

Instruction Data Selection via Answer Divergence

Bo Li^{1,2,3}, Mingda Wang⁴, Shikun Zhang¹, Wei Ye^{1*}

¹ National Engineering Research Center for Software Engineering, Peking University

² School of Computer Science, Peking University

³ PKU-CMCC(Hubei) Joint Research Lab for LLM Industrial Applications

⁴ School of Health Sciences and Biomedical Engineering, Hebei University of Technology
deepblue.lb@gmail.com, wye@pku.edu.cn

 WisdomShell/ADG  ADG Project  HuggingFace Model

Abstract

Instruction tuning relies on large instruction–response corpora whose quality and composition strongly affect downstream performance. We propose **Answer Divergence-Guided Selection (ADG)**, which selects instruction data based on the geometric structure of multi-sample outputs. ADG draws several high-temperature generations per instruction, maps responses into an embedding space, and computes an output divergence score that jointly encodes dispersion magnitude and shape anisotropy. High scores correspond to instructions whose answers are both far apart and multi-modal, rather than clustered paraphrases along a single direction. Across two backbones and three public instruction pools, fine-tuning on only 10K ADG-selected examples consistently outperforms strong selectors on six benchmarks spanning reasoning, knowledge, and coding. Analyses further show that both dispersion magnitude and shape anisotropy are necessary, supporting answer divergence as a practical signal for instruction data selection. Code and appendix are included in the supplementary materials.

1 Introduction

Large language models are typically turned into helpful assistants by instruction tuning on collections of instruction–response pairs (Ouyang et al., 2022; Wei et al., 2022a; Sanh et al., 2022). Beyond scaling corpus size and diversity, recent work shows that performance is highly sensitive to *which* examples are used under the same budget, making data selection a first-order design choice for efficient alignment (Wang et al., 2023; Zhou et al., 2023; Li et al., 2024a; He et al., 2025; Chen et al., 2024; Wang et al., 2024; Liu et al., 2025; Li et al., 2026; Zhao and Yan, 2026; Liu et al., 2026).

Existing selection approaches largely follow two directions. **Data-centric** methods improve the cor-

pus directly, for example by manual curation, filtering, rewriting, or enforcing stylistic constraints. These strategies are broadly applicable, but they do not explicitly reflect what a particular base model is missing. (Zhou et al., 2023; Li et al., 2024a; Chen et al., 2024; He et al., 2025; Li et al., 2025) **Model-centric** methods attempt to read such information from the model itself, scoring each instruction by loss, uncertainty, learning progress, or influence, and then selecting a top-scoring subset for fine-tuning. In practice, a common scoring design is to measure error against a *single* teacher response (Li et al., 2024b; Liu et al., 2024; Xia et al., 2024a; Chen et al., 2025). However, single-reference scoring has a structural limitation in instruction tuning. Many instructions admit multiple valid answers, and the “reference” response is only one realization among many. A high loss may therefore reflect benign differences in format, tone, or reasoning style rather than genuine gaps in the model’s competence.

There is growing evidence that multi-sample behavior carries richer signals about what a model has and has not learned. Classical query-by-committee targets points with maximal disagreement across hypotheses and shows that such points concentrate near decision boundaries (Seung et al., 1992). Deep ensembles interpret predictive disagreement across independently trained networks as a proxy for epistemic uncertainty (Lakshminarayanan et al., 2016). In LLMs, self-consistency and Tree-of-Thoughts explicitly exploit the diversity among multiple reasoning chains for the same query to obtain more reliable or stronger answers (Wang et al., 2022; Yao et al., 2023). These lines of work inspire us that data selection should also be based on how a model’s outputs for the same instruction populate representation space, rather than judging an instruction by its fit to one reference.

We introduce **Answer Divergence-Guided Selection (ADG)**, a data selection framework built

*Corresponding author

around this idea. ADG treats the set of responses generated at a relatively high temperature for a given instruction as a geometric object. It then computes an output divergence score that captures two complementary aspects: how widely the responses spread, and whether the spread reflects genuinely multi-directional structure rather than a near one-dimensional drift. Intuitively, high scores indicate prompts where the model’s behavior is diverse in a structured way, which is more likely to reveal boundary cases than redundant supervision. To prevent a global top- k from collapsing onto a few dense regions, ADG clusters instructions using semantic embeddings, and performs bin-wise selection by ranking examples within each bin according to their divergence scores.

We evaluate ADG on two backbones and three selection pools with a fixed 10K budget, and assess reasoning, knowledge, and coding on six standard benchmarks. Across settings, ADG achieves the best or tied-best average performance and consistently outperforms strong selectors. We also conduct ablations and data-level analyses to interpret what ADG selects, including task-type shifts and a quadrant-based case study. In summary, our main contributions are as follows:

- We propose ADG, a data selection method that scores instructions using multi-sample response behavior for effective instruction tuning under a fixed budget.
- We demonstrate consistent gains over strong selectors across two backbones, three selection pools, and six benchmarks, and provide ablations and data-level analyses that validate key design choices and characterize the selected instructions.

2 Answer Divergence-Guided Selection

2.1 Overview of ADG

We aim to select the most training-effective instructions from a large pool. **Answer Divergence-Guided Selection (ADG)** scores an instruction by the model’s *multi-sample response behavior* under stochastic decoding. Given an instruction x , we sample K answers at a relatively high temperature and embed each answer into a hidden representation, yielding a set of response vectors. Near-coincident vectors indicate near-deterministic behavior, while diverse vectors indicate competing hypotheses for resolving x .

ADG summarizes this set with two complementary cues: *dispersion magnitude*, measuring how widely responses spread, and *shape anisotropy*, measuring whether the spread is multi-directional rather than dominated by a single direction. Instructions with both high dispersion and high anisotropy receive higher scores. Subsequent sections detail the computation from hidden states, score aggregation, and bin-wise selection for coverage.

2.2 Answer Sampling and Representation

For each instruction x , ADG samples K stochastic answers in parallel from the base model at a temperature of T :

$$A(x) = \{a_1, \dots, a_K\}, \quad a_k \sim f_\theta(\cdot | x; T).$$

This multi-sample protocol exposes the range of answers and reasoning modes the model currently considers plausible for the same instruction, and serves as the basis for our subsequent geometric analysis.

We represent each answer a_k using hidden states of its output tokens from the last few transformer layers \mathcal{L} (e.g., the final four layers). Let $H_k^{(l)} \in \mathbb{R}^{T_k \times d}$ be the hidden matrix at layer $l \in \mathcal{L}$. We compute an answer embedding by averaging over tokens and layers:

$$h_k = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \left(\frac{1}{T_k} \sum_{t=1}^{T_k} H_{k,t}^{(l)} \right), \quad (1)$$

then L_2 -normalize:

$$v_k = \frac{h_k}{\|h_k\|_2}, \quad k = 1, \dots, K. \quad (2)$$

Stacking the embeddings yields

$$V(x) = \begin{bmatrix} v_1^\top \\ \vdots \\ v_K^\top \end{bmatrix} \in \mathbb{R}^{K \times d}. \quad (3)$$

To remove the mean effect, we form the Gram matrix $S = V(x)V(x)^\top$ and apply the centering matrix

$$C = I_K - \frac{1}{K} \mathbf{1}_K \mathbf{1}_K^\top,$$

where $\mathbf{1}_K \in \mathbb{R}^K$ denotes the all-ones column vector, to remove the effect of the mean representation and obtain the centered similarity matrix

$$S_c = CSC = (CV(x))(CV(x))^\top. \quad (4)$$

All divergence statistics in Section 2.3 are computed from S_c , and are therefore invariant to global translations of the answer representations.

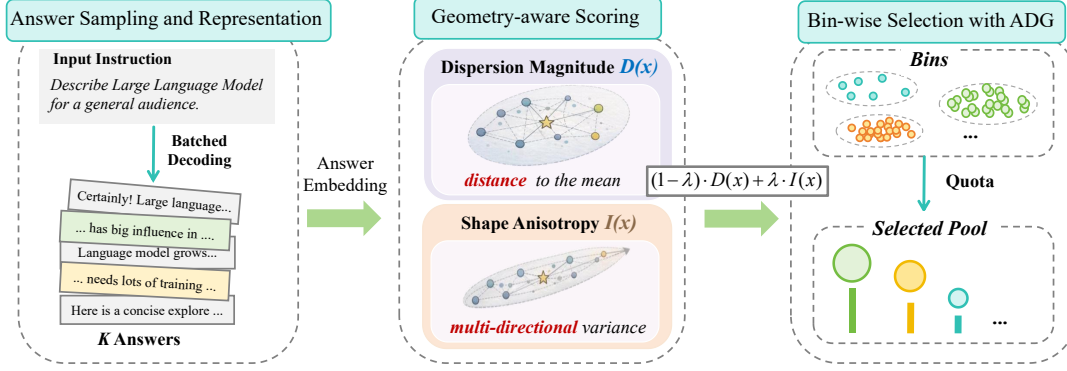


Figure 1: Overview of ADG: sample K answers per instruction, embed answers from output hidden states, score with $D(x)$ (dispersion magnitude) and $I(x)$ (shape anisotropy), and select top examples within semantic bins.

2.3 Geometry-aware Scoring

Using the centered similarity matrix S_c , ADG summarizes the multi-sample answer behavior for each instruction with two complementary statistics: overall spread and directional shape.

Dispersion magnitude. Let $\mu = \frac{1}{K} \sum_{k=1}^K v_k$ be the mean of the unit-norm answer embeddings. We measure how far the K answers spread around their mean by

$$D(x) = \frac{1}{K} \text{tr}(S_c) = \frac{1}{K} \sum_{k=1}^K \|v_k - \mu\|_2^2 \quad (5)$$

$$= 1 - \|\mu\|_2^2.$$

When answers are nearly identical, μ is close to a unit vector and $D(x)$ is small; when answers are diverse, $\|\mu\|_2$ decreases and $D(x)$ increases.

Shape anisotropy. Let $\gamma_1 \geq \gamma_2 \geq \dots \geq \gamma_K \geq 0$ be the eigenvalues of S_c . We define

$$I(x) = 1 - \frac{\gamma_1}{\sum_{j=1}^K \gamma_j}. \quad (6)$$

This is the complement of the fraction of variance explained by the leading direction: $I(x)$ is near zero when variation is dominated by a single direction, and becomes larger when variance is distributed across multiple directions. When $\sum_{j=1}^K \gamma_j = 0$ (equivalently $S_c = 0$), we set $I(x) = 0$.

Combined ADG score. $D(x)$ captures the amount of variation, while $I(x)$ captures whether that variation is multi-directional. ADG combines them as

$$s(x) = (1 - \lambda)D(x) + \lambda I(x), \quad (7)$$

where $\lambda \in [0, 1]$ controls the trade-off between spread and shape.

2.4 Bin-wise Selection with ADG

Given a pool of instructions $\{x_i\}_{i=1}^N$, ADG performs selection within semantic bins to avoid a single global top- k collapsing onto a few dense regions. We compute a frozen semantic embedding $z_i = g(x_i) \in \mathbb{R}^p$ for each instruction using an encoder g , then cluster $\{z_i\}_{i=1}^N$ into B bins $\{\mathcal{B}_1, \dots, \mathcal{B}_B\}$ via k -means.

For each instruction x_i , we compute its ADG score $s(x_i)$. Let \mathcal{B}_b denote the index set of bin b , with size $|\mathcal{B}_b|$. Given a global budget M , we allocate a bin quota

$$m_b = \text{round}\left(M \cdot \frac{|\mathcal{B}_b|}{N}\right), \quad b = 1, \dots, B, \quad (8)$$

and select the top m_b instructions in \mathcal{B}_b ranked by $s(x_i)$. The final subset \mathcal{S} is the union of these bin-wise selections. This strategy preserves coverage across semantic regions while prioritizing, within each region, instructions whose sampled answers exhibit the strongest and most structurally diverse divergence.

3 Experimental Setup and Main Results

3.1 Experimental Setup

Selection Pools and Budget. We consider three widely-used instruction corpora as selection pools: Alpaca-GPT4 (52K) (Peng et al., 2023), WizardLM (70K) (Xu et al., 2024), and CoT (100K) (Wei et al., 2022b). Unless otherwise specified, all methods select a fixed budget of $M=10K$ instances from each corpus *individually*.

Evaluation Benchmarks and Metrics. We evaluate broad capabilities spanning reasoning, knowledge, and code generation. For reasoning, we use BBH (Suzgun et al., 2023) and GSM8K (Cobbe

	Alpaca-GPT4				WizardLM				CoT			
	Reason.	Know.	Code.	Avg.Score	Reason.	Know.	Code.	Avg.Score	Reason.	Know.	Code.	Avg.Score
<i>Backbone: LLaMA3-8B</i>												
All Data	45.95	57.63	37.87	47.15	51.60	54.20	42.85	49.55	46.18	47.60	26.88	40.22
Random	42.81	57.07	38.30	46.06	48.38	56.89	38.81	48.03	39.58	47.00	29.38	38.65
IFD	43.19	59.65	43.93	48.92	50.34	57.73	42.52	50.19	37.09	56.63	39.19	44.30
SuperFiltering	46.80	58.46	42.93	<u>49.39</u>	51.09	57.87	42.11	<u>50.35</u>	44.89	49.04	38.19	<u>44.03</u>
SelectIT	42.28	56.23	37.57	45.36	45.02	53.42	37.67	45.37	35.52	45.41	29.80	36.91
Rethinking	44.69	58.30	38.20	47.06	45.05	54.03	36.68	45.25	42.53	50.22	32.71	41.82
ZIP	44.50	56.79	38.90	46.73	44.66	56.70	36.47	45.94	43.60	51.37	33.64	42.87
MIG	44.46	57.74	37.99	46.73	46.95	55.74	41.32	48.00	42.94	49.63	35.05	42.54
SCAR	48.10	59.99	35.87	47.99	45.04	54.21	33.64	44.29	40.77	48.75	29.07	39.53
ADG(Ours)	50.11	60.31	42.14	50.85	52.53	57.61	45.06	51.73	45.05	50.99	39.30	45.11
<i>Backbone: Qwen2.5-7B</i>												
All Data	69.46	65.38	57.23	64.02	70.79	64.00	55.11	63.30	65.22	59.23	53.82	59.42
Random	71.69	66.80	61.80	66.76	72.26	64.00	57.97	64.74	66.34	58.12	55.94	60.13
IFD	73.54	66.38	61.79	67.24	73.22	65.99	53.19	64.13	69.19	59.37	56.95	<u>61.84</u>
SuperFiltering	73.14	67.01	64.00	<u>68.05</u>	73.94	65.23	57.55	65.57	68.43	59.95	55.14	61.17
SelectIT	71.44	63.83	61.30	65.52	71.79	64.16	55.17	63.71	66.46	58.89	54.54	59.96
Rethinking	72.93	66.79	60.78	66.83	72.49	64.59	55.85	64.31	68.30	59.64	55.42	61.12
ZIP	72.97	67.42	63.31	67.90	71.12	65.22	57.26	64.53	66.90	58.62	56.67	60.73
MIG	74.78	67.13	60.88	67.59	74.10	65.68	59.28	<u>66.35</u>	67.90	59.24	56.03	61.05
SCAR	75.22	66.41	59.78	67.10	73.65	64.44	57.36	65.15	66.05	57.25	56.55	59.95
ADG(Ours)	75.43	67.36	64.40	69.06	75.07	66.67	61.19	67.64	70.55	59.40	58.17	62.71

Table 1: Main results with a fixed selection budget ($M=10K$) for selection methods across three selection datasets (Alpaca-GPT4, WizardLM, and CoT) and two backbones (LLaMA3-8B-Instruct and Qwen2.5-7B-Instruct). Reason., Know., and Code. denote the average performance on reasoning (BBH, GSM8K), knowledge (MMLU, TruthfulQA), and code generation (HumanEval, MBPP). **Avg.Score** is computed as the unweighted average of the three capability scores. All selection methods select 10K instances from the same pool, while All Data uses the full pool. All models are fine-tuned and evaluated with identical training and decoding configurations, and we report the mean over five random seeds to reduce randomness.

et al., 2021) with exact match (EM). For knowledge, we use MMLU (Hendrycks et al., 2021) and TruthfulQA (Lin et al., 2021) with accuracy. For coding, we use HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) with pass@1. We adopt standard evaluation scripts for each benchmark, keep decoding parameters identical across methods. See Appendix A for more details.

3.2 Backbone Models and Training Details

Backbones. We conduct experiments on LLaMA3-8B-Instruct (Dubey et al., 2024) and Qwen2.5-7B-Instruct (Yang et al., 2024). We further include results on the much smaller LLaMA3.2-1B-Instruct in the Appendix J to verify that ADG remains effective in the low-capacity regime and does not rely on large-model-specific behaviors. For each backbone, we compute the ADG score using the same backbone and then fine-tune it on the selected subset.

ADG Scoring and Selection Details. We score each instruction using the proposed ADG pipeline. Specifically, we sample $K=5$ candidate answers per instruction with temperature $T=1.4$, top- $p=0.9$, and max_new_tokens=180 (for ADG scoring only). These settings are used only for response sampling during ADG scoring. We obtain answer embeddings by pooling output-token hidden states and averaging the last four transformer layers. After scoring the full pool, we apply k -means binning with 1,000 clusters and select top-ranked instances within each bin under proportional quotas, finally merging all bins to form the selected subset.

Fine-tuning Configuration. We use a unified full fine-tuning recipe across all methods: 3 epochs, learning rate $1e-5$, cosine scheduling with warmup ratio 0.03, per-device batch size 2, gradient accumulation 8, and maximum sequence length 2048. We keep the optimization and system settings (e.g.,

precision and distributed training) identical across methods. For every method and setting, we run fine-tuning and evaluation five times with different random seeds, and report the mean results.

3.3 Baselines

Compared Methods. We compare ADG with a comprehensive set of strong data selection baselines, including **All Data**, **Random**, **IFD** (Li et al., 2024b), **SuperFiltering** (Li et al., 2024a), **SelectIT** (Liu et al., 2024), **Rethinking** (Xia et al., 2024b), **ZIP** (Yin et al., 2024), **MIG** (Chen et al., 2025), and **SCAR** (Li et al., 2025). For each method, we use the official open-source implementation released by the authors. We also provide concise descriptions for all baselines in the appendix K.

Fairness Controls. We keep the fine-tuning configuration, selection budget, and evaluation protocol identical across methods; the only difference is the selected subset produced by each selector. We follow the default hyperparameters recommended by each baseline.

3.4 Main Results

Table 1 reports results under a fixed selection budget ($M=10K$) across three pools and two backbones.¹ ADG achieves the best average score in every setting, demonstrating that the proposed geometry-aware criterion generalizes across both model families and selection pools. Specifically, on LLaMA3-8B-Instruct, ADG improves the average score by about 2%–3% (relative) over the best baseline depending on the pool; on Qwen2.5-7B-Instruct, gains are typically 1%–2%. These results indicate that ADG selects consistently more training-effective instructions under the same data budget.

The gains are most pronounced on noisier pools such as CoT: ADG surpasses All Data by +4.89 Avg.Score on LLaMA3 and +3.29 on Qwen2.5, supporting our motivation that more data is not necessarily better when supervision is redundant or unstable. Moreover, ADG yields more balanced improvements across capability groups, strengthening reasoning and coding while maintaining competitive knowledge performance, rather than trading off one axis for another.

¹The detailed results on each dataset are provided in the Appendix D.

Robustness to Chinese instruction data. We further evaluate ADG on a public Chinese instruction pool under the same setting, and observe consistent improvements over strong baselines. Full results are reported in Appendix C.

4 Analysis

4.1 Ablation Study

Setup. All ablations are conducted on **LLaMA3-8B-Instruct** with **Alpaca-GPT4 (52K)** as the selection pool. We keep the selection budget fixed to 10K and use the same setting in the main experiments, reporting the mean over five runs.

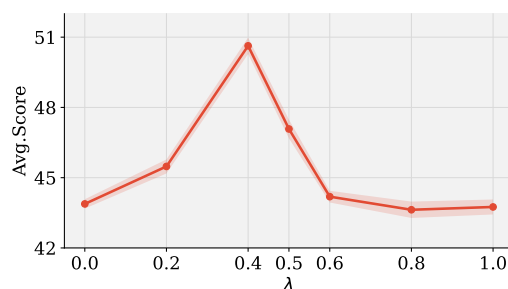


Figure 2: Effect of the fusion weight λ on ADG. We report the mean Avg.Score over five runs, with the shaded band indicating standard deviation across five runs.

Ablation on the fusion weight λ . Figure 2 shows a clear unimodal behavior with the best performance at $\lambda=0.4$. Both extremes degrade performance, indicating that the two scoring components provide complementary signals and that a moderate fusion yields the most reliable selection quality.

Ablation on selection ranks (Top/Middle/Tail 10K). Selecting the *top*-ranked 10K consistently outperforms the *middle* and *tail* segments by a large margin. This gap confirms that ADG induces a meaningful ordering that separates training-effective instructions from lower-value supervision.

Ablation on sampling temperature for scoring. Using a moderate temperature ($T=1.4$) yields the best results, while lower ($T=1.0$) or higher ($T=2.0$) temperatures hurt performance. This suggests ADG benefits from a suitable level of output diversity: too low under-exposes informative variation, while too high adds noisy dispersion.

Ablation on the number of bins. The stratified selection stage exhibits a sweet spot at $B=1000$. Fewer bins reduce coverage and increase redundancy, whereas too many bins over-fragment the

Ablation Setting	Avg.Score
(A) Score-rank	
Top 10K (default)	50.85
Middle 10K	42.69
Tail 10K	37.93
(B) Sampling temperature for scoring (top-$p=0.9$)	
$T=1.0$	48.49
$T=1.4$ (default)	50.85
$T=2.0$	42.92
(C) Number of bins for stratified selection	
$B=200$	45.23
$B=500$	46.23
$B=1000$ (default)	50.85
$B=2000$	45.91
(D) Scoring components	
$D(x)$ only ($\lambda=0$)	43.75
$I(x)$ only ($\lambda=1$)	43.88

Table 2: Ablations on key components and hyperparameters of ADG: (A) selecting 10K examples from different score-rank segments, (B) the temperature used for multi-sample answer generation, (C) the number of bins (B) for stratified top selection, and (D) the necessity of combining the two complementary scoring components. We report mean Avg.Score over five runs.

pool and weaken within-bin top selection. Overall, moderate bin granularity best balances coverage and exploitation under a fixed budget.

Ablation on scoring components. We ablate the scoring function by using only $D(x)$ or only $I(x)$, with all other settings fixed. Both variants underperform the full score, indicating that dispersion magnitude or shape anisotropy alone is insufficient. Combining $D(x)$ and $I(x)$ yields the best Avg.Score, confirming their complementarity.

4.2 Cross-backbone Generalization

Setup. We study whether ADG rankings transfer across backbones when the model used for scoring differs from the model used for finetuning. We conduct this experiment on Alpaca-GPT4 dataset with a fixed 10K selection budget. Concretely, we first run ADG with a *selection backbone* to produce a 10K subset. We then use a different *fine-tune backbone* on the selected subset with the same training recipe as in our main experiments.

Results. Table 3 reports the cross-backbone transfer matrix between Qwen2.5 and LLaMA3. Overall, using a different backbone for scoring and selection results in only a small drop in Avg.Score, suggesting that ADG captures a largely transferable ranking signal rather than one that is tightly cou-

Fine-tune Selection		Reason.	Know.	Code.	Avg.Score
Qwen2.5	Qwen2.5	75.43	67.36	64.40	69.06
Qwen2.5	LLaMA3	76.09	67.11	62.69	68.63
LLaMA3	LLaMA3	50.11	60.31	42.14	50.85
LLaMA3	Qwen2.5	49.84	59.61	41.11	50.19

Table 3: Cross-backbone generalization of ADG on Alpaca-GPT4 dataset with a 10K budget: we score and select data using the specified selection backbone, then fine-tune the specified backbone.

pled to a specific backbone. Notably, the transfer gap is more pronounced on coding than on reasoning/knowledge. A plausible explanation is that coding instructions are more sensitive to backbone-specific inductive biases about program structure and formatting; consequently, divergence geometry estimated by a different backbone may place relatively more weight on stylistic variability and less on correctness-relevant signals, leading to a slightly larger mismatch in the selected subset.

4.3 Selection Efficiency

Setup We report the end-to-end selection time on Alpaca-GPT4 ($N=52,002$) for constructing a 10K subset. All methods are run on the same machine with 4 NVIDIA A800 GPUs in a consistent runtime environment. The reported wall-clock time covers the full pipeline, including data loading, scoring (and answer sampling/generation when applicable), clustering/binning, and ranking, thus reflecting both GPU and CPU stages.

Selector	Time (h)↓	Ex/s↑
SuperFiltering	1.05	13.76
ADG (ours)	1.78	8.10
ZIP	3.17	4.56
MIG	3.92	3.69

Table 4: End-to-end selection efficiency on Alpaca-GPT4 ($N=52,002$) for producing a 10K subset using 4 GPUs. Ex/s is computed as $N/(\text{time} \times 3600)$.

Results. Table 4 shows that ADG incurs a moderate selection cost. ADG completes end-to-end selection in 1.78 hours (8.10 ex/s). The overhead mainly comes from multi-sample decoding and representation extraction, which are necessary for computing divergence-based scores. In practice, selection is an offline, one-time preprocessing step, and ADG offers a flexible cost-quality trade-off via K and the maximum generation length (see Appendix I).

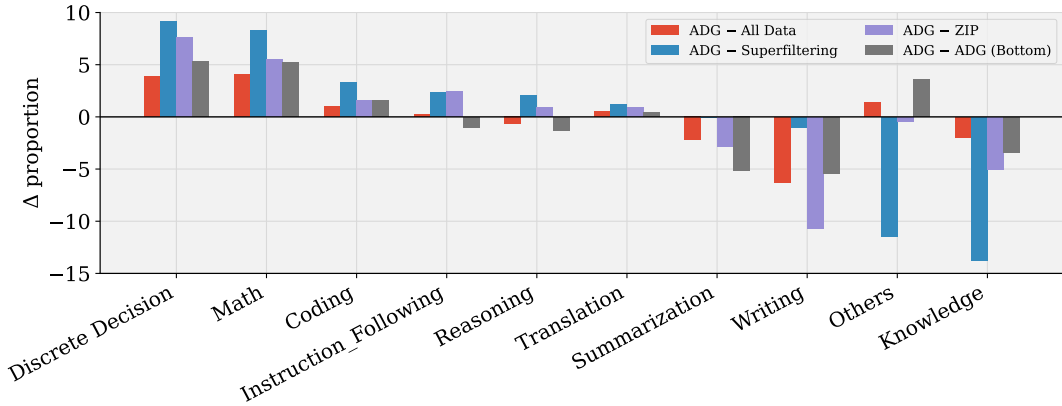


Figure 3: **Task-type composition shift of ADG-selected data.** We report Δ proportion (percentage-point change) between ADG and four reference sets (All Data, SuperFiltering, ZIP, and ADG (Bottom)) on Alpaca-GPT4. Task types are ordered by $\Delta(\text{ADG} - \text{SuperFiltering})$ from high to low.

4.4 Robustness to Alternative Output Representation Choices

Setup. Our default ADG setting represents each sampled response by averaging the output hidden states from the last four transformer layers. We use this same configuration across all selection pools and backbones, without any per-dataset tuning. This choice is also consistent with recent findings that middle-to-late layers of LLMs often provide stronger and more stable representations for downstream adaptation and transfer than early layers (Raja and Vats, 2025; Skean et al., 2025). To examine the robustness of ADG to alternative representation choices, we shift the 4-layer embedding window across the network. We use LLaMA3-8B-Instruct as the backbone and Alpaca-GPT4 as the instruction pool, with the same 10K selection budget as in the main experiments. All other settings are kept identical to the default ADG configuration.

Results. Table 5 shows that the best performance is achieved by the last-four-layer window (28–31), which reaches an Avg.Score of 50.85. Nearby higher-layer choices remain competitive, such as 24–27 (49.70) and 16–19 (49.10), whereas early-layer windows perform clearly worse, such as 0–3 (46.28) and 8–11 (43.82). These results suggest that ADG is not overly sensitive to the exact layer choice as long as higher-layer representations are used, and that averaging the last four layers provides a simple and robust default.

4.5 Task-type Distribution of Selected Instructions

Setup. We categorize Alpaca-GPT4 instructions into ten coarse task types using GPT-4o as a classi-

Layer	Reason.	Know.	Code.	Avg.Score
0–3	44.46	57.52	36.87	46.28
4–7	42.81	57.54	36.59	45.65
8–11	38.37	57.11	35.97	43.82
12–15	42.74	56.92	38.08	45.91
16–19	47.62	59.27	40.41	49.10
20–23	50.06	54.82	39.80	48.23
24–27	48.52	57.96	42.63	49.70
28–31 (ours)	50.11	60.31	42.14	50.85

Table 5: Robustness of ADG to different 4-layer output representation windows on LLaMA3-8B-Instruct with Alpaca-GPT4 under a fixed 10K selection budget. Reason., Know., and Code. denote the average performance on reasoning (BBH, GSM8K), knowledge (MMLU, TruthfulQA), and code generation (HumanEval, MBPP), respectively. Avg.Score is the unweighted average of the three capability scores.

fier, and analyze how different selectors change the *composition* of the resulting 10K subset. Figure 3 reports Δ proportion (percentage-point change) between the ADG-selected subset and four reference sets (All Data, SuperFiltering, ZIP, and ADG Bottom), with task types ordered by $\Delta(\text{ADG} - \text{SuperFiltering})$ from high to low. The GPT-4o classification prompt is provided in the Appendix B.

ADG rebalances the selected subset toward structured tasks. As shown in Figure 3, ADG consistently increases the share of Discrete Decision and Math (with a smaller gain on Coding), while mainly reducing writing and Knowledge-heavy prompts. The same pattern holds when comparing against the full pool and the ADG-bottom subset, suggesting that the shift is robust to the choice of reference.

Why Discrete Decision and Math are favored.

This trend aligns with ADG’s multi-sample divergence scoring: Discrete Decision tasks often yield a few well-separated output modes corresponding to competing choices, and Math tasks amplify small reasoning differences into different final answers. In both cases, higher divergence more often reflects verifiable disagreement than surface-level paraphrasing, making these instructions more boundary-revealing and learnable.

Overall, ADG tends to prioritize regimes where response disagreement more often corresponds to *learnable decision boundaries* and *verifiable reasoning*, providing a data-level perspective on its stronger downstream generalization. Appendix G further provides an LLM-based sample-property analysis along five dimensions, offering complementary evidence about the characteristics of ADG-selected data.

4.6 Quadrant-based Case Study

Setup. We interpret ADG by mapping each instruction onto a 2D plane of $D(x)$ (Dispersion magnitude) and $I(x)$ (Shape anisotropy). For each instruction, we sample five responses and illustrate the disagreement patterns; Figure 4 shows one representative example per quadrant. The four regions correspond to distinct types of behavioral variation, explaining why ADG favors the high- $D(x)$ and high- $I(x)$ regime.

Low $I(x)$, Low $D(x)$: single-mode, low-signal. Outputs nearly collapse to one answer (often with minor formatting noise), so sampling reveals little meaningful disagreement and mainly reinforces already-stable behavior.

High $I(x)$, Low $D(x)$: template-stable with a localized split. Responses share a consistent structure, but differ on a specific semantic pivot (e.g., who/when/what), yielding a pronounced shape (high $I(x)$) yet limited spread. Such instances often indicate a localized decision boundary: the template is learned, while the key choice remains uncertain.

Low $I(x)$, High $D(x)$: diffuse spread from style/trajectory. Responses vary widely in verbosity or reasoning style (short vs. long, different phrasing), producing high dispersion without a clear anisotropic structure. This shows that $D(x)$ alone can be dominated by non-semantic variation,

and $I(x)$ helps separate structured disagreement from expression-level drift.

High $I(x)$, High $D(x)$: high-signal disagreement (selected). Samples are both widely spread and clearly structured, suggesting competing semantic routes rather than mere paraphrases. These instances are boundary-revealing and provide higher supervision gain under a fixed budget by pushing the model toward consistent decisions across genuinely multi-modal solution spaces.

Overall, neither high dispersion nor high anisotropy alone is sufficient. ADG prioritizes instances with *both* high $D(x)$ and high $I(x)$, where disagreement is strong yet structured, making it more likely to correspond to learnable decision boundaries instead of noisy stylistic variability.

5 Related Work

Early instruction tuning showed that curated instruction–response pairs can substantially improve alignment and generalization, motivating *data-centric* pipelines that enhance corpus quality via manual or LLM-assisted cleaning, rewriting, and diversity control (Wang et al., 2023; Zhou et al., 2023; Chen et al., 2024; Li et al., 2024a; He et al., 2025; Li et al., 2024b; Zhou et al., 2026; Li et al., 2026). In parallel, *model-centric* selectors score examples using a reference model (e.g., loss/uncertainty/influence) and then select top-ranked subsets, with recent work also analyzing their behavior and biases (Liu et al., 2024; Xia et al., 2024a; Mekala et al., 2024; Liu et al., 2025; Wang et al., 2024; Xia et al., 2024b). Beyond per-example scoring, several methods incorporate global information structure or stylistic signals into selection (Chen et al., 2025; Li et al., 2025). Despite these advances, most methods still reduce each example to a single scalar from a single forward pass and rely on global top- k selection, making them sensitive to format, domain, and length. ADG adapts multi-sample disagreement ideas to data selection (Seung et al., 1992; Lakshminarayanan et al., 2016; Wang et al., 2022; Yao et al., 2023). It scores each *instruction* by the geometric structure of its multi-sample outputs in an embedding space, using dispersion magnitude and shape anisotropy with selection with proportional quotas, providing a distribution-aware signal complementary to scalar loss/uncertainty and style-based selectors.

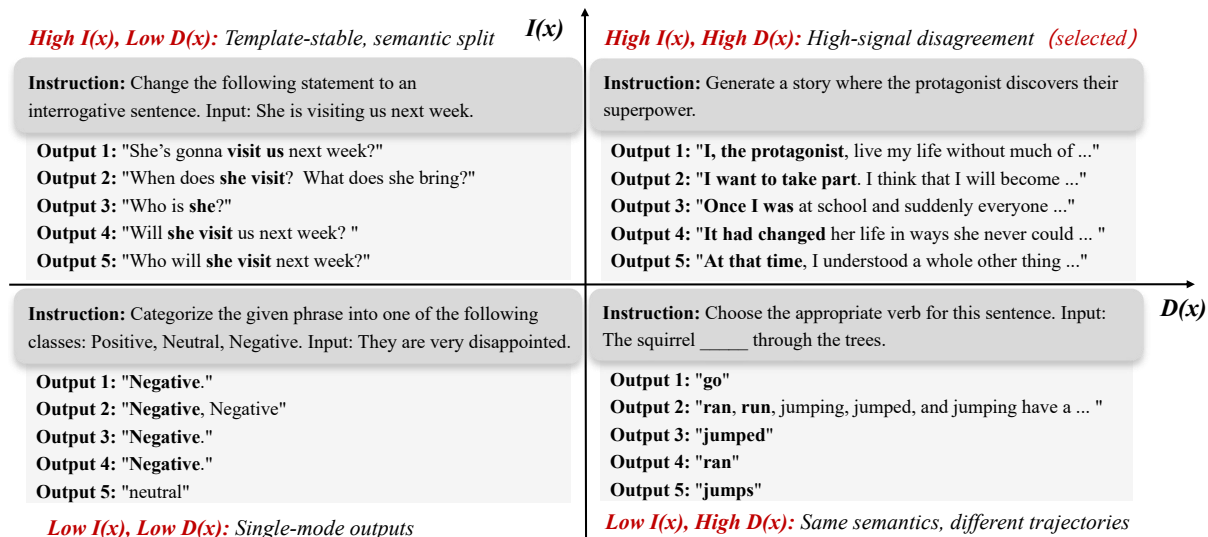


Figure 4: A quadrant-based case study of ADG using $D(x)$ (Dispersion magnitude) and $I(x)$ (Shape anisotropy). Each quadrant shows one instruction with five sampled outputs, illustrating distinct disagreement patterns; ADG favors the high- $D(x)$ and high- $I(x)$ region (top-right).

6 Conclusion

We introduced ADG, a geometry-based, model-centric data selection method that leverages multi-sample behavior to identify training-effective instructions. By jointly modeling dispersion magnitude and shape anisotropy and selecting within semantic bins, ADG prioritizes instances where disagreement is both strong and structured, while avoiding collapse onto length- or domain-biased regions. Experiments across two backbones and three instruction pools show consistent gains with only 10K selected examples, and analyses further validate the complementarity of the two geometric cues. These results suggest answer divergence is a simple, scalable, and broadly applicable organizing principle for efficient instruction data selection.

Limitations

First, ADG uses a fixed design for answer-embedding extraction (mean-pooling output-token hidden states and averaging late layers). While alternative pooling and layer choices may further improve the signal-to-noise trade-off, we follow common settings in prior work and validate key choices with targeted ablations. Second, ADG uses multi-sample disagreement structure as an indirect proxy for training value. In some cases, high divergence may also stem from prompt ambiguity or underspecified supervision. Combining ADG with lightweight quality or clarity filters is a promising direction for future work.

References

- Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. [Program synthesis with large language models](#). *ArXiv*, abs/2108.07732.
- Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. [Alpagasus: Training a better alpaca with fewer data](#). In *The Twelfth International Conference on Learning Representations*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 34 others. 2021. [Evaluating large language models trained on code](#). *ArXiv*, abs/2107.03374.
- Yicheng Chen, Yining Li, Kai Hu, Zerun Ma, Haochen Ye, and Kai Chen. 2025. [Mig: Automatic data selection for instruction tuning by maximizing information gain in semantic space](#). In *Annual Meeting of the Association for Computational Linguistics*.
- Karl Cobbe, Vineet Kosaraju, Mo 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*, abs/2110.14168.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, and et al. 2024. [The llama 3 herd of models](#).

- Junliang He, Ziyue Fan, Shaohui Kuang, Li Xiaoqing, Kai Song, Yaqian Zhou, and Xipeng Qiu. 2025. Fine: Filtering and improving noisy data elaborately with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8686–8707.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi lei, Yao Fu, Maosong Sun, and Junxian He. 2023. [C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models](#). In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2016. [Simple and scalable predictive uncertainty estimation using deep ensembles](#). In *Neural Information Processing Systems*.
- Bo Li, Shikun Zhang, and Wei Ye. 2026. [Data selection for multi-turn dialogue instruction tuning](#). *Preprint*, arXiv:2604.07892.
- Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Tim Baldwin. 2023. [Cmmlu: Measuring massive multi-task language understanding in chinese](#). *ArXiv*, abs/2306.09212.
- 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](#). *arXiv preprint arXiv:2402.00530*.
- 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.
- Zhuang Li, Yuncheng Hua, Thuy-Trang Vu, Haolan Zhan, Lizhen Qu, and Gholamreza Haffari. 2025. [SCAR: Data selection via style consistency-aware response ranking for efficient instruction-tuning of large language models](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12756–12790, Vienna, Austria. Association for Computational Linguistics.
- Stephanie C. Lin, Jacob Hilton, and Owain Evans. 2021. [Truthfulqa: Measuring how models mimic human falsehoods](#). In *Annual Meeting of the Association for Computational Linguistics*.
- Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. 2024. [Selectit: Selective instruction tuning for llms via uncertainty-aware self-reflection](#). *Advances in Neural Information Processing Systems*, 37:97800–97825.
- Peiyang Liu, Zhirui Chen, Xi Wang, Di Liang, Youru Li, Zhi Cai, and Wei Ye. 2026. [Learning from contrasts: Synthesizing reasoning paths from diverse search trajectories](#). *Preprint*, arXiv:2604.11365.
- Ziche Liu, Rui Ke, Yajiao Liu, Feng Jiang, and Haizhou Li. 2025. [Take the essence and discard the dross: A rethinking on data selection for fine-tuning large language models](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6595–6611, Albuquerque, New Mexico. Association for Computational Linguistics.
- Dheeraj Mekala, Alex Nguyen, and Jingbo Shang. 2024. [Smaller language models are capable of selecting instruction-tuning training data for larger language models](#). *ArXiv*, abs/2402.10430.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. 2022. [Training language models to follow instructions with human feedback](#). *ArXiv*, abs/2203.02155.
- Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. [Instruction tuning with gpt-4](#). *ArXiv*, abs/2304.03277.
- Rahul Raja and Arpita Vats. 2025. [Evaluating generalization and representation stability in small llms via prompting, fine-tuning and out-of-distribution prompts](#). *ArXiv*, abs/2506.17289.
- Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, and 21 others. 2022. [Multitask prompted training enables zero-shot task generalization](#). In *International Conference on Learning Representations*.
- H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. 1992. [Query by committee](#). In *Annual Conference Computational Learning Theory*.

- Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. 2025. [Layer by layer: Uncovering hidden representations in language models](#). *ArXiv*, abs/2502.02013.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. [Challenging BIG-bench tasks and whether chain-of-thought can solve them](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.
- Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. 2024. [A survey on data selection for llm instruction tuning](#). *ArXiv*, abs/2402.05123.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed H. Chi, and Denny Zhou. 2022. [Self-consistency improves chain of thought reasoning in language models](#). *ArXiv*, abs/2203.11171.
- 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.
- Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. [Finetuned language models are zero-shot learners](#). In *International Conference on Learning Representations*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022b. [Chain of thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems*.
- 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*.
- Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. 2024b. [Re-thinking data selection at scale: Random selection is almost all you need](#). *ArXiv*, abs/2410.09335.
- Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. [WizardLM: Empowering large pre-trained language models to follow complex instructions](#). In *The Twelfth International Conference on Learning Representations*.
- Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024. [Qwen2.5 technical report](#). *ArXiv*, abs/2412.15115.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of thoughts: Deliberate problem solving with large language models](#). *ArXiv*, abs/2305.10601.
- 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](#). *ArXiv*, abs/2407.06645.
- Yiheng Zhao and Jun Yan. 2026. [Generating effective cot traces for mitigating causal hallucination](#). *Preprint*, arXiv:2604.12748.
- Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023. [Lima: Less is more for alignment](#). *Advances in Neural Information Processing Systems*, 36:55006–55021.
- Yixi Zhou, Fan Zhang, Yu Chen, Haipeng Zhang, Preslav Nakov, and Zhuohan Xie. 2026. [Fincards: Card-based analyst reranking for financial document question answering](#). *Preprint*, arXiv:2601.06992.

Appendix

A Dataset and Evaluation Protocol

Dataset	Number Instances
<i>Training</i>	
Alpaca-GPT4	52,002
WizardLM	70,000
CoT	100,000
Alpaca-GPT4-ZH	42,677
<i>Evaluation</i>	
BBH	6,511
GSM8K	1,319
HumanEval	164
MBPP	500
MMLU	56,168
TruthfulQA	5,882
CMMLU(ZH)	23,232
C-Eval(ZH)	2,764

Table 6: Datasets used in our experiments. We report the training selection pools and evaluation benchmarks, together with the number of instances for each dataset.

Dataset. We evaluate ADG on three English instruction-tuning pools (Alpaca-GPT4, WizardLM, and CoT) and additionally include a public Chinese instruction pool (Alpaca-GPT4-ZH) for robustness under language shift. For evaluation, we cover a broad set of widely used benchmarks spanning reasoning (BBH, GSM8K), coding (HumanEval, MBPP), and knowledge (MMLU, TruthfulQA), and further report results on two Chinese benchmarks (CMMLU and C-Eval). Table 6 lists the dataset sizes used throughout the paper.

Evaluation Protocol. We evaluate all fine-tuned models using the open-source *EleutherAI Language Model Evaluation Harness* (lm-evaluation-harness).² To ensure fair comparison across selection methods, we keep the evaluation pipeline fixed and use the harness’ default task configurations for all. In particular, we adopt the official few-shot settings for each benchmark: BBH (3-shot), GSM8K (8-shot), TruthfulQA-MC2 (0-shot), MBPP (1-shot), HumanEval (0-shot), and MMLU (4-shot). We use the harness’ default decoding and scoring options for each task. All reported results are obtained under this unified evaluation setup.

²<https://github.com/EleutherAI/lm-evaluation-harness>

B Task-Type Classification Prompt

<p>Prompt: Please classify the following instruction into ONE of these categories:</p> <p>Categories:</p> <ul style="list-style-type: none">- Math: Mathematical problems, calculations, equations- Knowledge: Factual questions, general knowledge queries- Reasoning: Logic puzzles, analytical thinking, problem-solving- Writing: Creative writing, article writing, content generation- Coding: Programming tasks, code generation, debugging- Summarization: Text summarization, content condensing- Translation: Language translation tasks- Discrete Decision: Categorization, labeling tasks- Instruction Following: Following specific instructions or procedures- Others: Tasks that don't fit the above categories <p>Instruction: {instruction}</p> <p>Please respond with ONLY the category name (e.g., "Math" or "Coding"). No other text.</p>

Figure 5: Task-Type classification prompt used in this paper.

To analyze task-type distribution shifts (Section 4.5), we use GPT-4o to assign each Alpaca-GPT4 instruction to one of ten coarse categories (e.g., Math, Coding, and Others). Figure 5 shows the exact prompt template. We require the model to output *only* the category name to reduce formatting variance and ease automatic parsing.

C Robustness to Chinese Instructions

Selector	Avg.Score	CMMLU	C-Eval
All Data	40.84	41.11	40.56
Random	41.32	42.08	40.56
SuperFiltering	44.50	44.02	44.97
Rethinking	42.40	42.53	42.27
ADG (Ours)	45.54	45.09	45.99

Table 7: Results on Chinese benchmarks. We select 10K instances from Alpaca-GPT4-ZH, fine-tune LLaMA3-8B-Instruct and evaluate on CMMLU and C-Eval. We report the mean over five runs.

Setup. To assess robustness beyond English, we additionally evaluate on Alpaca-GPT4-ZH, a publicly available Chinese instruction-tuning pool with 42.6K instances (Peng et al., 2023). We then select 10K samples using each selector, fine-tune LLaMA3-8B-Instruct under the same recipe as the main experiments, and evaluate the resulting models on two Chinese knowledge benchmarks, CMMLU (Li et al., 2023) and C-Eval (Huang et al., 2023), and we report the average score over five runs.

Results and discussion. As shown in Table 7, ADG achieves the best performance on the aggregated score and on both Chinese benchmarks, demonstrating that our selection criterion transfers

well beyond the original English evaluation suite. Compared with the strongest baseline selector (SuperFiltering), ADG improves the overall Avg.Score by about **+2.3% relative**, with consistent gains on CMMLU (+2.4%) and C-Eval (+2.3%). Notably, training on the Chinese instruction-tuning pool is not always optimal, and even random selection can be competitive, suggesting that the Chinese instruction pool still contains redundancy and noisy supervision. These results suggest that ADG captures model-behavior signals that remain informative when the instruction language changes, rather than relying on language-specific heuristics.

D Full Benchmark Results

To provide a complete view beyond the aggregated averages reported in the main text, we present detailed results for **both backbones on all six evaluation benchmarks** in this appendix. Specifically, we report per-benchmark scores for models fine-tuned on the 10K subsets selected by each method in Table 11 and Table 12, using the same fine-tuning budget and evaluation protocol as in Section 3.4. These tables clarify where ADG yields the largest gains and help verify that improvements are not driven by a single benchmark, but are consistently observed across reasoning, knowledge, and coding evaluations.

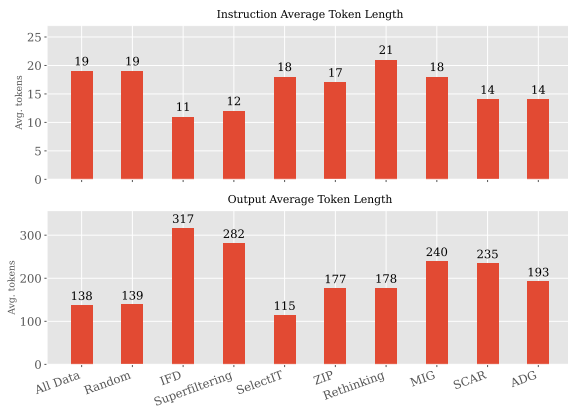


Figure 6: Average token length of the selected subsets, reporting instruction (input) and response (output) lengths under the same tokenizer.

E Instruction and Response Length Statistics

Figure 6 summarizes the *average token length* of the selected subsets, measured separately for the instruction (input) and the model-generated response (output) using the same tokenizer. Overall, ADG

produces **moderate-length** training data: its selected instructions are shorter than the full pool (14 vs. 19 tokens on average), while its responses are *not* excessively long (193 tokens), staying close to ZIP/Rethinking (177–178) and far below the very long-response subsets (e.g., IFD at 317 and SuperFiltering at 282). In contrast, some baselines exhibit clearer length skew, such as selecting very short instructions (IFD, SuperFiltering) or very short outputs (SelectIT at 115). These patterns suggest that ADG’s improvements are unlikely to come from simply shifting toward longer (or shorter) generations; instead, ADG selects samples with more informative multi-answer behavior while keeping response verbosity within a reasonable range.

F Behavior under Different Selection Budgets

Setup. We study how ADG behaves as the selection budget increases. Due to rebuttal-time constraints, we conduct a focused budget-scaling experiment on LLaMA3-8B-Instruct using AlpacaGPT4 (52K) as the selection pool. We evaluate one representative benchmark for each capability group: BBH for reasoning, MMLU for knowledge, and HumanEval for coding. We vary the selection budget over $M \in \{3K, 5K, 10K, 20K, 25K\}$, corresponding to approximately 6%, 10%, 20%, 39%, and 48% of the full pool, respectively. All methods use the same SFT protocol, and we compare ADG against Random and SuperFiltering under identical settings.

Results. Table 10 shows that ADG consistently outperforms Random across all selection budgets, and also surpasses SuperFiltering in Avg.Score at every budget we test. The gains are already visible under small budgets, where ADG achieves 52.89 at 3K and 52.69 at 5K, and remain strong at larger budgets, including 54.04 at 10K and 54.00 at 25K. At the same time, the gains do not increase monotonically with the budget. In particular, performance improves from 3K to 10K, but does not continue to rise thereafter, suggesting that the benefit of simply increasing the selected subset gradually saturates. This trend is consistent with our overall motivation: under a fixed training recipe, selection quality matters more than selecting more data indiscriminately.

Method	Cognitive Complexity	Instruction Precision	Response Quality	Educational Value	Scalability Potential
Random	3.94	8.10	7.64	5.00	5.56
SuperFiltering	5.79	8.56	8.07	6.83	7.27
ADG_Bottom	3.02	8.02	7.74	4.13	4.80
ADG_Top(default)	6.37	8.64	8.16	7.11	7.48

Table 8: LLM-based rating scores (GPT-4o) on five sample-quality dimensions. Each value is the mean over 1,000 randomly sampled instruction–response pairs from the corresponding subset.

K	Avg.Score	BBH	GSM8K	MMLU	TruthfulQA	HumanEval	MBPP
3	49.32	53.05	42.00	62.40	54.32	37.80	46.40
5 (default)	50.85	57.38	42.84	62.67	57.94	42.07	42.20
7	46.79	54.31	34.87	61.24	54.77	34.76	40.80
10	45.72	50.38	37.15	59.93	55.73	33.54	37.60

Table 9: Ablation on the number of sampled answers K for ADG scoring and selection, evaluated with a LLaMA3-8B-Instruct backbone on the Alpaca-GPT4 pool. Bold indicates the best score per column.

M	Method	BBH	MMLU	HumEval	Avg.Score
3K	Random	51.37	61.26	34.76	49.13
	SuperFiltering	51.94	61.65	42.68	52.09
	ADG	52.46	62.92	43.29	52.89
5K	Random	51.26	61.72	35.80	49.59
	SuperFiltering	54.04	59.97	37.80	50.60
	ADG	53.68	61.06	43.34	52.69
10K	Random	52.35	60.48	37.80	50.21
	SuperFiltering	53.86	60.13	40.85	51.61
	ADG	57.38	62.67	42.07	54.04
20K	Random	53.37	61.54	36.59	50.50
	SuperFiltering	55.07	61.12	35.37	50.52
	ADG	56.44	62.43	35.98	51.62
25K	Random	54.58	61.71	31.71	49.33
	SuperFiltering	51.97	61.85	42.73	52.18
	ADG	55.66	61.37	44.98	54.00

Table 10: Budget-scaling results on LLaMA3-8B-Instruct with Alpaca-GPT4 as the selection pool. Avg.Score is the unweighted average of the three representative benchmarks.

G Sample Property Analysis via LLM-based Rating

Goal and Evaluation protocol. We analyze whether examples selected by ADG exhibit known properties of high-quality instruction-tuning data, beyond downstream benchmark scores. In particular, we test if ADG tends to surface instructions that are (i) cognitively challenging yet learnable, (ii) well-specified, and (iii) paired with strong responses that are useful for generalization.

We use GPT-4o as an automatic rater and apply

a uniform sampling protocol across all compared subsets. For each subset produced by a selection method (e.g., Random, SuperFiltering, ADG_Top, and ADG_Bottom), we **randomly sample 1,000 examples** and ask GPT-4o to assign *pointwise* scores to each example along the five dimensions defined below. We then report the **mean rating** over the 1,000 scored examples for each subset. This design keeps the evaluation cost comparable across methods and avoids bias from manually selecting only the best-looking samples.

Rating dimensions. Each instruction–response pair is rated on a 1–10 scale over five aspects:

- **Cognitive complexity**, reflecting the degree of multi-step reasoning, abstraction, or domain knowledge required;
- **Instruction precision**, measuring clarity, specificity, and operationality of the instruction;
- **Response quality**, measuring correctness, completeness, and directness of the response;
- **Educational value**, assessing whether the pair teaches useful reasoning patterns or domain knowledge;
- **Scalability potential**, estimating how broadly the learned pattern can transfer to similar tasks.

Findings. Table 8 shows that ADG-selected examples score consistently higher than baselines across all five dimensions. Compared with Random and the ADG_Bottom subset, ADG_Top exhibits notably higher *cognitive complexity* and *educational value*, suggesting that ADG tends to prioritize training signals that are both challenging and instructive rather than trivial or low-yield. ADG_Top also maintains strong *instruction precision* and *response quality*, indicating that the higher complexity does not come from ambiguous prompts or poorly formed supervision. Overall, this analysis provides an interpretable, data-level view of *why* ADG-selected data leads to better downstream performance: it preferentially surfaces examples that are clearer, more instructive, and more transferable.

H Effect of the Number of Sampled Answers K

We ablate the number of sampled answers K , which controls how many outputs we generate per instruction to estimate the divergence-based score. We run this study on the Alpaca-GPT4 pool using LLaMA3-8B-Instruct as the backbone. All other settings follow the best configuration reported in the main paper, including the same decoding setup for answer sampling, the same feature extraction, the same binning strategy, and the same selection budget.

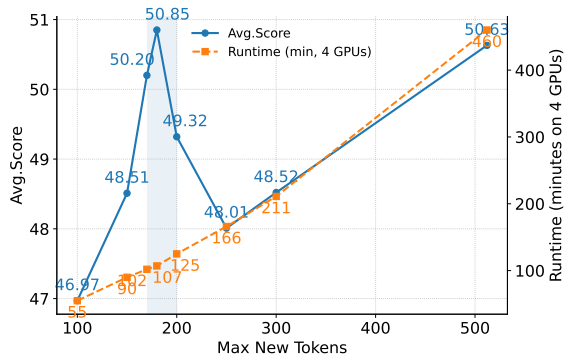


Figure 7: Trade-off of ADG scoring w.r.t. response sampling length on Alpaca-GPT4 dataset.

Table 9 shows a clear sweet spot at $K=5$, which achieves the best average score and leads on most benchmarks. When K is too small (for example, $K=3$), the estimated dispersion and shape statistics become less stable, which can weaken ranking quality under a fixed 10K budget, even though some tasks (such as MBPP) may still benefit. When

K is too large (for example, $K \geq 7$), performance drops consistently. A plausible reason is that excessive sampling introduces more off-manifold or stylistically drifting outputs, which can distort the divergence geometry and increase noise in the selection signal, while also increasing computation. Overall, moderate sampling provides enough diversity to reliably estimate the divergence structure while avoiding unnecessary noise and cost.

I Trade-off of Response Sampling Length

Setup. We ablate the maximum generation length (max new tokens) used when sampling multiple responses per instruction for ADG scoring. All other settings are kept identical to our main pipeline (same pool, scoring procedure, and 10K selection budget). We report the resulting Avg.Score together with the wall-clock runtime measured on 4 GPUs.

Findings. Figure 7 shows a clear trade-off between selection quality and scoring cost. As the generation cap increases from 100 to around 170–200 tokens, Avg.Score improves substantially. Beyond this range, the runtime grows rapidly while the quality gains become unstable, indicating diminishing returns for longer sampled responses. Based on this trade-off, we adopt max_new_tokens=180 as a practical default in our experiments.

J Results with a Small Backbone (LLaMA3.2-1B)

To test whether ADG remains effective for lightweight models, we repeat the main selection experiments using LLaMA3.2-1B-Instruct as the training backbone. We follow the same evaluation protocol as in the main paper: selecting a 10K subset from each pool (Alpaca-GPT4, WizardLM, and CoT), training with the same recipe, and evaluating on the same suite covering reasoning (BBH, GSM8K), knowledge (MMLU, TruthfulQA), and coding (HumanEval, MBPP). All baselines are run under the same budget and pipeline for a fair comparison.

Table 13 shows that ADG continues to deliver consistent gains under the 1B-scale backbone, achieving the best (or competitive-best) Avg.Score across the three pools. This suggests that the proposed divergence signal is not tied to large-capacity backbones: even when the model is small and its raw generation quality is weaker, the multi-sample output geometry still provides a useful proxy for

identifying training-effective instructions. In particular, the improvements on Alpaca-GPT4 and CoT indicate that ADG helps a small model avoid spending budget on redundant or weakly informative supervision, while preserving instructions that better shape structured reasoning and coding behaviors. Overall, these results support the robustness of ADG across model scales.

K Baseline Details

In this subsection, we provide concise descriptions for all baselines.

1) MIG: The method constructs a label graph in semantic space (labels as nodes and semantic relations as edges) and assigns each instruction example to its corresponding labels with quality-weighted contributions. It then performs selection by maximizing marginal information gain under a diminishing-returns objective, encouraging both high-quality signals and broad semantic coverage.

2) SCAR: The method decomposes response style into two key elements, linguistic form and instructional surprisal, and uses their consistency signals to identify instruction–response pairs that are more beneficial for efficient SFT under comparable quality. It trains a style consistency-aware response ranking model to score and rank examples, then selects a small but highly style-consistent subset for instruction tuning.

3) IFD: The method gives the target LLM a brief “experience” by training on a small subset that maintains semantic coverage, and then scores each candidate example with an Instruction-Following Difficulty (IFD) metric. The IFD score is computed from the loss difference between generating the answer with vs. without the instruction context, and the method selects the higher-IFD examples as training data.

4) Superfiltering: The method proposes a weak-to-strong data filtering pipeline where a much smaller filter model scores instruction data using difficulty-based signals. Motivated by the observed rank consistency of these difficulty scores across weak and strong models, it uses the weak model’s ranking to select a small subset for fine-tuning the larger target model at lower filtering cost.

5) ZIP: ZIP treats the compression ratio of a candidate set as a proxy for redundancy and greedily constructs a subset that minimizes the compression ratio to keep information dense and less repetitive. It uses a lightweight multi-stage procedure (can-

didate pre-filtering followed by cascaded greedy selection) to efficiently build a low-redundancy subset without relying on expensive strong-model scoring.

6) SelectIT: SelectIT ranks instruction-tuning examples by exploiting the target LLM’s own uncertainty as a self-reflection signal, combining token-level and sentence-level uncertainty across prompts to make scoring more robust. It then selects the highest-scoring subset for SFT, improving fine-tuning efficiency without requiring an extra external scoring model.

7) Rethinking: The method first partitions the instruction pool into N semantic clusters via k -means to preserve broad coverage, then allocates each cluster a quota proportional to its size. Within each cluster, it selects the examples with the largest token length, yielding a long-text-focused subset while maintaining semantic diversity through cluster-wise quotas.

	Reasoning		Knowledge		Coding		Avg.Score
	BBH	GSM8K	MMLU	TruthfulQA	HumanEval	MBPP	
<i>Alpaca-GPT4(52K)</i>							
All Data	55.28	36.62	60.42	54.84	32.93	42.80	47.15
Random	52.35	33.28	60.48	53.65	37.8	38.80	46.06
IFD	40.35	46.02	63.21	56.08	41.46	46.40	48.92
SuperFiltering	53.86	39.73	60.13	56.79	40.85	45.00	49.39
SelectIT	50.81	33.74	58.57	53.89	33.54	41.60	45.36
MIG	51.93	37.00	60.08	55.40	35.98	40.00	46.73
SCAR	53.74	42.46	61.43	58.54	33.54	38.20	47.99
Rethinking	53.05	36.32	60.56	56.04	37.80	38.60	47.06
ZIP	51.77	37.23	61.24	52.33	37.20	40.60	46.73
ADG(Ours)	57.83	42.84	62.67	57.94	42.07	42.20	50.85
<i>WizardLM(70K)</i>							
All Data	56.73	46.47	60.50	47.90	43.29	42.40	49.55
Random	55.06	41.70	61.47	52.31	38.41	39.20	48.03
IFD	58.29	42.38	61.64	53.82	39.63	45.40	50.19
SuperFiltering	58.58	43.59	61.94	53.79	39.02	45.20	50.35
SelectIT	54.25	35.78	60.58	46.25	33.54	41.80	45.37
MIG	55.38	38.51	61.11	50.36	40.85	41.80	48.00
SCAR	52.54	37.53	58.84	49.57	29.88	37.40	44.29
Rethinking	52.19	37.91	61.24	46.82	34.15	39.20	45.25
ZIP	54.37	34.95	61.06	52.33	32.93	40.00	45.94
ADG(Ours)	57.90	47.16	62.55	52.67	44.51	45.60	51.73
<i>CoT(100K)</i>							
All Data	44.97	47.38	51.91	43.29	22.56	31.20	40.22
Random	46.40	32.75	53.61	40.39	22.56	36.20	38.65
IFD	39.30	34.87	60.60	52.66	35.37	43.00	44.30
SuperFiltering	52.60	37.15	59.32	38.75	35.37	41.00	44.03
SelectIT	41.31	29.72	53.08	37.73	24.39	35.20	36.91
MIG	49.49	36.39	57.10	42.15	31.10	39.00	42.54
SCAR	44.92	36.62	54.76	42.74	21.34	36.80	39.53
Rethinking	47.67	37.38	57.38	43.06	26.22	39.20	41.82
ZIP	50.12	37.07	56.45	46.28	29.27	38.00	42.87
ADG(Ours)	53.79	36.31	58.04	43.94	37.19	41.40	45.11

Table 11: Detailed results on the LLaMA3-8B-Instruct backbone.

	Reasoning		Knowledge		Coding		Avg.Score
	BBH	GSM8K	MMLU	TruthfulQA	HumanEval	MBPP	
<i>Alpaca-GPT4(52K)</i>							
All Data	60.21	78.70	71.79	58.97	53.66	60.80	64.02
Random	62.17	81.20	73.04	60.56	61.59	62.00	66.76
IFD	63.23	83.85	73.55	59.20	60.98	62.60	67.24
SuperFiltering	63.71	82.56	73.29	60.73	61.59	66.40	68.05
SelectIT	63.48	79.39	73.47	54.18	62.20	60.40	65.52
MIG	65.85	83.70	73.55	60.70	59.15	62.60	67.59
SCAR	66.89	83.55	73.47	59.35	56.15	63.20	67.10
Rethinking	64.28	81.58	73.84	59.73	59.15	62.40	66.83
ZIP	65.04	80.89	74.04	60.79	62.02	64.60	67.90
ADG(Ours)	66.64	84.21	73.98	60.73	62.80	66.00	69.06
<i>WizardLM(70K)</i>							
All Data	62.88	78.70	72.61	55.38	63.41	46.80	63.30
Random	63.85	80.67	73.82	54.18	58.54	57.40	64.74
IFD	65.54	80.89	73.19	58.79	60.37	46.00	64.13
SuperFiltering	66.99	80.89	73.12	57.34	56.10	59.00	65.57
SelectIT	65.40	78.17	73.79	54.53	58.54	51.80	63.71
MIG	67.01	81.18	73.85	57.50	58.15	60.40	66.35
SCAR	66.26	81.04	72.97	55.90	56.71	58.00	65.15
Rethinking	65.15	79.83	73.86	55.32	56.10	55.60	64.31
ZIP	65.35	76.88	73.49	56.95	56.71	57.80	64.53
ADG(Ours)	67.65	82.49	74.28	59.05	60.37	62.00	67.64
<i>CoT(100K)</i>							
All Data	60.76	69.67	70.50	47.95	52.44	55.20	59.42
Random	62.62	70.05	73.19	43.04	54.27	57.60	60.13
IFD	61.97	76.41	72.85	45.89	56.1	57.80	61.84
SuperFiltering	62.48	74.37	72.80	47.09	54.88	55.40	61.17
SelectIT	62.17	70.74	72.52	45.25	54.27	54.80	59.96
MIG	64.08	71.72	73.37	45.11	53.66	58.40	61.05
SCAR	59.84	72.25	73.14	41.36	56.09	57.00	59.95
Rethinking	63.28	73.31	72.66	46.62	52.44	58.40	61.12
ZIP	60.87	72.93	72.45	44.79	57.93	55.40	60.73
ADG(Ours)	64.28	76.82	74.00	44.80	58.54	57.80	62.71

Table 12: Detailed results on the Qwen2.5-7B-Instruct backbone.

	Reasoning		Knowledge		Coding		Avg.Score
	BBH	GSM8K	MMLU	TruthfulQA	HumanEval	MBPP	
<i>Alpaca-GPT4(52K)</i>							
All Data	29.67	5.38	29.53	44.91	17.07	21.80	24.73
Random	31.07	6.75	31.76	38.43	17.07	22.20	24.55
IFD	31.98	7.28	30.52	39.12	18.9	24.20	25.33
SuperFiltering	31.77	6.82	31.8	41.92	17.68	26.20	26.03
SelectIT	30.29	7.05	34.23	42.05	18.29	23.20	25.85
MIG	30.21	4.70	30.34	41.49	17.68	23.60	24.67
SCAR	30.75	5.69	30.76	42.66	17.68	22.40	24.99
Rethinking	30.02	6.28	31.74	42.07	20.12	23.20	25.57
ZIP	30.22	6.90	33.62	42.05	20.73	22.20	25.95
ADG(Ours)	30.39	7.28	34.28	43.86	20.62	24.20	26.77
<i>WizardLM(70K)</i>							
All Data	29.77	6.67	29.75	40.19	17.68	21.20	24.21
Random	30.59	6.22	30.96	39.78	17.68	24.80	25.01
IFD	31.24	5.31	28.98	40.31	18.90	22.40	24.52
SuperFiltering	31.25	6.22	27.60	38.77	20.12	24.40	24.73
SelectIT	30.9	6.22	31.02	39.49	19.51	24.80	25.32
MIG	32.27	5.16	32.08	40.78	18.26	21.80	25.06
SCAR	31.52	6.22	32.29	42.19	17.07	21.60	25.14
Rethinking	31.53	6.52	27.92	40.29	20.12	22.40	24.80
ZIP	30.46	6.60	30.33	38.42	19.51	23.40	24.79
ADG(Ours)	32.13	6.84	32.78	39.53	20.73	24.80	26.13
<i>CoT(100K)</i>							
All Data	25.08	10.25	35.07	41.54	11.59	15.20	23.12
Random	26.94	9.93	34.21	36.91	17.68	24.00	24.95
IFD	27.29	9.02	29.30	40.10	19.51	23.20	24.74
SuperFiltering	27.80	6.60	34.28	38.15	17.68	25.40	24.99
SelectIT	27.42	7.96	33.41	37.79	18.90	22.60	24.68
MIG	28.50	9.40	30.05	39.22	15.24	22.80	24.20
SCAR	28.47	9.55	31.91	41.08	15.24	22.20	24.74
Rethinking	29.27	9.86	34.20	39.28	17.07	23.00	25.45
ZIP	27.35	11.60	35.48	37.20	15.85	26.20	25.61
ADG(Ours)	28.69	10.79	36.10	40.30	17.07	24.40	26.23

Table 13: Detailed results under a small backbone (LLaMA3.2-1B-Instruct). We report reasoning, knowledge, and coding performance on three selection pools (Alpaca-GPT4, WizardLM, and CoT) with a fixed 10K budget. ADG consistently improves the average score over strong selection baselines, indicating that divergence-guided selection remains effective even for a lightweight model.