \sigma^2 \mathbb{E}[g'(z)]. \quad (4)$$

Employing Stein's lemma, by identifying $\varepsilon = z$ and $g(\varepsilon) = \hat{f}(x, y) - f(x) = \hat{f}(x, f(x) + \varepsilon) - f(x)$, results in the following:

$$\begin{aligned} \mathbb{E}[\varepsilon(\hat{f} - f)] &= \mathbb{E}[(\varepsilon - \mu)(\hat{f} - f)] + \mu\,\mathbb{E}[\hat{f} - f] \\ &= \sigma^2 \mathbb{E}\left[\frac{\partial(\hat{f} - f)}{\partial \varepsilon}\right] + \mu\,\mathbb{E}[\hat{f} - f] \\ &= \sigma^2 \mathbb{E}[\hat{f}'(y)] + \mu\,\mathbb{E}[\hat{f} - f] \end{aligned} \quad (5)$$

The last term is neglected because the expectation is anticipated to become sufficiently small as training converge. Substituting Eq. 3 yields:

$$\mathcal{L}_{\hat{f}} \approx \mathbb{E}[(\hat{f} - f)^2] + \sigma^2 + \mu^2 - 2\sigma^2 \mathbb{E}[\hat{f}'(y)] \quad (6)$$

Category-wise Analysis. Applying Monte-Carlo estimation within category $\mathcal{D}_t$ in Eq. 6, we obtain:

$$\mathcal{L}_{\hat{f}}^{(t)} \approx \sum_i \frac{(\hat{f}_i - f_i)^2}{|\mathcal{D}_t|} + \sigma_t^2 + \mu_t^2 - \frac{2\sigma_t^2}{|\mathcal{D}_t|} \sum_{i \in \mathcal{D}_t} \frac{\partial \hat{f}}{\partial y_i} \quad (7)$$

The most extreme case $\hat{f}(x) \equiv y$ derives that $\mathbb{E}[(\hat{f} - f)^2] = \sigma_t^2$, $\frac{\partial \hat{f}}{\partial y_i} = 1$ and $\mathcal{L}_{\hat{f}}^{(t)} \approx \mu_t^2$. Therefore, when moving to general situations, we have:

$$\mathcal{L}_{\hat{f}}^{(t)} \approx k\sigma_t^2 + \mu_t^2, \qquad k > 0. \quad (8)$$

Implications. Eq. 8 indicates that larger category-level bias μ_t and diversity σ_t lead to higher loss. Therefore, after training the proxy learner $\hat{f}$ on the selected subset $\tilde{\mathcal{D}}$, high-loss samples on $\hat{f}$ are expected to be the most informative complementary candidates for rectifying current subset bias and improving generalization.

3.2 Homologous Corrected Cognitive Gaps

The pure proxy learner loss-based method for identifying complementary samples is built on the assumption that all data have gold labels. Yet, the challenge stems from the presence of low-quality and overly hard outliers, which exhibit substantial divergence from the subset but should be excluded.

To mitigate this issue, we use the expert variant of the compact learner and treat its loss as an auxiliary anchor of the knowledge boundary. INSTRUCT models, typically released alongside their BASE counterparts, usually deliver stronger downstream performance and thus serve as natural experts.

The expert model is trained on massive data and is one training stage ahead of the proxy learner. Categories that still obviously confuse the expert are penalized as noisy or overly hard, rather than selected as precisely hard samples for the learner.

After comprehensive consideration (Appendix A.2.2), we introduce a learner-aligned data utility metric, termed ***H**omologous **C**orrected **C**ognitive **G**aps* (HCCG). Given the base model f_b and a fixed expert model f_e as anchor, the HCCG score $\mathcal{Q}$ for a sample (x, y) is defined as:

$$\mathcal{Q}(x, y) = \ln|y| \left[\, \mathcal{L}(f_b, x, y) - \alpha \mathcal{L}(f_e, x, y) \,\right], \ \alpha \geq 1 \quad (9)$$

It includes two corrections for observed bias:

Length Correction. The cross-entropy loss averaged across tokens tends to have higher variance on short sequences. We therefore add a length correction term, $\ln|y|$, to reduce length-induced distortion in the loss distribution.

Preference Penalty. The preference penalty α downweights (i) overly hard samples that even the expert struggles with and (ii) samples whose style distribution is misaligned with the learner and the expert. The second effect is supported by our empirical findings (Section 4.4): *synthetic data produced by a homogeneous, smaller-scale expert improves compact learners more than data from a much stronger but heterogeneous model*. This suggests that matching the learner's native distribution is more important for data selection than relying on seemingly objective signals such as expertise.

The loss-based HCCG is crucial for transferability within a learner family. It is highly consistent across similarly sized models. HCCG from QWEN3-0.6B BASE & INSTRUCT closely matches that of 1.7B, 4B, NANBEIGE4-3B, and OLMO3-7B (Figure 7), enabling the selected data to transfer both efficiency and efficacy across the family.

The HCCG metric is also computationally efficient, as using VLLM (Kwon et al., 2023) accelerates loss computation without sacrificing precision.

3.3 Segment-wise Subset Rectification

Selection methods anchored to the initial model can overlook the learner's actual progress on the current data, which biases the selected subset. LERS treats the currently selected subset as potentially biased and applies a segment-wise rectification strategy that progressively adds complementary samples. In each segment, LERS trains a proxy learner on the current subset to construct pseudo-checkpoints and tracks HCCG dynamically as the subset expands.

Specifically, LERS partitions the selection horizon of $|\tilde{\mathcal{D}}| = \tilde{N}$ into $K+1$ segments with dividing points $T_k = k\frac{\tilde{N}}{K+1}$ for $k \in \{1, 2, \ldots, K\}$. At T_k, the selected subset $\tilde{\mathcal{D}}_{T_k}$ contains T_k samples, and $\hat{f}_{T_k}$ denotes the proxy learner trained on this subset. To estimate the learner state at T_{k+1}, we continue training $\hat{f}_{T_k}$ on an additional $T_{k+1} - T_k$ samples in $\tilde{\mathcal{D}}_{T_k}$ and obtain a pseudo-checkpoint $\hat{g}_{T_{k+1}}$. LERS evaluates HCCG $\mathcal{Q}_{T_k}, \mathcal{Q}_{T_{k+1}}$ on $\hat{f}_{T_k}$ and $\hat{g}_{T_{k+1}}$, respectively (Figure 2). For iterations $t \in (T_k, T_{k+1}]$, it linearly interpolates the HCCG score $\mathcal{Q}_t(x, y)$ as

$$\hat{\mathcal{Q}}_t = \mathcal{Q}_{T_k} + \frac{t - T_k}{T_{k+1} - T_k}\left(\mathcal{Q}_{T_{k+1}} - \mathcal{Q}_{T_k}\right). \quad (10)$$

At each step, the top-`batch_size` samples are selected and removed from the candidate pool. The full procedure is summarized in Algorithm 1.

The estimate error of interpolation is bounded. Drawing upon the power laws in data scaling (Hoffmann et al., 2022; Luo et al., 2025) and assuming the influence of the learning rate schedule is marginal, the loss trajectory can be modeled as:

$$\mathcal{L}_T \approx \mathcal{L}_0 + \frac{B}{T^\beta} \quad (11)$$

where $\mathcal{L}_0$, B, and β are constant coefficients.

By applying the Mean Value Theorem, the interpolation error can be uniformly bounded as:

$$\max_{t \in [T_k, T_{k+1}]} \frac{|\hat{\mathcal{Q}}_t - \mathcal{Q}_t|}{\ln |y_t|} \leq B\frac{\beta(\beta+1)}{8K^2 T_k^\beta} \quad (12)$$

Thus, the absolute interpolation error is uniformly controlled by K, decaying at a rate of $O(K^{-2})$.

4 Experiment

4.1 Experimental Setup

Data. We use DEEPSEEK-R1 to generate large-scale reasoning datasets with <THINK> labels from pools of mathematical and programming prompts. We perform preliminary cleaning and retain samples with lengths less than 3,500 tokens, resulting in 100K candidate datasets. They consist of: (1) Math: containing verifiable AoPS (Mahdavi et al., 2025) problems, exhibiting relatively homogeneous sources. (2) Code: a programming dataset composed of diverse sources and prompt types drawn from KODCODE (Xu et al., 2025).

Model. We employ open-source QWEN3 series (Yang et al., 2025a), instantiating the proxy learner model f_b and the expert model f_e in LERS with 0.6B BASE and 0.6B INSTRUCT. The BASE version of QWEN3-0.6B, 1.7B, and 4B are fine-tuned with the selected data, along with heterologous model series NANBEIGE4-3B and OLMO3-7B.

Metric. The performance of the post-trained model directly manifests the effectiveness of selected datasets. Mathematical reasoning capability is evaluated on MATH-500, GSM8K, SVAMP, and related benchmarks, while code reasoning capability is assessed on HumanEval (+), MBPP (+), and LiveCodeBench. All results are reported using accuracy (ACC). See Appendix A.3.2 for details.

Baselines. Early studies focused predominantly on crafting static filtering metrics, assigning fixed quality scores and integrating dataset diversity, from which we select one representative method per category serving as baselines.

- **Deita** (Liu et al., 2024b) automatically selects high-value instruction samples by jointly maximizing evolved complexity and quality scores while enforcing embedding-based diversity.
- **Superfiltering** (Li et al., 2024a) employs a small model to compute instruction-following difficulty (IFD) and select top-ranked data.
- **CaR** (Ge et al., 2024) ranks instruction pairs with a lightweight scorer and clusters them to obtain a high-quality and diverse subset.

We also benchmark a series of learner-based, expert-anchored selection variants, using a spectrum of static filtering metrics as baselines, including loss-difference ($\Delta\mathcal{L}$) (Cao et al., 2024), KL divergence (KL), gradient-cosine similarity (GCS) (Morris et al., 2025), and gradient projection (GP), from which $\tilde{N}$ instances with the highest scores are retained (Appendix A.3.1).

Table 1: Overall Downstream Fine-tuning Performance of Selected Datasets on QWEN3-0.6B BASE (the proxy learner). Each dataset contains 10K (10%) samples filtered from a synthetic 100K pool for mathematics or programming. CM.BaI denotes the Beginning_and_Intermediate_Algebra split of CollegeMath. For HumanEval and MBPP, we report both the original and "+" (augmented test-case) versions. All results are reported as accuracy.

Method		MATH								CODE		
		MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.	HumanEval (+)	MBPP (+)	Avg (+).
RANDOM	10%	37.0	63.1	79.8	8.1	51.2	35.6	38.3	44.7	12.8 (11.6)	21.1 (18.8)	17.0 (15.2)
	50%	42.4	68.2	83.8	9.2	52.0	38.7	39.5	47.7	15.9 (13.4)	24.8 (21.6)	20.4 (17.5)
	100%	41.6	68.8	84.3	8.8	56.7	41.6	41.9	49.1	17.1 (16.5)	26.3 (22.1)	21.7 (19.3)
DATA -BASED	Deita	38.2	62.5	77.1	8.8	50.4	37.1	35.5	44.2	15.9 (13.4)	28.1 (23.8)	22.0 (18.6)
	CaR	37.4	63.2	79.2	6.6	44.9	36.9	39.5	44.0	18.3 (15.9)	28.1 (25.1)	23.2 (20.5)
	SupF	40.0	61.5	69.9	9.2	51.6	**39.2**	36.4	44.0	12.2 (10.4)	18.3 (16.0)	15.3 (13.2)
LEARNER -BASED	$\Delta\mathcal{L}$	36.2	64.4	**83.8**	7.4	53.9	37.1	36.3	45.6	17.1 (16.5)	25.6 (23.3)	21.4 (19.9)
	KL	35.0	61.9	82.0	9.9	49.2	34.3	35.9	44.0	12.2 (11.0)	22.8 (18.8)	17.5 (14.9)
	GCS	39.0	**64.7**	80.3	10.7	49.6	36.4	38.6	45.6	17.1 (15.9)	23.1 (20.8)	20.1 (18.4)
	GP	37.6	64.6	80.5	9.2	47.2	36.6	**40.6**	45.2	12.8 (11.0)	24.1 (21.3)	18.5 (16.2)
	LERS	**40.8**	**64.7**	77.5	**12.1**	**57.9**	38.2	38.7	**47.1**	**23.8 (20.7)**	**30.3 (25.1)**	**27.1 (22.9)**

Implementation. We select $\tilde{N} = 10$K samples from the full dataset of size $N = 100$K. For each model, we perform a grid search (Appendix A.3.3) over learning rates $\{7, 5, 3, 1\} \times 10^{-5}$ and choose the best setting based on runs trained with random samples. We use a total batch size of 16, weight decay of 0.01, and a warm-up ratio of 0.05 with a cosine scheduler. All models are trained for 3 epochs, which we found well-converged in the grid search. We fix the random seed across all runs. In LERS, the number of segment dividing points is set to $K = 8$. For the preference penalty, we use $\alpha = 1.5$ for mathematical tasks and $\alpha = 1.0$ for coding tasks, because QWEN3-0.6B INSTRUCT shows stronger expertise in math than in code.

4.2 Main Results

Effects on the Proxy Learner. Table 1 reports the SFT results on QWEN3-0.6B BASE across 10K-sample subsets selected by different strategies.

On the code data pool—a heterogeneous mixture including Code Completion, Codeforces, LeetCode, Prefill, and other sources—LERS shows a clear advantage, achieving up to +10.1% over random selection and +3.9% over the best baseline.

In the mathematics domain, which consists exclusively of synthetic, verifiable AoPS problems, limited category diversity makes data selection inherently harder than in mixture corpora. Nevertheless, LERS still surpasses the best baseline by +1.5% on average. These results highlight the strong generalizability of LERS across datasets with different characteristics.

The data-based selection yields only marginal gains over random selection in both math and code settings, suggesting an overemphasis on surface structure rather than learner utility, while the static inductive bias and noise of learner-based selection limit its generalization. Detailed analyses are provided in Appendix A.2.1.

Data Efficiency and Efficacy Improvements Across the Learner Family. To quantify the data efficiency and efficacy gains conferred by LERS, we fine-tune models on 10%, 50%, and 100% of the available data and compare their performance with that of using 10% data selected by LERS. The results are shown in Figure 1 for homologous models (QWEN3-0.6B/1.7B/4B BASE) and in Figure 3 for heterologous models (NANBEIGE4-3B BASE, OLMO3-7B BASE) using the proxy.

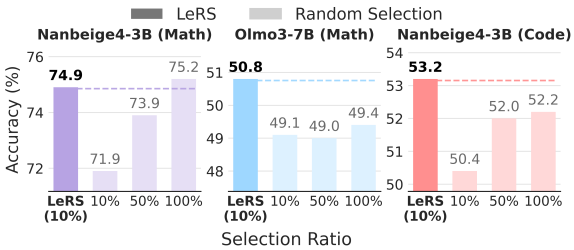


Figure 3: Data efficiency and efficacy comparison on heterologous learners (NANBEIGE4-3B BASE and OLMO3-7B BASE). Details are in Table 10 and 11.

On the math corpus, where prompts are more homogeneous, LERS delivers a consistent 5× gain in data efficiency, matching 50% data performance at one-fifth of the training cost on QWEN3 learners, and matching 100% data performance on heterologous learners. In the multi-source code dataset, enlarging the training pool yields little benefit, while LERS-selected samples outperform models trained on the full dataset. Overall, LERS improves both data efficiency and efficacy by filtering out uninformative, hard-to-learn samples.

Improvements in data efficiency and efficacy transfer well within a learner family of similarly capable models, yielding consistent gains across model variants. This transferability is supported

Table 2: Transferability within the Learner Family. We transfer data selected by LERS using a proxy model to larger homologous models (QWEN3-1.7B/4B BASE) for fine-tuning. LCBv1 denotes LiveCodeBench v1.

Model	Method	MATH								CODE			
		MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.	HumanEval (+)	MBPP (+)	LCBv1	Avg (+).
QWEN3-1.7B BASE	Random	61.2	82.7	92.5	20.6	70.1	54.5	56.6	62.6	42.1 (39.6)	52.9 (44.9)	8.5	34.5 (31.0)
	Deita	64.0	83.5	**93.6**	21.0	73.2	**57.1**	56.2	64.1	45.1 (42.1)	50.9 (43.6)	10.0	35.3 (31.9)
	CaR	62.0	82.8	92.3	19.5	75.2	51.2	56.0	62.7	50.0 (45.7)	53.1 (45.1)	10.5	37.9 (33.8)
	$\Delta\mathcal{L}$	54.0	83.2	92.5	**25.0**	74.4	49.4	52.5	61.6	45.7 (42.1)	52.1 (45.1)	**14.8**	37.5 (34.0)
	GCS	64.6	84.8	**93.6**	23.2	74.4	54.8	56.6	64.6	45.1 (41.5)	52.6 (**45.9**)	8.0	35.2 (31.8)
	LERS	**66.2**	**85.1**	92.4	22.8	**76.8**	56.1	**59.2**	**65.5**	**51.2 (47.0)**	**53.4** (45.4)	10.5	**38.4 (34.3)**
QWEN3-4B BASE	Random	82.0	93.4	95.7	36.8	78.3	73.2	**67.8**	75.3	64.6 (59.8)	69.4 (59.1)	30.8	54.9 (49.9)
	Deita	85.0	**93.9**	95.7	41.2	77.2	74.8	66.4	76.3	65.9 (62.8)	69.4 (59.6)	33.0	56.1 (51.8)
	CaR	81.4	92.9	95.8	37.5	69.3	73.8	65.8	73.8	62.2 (58.5)	69.2 (59.4)	23.8	51.7 (47.2)
	$\Delta\mathcal{L}$	79.2	92.3	**96.0**	37.5	**81.5**	68.6	64.4	74.2	65.9 (60.4)	63.9 (55.1)	37.0	55.6 (50.8)
	GCS	83.4	92.3	94.7	38.2	76.8	71.9	66.9	74.9	65.9 (61.0)	**71.2** (**60.9**)	30.5	55.9 (50.8)
	LERS	**85.6**	93.4	95.6	**42.6**	75.6	**75.8**	67.3	**76.6**	**66.5 (62.8)**	68.2 (58.9)	**37.3**	**57.3 (53.0)**

by the high within-family HCCG similarity (Appendix A.5), which leads to similar data preferences, thereby enabling one-shot data selection with a proxy learner for broad reuse.

Table 2 reports the transfer performance for selection baselines. LERS remains the strongest, whereas CaR, Deita (data-based) and GCS, $\Delta\mathcal{L}$ (learner-based) do not transfer stably: the former does not evaluate learner state, and the latter relies on noisy gradients without transferability guarantees, leading to unpredictable transfer outcomes.

4.3 Ablation Study

To disentangle the contributions of LERS's core components, we conduct ablations on QWEN3-0.6B across both math and code benchmarks, where the capacity gap between learner and generator makes selection effects more pronounced. The aggregated results are summarized in Table 3.

Table 3: Ablation Study of LERS on QWEN3-0.6B across Mathematical and Code Benchmarks

Benchmark	LERS	w/o Length Corr.	w/o Segments	w/o Int.	w/o Expert
MATH-500	**40.8**	37.0	39.8	40.2	29.2
GSM8K	64.7	58.2	63.2	**65.7**	59.1
SVAMP	77.5	72.1	**79.9**	77.5	80.1
MINERVA	**12.1**	8.8	10.3	8.8	5.1
AQUA	**57.9**	49.2	51.6	49.2	51.2
GAOKAO	38.2	39.2	36.9	**40.5**	28.8
CM.BaI	38.7	39.5	40.9	**41.7**	31.9
HumanEval	**23.8**	17.7	11.6	18.3	13.4
MBPP	**30.3**	22.1	22.3	22.1	30.1
Math Avg.	**47.1**	43.4	46.1	46.2	40.8
Code Avg.	**27.1**	19.9	17.0	20.2	21.8

w/o Length Corr. Removing length correction biases selection toward shorter samples, as token-level loss noise disproportionately affects longer sequences, leading to weak chain-of-thought representations and degraded performance.

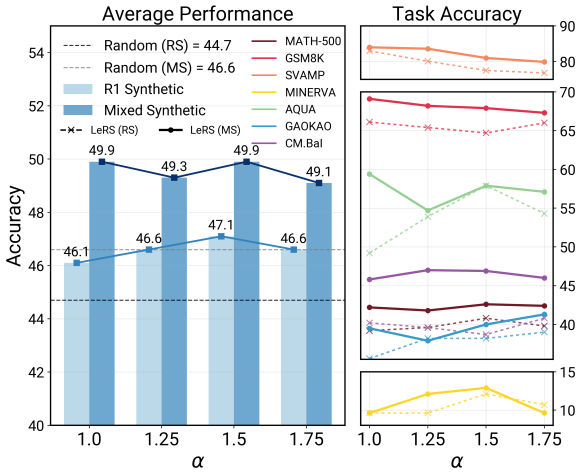
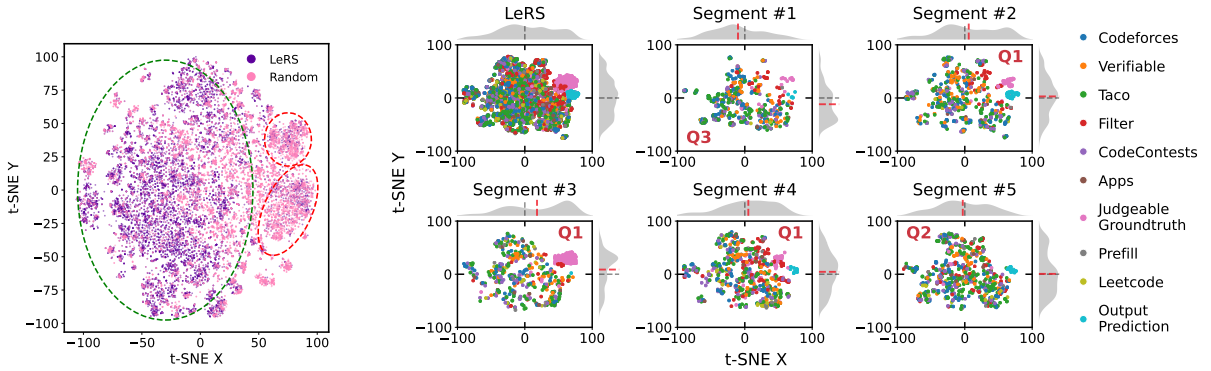


Figure 4: LERS performance on mathematical tasks under varying penalty α. We evaluate two distinct configurations: (i) R1 Synthetic (RS), containing responses exclusively from DEEPSEEK-R1; and (ii) Mixed Synthetic (MS), a balanced 1:1 mixture of responses from QWEN3-1.7B INSTRUCT and DEEPSEEK-R1.

w/o Segments Omitting segment-wise rectification skews performance toward over-represented domains. Segment-wise rectification, by replenishing underrepresented features on the fly, yields a more diverse dataset tailored to learners.

w/o Int. Disabling linear interpolation alone causes a modest decline, but performance remains unbalanced, indicating that linear interpolation yields more precise and adaptive HCCG estimates.

w/o Expert (Rectification with Loss) Excluding the expert anchor substantially reduces data utility and causes notable performance degradation. Relying solely on the proxy learner's judgment can severely misrate noisy and overly hard data, i.e., outliers that may in turn mislead even the expert.



(a) Random and LERS-selected data distribution comparison.

(b) Distribution of selected samples across rectification segments in LERS. We highlight the average of points in red lines and annotate the corresponding mean quadrant.

Figure 5: Embedding t-SNE visualization of code training samples. (a) LERS yields a more uniform distribution than random sampling. (b) LERS achieves this by rectifying distributional biases through segment-wise complements.

4.4 Learner-Aligned Data Highly Matters

In this section, we present counterintuitive evidence highlighting the substantial impact of the learner's style preferences and provide a comprehensive analysis of the role of the preference penalty α.

Alongside the verifiable mathematical data pool synthesized by DEEPSEEK-R1 (noted as "R1 Synthetic"), we generate an additional split by prompting QWEN3-1.7B INSTRUCT with identically distributed questions. These are merged into a 100K "Mixed Synthetic" (MS) dataset at a 1:1 ratio.

Although DEEPSEEK-R1 is much stronger than QWEN3-1.7B INSTRUCT in mathematical reasoning, a 0.6B learner trained on the latter's synthetic data outperforms one trained only on DEEPSEEK-R1 outputs by about +3%. This result shows that endogenous token priors in the learner strongly shape optimization: preference-aligned samples are easier to absorb than objectively stronger but more complex ones. We therefore introduce the penalty α to counter this intrinsic bias.

Figure 4 shows that training on MS consistently outperforms RS across tasks and selection methods. In RS, where learner preferences are less polarized, applying the α penalty recovers a subset of high-utility samples. In MS, where learner-aligned samples are more prevalent, the gain from tuning the preference penalty is marginal. These results support the effectiveness of using α to penalize hard-to-grasp, style-misaligned samples.

4.5 Analysis of the Rectification Mechanism

To understand why a selected subset in LERS improves data efficiency and efficacy, we analyze its selection dynamics: Figure 5 shows (a) embeddings of samples selected by random sampling versus LERS, and (b) the evolution of LERS selections over the first five rectification segments.

In Figure 5a, random selection inherits the dataset's distributional skew and forms a high-density cluster (red ellipse), inducing local overfitting. In contrast, the LERS cohort (green ellipse) covers the manifold more uniformly. Figure 5b shows how LERS rectifies this bias: data selected in segment #1 overrepresents the third-quadrant mass, so segments #2–#3 compensate by selecting first-quadrant instances. Once this bias is largely corrected, segment #4 applies a much milder first-quadrant compensation, and the process stabilizes by segment #5. This indicates that LERS uses the proxy learner's real-time learning state on the subset to identify under-mastered features and add precisely challenging, highly informative samples.

5 Conclusion

In this paper, we introduce LERS, a learner-centric data selection framework that improves data efficiency and efficacy for learner families by addressing redundancy, imbalance, and poor learnability in synthetic SFT datasets. LERS estimates the Homologous Corrected Cognitive Gap (HCCG), enabling a compact proxy learner to adaptively identify high-utility samples that rectify distributional bias and enrich the selected subset with suitably challenging data during selection. Extensive experiments show that LERS consistently outperforms all baselines, achieves up to $5\times$ higher data efficiency, and surpasses full-data training in multi-source settings with markedly improved data efficacy. These gains can generalize across models with similar capability levels. Our findings show that prioritizing data aligned with the learner's evolving needs is fundamental to improving data efficiency and efficacy in supervised fine-tuning.

6 Limitation

We show that learning-worthy data is a learner-centric concept and validate LERS's ability to improve data efficiency and efficacy, especially for widely used small-scale models whose capacities diverge most from synthetic-data generators. While LERS provides an effective selection strategy for resource-constrained model customization, its scalability to larger models remains an open question. Such generalization likely depends on a complex interplay between data sources and learner capabilities. Moreover, several phenomena identified in this study—such as the impact of learners' endogenous preferences on learning and their trade-off with sample robustness—warrant systematic analysis across models and scenarios. Finally, although we identified an optimal segment count through ablation, future work should examine whether dynamic or non-uniform segmentation tailored to the data pool can better fit the HCCG curve while reducing computational overhead.

This work also cautions against indiscriminately reusing training data generated by SOTA LLMs, which may cause training degradation and diminishing returns. We encourage further exploration of customized resources and data governance strategies tailored to the learner model's specific state.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Ning Bian, Hongyu Lin, Yaojie Lu, Xianpei Han, Le Sun, and Ben He. 2023. Chatalpaca: A multi-turn dialogue corpus based on alpaca instructions. https://github.com/cascip/ChatAlpaca.

Wenrui Cai, Chengyu Wang, Junbing Yan, Jun Huang, and Xiangzhong Fang. 2025. Enhancing reasoning abilities of small LLMs with cognitive alignment. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7434–7449, Suzhou, China. Association for Computational Linguistics.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. 2024. Instruction mining: Instruction data selection for tuning large language models. *Preprint*, arXiv:2307.06290.

Jie Chen, Yupeng Zhang, Bingning Wang, Wayne Xin Zhao, Ji-Rong Wen, and Weipeng Chen. 2024. Unveiling the flaws: exploring imperfections in synthetic data and mitigation strategies for large language models. *arXiv preprint arXiv:2406.12397*.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and 1 others. 2023. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. 2025. Self-evolving curriculum for llm reasoning. *Preprint*, arXiv:2505.14970.

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. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Mahong Xia, Zhang Li, Boxing Chen, Hao Yang, Bei Li, Tong Xiao, and JingBo Zhu. 2024. Clustering and ranking: Diversity-preserved instruction selection through expert-aligned quality estimation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 464–478, Miami, Florida, USA. Association for Computational Linguistics.

Benyamin Ghojogh and Mark Crowley. 2023. The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial. *Preprint*, arXiv:1905.12787.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 31 others. 2025. Openthoughts: Data recipes for reasoning models. *Preprint*, arXiv:2506.04178.

Yuto Harada, Yusuke Yamauchi, Yusuke Oda, Yohei Oseki, Yusuke Miyao, and Yu Takagi. 2025. Massive supervised fine-tuning experiments reveal how data, layer, and training factors shape LLM alignment quality. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 22371–22392, Suzhou, China. Association for Computational Linguistics.

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

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, and 5 others. 2024. Qwen2.5-coder technical report. *Preprint*, arXiv:2409.12186.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint*.

Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. 2021. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Preprint*, arXiv:2206.14858.

Dawei Li, Renliang Sun, Yue Huang, Ming Zhong, Bohan Jiang, Jiawei Han, Xiangliang Zhang, Wei Wang, and Huan Liu. 2025. Preference leakage: A contamination problem in llm-as-a-judge. *arXiv preprint arXiv:2502.01534*.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024a. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14255–14273, Bangkok, Thailand. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024b. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7602–7635, Mexico City, Mexico. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.

Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2022. Wanli: Worker and ai collaboration for natural language inference dataset creation. *arXiv preprint arXiv:2201.05955*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024a. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *Preprint*, arXiv:2312.15685.

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

Kairong Luo, Haodong Wen, Shengding Hu, Zhenbo Sun, Zhiyuan Liu, Maosong Sun, Kaifeng Lyu, and Wenguang Chen. 2025. A multi-power law for loss curve prediction across learning rate schedules. *Preprint*, arXiv:2503.12811.

Sadegh Mahdavi, Muchen Li, Kaiwen Liu, Christos Thrampoulidis, Leonid Sigal, and Renjie Liao. 2025. Leveraging online olympiad-level math problems for llms training and contamination-resistant evaluation. *Preprint*, arXiv:2501.14275.

John X. Morris, Junjie Oscar Yin, Woojeong Kim, Vitaly Shmatikov, and Alexander M. Rush. 2025. Approximating language model training data from weights. *Preprint*, arXiv:2506.15553.

Mihai Nadăș, Laura Dioșan, and Andreea Tomescu. 2025. Synthetic data generation using large language models: Advances in text and code. *IEEE Access*.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, and 1 others. 2025. Olmo 3. *arXiv preprint arXiv:2512.13961*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only. *Preprint*, arXiv:2306.01116.

Ilija Radosavovic, Piotr Dollár, Ross Girshick, Georgia Gkioxari, and Kaiming He. 2018. Data distillation: Towards omni-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4119–4128.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Charles M Stein. 1981. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151.

Qiushi Sun, Jinyang Gong, Lei Li, Qipeng Guo, and Fei Yuan. 2025. Codeevo: Interaction-driven synthesis of code-centric data through hybrid and iterative feedback. *Preprint*, arXiv:2507.22080.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *Preprint*, arXiv:2407.13690.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508.

Maurice Weber, Dan Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, and 1 others. 2024. Redpajama: an open dataset for training large language models. *Advances in neural information processing systems*, 37:116462–116492.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. Qurating: Selecting high-quality data for training language models. *arXiv preprint arXiv:2402.09739*.

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

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

Xinyi Xu, Zhaoxuan Wu, Rui Qiao, Arun Verma, Yao Shu, Jingtan Wang, Xinyuan Niu, Zhenfeng He, Jiangwei Chen, Zijian Zhou, and 1 others. 2024. Data-centric ai in the age of large language models. *arXiv preprint arXiv:2406.14473*.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. 2025. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024a. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Bohao Yang, Chen Tang, Kun Zhao, Chenghao Xiao, and Chenghua Lin. 2024b. Effective distillation of table-based reasoning ability from llms. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5538–5550.

Chen Yang, Guangyue Peng, Jiaying Zhu, Ran Le, Ruixiang Feng, Tao Zhang, Wei Ruan, Xiaoqi Liu, Xiaoxue Cheng, Xiyun Xu, and 1 others. 2025b. Nanbeige4-3b technical report: Exploring the frontier of small language models. *arXiv preprint arXiv:2512.06266*.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E Gonzalez, and Ion Stoica. 2023. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *Preprint*, arXiv:2502.03387.

Mingjia Yin, Chuhan Wu, Yufei Wang, Hao Wang, Wei Guo, Yasheng Wang, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. 2024. Entropy law: The story behind data compression and llm performance. *Preprint*, arXiv:2407.06645.

Weihao Zeng, Can Xu, Yingxiu Zhao, Jian-Guang Lou, and Weizhu Chen. 2024. Automatic instruction evolving for large language models. *Preprint*, arXiv:2406.00770.

Dylan Zhang, Qirun Dai, and Hao Peng. 2025a. The best instruction-tuning data are those that fit. *arXiv preprint arXiv:2502.04194*.

Jianguo Zhang, Kun Qian, Zhiwei Liu, Shelby Heinecke, Rui Meng, Ye Liu, Zhou Yu, Huan Wang, Silvio Savarese, and Caiming Xiong. 2024a. Dialogstudio: Towards richest and most diverse unified dataset collection for conversational ai. *Preprint*, arXiv:2307.10172.

Jipeng Zhang, Yaxuan Qin, Renjie Pi, Weizhong Zhang, Rui Pan, and Tong Zhang. 2025b. Tagcos: Task-agnostic gradient clustered coreset selection for instruction tuning data. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4671–4686.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. 2023. Evaluating the performance of large language models on gaokao benchmark.

Yifei Zhang, Bo Pan, Chen Ling, Yuntong Hu, and Liang Zhao. 2024b. Elad: Explanation-guided large language models active distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, page 4463–4475. Association for Computational Linguistics.

Yingxiu Zhao, Bowen Yu, Binyuan Hui, Haiyang Yu, Minghao Li, Fei Huang, Nevin L. Zhang, and Yongbin Li. 2024. Tree-instruct: A preliminary study of the intrinsic relationship between complexity and alignment. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 16776–16789, Torino, Italia. ELRA and ICCL.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. 2025. Opencodeinterpreter: Integrating code generation with execution and refinement. *Preprint*, arXiv:2402.14658.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

A Appendix

A.1 Derivations

Proof of Stein's Lemma For a Gaussian random variable $X \sim \mathcal{N}(\mu, \sigma^2)$ and any absolutely continuous function g with an integrable derivative, the standard integral expression is

$$\mathbb{E}[g'(X)] = \int_{-\infty}^{\infty} g'(x)\,\varphi(x)\,\mathrm{d}x$$
$$\varphi(x) = \mathcal{N}(x; \mu, \sigma^2)$$

where φ denotes the Gaussian probability density function. Due to the exponential decay of the Gaussian density, the boundary term $g(x)\varphi(x)$ vanishes as $x \to \pm\infty$, ensuring that the integration-by-parts formula applies. Thus,

$$\mathbb{E}[g'(X)] = \Big[g(x)\varphi(x)\Big]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} g(x)\,\varphi'(x)\,\mathrm{d}x$$
$$= -\int_{-\infty}^{\infty} g(x)\Big(-\tfrac{x-\mu}{\sigma^2}\varphi(x)\Big)\,\mathrm{d}x,$$

where the second equality uses the identity $\varphi'(x) = -\frac{x-\mu}{\sigma^2}\varphi(x)$, which follows from directly differentiating the Gaussian density. Recognizing the resulting integral as an expectation, we obtain

$$\mathbb{E}[g'(X)] = \tfrac{1}{\sigma^2}\,\mathbb{E}[g(X)(X-\mu)]\,.$$

This completes the proof.

Bias-Variance Decomposition of Cross-Entropy Loss Let $x \in \mathcal{X}$ be an input, and the ground-truth conditional distribution be $p(\cdot \mid x)$ over labels $y \in \mathcal{Y}$. Consider a model that produces a conditional distribution $q(\cdot \mid x; \theta)$. The cross-entropy of model output q under distribution p is

$$\mathcal{L}_{CE}\big(p, q(\cdot \mid x; \theta)\big) \;=\; -\sum_{y \in \mathcal{Y}} p(y \mid x) \log q(y \mid x; \theta).$$

Taking the expectation over a data category $\mathcal{D}_i$ (as defined in Section 3.1), and using the decomposition $\mathcal{L}_{CE}(p, q) = H(p) + \mathrm{KL}(p \| q)$, we obtain

$$\begin{aligned}\mathbb{E}_{x\sim\mathcal{D}_i}[\mathcal{L}_{CE}(p, q(\cdot \mid x))] &= \mathbb{E}_{x\sim\mathcal{D}_i}[H(p(\cdot \mid x))] \\ &\quad + \mathbb{E}_{x\sim\mathcal{D}_i}[\mathrm{KL}(p(\cdot \mid x) \,\|\, q(\cdot \mid x))]\,,\end{aligned} \tag{13}$$

which rewrites the expected cross-entropy loss as the sum of the conditional entropy and the expected KL divergence between the ground-truth distribution and the model prediction.

To further decompose the expected KL-divergence, we separate it into a *bias* term, measuring the discrepancy between the truth and the mean model prediction, and a *variance-like* term that reflects prediction variability under data sampling. Define the mean prediction over dataset $\mathcal{D}_i$ as

$$\bar{q}(y \mid x) = \mathbb{E}_{x\sim D_i}\big[q(y \mid x)\big].$$

By direct algebra, we obtain

$$\begin{aligned}&\mathbb{E}_{x\sim\mathcal{D}_i}[\mathrm{KL}(p\|q)] = \mathrm{KL}(p \,\|\, \bar{q}) \\ &+ \underbrace{\mathbb{E}_{x\sim\mathcal{D}_i}\left[\sum_y p(y \mid x) \log \frac{\bar{q}(y \mid x)}{q(y \mid x)}\right]}_{V(p,\{q\},\mathcal{D}_i)\text{: variance-like term}},\end{aligned} \tag{14}$$

where the first term captures systematic bias and the second aggregates variability in predictions relative to the mean predictor.

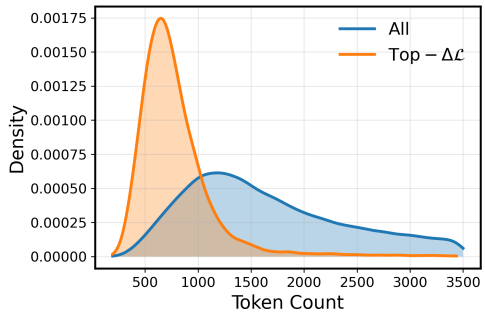
A.2 Discussions

A.2.1 Potential Issues of Other Scoring Metrics

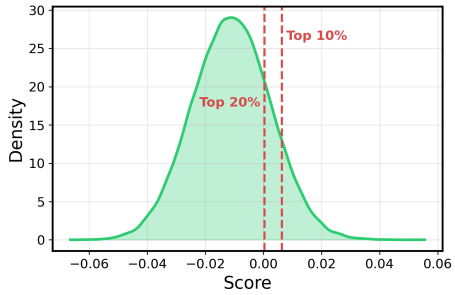
In this section, we analyze the robustness several established model-based data quality metrics. We examine their intrinsic limitations, and articulate how these issues motivate our proposed approach.

For the loss-gap metric $\Delta\mathcal{L}$, prior work (Cao et al., 2024) proposes a similar metric, suggesting that larger values indicate samples from which the model can benefit more, implying higher learning value. While this intuition is broadly correct, it introduces a critical issue in our setting: because cross-entropy loss is computed token-wise, shorter samples exhibit substantially higher variance in their scores. As illustrated in Fig. 6a, this bias causes the metric to disproportionately favor short examples, ultimately harming generalization. Consequently, HCCG adopts a length-corrected variant to mitigate this artifact.

For Gradient Cosine Similarity (GCS), a metric widely used in computer-vision data selection and recently adapted to NLP by Morris et al. (2025), the main challenge lies in its prohibitive computational cost when it is evaluated on full-model gradients for large language models. In practice, one must rely on aggressive approximations such as low-rank gradient compression or single-layer extraction. These approximations discard informative gradient structure, yielding scores that resemble near-Gaussian noise with extremely small variance (Fig. 6b). Moreover, only a small fraction of samples exhibit gradients in the positive direction. As a result, the metric provides limited actionable signal for data selection in LLMs.



(a) Comparison of the token count distributions between the $\Delta\mathcal{L}$-selected subset and the full dataset.



(b) Gradient cosine similarity score distribution.

Figure 6: Density distribution plots. $\Delta\mathcal{L}$ tends to prefer shorter responses, while Gradient Cosine Similarity scores behave like Gaussian noise.

A.2.2 Considerations behind HCCG metric

Although HCCG is simple in form, its design is carefully motivated. We discuss several potential concerns and explain why HCCG can overcome them.

Does HCCG over-reward learner–expert style mismatch? A possible concern is that HCCG may over-reward style mismatch, because the loss gap between a BASE learner and an INSTRUCT expert can partially reflect formatting preferences (e.g., reasoning markers or safety phrasing). However, we show style is treated as a functional component of data utility rather than a purely spurious factor, for the following reasons:

- **User alignment:** Response format (e.g., structured reasoning and safety phrasing) is often part of what users value, and is therefore learning-relevant.
- **Absorbability effect:** Section 4.4 shows that data synthesized by QWEN3-1.7B INSTRUCT, despite weaker raw reasoning quality than DEEPSEEK-R1, yields about 3% higher average gains for a 0.6B learner, indicating that compatible style, verified through the expert anchor, can improve learning efficiency.
- **Selection coverage:** Multi-source settings contains complex questions and responses style coverage, but LERS does not collapse to superficial style imitation.
- **Signal salience:** INSTRUCT experts are trained on large-scale heterogeneous mixtures start from BASE checkpoints, making purely superficial style preference less likely to dominate selection behavior.

Does penalizing learner perference filters out valuable-hard samples? As discussed in Section 4.4, sample difficulty is beneficial only when the reasoning pattern is matched to the learner's current preference. In our setting, the expert is expected to stay roughly one training stage ahead of the base learner. Therefore, if a sample already causes a clear loss increase for the expert (i.e., it can substantially confuse the expert), the sample is unlikely to be a valuable-hard instance for the current learner stage. Instead, it is more likely to be out-of-stage, noisy, or mismatched. Under this criterion, the learner preference penalty is used to suppress such misaligned difficulty, while segment-wise rectification still recovers genuinely informative hard samples that are difficult for the learner but remain learnable under expert-referenced guidance.

A.3 Experimental Details

A.3.1 Baseline Details

For the learner-based and expert-anchored selection setting, we adopt static metric variants as baselines. Let the parameters of the base model f_b and the expert model f_e be θ_b and θ_e, respectively. They are:

- **Loss Difference ($\Delta\mathcal{L}$):** the difference in cross-entropy loss, i.e., $\mathcal{Q} = \mathcal{L}(f_b, x, y) - \mathcal{L}(f_e, x, y)$.
- **KL Divergence (KL):** the KL divergence from the base model's likelihood distribution to the expert model's likelihood distribution,

$$\mathcal{Q} = D_{\mathrm{KL}}\big(f_e(\cdot|x) \parallel f_b(\cdot|x)\big) = \sum_{y\in\mathcal{V}} f_e(y|x) \log \frac{f_e(y|x)}{f_b(y|x)}.$$

- **Gradient Cosine Similarity (GCS):** cosine similarity between the per-sample gradient and the parameter discrepancy vector,

$$\mathcal{Q} = \frac{\langle \nabla_{\theta_b}\mathcal{L},\, \theta_e - \theta_b\rangle}{\|\nabla_{\theta_b}\mathcal{L}\|\, \|\theta_e - \theta_b\|}.$$

To retain computational feasibility, we follow Morris et al. (2025) and approximate the gradient using the `lm_head` layer under a 4096-dimensional low-rank projection.

- **Gradient Projection (GP):** the scalar projection of the per-sample gradient onto the parameter discrepancy vector,

$$\mathcal{Q} = \frac{\langle \nabla_{\theta_b} \mathcal{L},\, \theta_e - \theta_b \rangle}{\|\theta_e - \theta_b\|}.$$

A.3.2 Implement Details

The hyperparameters and configuration selections presented in Section 4.1 are systematically compiled and uniformly displayed in Table 4.

The evaluation details are enumerated in Table 5. Owing to the incomplete public release of the evaluation pipeline for QWEN3, we implemented the evaluation on top of the libraries released with Qwen2.5-Math (Yang et al., 2024a) and Qwen2.5-Coder (Hui et al., 2024). Every benchmark is executed under 0-shot mode with max_length=16,384, temperature=0.6, and top-p=0.95.

A.3.3 Grid Search Experiments and Training Convergence Analysis

To examine possible under-training across different data volumes and ensure fair comparisons, we explicitly analyze convergence behavior under exhaustive hyperparameter tuning. In main comparisons (10%, 50%, and 100% data), we apply grid-search over learning rates and epochs, and report each experiment with its best-performing setting at epoch 3, where the training has fully converged.

As an example, we inspect the training dynamics of QWEN3-1.7B-BASE on the 100% math training set. Table 6 shows validation loss and downstream math performance under different learning rates and epochs. The validation loss consistently reaches its minimum around epoch 3, while gains after epoch 3 are marginal or negative, indicating convergence under the training recipe.

To further verify that code datasets, which are surpassed by 10% LERS-selected dataset, are not disadvantaged by hyperparameters, we additionally conduct a dedicated grid search on code data for QWEN3-0.6B-BASE across 10%, 50%, and 100% settings. Results are summarized in Table 7. We choose the best score of baseline to report in the main results. This confirms that our previous full-data baselines are already near their tuned optima, and the advantage of the 10K LERS-selected subset is not an artifact of insufficient training.

Overall, these analyses support that all data-size baselines in this work are tuned to converged or near-converged configurations through grid search. Therefore, performance differences between 10%, 50%, and 100% settings reflect true data utility differences, rather than optimization under-training.

A.4 Ablations

A.4.1 Ablation of Segments Count K

Table 12 reports the fine-tuning performance of QWEN3-0.6B as the number of segment cut points K growth. Across all segment counts, the performance of LERS selected dataset consistently surpasses that obtained by randomly selecting 10K training examples. Accuracy first improves and then decreases along with segments increases, indicating that moderately increasing the number of segments helps mitigate the selection bias of the training set. However, an excessive number of segments inflate computational overhead but amplifies the noise inherent in the optimization process.

A.4.2 Ablation of Proxy Scale

To assess proxy-scale sensitivity in LERS, we additionally replace the proxy models (including the learner and expert) with QWEN3-1.7B BASE/INSTRUCT and compare it against the QWEN3-0.6B BASE/INSTRUCT pairs. As shown in Table 8, scaling up the proxy learner yields only marginal gains in aggregate performance, with improvements that are generally limited and not uniformly consistent across downstream settings. In contrast, the larger proxy increases LERS selection time from about 5.5 hours to approximately 10 hours. This cost–benefit pattern suggests that the 0.6B proxy is already sufficient to approximate the HCCG improvements obtained with larger experts in our setup. This interpretation is further consistent with Figure 7, where HCCG scores exhibit high cross-model similarity, indicating that compact proxies can capture the dominant learner-relevant signal for selection.

A.5 Cross-Model Transferability of Data Grounded by HCCG Similarity

To gain a deeper understanding of the generalizability of LERS-selected data, and to analyze the transferability of compact models (e.g., QWEN3-0.6B BASE/INSTRUCT) as efficient proxies, we investigate the consistency of the HCCG metric

Table 4: Hyperparameter and Implementation Details for Main Experiments.

Stage	Parameter	Value	Notes
Data	Synthetic dataset size N	100K	
	Selected subset $\tilde{N}$	10K	
	Data types	Math / Code	
	Context length	$\leq$ 3500 tokens	
Selection	Student model	QWEN3-0.6B BASE	Model f_b
	Expert model	QWEN3-0.6B INSTRUCT	Model f_e
	Segment cut points K	8	For segmentation
	Preference penalty α (Math)	1.5	Due to expert model capability differences
	Preference penalty α (Code)	1.0	
Fine-Tuning	LR grid search candidates	$\{7\times10^{-5}, 5\times10^{-5}, 3\times10^{-5}, 1\times10^{-5}\}$	Grid search range
	Selected LR (QWEN3-0.6B)	7×10^{-5}	
	Selected LR (QWEN3-1.7B)	3×10^{-5} (Math) / 1×10^{-5} (Code)	
	Selected LR (QWEN3-4B)	1×10^{-5}	
	Selected LR (NANBEIGE4-3B)	2×10^{-5} (Math) / 1×10^{-5} (Code)	
	Selected LR (OLMO3-7B)	1×10^{-5}	
	Batch size	16	Total batch size
	Weight decay	0.01	
	Warmup ratio	0.05	w Cosine Scheduler
	Epoch	3	
	Epoch grid search candidates	$\{1, 2, 3, 4\}$	
	Seed	45	Fixed for all runs

Table 5: Details of the Benchmarks Used for Evaluation.

Task	Benchmarks	Category
Math	MATH-500 (Hendrycks et al., 2021; Lightman et al., 2023), GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), MINERVA_MATH (Lewkowycz et al., 2022), GAOKAO2023EN (Zhang et al., 2023), CollegeMath.Beginning_and_Intermediate_Algebra (Tong et al., 2024)	Problem Solving
	AQUA (Ling et al., 2017)	Multiple-Choice
Code	HumanEval(+) (Chen et al., 2021), MBPP(+) (Austin et al., 2021), LiveCodeBenchv1 (Jain et al., 2024)	Code Generation

across diverse model architectures. Specifically, we randomly sample 10,000 mathematical data and computed the pearson correlation coefficients of HCCG metric across various learner-expert pairs. These pairs include the homogeneous QWEN3 series (0.6B, 1.7B, 4B, and 8B), as well as heterogeneous models with distinct pre-training backgrounds, such as NANBEIGE4-3B BASE & THINKING (Yang et al., 2025b) and OLMO3-7B BASE & INSTRUCT (Olmo et al., 2025).

The resulting correlation heatmap is presented in Figure 7. The results indicate that when using QWEN3-0.6B as the proxy, the derived HCCG exhibits strong correlations with models in the learner family. It computed HCCG with compact proxies can effectively reflect the cognitive gaps in models at similar capability background. However, we observe a decline in similarity as the model size increases, which underscores the necessity of learner-centric data selection - models of different scales possess distinct cognitive biases.

A.6 Cost Measurement

Taking the experimental setup in Section 4.1 as an example, we report the end-to-end runtime of different data selection methods and the training cost of models at various scales. All measurements are obtained on NVIDIA H100 GPUs, and the results are summarized in Figure 8.

Although LERS performs segment-wise subset distribution rectification and offers a more robust data selection pipeline, the overall computational overhead remains low due to the lightweight HCCG metric computation, which only requires forward

Table 6: Validation loss and math-task performance during grid search for fine-tuning QWEN3-1.7B-BASE on full 100K math data.

Learning Rate	Metric	Epoch 1	Epoch 2	Epoch 3	Epoch 4
1×10^{-5}	Val. loss	0.367	0.355	**0.353**	0.358
	Math perf.	61.7	65.5	66.9	**67.0**
3×10^{-5}	Val. loss	0.375	0.369	**0.357**	0.436
	Math perf.	57.7	62.4	**66.4**	65.9
5×10^{-5}	Val. loss	0.387	**0.363**	0.374	0.489
	Math perf.	52.7	59.8	**64.4**	64.2

Table 7: Grid search results of QWEN3-0.6B-BASE on code data with different training-data sizes, ensuring the our improvements against larger data volumes are solid

Data Size	LR	Epoch 1	Epoch 2	Epoch 3	Epoch 4
10K	7×10^{-5}	–	–	17.0	–
10K (LERS)	7×10^{-5}	–	–	27.0	–
50K	3×10^{-5}	12.8	14.5	14.0	13.8
	5×10^{-5}	12.9	14.9	14.6	16.2
	7×10^{-5}	15.2	13.7	**18.1**	17.2
100K	3×10^{-5}	11.3	12.3	15.8	13.9
	5×10^{-5}	13.8	15.6	18.2	18.0
	7×10^{-5}	15.8	21.4	**21.7**	21.5

propagation to calculate, and benefit from the acceleration by vLLM. The executing time is much lower than that of LLM evaluator-based methods such as DEITA.

In our experiments, LERS achieves up to a $5\times$ improvement in data efficiency, substantially reducing the training time required to reach comparable performance. Moreover, the selected dataset from a single pass can be reused to fine-tune suites of homologous models, providing a readily reusable resource for the community.

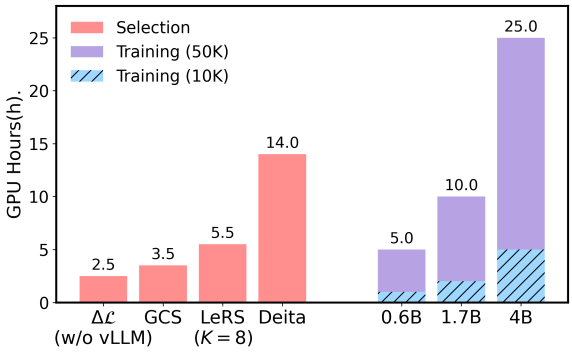


Figure 8: GPU-hour comparison. LERS achieves $5\times$ data efficiency in mathematical-task fine-tuning with low additional data-selection cost.

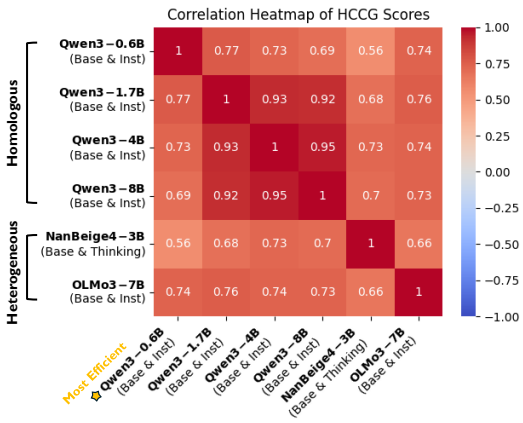


Figure 7: Cross-model correlation of HCCG across diverse learner–expert pairs. Despite variations in pre-training backgrounds and architectures, the observed cognitive gaps remain highly similar. This justifies the use of compact models as computationally efficient proxies and demonstrates that the utility of selected data transfers effectively across learners at similar capability levels.

Table 8: Ablation Study on Proxy Scales. Under the current setup, increasing proxy scale provides limited aggregate gains while increasing selection time. The compact learner proxy (0.6B) captures HCCG signal efficiently.

Finetuned Model	Proxy Scale	MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.	HumanEval	MBPP	LCBv1	Avg.
QWEN3-1.7B	0.6B	66.2	85.1	92.4	22.8	76.8	56.1	59.2	65.5	51.2	53.4	10.5	**38.4**
	1.7B	65.4	86.7	91.7	23.9	74.8	57.1	59.4	**65.6**	51.8	52.9	9.8	38.2
QWEN3-4B	0.6B	85.6	93.4	95.6	42.6	75.6	75.8	67.3	76.6	66.5	68.2	37.3	**57.3**
	1.7B	85.8	93.6	95.4	38.2	87.4	76.1	67.0	**77.6**	65.2	71.7	34.5	57.1

Table 9: Details of the Preference Penalty Ablation Study in Figure 4.

Dataset	α	MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.
R1 Synthetic	1.0	39.2	**66.1**	**83.0**	9.6	49.2	35.6	40.2	46.1
	1.25	39.6	65.4	80.1	9.6	53.9	38.2	39.6	46.6
	1.5	**40.8**	64.7	77.5	**12.1**	**57.9**	38.2	38.7	**47.1**
	1.75	39.8	66.0	76.8	10.7	54.3	**39.0**	**40.8**	46.8
Mixed Synthetic	1.0	42.2	**69.1**	**84.0**	9.6	**59.4**	39.5	45.8	**49.9**
	1.25	41.8	68.2	83.6	12.1	54.7	37.9	**47.0**	49.3
	1.5	**42.6**	67.9	81.0	**12.9**	57.9	40.0	46.9	**49.9**
	1.75	42.4	67.3	79.9	9.6	57.1	**41.3**	46.0	49.1

Algorithm 1 LERS

Input: Candidate dataset $\mathcal{D}$, total sample size $\tilde{N}$, segment division numbers K, batch size B
for $k = 0$ to K **do**
  $\tilde{\mathcal{D}}_+ \leftarrow \text{random}(\tilde{\mathcal{D}}, T_{k+1} - T_k)$
  $\hat{g}_{T_k} \leftarrow \hat{f}_{T_k}$
  for every batch $\mathcal{B}_t$ in $\tilde{\mathcal{D}}_+$ **do**
    $\hat{g}_t \leftarrow \hat{g}_{t-B} - \eta \nabla \mathcal{L}(\mathcal{B}_t)$
  end for
  Compute $\mathcal{Q}_{T_k}$ on checkpoint $\hat{f}_{T_k}$
  Compute $\mathcal{Q}_{T_{k+1}}$ on checkpoint $\hat{g}_{T_{k+1}}$
  $\Delta\mathcal{Q} \leftarrow (\mathcal{Q}_{T_{k+1}} - \mathcal{Q}_{T_k})/(T_{k+1} - T_k)$
  $\hat{\mathcal{Q}} \leftarrow \mathcal{Q}_{T_k}$
  for each step $T \in (T_k, T_{k+1}]$ **do**
    $\mathcal{B} \leftarrow \text{Top-K Indices}\left(\hat{\mathcal{Q}}_T, \text{Mask}\right)$
    $\hat{f}_T \leftarrow \hat{f}_{T-B} - \eta \nabla \mathcal{L}(\mathcal{B})$
    $\text{Mask}[\mathcal{B}] \leftarrow 1,\ \hat{\mathcal{D}} \leftarrow \hat{\mathcal{D}} \cup \mathcal{B}$
    $\hat{\mathcal{Q}}_{T+B} \leftarrow \hat{\mathcal{Q}} + \Delta\mathcal{Q},\ T \leftarrow T + B$
  end for
end for

Table 10: Experimental Results on Data-efficiency Cross-model Transferability for Mathematical Tasks. LERS-selected data maintains strong data-efficiency gains across varying model architectures and scales.

Model	Method	Percent (%)	MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.
NANBEIGE4-3B BASE	RANDOM	10%	76.2	89.2	93.3	31.2	81.1	69.9	62.6	71.9
		50%	79.4	90.3	94.7	29.0	86.2	71.9	65.6	73.9
		100%	**82.8**	**90.9**	**95.4**	29.4	**86.6**	73.8	**67.4**	**75.2**
	LERS	10%	82.0	89.5	93.0	**35.3**	83.9	**74.8**	65.9	74.9
OLMO3-7B BASE	RANDOM	10%	37.0	77.3	86.3	**9.6**	54.7	35.3	43.4	49.1
		50%	37.6	76.0	86.9	8.1	55.5	35.5	43.7	49.0
		100%	**38.0**	75.3	87.6	9.2	53.9	**37.1**	44.8	49.4
	LERS	10%	36.2	**80.7**	**88.2**	**9.6**	**60.6**	35.1	**45.3**	**50.8**

Table 11: Experimental Results of Data Efficacy Cross-Model Transferability on Coding Tasks.

Model	Method	Percent(%)	HumanEval (+)	MBPP (+)	LCBv1	Avg (+).
NANBEIGE4-3B BASE	RANDOM	10%	59.1 (55.5)	64.2 (**54.6**)	28.0	50.4 (46.0)
		50%	64.0 (**60.4**)	60.9 (53.6)	31.0	52.0 (**48.3**)
		100%	59.1 (53.7)	65.9 (56.1)	**31.5**	52.2 (47.1)
	LERS	10%	**64.0** (58.5)	**64.4** (54.1)	31.3	**53.2** (48.0)

Table 12: Ablation Experiments of the Sensitivity to LERS Rectification Segment Granularity in QWEN3-0.6B Fine-Tuning.

K	MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.
0	39.8	63.2	**79.9**	10.3	51.6	36.9	40.9	46.0
4	41.8	64.3	76.3	10.3	54.3	36.1	42.0	46.4
8	40.8	64.7	77.5	**12.1**	**57.9**	**38.2**	38.7	**47.1**
12	**42.4**	65.0	77.3	10.7	52.0	37.4	40.0	46.4
16	40.2	**66.3**	77.5	8.5	52.8	37.7	**42.8**	46.5

Table 13: Overall Comparison between Random-Selected and LERS-Selected Data Fine-Tuning Performance.

Model	Method	Percent(%)	MATH-500	GSM8K	SVAMP	MINERVA	AQUA	GAOKAO	CM.BaI	Avg.	HumanEval (+)	MBPP (+)	LCBv1	Avg (+).
QWEN3-0.6B BASE	RANDOM	10%	37.0	63.1	79.8	8.1	51.2	35.6	38.3	44.7	12.8 (11.6)	21.1 (18.8)	-	17.0 (15.2)
		50%	**42.4**	68.2	83.8	9.2	52.0	38.7	39.5	47.7	15.9 (13.4)	24.8 (21.6)	-	20.4 (17.5)
		100%	41.6	**68.8**	**84.3**	8.8	56.7	**41.6**	**41.9**	**49.1**	17.1 (16.5)	26.3 (22.1)	-	21.7 (19.3)
	LERS	10%	40.8	64.7	77.5	**12.1**	**57.9**	38.2	38.7	47.1	**23.8 (20.7)**	**30.3 (25.1)**	-	**27.1 (22.9)**
	vs. RDM $\Delta_{10\%}$		*+3.8*	*+1.6*	*-2.3*	*+4.0*	*+6.7*	*+2.6*	*+0.4*	*+2.4*	*+11.0 (+9.1)*	*+9.2 (+6.3)*	-	*+10.1 (+7.7)*
	vs. RDM $\Delta_{100\%}$		*-0.8*	*-4.1*	*-6.8*	*+3.3*	*+1.2*	*-3.4*	*-3.2*	*-2.0*	*+6.7 (+4.2)*	*+4.0 (+3.0)*	-	*+5.4 (+3.6)*
QWEN3-1.7B BASE	RANDOM	10%	61.2	82.7	92.5	20.6	70.1	54.5	56.6	62.6	42.1 (39.6)	52.9 (44.9)	8.5	34.5 (31.0)
		50%	63.8	83.7	93.2	**24.6**	73.6	56.6	58.3	64.8	46.3 (42.1)	51.9 (44.9)	9.5	35.9 (32.2)
		100%	64.8	**86.2**	**95.0**	23.2	73.2	**57.7**	**63.1**	**66.2**	51.2 (45.7)	51.1 (44.6)	6.5	36.2 (32.3)
	LERS	10%	**66.2**	85.1	92.4	22.8	**76.8**	56.1	59.2	65.5	**51.2 (47.0)**	**53.4 (45.4)**	**10.5**	**38.4 (34.3)**
	vs. RDM $\Delta_{10\%}$		*+5.0*	*+2.4*	*-0.1*	*+2.2*	*+6.7*	*+1.6*	*+2.6*	*+2.9*	*+9.1 (+7.4)*	*+0.5 (+0.5)*	*+2.0*	*+3.9 (+3.3)*
	vs. RDM $\Delta_{100\%}$		*+1.4*	*-1.1*	*-2.6*	*-0.4*	*+3.6*	*-1.6*	*-3.9*	*-0.7*	*0.0 (+1.3)*	*+2.3 (+0.8)*	*+4.0*	*+2.2 (+2.0)*
QWEN3-4B BASE	RANDOM	10%	82.0	93.4	**95.7**	36.8	78.3	73.2	67.8	75.3	64.6 (59.8)	69.4 (59.1)	30.8	54.9 (49.9)
		50%	**85.6**	93.9	**95.7**	39.3	**83.1**	74.0	**68.2**	**77.1**	63.4 (57.9)	**71.4** (60.2)	33.5	56.1 (50.5)
		100%	84.0	**95.2**	95.5	40.8	80.7	75.3	68.0	**77.1**	65.2 (60.4)	69.1 (**60.7**)	34.8	56.4 (52.0)
	LERS	10%	**85.6**	93.4	95.6	**42.6**	75.6	**75.8**	67.3	76.6	**66.5 (62.8)**	68.2 (58.9)	**37.3**	**57.3 (53.0)**
	vs. RDM $\Delta_{10\%}$		*+3.6*	*+0.0*	*-0.1*	*+5.8*	*-2.7*	*+2.6*	*-0.5*	*+1.3*	*+1.9 (+3.0)*	*-1.2 (-0.2)*	*+6.5*	*+2.4 (+3.1)*
	vs. RDM $\Delta_{100\%}$		*+1.4*	*-1.8*	*+0.1*	*+1.8*	*-5.1*	*+0.5*	*-0.7*	*-0.5*	*+1.3 (+2.4)*	*-0.9 (-1.8)*	*+2.5*	*+0.9 (+1.0)*