MoReC: Mode-aware Relevance-Gated Coresets for Targeted Instruction Tuning

Anonymous authors
Paper under double-blind review

Abstract

Instruction tuning is widely used to adapt LLMs to specialized tasks, yet its effectiveness is often constrained by the limited availability of high-quality target-specific data. To alleviate this limitation, existing data selection methods identify, from general instruction corpora, examples whose learning effects align with the target task. However, they often treat the target set as homogeneous and evaluate training examples independently, which may obscure fine-grained target capability requirements and produce redundant subsets. We propose **MoReC**, a **Mo**de-aware **Re**levance-gated **C**oreset selection framework. Built upon optimizer-aware gradient representations, MoReC first discovers latent capability modes in the target validation set and retrieves relevant training examples for each mode, constructing a candidate pool that preserves diverse target requirements. It then performs set-level selection through a facility-location objective that jointly optimizes target relevance and coverage. The overall objective is monotone submodular and admits an efficient greedy algorithm with a $(1 - 1/e)$ approximation guarantee. Experiments on benchmarks spanning knowledge, reasoning, and code generation show that, using only 5% of the instruction data, MoReC achieves competitive or state-of-the-art performance, while yielding subsets with lower intra-subset similarity. These results indicate that covering the target task's diverse capability modes is more effective than concentrating the budget on top-ranked examples.

1 Introduction

Instruction tuning has become a standard approach for adapting LLMs to follow human instructions and generalize across diverse tasks (Wei et al., 2021; Sanh et al., 2021; Longpre et al., 2023). In many practical applications, however, the goal is to enhance a specific domain or capability, such as reasoning or domain-specific knowledge, rather than improving all capabilities uniformly, motivating the setting of *targeted instruction tuning* (Peng et al., 2023; Xia et al., 2024). Since collecting large-scale, high-quality data for the target task is costly, a promising alternative is to identify useful training examples from existing general-purpose instruction corpora.

The effectiveness of instruction tuning depends critically on the composition of the training data. Large instruction datasets are commonly constructed from different sources, including human annotations and model-generated responses, and thus vary widely in quality. Prior studies have shown that a small amount of carefully curated data can achieve performance comparable to, or even better than, tuning on substantially larger datasets (Zhou et al., 2023; Chen et al., 2024; Li et al., 2024b). This phenomenon is especially important in targeted settings: an example may be intrinsically high-quality yet contribute little to a particular target capability, while irrelevant or redundant examples may dilute useful training signals. Consequently, targeted instruction tuning requires identifying not only generally good data, but data whose learning effects align with the target task.

Recent data selection methods typically estimate the utility of training examples by measuring their similarity to or influence on a held-out validation set (Xia et al., 2024; San Joaquin et al., 2024; Kwon et al., 2024). However, these methods often oversimplify both sides of the selection problem. On the target side, validation examples are commonly modeled as a homogeneous set without explicitly accounting for its latent structures, which may obscure the fine-grained capabilities required by the target task. On the training side, examples are typically chosen independently, ignoring their joint distribution and interactions within the selected subset (Zhao et al., 2024; 2025). As a result, point-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

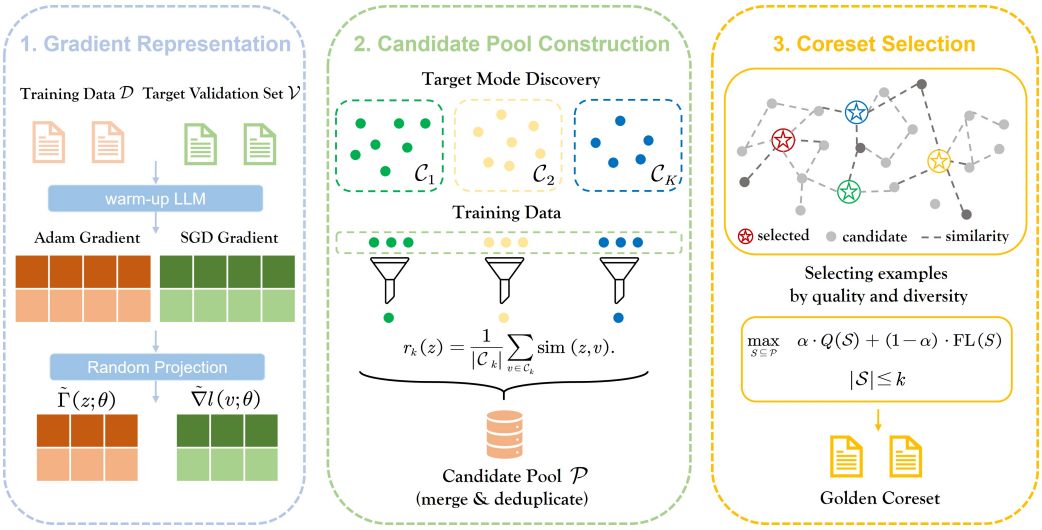


Figure 1: Overview of **MoReC**. Given an instruction training pool $\mathcal{D}$ and a target validation set $\mathcal{V}$, **(1) Gradient Representation** computes optimizer-aware gradient features via a warmup model and random projection, using Adam gradients for training examples and SGD gradients for validation examples. **(2) Candidate Pool Construction** discovers latent capability modes $\{\mathcal{C}_k\}$ from the gradients, and subsequently retrieves the most relevant training examples for each mode, which are then merged into a candidate pool $\mathcal{P}$. **(3) Coreset Selection** selects the final coreset from $\mathcal{P}$ by jointly optimizing quality and diversity.

wise ranking may repeatedly select similar examples from dominant regions, producing a redundant subset with limited coverage of full target capabilities.

To address these challenges, we propose **MoReC**, a **Mo**de-aware **Re**levance-gated **C**oreset selection framework for targeted instruction tuning. Built upon optimizer-aware gradient representations, MoReC discovers latent capability modes in the target validation set and constructs a mode-aware candidate pool that preserves the heterogeneous capability requirements. It then selects a coreset by jointly optimizing target relevance and coverage, thereby encouraging diversity among target-relevant examples. The resulting objective is monotone submodular and can be efficiently optimized by a greedy algorithm with a $(1 - 1/e)$ approximation guarantee.

We evaluate MoReC on multiple benchmarks spanning knowledge, reasoning, and code generation, using both Llama-3.1-8B and Mistral-7B as backbone models. Using only 5% of the instruction data, MoReC achieves competitive or state-of-the-art performance compared with strong baselines. Furthermore, the selected subsets exhibit lower intra-subset similarity, indicating reduced redundancy and broader coverage of the target capabilities. These results support our central claim: for targeted instruction tuning, covering the diverse capability modes is more effective than concentrating the selection budget on top-ranked examples.

2 Related Work

Target-agnostic data selection. A large body of work selects instruction data without reference to any specific downstream task, focusing on data quality or subset diversity. Quality-oriented methods score each example independently using signals such as perplexity, training loss, instruction-following difficulty, or judgments from stronger language and reward models (Li et al., 2024b;a; Liu et al., 2024; Bukharin et al., 2024). For instance, IFD (Li et al., 2024b) measures how much an instruction aids the generation of its response, and weak-to-strong methods approximate the rankings of larger models with smaller ones (Li et al., 2024a). Diversity-oriented methods instead reduce redundancy and preserve corpus coverage through clustering in semantic or tag spaces (Lu et al., 2024), information-gain maximization (Chen et al., 2025), or submodular objectives such as facility location (Lin & Bilmes, 2011; Wei et al., 2015). Closest to our formulation, TAGCOS (Zhang et al., 2025) clusters training gradients and applies submodular selection to build a representative

coreset. While effective at compressing a corpus, these methods optimize coverage of the *training distribution* and provide no mechanism to steer selection toward a particular target capability.

Target-aware data selection. Target-aware methods use a small validation or demonstration set to identify examples useful for a specific task. Retrieval-based approaches rank candidates by lexical or representation similarity, whereas model-aware approaches estimate example effects through gradients or influence functions. LESS (Xia et al., 2024) builds reusable low-dimensional gradient features and selects examples by their gradient similarity to the target set, and In2Core (San Joaquin et al., 2024) uses influence estimates to form a compact coreset. G2IS (Zhao et al., 2025) moves beyond independent scoring by extracting principal directions from validation gradients and traversing a gradient-based instruction graph. However, most target-aware methods score training examples independently and thus overlook set-level redundancy, while the set-level methods above are target-agnostic. MoReC bridges these two lines: it retains gradient-based target relevance but (i) replaces a single averaged target representation with *latent capability modes* discovered from validation gradients, ensuring that each target capability is represented in the candidate pool, and (ii) formulates final selection as a relevance-gated facility-location problem that optimizes target-conditioned coverage rather than independent ranking.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

Given a large-scale dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ consisting of instruction-response pairs and a small validation set $\mathcal{V}$ sampled from the target task, we aim to select a subset $\mathcal{S} \subseteq \mathcal{D}$ with $|\mathcal{S}| \leq k$ to maximize downstream performance.

3.2 GRADIENT-BASED DATA REPRESENTATION

During instruction tuning, gradients directly determine how each instruction-response pair updates the model parameters. Therefore, gradients provide a natural model-aware representation of data (Zhao et al., 2025): they not only reflect the semantic similarity between samples, but also encode how each sample contributes to the optimization dynamics of the model (Hammoudeh & Lowd, 2024; Choe et al., 2025). Compared with static embeddings, gradient representations are more suitable for data selection because they characterize the learning effect of each example with respect to the current model.

Prior work shows that gradient representations computed directly from the base model may be imprecise (Zhao et al., 2025). To mitigate this issue, we first conduct a warmup training stage following prior work (Xia et al., 2024). Specifically, we randomly sample a small subset $\mathcal{D}_{\text{warm}} \subset \mathcal{D}$, fine-tune the base model with LoRA (Hu et al., 2022), and save the intermediate checkpoints $\{\theta_t\}_{t=1}^{T}$ together with the corresponding Adam optimizer states. This procedure adapts the model to the instruction-tuning distribution and provides more reliable gradient representations for data selection.

For each training example $z \in \mathcal{D}$, we use the Adam update direction as its gradient representation. This is because instruction tuning is typically optimized with Adam (Kingma & Ba, 2014), and the raw gradient $\nabla \ell(z; \theta_t)$ does not fully reflect the actual parameter update direction. Given the first- and second-order moment estimates m_t and v_t from the warmup trajectory, the Adam gradient $\Gamma(z; \theta_t)$ is defined as

$$\theta_{t+1} - \theta_t = -\eta_t \Gamma(z; \theta_t) \tag{1}$$

$$\Gamma(z; \theta_t) = \frac{m_{t+1}}{\sqrt{v_{t+1}} + \epsilon}, \tag{2}$$

$$m_{t+1} = \frac{\beta_1 m_t + (1 - \beta_1) \nabla \ell(z; \theta_t)}{1 - \beta_1^t}, \tag{3}$$

$$v_{t+1} = \frac{\beta_2 v_t + (1 - \beta_2)(\nabla \ell(z; \theta_t))^2}{1 - \beta_2^t}. \tag{4}$$

Here $\ell(z;\theta_t)$ denotes the sequence-level loss of example z at checkpoint θ_t, and all operations are applied element-wise.

For each validation example $v \in \mathcal{V}$, we compute the standard first-order gradient $\nabla\ell(v;\theta_t)$. Intuitively, the validation gradient indicates the direction that reduces the target validation loss, while the Adam training gradient indicates how a candidate instruction would update the model during training. Therefore, the alignment between these two gradients provides an estimate of how useful a training example is for the target task.

Since storing gradient vectors for all training and validation examples is still expensive even under LoRA, we further apply random projection (Johnson & Lindenstrauss, 1984; Park et al., 2023) to obtain low-dimensional gradient features. We denote the projected Adam training gradient as $\tilde{\Gamma}(z;\theta_t)$ and the projected validation gradient as $\tilde{\nabla}\ell(v;\theta_t)$. For simplicity, we use g_z and h_v to denote the normalized aggregation of projected gradients over all warmup checkpoints for training example z and validation example v, respectively. The relevance between z and v is then measured by cosine similarity:

$$\mathrm{sim}(z,v) = \cos(g_z, h_v). \tag{5}$$

The resulting gradient features are used in the following sections for mode discovery, candidate retrieval, and coreset selection.

3.3 Mode-aware Candidate Pool Construction

The target validation set $\mathcal{V}$ is usually a small set sampled from the target task, serving as a proxy for the capabilities required by the task. Existing selection methods often treat validation examples as independent and identically distributed, using their average gradient as an overall representation of the target task (Xia et al., 2024; San Joaquin et al., 2024). However, a target task may not require a single homogeneous capability. For example, solving math problems may involve algebraic manipulation, geometric reasoning, and logical inference. Simply averaging validation gradients may obscure such fine-grained structures. To explicitly capture these heterogeneous capabilities, we first discover latent target modes and then retrieve candidates for each mode separately to construct a candidate pool.

Target mode discovery. Given the validation gradient representations $\mathcal{H} = \{h_v \mid v \in \mathcal{V}\}$, we perform K-means clustering over $\mathcal{H}$ and obtain K target modes, denoted as $\{\mathcal{C}_k\}_{k=1}^{K}$. Each mode $\mathcal{C}_k$ contains a group of validation examples with consistent gradient directions and is expected to correspond to a core capability required by the target task.

Mode-wise candidate retrieval. To ensure that each core capability has corresponding training examples in the selected subset, we construct a candidate pool with size at most βk, where k is the final selection budget and $\beta > 1$ controls the recall size. For each target mode $\mathcal{C}_k$, we compute the relevance score of a training example $z \in \mathcal{D}$ by averaging its similarity to all examples in this mode:

$$r_k(z) = \frac{1}{|\mathcal{C}_k|} \sum_{v \in \mathcal{C}_k} \mathrm{sim}(z,v). \tag{6}$$

Then, we retrieve the top-L_k most relevant examples for each capability mode:

$$\mathcal{P}_k = \mathrm{Top}_{L_k}\left(\{z \in \mathcal{D} \mid r_k(z)\}\right), \tag{7}$$

where $L_k = \lfloor \beta k/K \rfloor$. The final candidate pool is obtained by merging and deduplicating the retrieved examples from all modes:

$$\mathcal{P} = \bigcup_{k=1}^{K} \mathcal{P}_k \tag{8}$$

This mode-wise retrieval strategy reduces the search space for subsequent coreset selection while preserving examples aligned with different target capability modes.

3.4 CORESET SELECTION VIA FACILITY LOCATION

The mode-wise retrieval stage constructs a target-relevant candidate pool $\mathcal{P}$, but the retrieved examples may still contain substantial redundancy. Directly selecting the top-k examples by relevance may over-sample highly similar instructions and fail to cover diverse regions within the candidate pool. Therefore, we further perform set-wise coreset selection over $\mathcal{P}$, aiming to balance data quality and diversity.

For each candidate example $z_i \in \mathcal{P}$, we define its quality score as the average similarity to the validation set $\mathcal{V}$:

$$\bar{r}_i = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \mathrm{sim}(z_i, v). \tag{9}$$

This score measures how well a training example aligns with the target task and serves as the quality term in our coreset objective.

To model diversity, we use a Facility Location Function over the candidate pool $\mathcal{P}$ (Lin & Bilmes, 2011; Wei et al., 2015). Specifically, for a selected subset $\mathcal{S} \subseteq \mathcal{P}$, we define

$$\mathrm{FL}(\mathcal{S}) = \sum_{z_j \in \mathcal{P}} \max_{z_i \in \mathcal{S}} \mathrm{sim}(z_i, z_j). \tag{10}$$

This function measures how well the selected subset $\mathcal{S}$ covers the candidate pool. If a region has already been covered by selected examples, adding another similar example yields limited marginal gain. Therefore, this term naturally discourages redundancy and promotes diversity.

The final coreset objective combines quality and diversity:

$$F(\mathcal{S}) = \alpha Q(\mathcal{S}) + (1 - \alpha)\,\mathrm{FL}(\mathcal{S}), \quad |\mathcal{S}| \le k. \tag{11}$$

where $Q(\mathcal{S}) = \sum_{z_i \in \mathcal{S}} \bar{r}_i$, and $\alpha \in [0, 1]$ controls the trade-off between data quality and diversity.

We select the final coreset by maximizing $F(\mathcal{S})$ under the cardinality constraint:

$$\mathcal{S}^* = \arg \max_{\mathcal{S} \subseteq \mathcal{P}, |\mathcal{S}| \le k} F(\mathcal{S}). \tag{12}$$

Since this combinatorial optimization problem is NP-hard, we follow standard practice in submodular data selection and optimize it with a greedy algorithm (Lin & Bilmes, 2011; Wei et al., 2015), as shown in Algorithm 1.

Algorithm 1 Greedy Coreset Selection

Input: Candidate pool $\mathcal{P}$, budget k
Output: Selected subset $\mathcal{S}$

```
Initialize S ← ∅
while |S| < k do
    for each z ∈ P \ S do
        Δ(z | S) = F(S ∪ {z}) − F(S)
    end for
    z* ← arg max_{z∈P\S} Δ(z | S)
    S ← S ∪ {z*}
end while
return S
```

Since $Q(\mathcal{S})$ is a non-negative modular function and $\mathrm{FL}(\mathcal{S})$ is monotone submodular under a non-negative similarity kernel, their non-negative weighted sum $F(\mathcal{S})$ is also monotone submodular. We clamp negative cosine similarities to zero to satisfy this condition. Thus, by the classical greedy guarantee (Nemhauser et al., 1978), the greedy solution satisfies

$$F(\mathcal{S}_{\mathrm{greedy}}) \ge \left(1 - \frac{1}{e}\right) F(\mathcal{S}^*), \tag{13}$$

where $\mathcal{S}^*$ is the optimal subset. We provide the complete proof in Appendix A.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Training pool. We use OpenHermes (Teknium, 2023) as the candidate instruction pool $\mathcal{D}$. It contains approximately 242K primarily GPT-4-generated instruction–response pairs drawn from general instruction, role-playing, domain-specific, and code-oriented datasets.

Table 1: Performance comparison of data selection methods on Llama-3.1-8B and Mistral-7B across five benchmarks. All subset-selection methods use 5% of the instruction pool, while FULL uses the entire dataset. **Bold** and underlined values denote the best and second-best results for each model, respectively.

Method	MMLU	ARC-C	GSM8K	BBH	MBPP	Average
Llama-3.1-8B						
FULL	60.82	53.58	53.53	35.68	46.80	50.08
RANDOM	64.27	56.48	55.34	40.65	49.00	53.15
IFD	63.82	55.03	45.19	36.69	47.00	49.55
TAGCOS	64.40	56.74	54.06	39.73	52.80	53.55
LESS	64.49	56.66	56.03	39.46	49.20	53.16
G2IS	64.45	57.08	54.59	41.50	51.40	53.80
MOREC (ours)	**64.67**	**59.46**	**58.45**	**42.76**	**53.64**	**55.80**
Mistral-7B						
FULL	56.22	51.71	39.73	34.51	36.80	43.79
RANDOM	60.29	53.58	38.13	35.46	41.60	45.81
IFD	60.14	55.03	34.80	35.20	34.80	44.00
TAGCOS	**61.42**	55.20	39.95	36.85	40.20	46.73
LESS	61.34	55.38	41.02	35.02	41.20	46.79
G2IS	61.30	55.97	37.68	35.48	42.80	46.65
MOREC (ours)	61.28	**57.34**	**42.60**	**36.89**	**44.01**	**48.42**

Target tasks. We evaluate on five benchmarks spanning knowledge, mathematical reasoning, general reasoning, and code generation: MMLU (Hendrycks et al., 2020), ARC-Challenge (ARC-C) (Clark et al., 2018), GSM8K (Cobbe et al., 2021), Big-Bench Hard (BBH) (Suzgun et al., 2023), and MBPP (Austin et al., 2021). For each benchmark, we construct a small target query set $\mathcal{V}$ that is disjoint from the test examples used for evaluation. Details of query-set construction and evaluation protocols are provided in Appendix B.1.

Models and training. We evaluate Llama-3.1-8B (Grattafiori et al., 2024) and Mistral-7B-v0.1 (Jiang et al., 2023) as the base models. For each backbone, following LESS (Xia et al., 2024), we randomly sample 5,000 examples from $\mathcal{D}$ and perform LoRA (Hu et al., 2022) warmup training for four epochs to obtain gradient representations. The warmup stage is used only for data selection. Final models are initialized from their corresponding original base models and trained independently on the selected subsets. Full training and gradient-computation details are provided in Appendix B.2.

Baselines. We compare MoReC with FULL, target-agnostic selection methods RANDOM, IFD (Li et al., 2024b), and TAGCOS (Zhang et al., 2025), and target-aware methods LESS (Xia et al., 2024) and G2IS (Zhao et al., 2025). All subset-selection methods use the same 5% data budget and final fine-tuning configuration. For each backbone, TAGCOS, LESS, G2IS, and MoReC use the same warmup checkpoints and projected gradient datastore to isolate the effect of the selection strategies.

Implementation details. For MoReC, we use a candidate-pool multiplier of $\beta = 5$, and set the relevance–coverage coefficient to $\alpha = 0.6$. We evaluate all models using lm-evaluation-harness under the same task-specific decoding and prompting configurations across methods.

4.2 MAIN RESULTS

Table 1 reports the performance of all selection methods on five benchmarks across two base models. MoReC achieves the best overall performance on both Llama-3.1-8B and Mistral-7B with only 5% of the instruction pool. On Llama-3.1-8B, MoReC attains an average score of 55.80, outperforming full-data training by 5.72 points and all baselines, and ranks first on all five benchmarks. On Mistral-7B, it achieves an average score of 48.42, outperforming full-data training by 4.63 points and all baselines, and ranks first on four of five benchmarks while remaining competitive on MMLU, within 0.14 points of the best method. These consistent gains across two different base models suggest that the effectiveness of MoReC is not specific to a particular model backbone. Notably, MoReC with

Table 2: Relevance and redundancy of subsets selected by LESS and MoReC on Llama-3.1-8B. For each benchmark, we report the *mean relevance* (average cosine similarity between selected examples and the target validation set; higher is better) and the *intra-subset similarity* (average pairwise cosine similarity within the selected subset; lower indicates less redundancy). *Retention* denotes MoReC's mean relevance relative to LESS, and *Reduction* denotes the relative decrease in intra-subset similarity.

	Mean Relevance ↑			**Intra-Subset Similarity ↓**		
Benchmark	**LESS**	**MoReC**	**Retention (%)**	**LESS**	**MoReC**	**Reduction (%)**
MMLU	0.0316	0.0299	94.8	0.0431	0.0329	23.8
ARC-C	0.0227	0.0227	100.3	0.0550	0.0437	20.6
GSM8K	0.0449	0.0438	97.4	0.1496	0.1103	26.3
BBH	0.0437	0.0428	98.0	0.1440	0.1099	23.7
MBPP	0.0862	0.0859	99.7	0.1174	0.0837	28.7
Average	**0.0458**	**0.0450**	**98.3**	**0.1018**	**0.0761**	**25.3**

only 5% of the data substantially outperforms full-data training on both models, highlighting the importance of selective data curation for targeted instruction tuning.

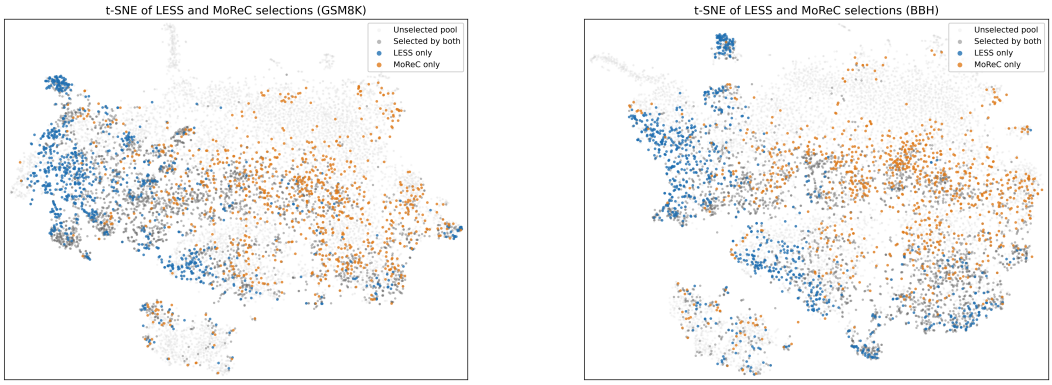
The comparison with TAGCOS, LESS, and G2IS provides further insight into the benefits of MoReC. TAGCOS constructs a task-agnostic coreset from the gradient space, whereas LESS prioritizes training examples according to their relevance to the target task. G2IS provides a stronger comparison by modeling dependencies among training examples through a gradient graph while extracting principal directions from the target gradients. Despite their different selection principles, these baselines achieve similar average performance on both models, and none consistently dominates across benchmarks. MoReC, by combining target-aware retrieval with set-level coverage, outperforms the baselines on nearly all model–benchmark combinations. These results suggest that target relevance and subset coverage provide complementary signals for targeted data selection, and that jointly accounting for both can yield more effective subsets. We further investigate this hypothesis through diversity analysis and controlled ablations in the following sections.

4.3 Diversity Analysis: Reduced Redundancy while Preserving Relevance

The main results establish that MoReC improves downstream performance, but they do not directly reveal *why*. Our central hypothesis is that the gain stems from better coverage of the target capability space rather than from higher per-example relevance. To test this, we characterize each selected subset along two complementary axes. *Mean relevance* measures the average cosine similarity between selected examples and the target validation set, quantifying how well the subset aligns with the target task. *Intra-subset similarity* measures the average pairwise cosine similarity among selected examples, serving as a proxy for redundancy: a subset that repeatedly picks near-duplicate instructions attains high intra-subset similarity and thus covers fewer distinct regions of the target space. An ideal targeted subset should keep mean relevance high while driving intra-subset similarity low; the two therefore constitute a trade-off that relevance-only selection cannot resolve.

As shown in Table 2, MoReC retains almost the entire target relevance of LESS (98.3% on average), indicating that mode-aware retrieval largely preserves target alignment. At the same time, MoReC reduces intra-subset similarity by 25.3%, showing substantially lower redundancy in the selected subset. This pattern is consistent across knowledge, reasoning, and code tasks, suggesting that the redundancy in relevance-only selection is systematic rather than task-specific, and that MoReC's coverage objective corrects it without sacrificing target alignment.

The largest redundancy reductions occur on MBPP (28.7%) and GSM8K (26.3%), where MoReC also yields clear downstream improvements over LESS in Table 1. Although these results do not establish a direct correlation between redundancy and downstream performance, they are consistent with our central hypothesis that allocating the selection budget across diverse target capability modes can be more effective than concentrating it on highly ranked but potentially redundant examples.



(a) GSM8K (b) BBH

Figure 2: t-SNE visualization of LESS and MoReC selections on GSM8K and BBH using Llama-3.1-8B gradient representations. Points are colored as LESS-only (blue), MoReC-only (orange), selected by both (gray), or unselected (light gray).

Table 3: Ablation of key components of MoReC on Llama-3.1-8B. "*w/o*" denotes removing the corresponding component. Best results are **bolded**.

Method	MMLU	ARC-C	GSM8K	BBH	MBPP	Average
w/o mode discovery	63.59	56.57	55.65	40.13	50.40	53.27
w/o set-level coverage	63.78	57.17	58.15	39.76	51.40	54.05
MoReC	**64.67**	**59.46**	**58.45**	**42.76**	**53.64**	**55.80**

Qualitative view of selection coverage. Figure 2 presents t-SNE visualizations of the subsets selected by LESS and MoReC. Under the same selection budget, LESS-only examples tend to concentrate in fewer regions, whereas MoReC-only examples are more broadly distributed on both GSM8K and BBH.

4.4 Ablation and Analysis

Component ablation. We ablate the two key components of MoReC to assess their individual contributions. For *w/o mode discovery*, we collapse the target validation set into a single global mode by setting $K = 1$, while retaining relevance-gated candidate retrieval and the subsequent set-level selection. For *w/o set-level coverage*, we preserve mode discovery and candidate retrieval but remove the coverage term from the selection objective by setting $\alpha = 1$, resulting in relevance-only selection within the candidate pool. Other settings are kept fixed.

As shown in Table 3, removing either component degrades the overall performance across all reported benchmarks. Compared with the full method, removing mode discovery decreases the average score by 2.53 points, while removing set-level coverage leads to a 1.75-point drop. The larger degradation from removing mode discovery suggests that representing the target validation set with multiple latent modes is important for preserving heterogeneous target requirements during candidate retrieval. Meanwhile, the consistent deterioration without set-level coverage shows that relevance alone is insufficient for constructing an effective subset, supporting the need to explicitly reduce redundancy among selected examples. Together, these results indicate that target mode discovery and set-level coverage make complementary contributions to MoReC.

Qualitative analysis of target modes. Figure 3a illustrates the correspondence between ARC-C target examples from different modes and training examples retrieved from OpenHermes. In the first pair, the connection lies in a shared causal mechanism: reduced biological CO_2 uptake contributes to greenhouse-gas accumulation. In the second pair, both examples require identifying a biological structure through a concise answer, although they concern different structures. These cases illustrate

Mode 1 Environmental causal reasoning

Target validation example
Question: Phytoplankton are plant-like microscopic organisms that float in bodies of water such as the ocean. They take in carbon dioxide during the process of photosynthesis. What would most likely happen if the population of phytoplankton was destroyed?
Reference answer: Greenhouse gases would increase.

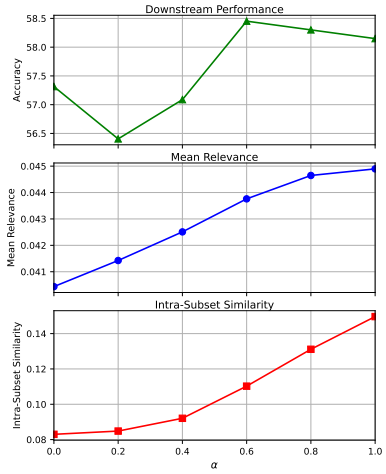
Retrieved training example
Instruction: Explain how **deforestation affects global climate**.
Response: Deforestation releases stored carbon and reduces CO_2 absorption, contributing to global warming and disruptions of the Earth's natural cycles.

Mode 2 Concise scientific concept identification

Target validation example
Question: Which of the following is found in a **eukaryotic cell but not in a prokaryotic cell**?
Reference answer: a nucleus

Retrieved training example
Instruction: The process of plants converting sunlight into energy through **photosynthesis** primarily occurs in which part of the plant?
Response: Leaves

(a) Target modes and retrieved examples.



(b) Sensitivity to α.

Figure 3: **Further analysis of MoReC. (a)** Representative ARC-C validation examples from two target modes and corresponding training examples retrieved from OpenHermes. The longer training response is summarized for brevity. **(b)** Effect of the relevance–coverage coefficient α on downstream performance, target relevance, and subset redundancy on GSM8K.

how target requirements can differ in both underlying knowledge and response form, and how mode-specific retrieval can identify training examples related to these local requirements.

Sensitivity to the relevance–coverage coefficient α. Figure 3b examines how α controls the balance between target relevance and subset coverage on GSM8K, with all other settings fixed. As α increases from 0 to 1, mean relevance increases monotonically from 0.0404 to 0.0449, while intra-subset similarity rises from 0.0830 to 0.1496. Thus, assigning more weight to relevance consistently selects examples that are better aligned with the target task, but also produces increasingly redundant subsets. This trend confirms the intended relevance–coverage trade-off in our objective: the two terms favor complementary properties of the selected subset rather than optimizing the same selection criterion.

Downstream performance exhibits a non-monotonic trend with respect to α, reaching the highest accuracy of 58.45 at the intermediate value $\alpha = 0.6$. In particular, compared with the relevance-only objective at $\alpha = 1$, $\alpha = 0.6$ retains 97.6% of its mean relevance while reducing intra-subset similarity by 26.3%, and improves accuracy from 58.15 to 58.45. Conversely, emphasizing coverage alone at $\alpha = 0$ yields the lowest intra-subset similarity but does not achieve the best downstream performance. These results suggest that neither maximizing relevance nor minimizing redundancy alone is sufficient; balancing the two leads to a more effective subset for targeted instruction tuning.

5 Conclusion

We propose **MoReC**, a mode-aware relevance-gated coreset selection framework for targeted instruction tuning. MoReC discovers latent capability modes in the target tasks and performs set-level selection with a facility-location objective, yielding a monotone submodular objective with an efficient greedy solution and a $(1 - 1/e)$ approximation guarantee. Experiments across knowledge, reasoning, and code generation tasks show that MoReC achieves strong performance on both Llama-3.1-8B and Mistral-7B using only 5% of the instruction data, while substantially reducing subset redundancy without sacrificing target relevance. Overall, our results demonstrate that effective targeted instruction tuning requires not only identifying relevant examples, but also allocating the limited training budget to cover the diverse capabilities required by the target task.

AI USE STATEMENT

In this work, we utilized AI tools to assist in improving the grammar, clarity, and overall readability of this manuscript. We have reviewed all AI-assisted work. The core ideas and scientific claims were conceived and articulated by the human authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Theoretical properties and the greedy approximation guarantee are proved in Appendices A.1 and A.2, respectively. Detailed experimental settings are provided in Appendix B, including target query sets and evaluation protocols (Appendix B.1), training and gradient computation (Appendix B.2), and baseline and selection configurations (Appendix B.3).

REFERENCES

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

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

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. In *International Conference on Learning Representations*, volume 2024, pp. 34767–34797, 2024.

Yicheng Chen, Yining Li, Kai Hu, Ma Zerun, Haochen Ye, and Kai Chen. MIG: Automatic data selection for instruction tuning by maximizing information gain in semantic space. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 9902–9915, 2025.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff G. Schneider, Eduard H. Hovy, Roger Baker Grosse, and Eric P. Xing. What is your data worth to GPT? LLM-scale data valuation with influence functions. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

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

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

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

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5):2351–2403, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

William B. Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206, 1984.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Y Zou. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 21921–21942, 2024.

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)*, pp. 14255–14273, 2024a.

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: Human Language Technologies (Volume 1: Long Papers)*, pp. 7602–7635, 2024b.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pp. 510–520, 2011.

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

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

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 *International Conference on Learning Representations*, volume 2024, pp. 36456–36474, 2024.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*, 2023.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

Ayrton San Joaquin, Bin Wang, Zhengyuan Liu, Nicholas Asher, Brian Lim, Philippe Muller, and Nancy Chen. In2core: Leveraging influence functions for coreset selection in instruction finetuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10324–10335, 2024.

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

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

Teknium. OpenHermes. `https://huggingface.co/datasets/teknium/openhermes`, 2023. Hugging Face dataset, accessed: 2026-07-13.

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

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International conference on machine learning*, pp. 1954–1963. PMLR, 2015.

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

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

Hanyu Zhao, Li Du, Yiming Ju, Chengwei Wu, and Tengfei Pan. Beyond iid: Optimizing instruction learning from the perspective of instruction interaction and dependency. *arXiv preprint arXiv:2409.07045*, 2024.

Yang Zhao, Li Du, Xiao Ding, Yangou Ouyang, Hepeng Wang, Kai Xiong, Jinglong Gao, Zhouhao Sun, Dongliang Xu, Qing Yang, et al. Beyond similarity: A gradient-based graph method for instruction tuning data selection. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24391–24404, 2025.

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.

A Theoretical Analysis

A.1 Monotonicity and Submodularity of the MoReC Objective

Recall the MoReC objective

$$F(\mathcal{S}) = \alpha Q(\mathcal{S}) + (1-\alpha)\,\mathrm{FL}(\mathcal{S}), \tag{14}$$

where

$$Q(\mathcal{S}) = \sum_{z_i \in \mathcal{S}} \bar{r}_i, \tag{15}$$

and

$$\mathrm{FL}(\mathcal{S}) = \sum_{z_j \in \mathcal{P}} \max_{z_i \in \mathcal{S}} \mathrm{sim}(z_i, z_j). \tag{16}$$

We define the maximum over the empty set as zero. Negative cosine similarities are clamped to zero, so $\bar{r}_i \geq 0$ and $\mathrm{sim}(z_i, z_j) \geq 0$.

Quality term. Since $Q(\mathcal{S})$ is a sum of fixed non-negative per-example scores, it is a non-negative modular function. For any $\mathcal{A} \subseteq \mathcal{B}$ and $z \notin \mathcal{B}$,

$$Q(\mathcal{A} \cup \{z\}) - Q(\mathcal{A}) = Q(\mathcal{B} \cup \{z\}) - Q(\mathcal{B}) = \bar{r}_z \geq 0. \tag{17}$$

Therefore, Q is both monotone and submodular.

Facility-location term. For each $z_j \in \mathcal{P}$, define

$$m_j(\mathcal{S}) = \max_{z_i \in \mathcal{S}} \mathrm{sim}(z_i, z_j). \tag{18}$$

The marginal gain of adding z to $\mathcal{S}$ is

$$\Delta_{\mathrm{FL}}(z \mid \mathcal{S}) = \mathrm{FL}(\mathcal{S} \cup \{z\}) - \mathrm{FL}(\mathcal{S}) \tag{19}$$

$$= \sum_{z_j \in \mathcal{P}} \max\left\{0,\, \mathrm{sim}(z, z_j) - m_j(\mathcal{S})\right\}. \tag{20}$$

For any $\mathcal{A} \subseteq \mathcal{B}$, we have $m_j(\mathcal{A}) \leq m_j(\mathcal{B})$ for every z_j. Hence,

$$\Delta_{\mathrm{FL}}(z \mid \mathcal{A}) \geq \Delta_{\mathrm{FL}}(z \mid \mathcal{B}), \tag{21}$$

which establishes the diminishing-returns property. Moreover, every marginal gain is non-negative, so FL is monotone.

Since $\alpha \in [0,1]$, $F(\mathcal{S})$ is a non-negative weighted sum of a monotone modular function and a monotone submodular function. Therefore, F is monotone submodular. In addition,

$$F(\emptyset) = 0, \tag{22}$$

so F is normalized.

A.2 Greedy Approximation Guarantee

Let $\mathcal{S}^*$ be an optimal subset under the cardinality constraint $|\mathcal{S}| \leq k$, and let

$$\mathcal{S}_0 = \emptyset, \mathcal{S}_1, \ldots, \mathcal{S}_k$$

be the sequence produced by the greedy algorithm, where

$$z_{t+1} \in \arg\max_{z \in \mathcal{P} \setminus \mathcal{S}_t} \Delta(z \mid \mathcal{S}_t) \tag{23}$$

and $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{z_{t+1}\}$.

By monotonicity and submodularity,

$$F(\mathcal{S}^*) - F(\mathcal{S}_t) \leq F(\mathcal{S}^* \cup \mathcal{S}_t) - F(\mathcal{S}_t) \tag{24}$$

$$\leq \sum_{z \in \mathcal{S}^* \setminus \mathcal{S}_t} \Delta(z \mid \mathcal{S}_t). \tag{25}$$

Since $|\mathcal{S}^*| \le k$, there exists some $z \in \mathcal{S}^* \setminus \mathcal{S}_t$ such that

$$\Delta(z \mid \mathcal{S}_t) \ge \frac{F(\mathcal{S}^*) - F(\mathcal{S}_t)}{k}. \tag{26}$$

By the greedy choice,

$$F(\mathcal{S}_{t+1}) - F(\mathcal{S}_t) \ge \frac{F(\mathcal{S}^*) - F(\mathcal{S}_t)}{k}. \tag{27}$$

Defining the residual

$$R_t = F(\mathcal{S}^*) - F(\mathcal{S}_t), \tag{28}$$

we obtain

$$R_{t+1} \le \left(1 - \frac{1}{k}\right) R_t. \tag{29}$$

Since $F(\emptyset) = 0$, we have $R_0 = F(\mathcal{S}^*)$, and therefore

$$R_k \le \left(1 - \frac{1}{k}\right)^k F(\mathcal{S}^*) \le \frac{1}{e} F(\mathcal{S}^*). \tag{30}$$

Hence,

$$F(\mathcal{S}_{\text{greedy}}) \ge \left(1 - \frac{1}{e}\right) F(\mathcal{S}^*). \tag{31}$$

Thus, greedy coreset selection achieves the classical $(1 - 1/e)$ approximation guarantee under the cardinality constraint.

B Additional Experimental Details

B.1 Target Query Sets and Evaluation Protocol

Table 4: Summary of the target benchmarks, evaluation metrics, prompting settings, and target query sets.

Dataset	Aspect	Metric	Prompting	$\lvert\mathcal{V}\rvert$
MMLU	Multi Task Understanding	Accuracy	5-shot	285
ARC-C	Commonsense Knowledge	Accuracy	0-shot	299
GSM8K	Math Reasoning	Exact Match	5-shot	300
BBH	Complex Reasoning	Exact Match	0-shot	81
MBPP	Code Generation	pass@1	3-shot	464

Table 4 summarizes the target benchmarks, evaluation metrics, prompting settings, and query-set sizes. For MMLU, we use five development examples from each of its 57 subjects. For ARC-C, we use the official validation split. For GSM8K, we randomly sample 300 examples from the training split. For BBH, we use 81 development examples covering its constituent tasks. For MBPP, we combine the training and validation splits. All query examples are disjoint from the test examples used for downstream evaluation.

The overall score reported in the main results is the unweighted average across the five benchmarks. All models are evaluated using lm-evaluation-harness under the same task-specific decoding and prompting configurations across methods.

B.2 Training and Gradient Computation

For each backbone, we randomly sample 5,000 examples from OpenHermes and perform LoRA warmup training for four epochs following LESS (Xia et al., 2024). We save the model checkpoint and Adam optimizer states after each epoch. The warmup stage is used only to construct gradient representations for data selection and is not used to initialize the final instruction-tuned models.

For training examples, we use the Adam update direction as the optimizer-aware gradient representation, while standard first-order gradients are computed for target query examples, as described

in Section 3. To reduce storage and computation costs, the resulting LoRA gradient features are projected to 4,096 dimensions using random projection and normalized before computing cosine similarity.

For final instruction tuning, each model is initialized from its corresponding original base model and fine-tuned on the selected subset for three epochs. We use a learning rate of 2×10^{-4}, an effective batch size of 32, a LoRA rank of $r = 16$, and a maximum sequence length of 2,048 tokens. The same final fine-tuning configuration is used across all subset-selection methods.

B.3 Baseline and Selection Details

We compare MoReC against both target-agnostic and target-aware data selection methods. FULL fine-tunes the model on the entire OpenHermes training pool, while RANDOM uniformly samples 5% of the pool. IFD (Li et al., 2024b) scores examples according to instruction-following difficulty and retains high-quality examples. TAGCOS (Zhang et al., 2025) clusters training examples and performs coreset selection within the resulting clusters. LESS (Xia et al., 2024) ranks training examples according to their gradient similarity to the target query set. G2IS (Zhao et al., 2025) applies PCA to the target gradients to extract principal directions and selects training examples through a gradient-graph walk subject to conflict and consistency constraints.

Unless otherwise specified, all subset-based methods select 5% of OpenHermes, corresponding to approximately 12.1K examples. For each backbone, TAGCOS, LESS, G2IS, and MoReC use the same warmup checkpoints and projected gradient datastore, so their differences arise from the subsequent selection strategies rather than from gradient computation.

For MoReC, we fix the number of target modes to $K = 8$ across all benchmarks and backbone models without task-specific tuning. We use a candidate-pool multiplier $\beta = 5$, resulting in a candidate pool of at most five times the final selection budget before merging and deduplication. We set the relevance–coverage coefficient to $\alpha = 0.6$ in the main experiments. Negative cosine similarities are clamped to zero, consistent with the non-negative similarity kernel used in our facility-location objective.