Learning Multi-Indicator Weights for Data Selection: A Joint Task-Model Adaptation Framework with Efficient Proxies

Jingze Song, Zihao Chen, Wenqing Chen and **Zibin Zheng**
School of Software Engineering, Sun Yat-sen University
{songjz5, chenzh636}@mail2.sysu.edu.cn, {chenwq95, zhzibin}@mail.sysu.edu.cn

Abstract

Data selection is a key component of efficient instruction tuning for large language models, as recent work has shown that data quality often matters more than data quantity. Accordingly, prior studies have introduced various multi-dimensional heuristics to evaluate and filter instruction data. However, most existing methods rely on static task-agnostic and model-agnostic weighting schemes, which overlook the varying requirements of specific downstream tasks and the differing pre-existing capabilities of models. In this paper, we propose a framework for learning multi-indicator weights that jointly adapts data selection to both the downstream task and the specific model. Our method identifies optimal weight configurations without full-scale fine-tuning by utilizing in-context learning (ICL) signals on compact tiny-validation sets. These signals serve as efficient performance proxies that ensure high-fidelity evaluation at minimal computational cost. Experiments across multiple benchmarks and model families, including Mistral, Qwen, and Llama, show that the approach achieves performance comparable to or exceeding full-dataset tuning while using only 30% of the training samples on GSM8K. Furthermore, our analysis reveals a trade-off between semantic diversity and logical complexity in reasoning tasks, highlighting the necessityf of joint task-model adaptation.

1 Introduction

Instruction tuning adjusts large language models (LLMs) to match human intent, and the choice of training data significantly influences model performance. Recent studies indicate that larger datasets may not necessarily lead to better performance, as noisy data can incur high computational costs and introduce hallucinations [Chen and Mueller, 2024; Si *et al.*, 2025]. Instead, the effectiveness of instruction fine-tuning mainly depends on the quality and diversity of the samples, rather than the size of the dataset [Zhou *et al.*, 2023; Chen *et al.*, 2024].

Previous work generally evaluates instruction tuning data from three dimensions: quality, which reflects the correctness and utility of responses [Liu *et al.*, 2024; Chen *et al.*, 2024]; complexity, which captures the logical depth or difficulty of queries [Li *et al.*, 2024b]; and diversity, which ensures broad coverage across various domains [Ge *et al.*, 2024; Lu *et al.*, 2024]. Consequently, most selection pipelines aggregate these indicators using static weighting schemes or predefined manual rules [Cao *et al.*, 2024]. However, two challenges persist in these approaches. First, static weighting schemes encounter the problem of poor generalizability [Diddee and Ippolito, 2025] and do not account for the varying requirements of different downstream tasks [Cao *et al.*, 2024] and different LLMs. The optimal weighting among indicators is universal across different tasks; for instance, mathematical reasoning tasks demand high logical complexity data, whereas general-purpose alignment may prioritize linguistic diversity. Second, while some data-driven approaches optimize the selection by identifying influential samples [Xia *et al.*, 2024; Tang *et al.*, 2025], they often incur high computational costs. These methods typically rely on "select-finetune-evaluate" loops or high-dimensional gradient calculations to score the data. To reduce training costs, some recent approaches employ in-context learning (ICL) as a proxy for fine-tuning performance [Wang *et al.*, 2024]. However, they remain constrained by their sample-level granularity, suffering from high inference time costs due to the traversal of the dataset.

In this paper, we propose a joint **T**ask-**M**odel **A**daptation framework with efficient **P**roxies (TMAP) for instruction data selection. To deal with the first challenge, unlike approaches that rely on static heuristic rules, TMAP treats data selection as an optimization problem by dynamically adjusting the weighting of various indicators, to match the requirements of both the downstream task and the base model. To deal with the second challenge, TMAP introduces an efficient performance proxy that utilizes ICL signals on a tiny validation sets, bypassing the prohibitive cost of fine-tuning an LLM. By leveraging these high-fidelity proxies, TMAP can rapidly evaluate weight configurations and update them without training the model. This approach significantly reduces execution time while ensuring the selected data remains effective for target tasks. Our main contributions are summarized as follows:

- We propose a joint task-model adaptation framework that tackle the problem of static heuristics by using learned weights over multiple selection indicators to tailor data subset to specific task requirements and model capabilities.
- We introduce efficient proxies of select-finetune-evaluate loops via performing ICL on tiny validation sets. By utilizing validation results as proxies, we enable efficient weight-level optimization without the need for full-scale finetuning and full-validation inference.
- Experiments across various model families and benchmarks demonstrate that our method achieves performance comparable to full-dataset tuning, reaching considerable results on reasoning tasks such as GSM8K while utilizing only 30% of the training data.

2 Related Work

Instruction data selection with different indicators. Current indicators for instruction tuning can be broadly categorized into three types: *heuristic-based*, *LLM-based*, and *ICL-based*. **Heuristic-based** approaches assess data quality using predefined heuristic metrics. A simple yet effective metric is selecting data with the longest output length [Zhao *et al.*, 2024]. [Li *et al.*, 2024b; Li *et al.*, 2024a] propose an instruction following difficulty (IFD) metric that leverages GPT-2 to compare the perplexity of model responses with and without instructions. **LLM-based** methods employ LLMs as scorers to identify high-quality data samples. For instance, ALPAGASUS [Chen *et al.*, 2024] and DEITA [Liu *et al.*, 2024] use template prompts to allow LLMs to rate data for both quality and complexity.

Some LLM-based selection use **data-driven optimization** to identify influential data best aligned with model requirements. LESS [Xia *et al.*, 2024] utilizes gradients from LLM to select data, while iterative methods like Middo [Tang *et al.*, 2025] continually update the training LLM with selected high complexity and quality data in each iteration, aiming to select data that is aligned with the current model's needs. Additionally, LASER [Mirza *et al.*, 2025] trains and employs data-specialized scorers to assess data quality.

ICL-based methods assess the quality of each candidate data point by measuring its usefulness as an ICL example for a given evaluation set. Nuggets [Li *et al.*, 2024c] computes a quality score by comparing the correctness gap between a zero-shot setting and a one-shot setting that uses the candidate data as the prompt. [Zhang *et al.*, 2025] propose an in-context approximation (ICA) approach, which estimates the loss of candidate examples by conditioning on a carefully selected in-context held-out set, and then re-weights gradient updates accordingly. Data Whisperer [Wang *et al.*, 2025a] randomly samples data from the candidate pool as demonstrations and queries, then computes the weighted evaluation score for each query based on the attention mechanism. The scores quantify the usefulness of the candidate data.

These data selection methods, however, often overlook the specific needs of diverse downstream applications. Furthermore, ICL-based methods incur high computational costs for new data, and the reliability of using pre-defined or randomly sampled evaluation sets is limited. Iterative approaches typically rely on "select-finetune-evaluate" loops that are computationally expensive and difficult to scale.

Benchmark compression for efficient evaluation. The increasing computational costs required for evaluating LLMs have spurred significant research into efficient benchmarking methods. Many studies have identified redundancy in benchmark items [Ye *et al.*, 2023; Perlitz *et al.*, 2024]. Consequently, methods now employ carefully selected small evaluation subsets to predict full benchmark performance. For instance, [Vivek *et al.*, 2024] proposes the Anchor Points method and identifies critical data points to assess models with far fewer examples [Polo *et al.*, 2024].introduces TinyBenchmarks, demonstrating that by training an Item Response Theory (IRT) model [Cai *et al.*, 2016] on just 319 curated items can reliably estimate full performance on MMLU within 2% error [Wang *et al.*, 2025b]. proposes the EssenceBench framework, which formulates benchmark compression as an optimization problem and addresses it through an iterative genetic algorithm. Our work differs by focusing on learning task-model-adaptive weights over a rich set of indicators, while sharing only the goal of reducing the cost of efficient evaluation.

3 Methodology

3.1 Problem Formulation

We formulate the problem of instruction data selection as follows. Given a candidate pool $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ containing N samples, for each sample d_i, we pre-calculate K heuristic-based and LLM-based indicators $\{s_{i,j}\}_{j=1}^{K}$. These indicators are then min-max normalized to form a feature vector $s_i \in [0,1]^K$. Our objective is to determine an optimal weight configuration $\mathbf{w} = [w_1, w_2, \ldots, w_K]^\top, w_i \in [0,1]$ to calculate a weighted score $S_i(\mathbf{w})$ for each sample:

$$S_i(\mathbf{w}) = \sum_{j=1}^{K} w_j \cdot s_{i,j}$$

Based on these scores, a training subset $\mathcal{D}_{\text{sub}}(\mathbf{w})$ is formed by selecting the top-M samples where $M < N$. Our goal is to find an optimal $\mathbf{w}$ that maximizes the target task performance of the target LLM after instruction tuning.

Efficient Proxies. Ideally, data-driven optimization for the selection problem is formulated as a select-finetune-evaluate loop. The optimization objective in this paradigm is typically denoted as follows:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \underbrace{\text{Eval}(\underbrace{\text{SFT}(\mathcal{M}, \mathcal{D}_{\text{sub}}(\mathbf{w}))}_{\text{High-cost}}, \mathcal{V}_{\text{full}})}_{\text{High-cost}}. \quad (1)$$

where $\text{SFT}(\cdot)$ denotes the supervised fine-tuning of a LLM $\mathcal{M}$ on the selected subset, and $\text{Eval}(\cdot, \mathcal{V}_{\text{full}})$ represents evaluating on the full downstream validation set. However, this process is computationally prohibitive due to the repeated LLM training and large-scale evaluation inference. To address these challenges, we redefine the selection process using two efficient proxies: ICL and tinybenchmark. Instead

of performing full-scale fine-tuning, we utilize few-shot ICL to measure the utility of $\mathcal{D}_{\text{sub}}(\mathbf{w})$. Instead of the full validation set $\mathcal{V}_{\text{full}}$, we evaluate performance on a tiny validation set $\mathcal{V}_{\text{tiny}} \subset \mathcal{V}_{\text{full}}$. We define a reward function $R(\mathbf{w})$ that serves as a proxy for the fine-tuning outcome:

$$R(\mathbf{w}) = \text{Eval}\left(\text{ICL}(\mathcal{M}, \mathcal{D}_{\text{sub}}(\mathbf{w})), \mathcal{V}_{\text{tiny}}\right). \tag{2}$$

By shifting the optimization target to $\arg\max_{\mathbf{w}} R(\mathbf{w})$, the framework finds approximately optimal configurations at the indicator level while reducing computational costs and execution time. The remaining problem is how to guarantee that the performance approximation via efficient proxies is reasonable. We introduce the details in Section 3.4

3.2 Overview of TMAP

As show in Figure 1, we treat data selection as a weight optimization problem. The TAMP pipeline consists of: (1) sampling a tiny validation set via K-means on IRT embeddings; (2) scoring the candidate data pool across an extensible set of indicators; (3) calculating weighted scores and selecting the highest-scoring data to form an ICL demonstration set; (4) iteratively optimizing indicator weights using Bayesian Optimization, guided by an ICL-based reward signal on the tiny validation set; and (5) selecting the final subset based on the optimized weights once optimization finished.

3.3 Multi-Dimensional Indicators

To capture multiple features of data quality, we score each candidate sample d_i with K dimensions to form a normalized feature vector $s_i \in [0,1]^K$. Following by prior work, we implement four key indicator groups: (1) **Semantic Quality**: LLM-based scores `alpagasus_score` and `deita_quality_score`, which capture the faithfulness and helpfulness of responses [Chen *et al.*, 2024; Liu *et al.*, 2024]. (2) **Complexity and Difficulty**: `ifd` and `deita_complexity_score`, which measure how challenging an instruction-response pair is for the model [Li *et al.*, 2024a; Liu *et al.*, 2024]. (3) **Linguistic Heuristics**: `output_length` [Zhao *et al.*, 2024] metric, which aligns with general-purpose filtering. (4) **Domain-specific metrics**: For the GSM8K dataset, we use `reason_steps` [Jin *et al.*, 2024a] to approximate the length of the underlying chain-of-thought. Note that TMAP is easy to adapt to different domains (e.g., dropping reasoning metrics for dialogue tasks).

3.4 Efficient Proxies

Proxy for Finetuning. To avoid the high computation cost of repeated SFT in the optimization cycle, we leverage ICL as an efficient proxy. Recent studies confirm that a sample's utility as a few-shot demonstration strongly correlates with its utility as a training instance [Yin *et al.*, 2024; Zhao *et al.*, 2025]. Note that, unlike prior work, we do not replicate exact SFT performance. Instead, our goal is to preserve the relative ranking of data quality. As shown in Appendix A[1], varying the number of ICL demonstrations consistently maintain the same performance trends as observed in SFT training. This demonstrates that ICL can serve as a reliable proxy of SFT in evaluating the influence of different data on the target model. To reduce the impact of ICL prompt order sensitivity on the reward signal $R(\mathbf{w})$, we partition the selected subset $\mathcal{D}_{\text{sub}}$ into m distinct groups of n-shot exemplars. We then evaluate the base model using these groups as prompts, deriving the reward from the aggregated accuracy.

Proxy for Full-validation Evaluating. To avoid time-consuming full-set inference, we construct a compact **tinybenchmark** $\mathcal{V}_{\text{tiny}}$ of S representative samples. Our target is to approximate the full-set performance via weighted evaluation on this subset:

$$R(\mathbf{w}) = \frac{1}{|\mathcal{V}_{\text{full}}|} \sum_{v \in \mathcal{V}_{\text{full}}} Y_v \approx \sum_{v \in \mathcal{V}_{\text{tiny}}} w_v Y_v. \tag{3}$$

$Y_{v\mathcal{M}} \in \{0,1\}$ is the correctness of sample v and w_v is the weight of sample v. The underlying principle is that an LLM can accurately answer a broad range of questions if and only if it correctly answers these representative samples. To select representative samples, we employ IRT to generate embedding vectors. In the context of TMAP, we adopt the Multidimensional Two-Parameter Logistic (M-2PL) model from IRT. The M-2PL model assumes that the probability of the LLM $\mathcal{M}$ correctly solve the validation sample v is:

$$p_{vl} = \frac{1}{1 + exp(-(\alpha_v^\top \theta_{\mathcal{M}} + \beta_v))}, \tag{4}$$

where $\theta_{\mathcal{M}}$ represents latent LLM abilities and (α_v, β_v) represent item difficulty and discrimination. We concatenate the point estimates parameters to form embedding vectors $E_v = (\hat{\alpha}, \hat{\beta})$ for sample v. We apply K-means clustering to identify centroids and select the item v^* closest to form V_{tiny}:

$$\mathcal{V}_{\text{tiny}} = \{v_1^*, \ldots, v_o^*\} \tag{5}$$

To further refine precision of tinybenchmark, following [Polo *et al.*, 2024], we combine the observed results and IRT prediction:

$$R(\mathbf{w}) = \lambda \sum_{i \in \mathcal{V}_{\text{tiny}}} w_i Y_i + (1-\lambda)(\delta_1 \sum_{i \in \mathcal{V}_{\text{tiny}}} Y_i + \delta_2 \sum_{j \notin \mathcal{V}_{\text{tiny}}} \hat{p}_i) \tag{6}$$

where λ is a learnable parameters, $\delta_1 = \frac{\hat{\lambda}}{|\mathcal{V}_{\text{tiny}}|}$, $\delta_2 = \frac{1-\hat{\lambda}}{|\mathcal{V}_{\text{full}} \setminus \mathcal{V}_{\text{tiny}}|}$ and $\hat{\lambda} = |\mathcal{V}_{\text{tiny}}|/|\mathcal{V}_{\text{full}}| \in [0,1]$. Experiment in Appendix B proves the reliability of the proxy for full-validation evaluating.

3.5 Task-Model Adaptive Weight Optimization

We formulate data selection as a black-box optimization problem: $\mathbf{w}^* = \arg\max_{\mathbf{w}} R(\mathbf{w})$, where $R(\mathbf{w})$ represents the proxy reward signal. To efficiently solve this non-differentiable objective, we employ Bayesian Optimization (BO) [Močkus, 1974].

As outlined in Algorithm 1, the process iteratively updates weights $\mathbf{w}$. In each iteration t, we compute weighted scores $S_i = \sum_{j=1}^{K} w_{t,j} \cdot s_{i,j}$ for all candidates, select the top-M instances, and partition them into m groups. These groups are evaluated via ICL on $\mathcal{V}_{\text{tiny}}$ to obtain a robust reward signal R_t, which guides the Bayesian optimizer to propose the next configuration $\mathbf{w}_{t+1}$.

[1]Due to space constraints, the appendix is available in our extended arXiv version: https://arxiv.org/abs/2605.09665.

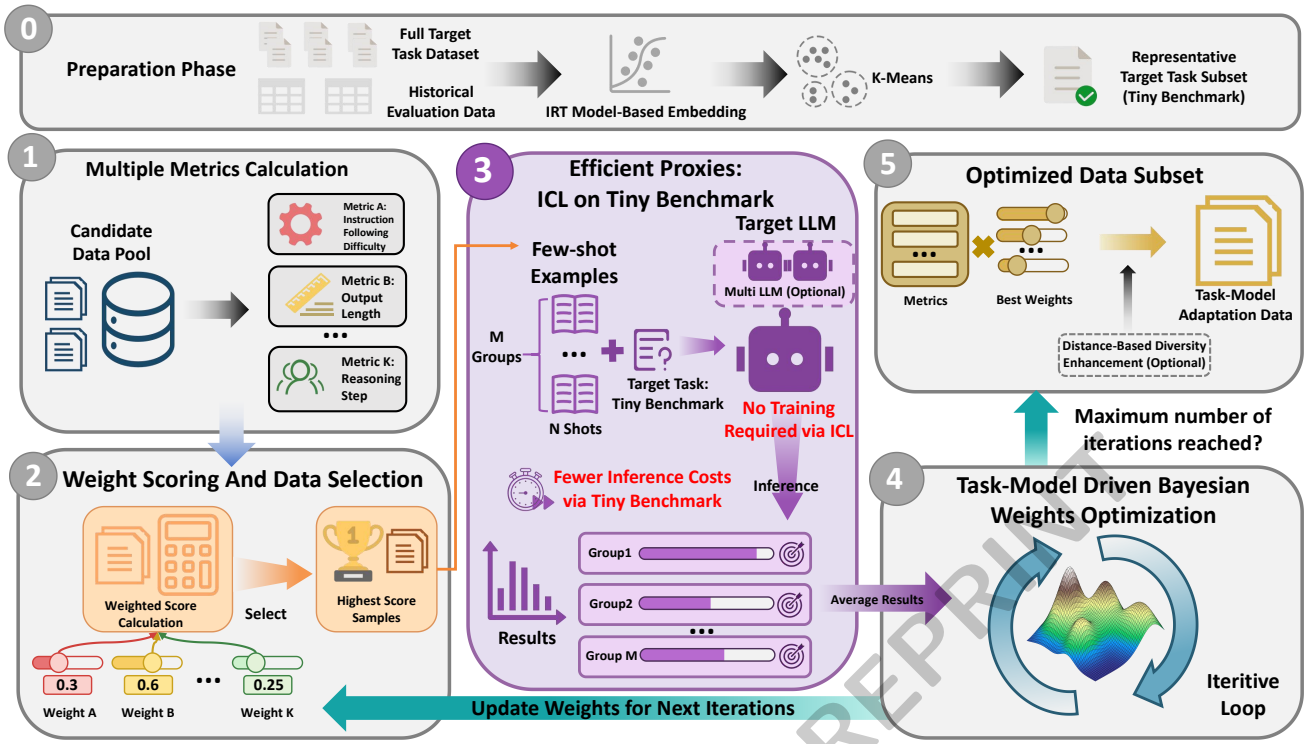

Figure 1: Overview of the Task-Model Adaptation with efficient Proxies (TMAP) data selection pipeline. The framework scores candidate instructions along multiple dimensions, optimizes task-specific weights, and selects a high-value subset for downstream fine-tuning.

Algorithm 1 Task-Adaptive Weight Optimization

Input: Candidate pool $\mathcal{D}$, Indicators $\{s_j\}_{j=1}^{K}$, Target task tiny validation set $\mathcal{V}_{\text{tiny}}$, Selection size M
Initialize: Weights $\mathbf{w}_0$, Bayesian Optimizer $\mathcal{B}$
for $t = 1$ **to** T **do**
  $S_i \leftarrow \sum_{j=1}^{K} w_{t,j} \cdot s_{i,j}$ for each $d_i \in \mathcal{D}$
  $\mathcal{D}_{sub} \leftarrow \text{Top-M}(\mathcal{D}, S_i)$
  $\{\mathcal{D}_{\text{sub}_l}\}_{l=1}^{m} \leftarrow \text{Partition}(\mathcal{D}_{\text{sub}})$
  $R_t \leftarrow \text{EvalICL}(\{\mathcal{D}_{\text{sub}_l}\}_{l=1}^{m}, \mathcal{V}_{\text{tiny}})$ {Average performance over m groups}
  $\mathbf{w}_{t+1} \leftarrow \text{Update}(\mathbf{w}_t, R_t)$
end for
Return $\mathbf{w}^*$

3.6 Final Filtering and Robustness Refinement

To capture the multifaceted nature of data quality, we score each candidate sample d_i across K dimensions to form a normalized feature vector $\mathbf{s}_i \in [0,1]^K$. Drawing from prior work, we implement four key indicator groups:

- **Semantic Quality**: LLM-based scores, specifically `alpagasus_score` [Chen *et al.*, 2024] and `deita_quality_score` [Liu *et al.*, 2024], which capture the faithfulness and helpfulness of responses.
- **Complexity and Difficulty**: Metrics like `ifd` and `deita_complexity_score`, which measure how challenging an instruction-response pair is for the model [Li *et al.*, 2024a; Liu *et al.*, 2024].
- **Linguistic Heuristics**: Simple statistics such as `output_length` [Zhao *et al.*, 2024], consistent with general-purpose filtering metrics.
- **Domain-Specific Metrics**: For reasoning tasks like GSM8K, we include `reason_steps` [Jin *et al.*, 2024a] to approximate the length of the underlying chain-of-thought.

Note that TMAP is designed to be extensible, allowing for easy adaptation to different domains (e.g., excluding reasoning metrics for dialogue tasks).

4 Experiments

4.1 Experimental Setup

Datasets. We focus on knowledge and reasoning benchmarks that are widely used for evaluating instruction-tuned LLMs. Specifically, we used the following datasets. **GSM8K**, a dataset of grade-school math word problems requiring multi-step reasoning [Cobbe *et al.*, 2021]. **ARC-C**, a challenging multiple-choice science QA subset with strong distractors [Clark *et al.*, 2018]. **MUSR**, a dataset for evaluating LLMs on multistep soft reasoning tasks specified in a natural language narrative [Sprague *et al.*, 2023]. **GPQA**, a multiple-choice dataset of hard questions created by experts in biology, physics, and chemistry. For the synthetic-data experiments, we additionally construct DeepSeek-distilled variants of GSM8K and ARC-C by augmenting each example

	Mistral-v0.2-7B			Qwen2.5-7B			Llama3-8B		
Method	**GSM8K**	**MUSR**	**GPQA**	**GSM8K**	**MUSR**	**GPQA**	**GSM8K**	**MUSR**	**GPQA**
Base	38.67	47.49	**28.86**	**79.08**	43.67	32.39	**72.10**	37.96	31.29
Full (100%)	**58.61**	46.30	28.78	75.36	**46.03**	**32.89**	64.67	40.21	**31.46**
Random	50.80	46.16	27.68	75.28	43.77	31.55	61.49	41.67	30.12
Metric-based Methods									
IFD	52.08	44.44	27.52	77.33	44.84	32.38	60.02	42.72	30.17
Longest	51.25	45.90	27.68	78.70	43.92	31.71	60.65	37.30	30.03
LLM-based Methods									
Alpagasus	52.77	46.43	27.36	76.57	44.02	32.21	60.88	42.86	30.08
DEITA	52.01	45.44	27.18	76.27	43.52	31.88	62.70	38.89	30.63
ICL-based Methods									
Data-Whisperer	49.58	47.89	27.77	78.77	44.56	32.55	60.73	41.80	30.20
TMAP (Ours)	55.42	**50.77**	27.94	79.00	45.90	32.74	65.96	**42.99**	30.73

Table 1: Performance on GSM8K/MUSR/GPQA datasets across different models. The best overall score in each column is highlighted in **bold**, while the best score among data selection methods (excluding Base and Full) is underlined. We omit the "Instruct" suffix in the model names for brevity. Base refers to the original model. Full means finetuning the model using the complete training set.

with model-generated chain-of-thought solutions, following the setup summarized in Section 2.

4.2 Baselines

We compared TMAP with several widely used data selection techniques: (i) Random, which randomly samples subsets from the dataset. (ii) IFD, which uses GPT-2-computed IFD scores to select high-quality samples. (iii) Longest, which selects data with longest output length. (iv) Alpagasus, which prompts a LLM with a hand-craft template to score and select data. (v) DEITA, which selects data with highest quality, complexity, and diversity. (vi) Data Whisperer, which leverages ICL and attention scores to select data.

4.3 Implementation Details

We use four backbone models of different scales and families: Mistral-v0.2-7B-Instruct [Jiang *et al.*, 2023], Qwen2.5-7B-Instruct, Llama3-8B-Instruct and Llama2-13B-Chat. For GSM8K and ARC-C, we train the model using 30% samples from their corresponding training set, aligning with [Wang *et al.*, 2025a]. For GPQA and MUSR, we use 10% samples from DS2-50K dataset for training following the settings of prior work [Pang *et al.*, 2025]. We report the Exact Match (EM) accuracy on the test set for GSM8K and the other three tasks are reported with their multi-choice accuracy. All experiments were performed on 8 NVIDIA A100 GPUs. We utilized LoRA [Hu *et al.*, 2022] for model fine-tuning. During the Bayesian optimization process, we set the ICL evaluation parameters to $m = 5$ groups and $n = 3$ exemplars per group. We employ 5 random initialization steps followed by 200 optimization iterations. Further details can be found in supporting materials.

4.4 Main Results: Effectiveness of TMAP

We evaluate the effectiveness of TMAP through multiple settings, including cross-model validation (Exp 1), synthetic data distillation (Exp 2), and multi-model generalization (Exp 3).

Performance Across Models and Tasks
As shown in Table 1, TMAP consistently outperforms heuristic baselines across all models. Notably, on Qwen2.5-7B with GSM8K, it achieves comparable performance to baselines while using only 30% of training data. These results demonstrate that learning task-specific weights for multi-dimensional heuristics is more effective than relying on a single fixed metric or uniform weighting.

Validation on Synthetic Data
In Exp 2, we evaluate TMAP on distilled datasets where reasoning paths are augmented using **DeepSeek** [Liu *et al.*, 2025]. We expand our evaluation to include both the mathematical reasoning benchmark **GSM8K** and the science reasoning benchmark **ARC-C**. For this setting, we select 30% of the synthetic data pool for fine-tuning Llama2-13B and Mistral-v0.2-7B. As shown in Table 2, TMAP consistently outperforms other data selection baselines. For Llama2-13B on GSM8K, TMAP achieves a score of **52.77**, surpassing DEITA (51.78) and Alpagasus (49.28). More notably, on the ARC-C benchmark using Mistral-v0.2-7B, our method reaches **77.05**, which is virtually equivalent to the performance of fine-tuning on the full synthetic dataset (77.22), despite using only 30% of the training samples. This result highlights TMAP's capability to filter high-quality reasoning traces from DeepSeek-generated data, effectively identifying the "gold" within synthetic pools while discarding redundant or lower-quality generations.

Multi-Model Generalization
In Exp 3, we replace the single-model reward with a multi-model consensus signal obtained by averaging the ICL-based rewards from Mistral-v0.2-7B, Qwen2.5-7B, and Llama3-8B. As shown in Table 3, this consensus reward leads to slightly lower peak scores on each individual model compared to the

	Llama2-13B		Mistral-v0.2-7B	
Method	**GSM8K**	**ARC-C**	**GSM8K**	**ARC-C**
Base	35.48	60.58	38.59	56.65
Full Synthetic	**59.51**	**67.23**	**75.36**	**77.22**
Random	48.52	64.33	67.48	69.37
IFD	48.67	64.76	71.04	74.15
Longest	49.66	64.25	70.89	74.40
Alpagasus	49.28	64.51	67.40	76.45
DEITA	51.78	65.52	70.89	75.10
TMAP (Ours)	<u>52.77</u>	<u>66.13</u>	<u>71.34</u>	<u>77.05</u>

Table 2: Filtering performance on DeepSeek-distilled GSM8K and ARC-C datasets (30% selection). The best overall score in each column is highlighted in **bold**, while the best score among data selection methods (excluding Base and Full) is <u>underlined</u>. TMAP shows improvements over the heuristic baselines across both reasoning tasks in our experiments.

Reward Scheme	**Mistral-v0.2**	**Qwen2.5**	**Llama3**
Single-Model Opt. (S)	**55.42**	**79.00**	**65.96**
Multi-Model Opt. (M)	53.60	78.46	64.06
Delta (M - S)	*-1.82*	*-0.54*	*-1.90*

Table 3: Comparison between Single-Model (S) reward and Multi-Model (M) consensus reward. Due to space limitations, we omit the model size. While single-model optimization yields the highest scores on each model, multi-model rewards provide more stable cross-model behavior in our experiments.

single-model optimal configuration. However, the consensus weights remain competitive with strong baselines and, more importantly, capture a more robust data mixture that transfers well across architectures. This highlights the cross-model generalizability of TMAP: even when no single model is explicitly favored during optimization, the learned weights deliver stable gains across Mistral, Qwen2.5, and Llama3.

4.5 Ablation Study: Impact of Diversity

In Exp 4, we investigate the impact of incorporating diversity-based filtering (following the DEITA approach [Liu *et al.*, 2024]) after determining the optimal weights via TMAP. Contrary to the prevailing intuition that diversity is always beneficial, our results in Table 4 reveal a consistent performance decline across all tested models when explicit diversity constraints are enforced. Specifically, the accuracy for Mistral drops by 4.17%, while Llama3 and Qwen2.5 see reductions of 1.90% and 0.38%, respectively. This confirms our hypothesis that for reasoning-dense tasks, the priority should be logic-complexity rather than semantic surface variation. In such reasoning-dense settings, explicit diversity filtering may prune semantically similar yet logically distinct hard examples that are crucial for strengthening step-by-step mathematical reasoning.

4.6 Ablation Study: Optimal Granularity m

To investigate the impact of partitioning granularity on reward signal stability, we evaluate our method across various

Model	**TMAP**	**TMAP + Diversity**	**Delta**
Mistral-v0.2-7B	**55.42**	51.25	-4.17
Qwen2.5-7B	**79.00**	78.62	-0.38
Llama3-8B	**65.96**	64.06	-1.90

Table 4: Ablation study on diversity-enhanced selection across different models. Incorporating explicit diversity filtering (similarity-threshold 0.9) consistently leads to a performance drop in reasoning tasks.

Granularity (m)	**MUSR**
16	45.25
8	46.00
4	44.50
2	44.50
1	44.05

Table 5: Performance comparison across different granularities m on GPQA and MUSR. Results are accuracy (%).

$m \in \{1, 2, 4, 8, 16\}$ on the MUSR datasets using Qwen2.5-7B.

Analysis of Granularity m. As illustrated in Table 5, the choice of granularity m significantly influences reasoning performance, exhibiting an "inverted U-shape" trend. We observe that performance improves as m increases from 1 to 8, peaking at 46.00%, before declining at $m = 16$. This suggests that extreme granularities are suboptimal: overly coarse granularity (e.g., $m = 1$) fails to capture sufficient variance for robust ranking, while excessively fine granularity (e.g., $m = 16$) introduces noise that masks underlying weight patterns, thereby hindering precise optimization. Considering the trade-off between performance and computational efficiency, we adopt $m = 5$ in our main experiments. Notably, the superior performance observed at $m = 8$ implies that our reported results do not represent the absolute ceiling, indicating that TMAP possesses an even higher performance upper bound.

4.7 Efficiency Analysis

To assess the computational scalability of TMAP, we benchmark its overhead against three baselines: (1) **Full-set Evaluation**, utilizing the complete validation set for ICL reward calculation; (2) **Data Whisperer**, which computes scores by traversing the entire candidate dataset; (3) **Standard SFT**, representing the wall-clock time for a single fine-tuning epoch. To simulate resource-constrained environments, all benchmarks are conducted on two NVIDIA A100 GPUs using the Llama3-8B backbone.

As presented in Table 6, TMAP ($m = 5$) requires approximately **3 minutes** per optimization iteration. A defining advantage of TMAP is its scalability regarding candidate pool size. While Data Whisperer exhibits linear complexity, where runtime surges from 11 hours to over 72 hours as the pool expands from GSM8K (7.5k) to DS2-50K (50k), TMAP maintains a nearly constant optimization cost ($\approx$ 3.0 to 3.5 minutes per iteration) across both settings.This efficiency stems from

Method	Granularity	Cost / Iter.	Total Cost *(200 Iters)*
Standard SFT	-	66 min	220 h
Full-set Evaluation	$m = 5$	38 min	127 h
DW (GSM8K)	-	-	11 h
DW (DS2-50K)	-	-	> 72 h
TMAP (GSM8K)	$m = 5$	**3.0 min**	**10 h**
TMAP (DS2-50K)	$m = 5$	**3.5 min**	**12 h**

Table 6: Efficiency comparison on Llama3-8B. DW is the abbreviation for Data Whisperer. Unlike sample-level scoring methods, **TMAP maintains a stable time cost regardless of candidate dataset size** (GSM8K vs. DS2-50K), as its complexity depends solely on the size of the tiny validation set $\mathcal{V}_{\text{tiny}}$.

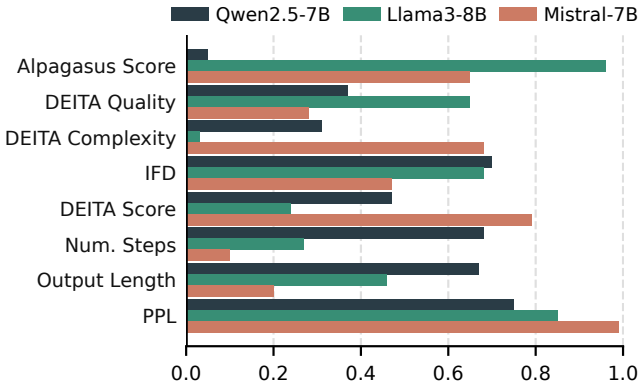

Figure 2: Column chart visualization of learned heuristic weights across Qwen2.5, Llama3, and Mistral on GSM8K.

the decoupling of the evaluation proxy from the size of the training set. TMAP's computational cost is determined solely by the inference complexity of the target task (specifically, generation length and the size of $\mathcal{V}_{\text{tiny}}$). By evaluating weight configurations via a compact proxy rather than scoring individual samples, TMAP achieves high-fidelity selection with significantly reduced overhead, facilitating scalability to massive instruction tuning datasets.

5 Weight Interpretation and Analysis

A core contribution of TMAP is discovering autonomous, model-specific weighting schemes. We analyze the converged weights for Qwen2.5, Llama3, and Mistral on GSM8K to understand their distinct definitions of "high-quality" data.

5.1 Model-Specific Heuristic Preferences

Table 7 and Figure 2 show that optimal weight configurations vary significantly, confirming that data quality is relative to the model's architecture rather than a static property.

Qwen2.5: Complexity and Length. Qwen2.5 assigns high weights to *IFD* (0.700), *Reasoning Steps* (0.679), and *Output Length* (0.674), while minimizing *Alpagasus Score* (0.050). As a strong reasoner, Qwen benefits most from structural complexity and hard data, relying less on external semantic judgments.

Indicator	Qwen2.5	Llama3	Mistral
Alpagasus Score [Chen *et al.*, 2024]	0.050	0.965	0.649
DEITA Complexity [Liu *et al.*, 2024]	0.314	0.029	0.675
DEITA Quality [Liu *et al.*, 2024]	0.371	0.652	0.281
DEITA Score (Joint) [Liu *et al.*, 2024]	0.468	0.237	0.787
IFD (PPL-based) [Li *et al.*, 2024a]	0.700	0.675	0.467
Number of Reasoning Steps [Jin *et al.*, 2024a]	0.679	0.271	0.101
Output Length [Zhao *et al.*, 2024]	0.674	0.458	0.199
Perplexity (PPL) [Jin *et al.*, 2024b]	0.752	0.854	0.991

Table 7: Learned optimal weights for multi-dimensional heuristics across different base models on GSM8K. Values are normalized to a $[0, 1]$ scale.

Llama3: Reliance on Strong Judges. Llama3 heavily weights *Alpagasus Score* (0.965) while ignoring *DEITA Complexity* (0.029). This indicates a priority for correctness and clear instruction-following over raw difficulty. TMAP steers selection toward authoritative samples to avoid introducing noise during alignment.

Mistral: High Information Density. Mistral prioritizes *Perplexity* (0.991) and *DEITA Score* (0.787) but ignores *Reasoning Steps* (0.101). Given the high-quality candidate pool, this preference for high perplexity suggests Mistral benefits from information density or surprisal, requiring data that diverges from its pre-training priors rather than predictable patterns.

5.2 The Necessity of Task-Adaptive Weighting

The fact that *DEITA Complexity* is prioritized by Mistral (0.675) but largely ignored by Llama3 (0.029) highlights a critical trade-off: a sample that is "appropriately difficult" for one model might be redundant or incorrectly scored for another. By bypassing manual rule-setting, TMAP identifies these hidden alignment points between data characteristics and model capabilities.

6 Conclusion

In this paper, we introduced TMAP, a joint task-model adaptation framework designed for instruction tuning data selection, which overcomes the limitations of static heuristic weighting and the high computation costs of existing data selection methods. By combining Bayesian Optimization with two efficient proxies In-Context Learning (ICL) and tinyBenchmark, TMAP iteratively searches optimal, multi-dimensional indicator weights tailored to specific target tasks and model. Experiments across various model families and benchmarks demonstrate that TMAP significantly outperforms static baselines with high computational efficiency. These results validate the necessity of adaptive, task-specific data selection strategies in instruction tuning. Future work will focus on extending TMAP to other instruction-tuning categories, such as writing or coding, as well as exploring more efficient proxy signals for continual model improvement.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62306344), Guangdong Basic and Applied Basic Research Foundation (2024A1515010253, 2026A1515011800), and the Open Research Fund of the State Key Laboratory of Blockchain and Data Security, Zhejiang University.

Contribution Statement

Jingze Song and Zihao Chen contributed equally to this work as co-first authors. They jointly developed the methodology, implemented the codebase, and conducted the experiments. Wenqing Chen supervised the research, provided methodological guidance, and served as the corresponding author. Zibin Zheng contributed to project administration and research support.

References

[Cai *et al.*, 2016] Li Cai, Kilchan Choi, Mark Hansen, and Lauren Harrell. Item response theory. *Annual Review of Statistics and Its Application*, 3(1):297–321, 2016.

[Cao *et al.*, 2024] Yihan Cao, Yanbin Kang, Lichao Sun, and Chi Wang. Instruction mining: Instruction data selection for tuning large language models. In *Proceedings of the First Conference on Language Modeling (COLM)*, 2024.

[Chen and Mueller, 2024] Jiuhai Chen and Jonas Mueller. Automated data curation for robust language model fine-tuning. *arXiv preprint arXiv:2403.12776*, 2024.

[Chen *et al.*, 2024] Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

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

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

[Diddee and Ippolito, 2025] Harshita Diddee and Daphne Ippolito. Chasing random: Instruction selection strategies fail to generalize. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1943–1957, 2025.

[Ge *et al.*, 2024] Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Mahong Xia, Zhang Li, Boxing Chen, Hao Yang, et al. 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, 2024.

[Hu *et al.*, 2022] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[Jiang *et al.*, 2023] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[Jin *et al.*, 2024a] Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. The impact of reasoning step length on large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1830–1842, 2024.

[Jin *et al.*, 2024b] Sigo Jin, Yanbing Wang, Shan Liu, Yue Zhang, and Wei Gu. Optimizing dataset creation: A general purpose data filtering system for training large language models. *Authorea Preprints*, 2024.

[Li *et al.*, 2024a] Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 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 14131–14151, 2024.

[Li *et al.*, 2024b] Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 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*, 2024.

[Li *et al.*, 2024c] Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Ling-Hao Chen, Junhao Liu, Tongliang Liu, et al. One-shot learning as instruction data prospector for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4586–4601, 2024.

[Liu *et al.*, 2024] Wei Liu, Weihao Zhu, Xingyuan Cheng, Yietong Shen, Kai Zhao, Qi Liu, Chaoqun Liang, Zeyu Hu, Wei Zeng, Ruifeng Liu, et al. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024.

[Liu *et al.*, 2025] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

[Lu *et al.*, 2024] Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The*

Twelfth International Conference on Learning Representations, 2024.

[Mirza *et al.*, 2025] Paramita Mirza, Lucas Weber, and Fabian Küch. Laser: Stratified selective sampling for instruction tuning with dedicated scoring strategy. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

[Močkus, 1974] Jonas Močkus. On bayesian methods for seeking the extremum. In *IFIP Technical Conference on Optimization Techniques*, pages 400–404. Springer, 1974.

[Pang *et al.*, 2025] Jinlong Pang, Na Di, Zhaowei Zhu, Jiaheng Wei, Hao Cheng, Chen Qian, and Yang Liu. Token cleaning: Fine-grained data selection for llm supervised fine-tuning. In *Forty-second International Conference on Machine Learning*, 2025.

[Perlitz *et al.*, 2024] Yotam Perlitz, Elron Bandel, Ariel Gera, Ofir Arviv, Liat Ein Dor, Eyal Shnarch, Noam Slonim, Michal Shmueli-Scheuer, and Leshem Choshen. Efficient benchmarking (of language models). 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 2519–2536, 2024.

[Polo *et al.*, 2024] Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinybenchmarks: evaluating llms with fewer examples. In *Proceedings of the 41st International Conference on Machine Learning*, pages 34303–34326, 2024.

[Si *et al.*, 2025] Shuzheng Si, Haozhe Zhao, Gang Chen, Cheng Gao, Yuzhuo Bai, Zhitong Wang, Kaikai An, Kangyang Luo, Chen Qian, Fanchao Qi, Baobao Chang, and Maosong Sun. Aligning large language models to follow instructions and hallucinate less via effective data filtering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

[Sprague *et al.*, 2023] Zayne Rea Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. In *The Twelfth International Conference on Learning Representations*, 2023.

[Tang *et al.*, 2025] Zinan Tang, Xin Gao, Qizhi Pei, Zhuoshi Pan, Mengzhang Cai, Jiang Wu, Conghui He, and Lijun Wu. Middo: Model-informed dynamic data optimization for enhanced llm fine-tuning via closed-loop learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

[Vivek *et al.*, 2024] Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. Anchor points: Benchmarking models with much fewer examples. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1576–1601, 2024.

[Wang *et al.*, 2024] Fei Wang, Ninareh Mehrabi, Palash Goyal, Rahul Gupta, Kai-Wei Chang, and Aram Galstyan. Data advisor: Dynamic data curation for safety alignment of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8227–8241, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[Wang *et al.*, 2025a] Shaobo Wang, Xiangqi Jin, Ziming Wang, Jize Wang, Jiajun Zhang, Kaixin Li, Zichen Wen, Zhong Li, Conghui He, Xuming Hu, and Linfeng Zhang. Data whisperer: Efficient data selection for task-specific LLM fine-tuning via few-shot in-context learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

[Wang *et al.*, 2025b] Shaobo Wang, Cong Wang, Wenjie Fu, Yue Min, Mingquan Feng, Isabel Guan, Xuming Hu, Conghui He, Cunxiang Wang, Kexin Yang, et al. Rethinking llm evaluation: Can we evaluate llms with 200x less data? *arXiv preprint arXiv:2510.10457*, 2025.

[Xia *et al.*, 2024] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 54104–54132, 2024.

[Ye *et al.*, 2023] Qinyuan Ye, Harvey Fu, Xiang Ren, and Robin Jia. How predictable are large language model capabilities? a case study on big-bench. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7493–7517, 2023.

[Yin *et al.*, 2024] Qingyu Yin, Xuzheng He, Chak Tou Leong, Fan Wang, Yanzhao Yan, Xiaoyu Shen, and Qiang Zhang. Deeper insights without updates: The power of in-context learning over fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.

[Zhang *et al.*, 2025] Ling Zhang, Xianliang Yang, Juwon Yu, Park Cheonyoung, Lei Song, and Jiang Bian. Holdout-loss-based data selection for llm finetuning via in-context learning. *arXiv preprint arXiv:2510.14459*, 2025.

[Zhao *et al.*, 2024] Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Long is more for alignment: a simple but tough-to-beat baseline for instruction fine-tuning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 60674–60703, 2024.

[Zhao *et al.*, 2025] Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Is in-context learning sufficient for instruction following in llms? In *The Thirteenth International Conference on Learning Representations*, 2025.

[Zhou *et al.*, 2023] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.