InstructDiff: Domain-Adaptive Data Selection via Contrastive Entropy for Efficient LLM Fine-Tuning

Junyou Su[1*], **He Zhu**[1*†], **Xiao Luo**[2], **Liyu Zhang**[1], **Hong-Yu Zhou**[3]
Yun Chen[4], **Peng Li**[3], **Yang Liu**[3], **Guanhua Chen**[5†]
[1]Peking University, [2]University of Wisconsin-Madison
[3]Tsinghua University,[4]Shanghai University of Finance and Economics
[5]Southern University of Science and Technology
{jysu25, zhuhe}@stu.pku.edu.cn, chengh3@sustech.edu.cn

Abstract

Supervised fine-tuning (SFT) is fundamental to adapting large language models, yet training on complete datasets incurs prohibitive costs with diminishing returns. Existing data selection methods suffer from severe domain specificity: techniques optimized for general instruction-following fail on reasoning tasks, and vice versa. We observe that measuring contrastive entropy between base models and minimally instruction-tuned calibrated models reveals a pattern—samples with the *lowest contrastive entropy* consistently yield optimal performance across domains, yet this principle manifests domain-adaptively: reasoning tasks favor entropy *increase* (cognitive expansion), while general tasks favor entropy *decrease* (cognitive compression). We introduce InstructDiff, a unified framework that operationalizes contrastive entropy as a domain-adaptive selection criterion through warmup calibration, bi-directional NLL filtering, and entropy-based ranking. Extensive experiments show that InstructDiff achieves 17% relative improvement over full data training on mathematical reasoning and 52% for general instruction-following, outperforming prior baselines while using only 10% of the data.

1 Introduction

Recent advances in large language models have demonstrated remarkable capabilities, achieving unprecedented performance on complex reasoning and language understanding tasks (OpenAI, 2025; Guo et al., 2025; Anthropic, 2025). Post-training has emerged as the predominant paradigm for unlocking these capabilities (Ouyang et al., 2022; Dubey et al., 2024), with supervised fine-tuning (SFT) serving as the cornerstone stage that shapes model behavior through instruction-response pairs. As post-training datasets continue to expand, often reaching millions of examples (Xu et al., 2024), a critical challenge arises: training on entire datasets incurs prohibitive computational costs while delivering diminishing returns. Recent evidence demonstrates that *data quality fundamentally dictates both training efficiency and model performance* (Zhou et al., 2023; Ye et al., 2025; Xiao et al., 2025). Carefully curated subsets can achieve comparable or superior results with dramatically fewer examples, reducing training time by orders of magnitude. This observation establishes *data curation*, the systematic selection of high-quality training examples, as an essential strategy for efficient and effective model adaptation.

Current data selection methods exhibit severe domain specificity, partitioning into siloed techniques that struggle to generalize across task structures. Existing approaches broadly fall into two categories: methods for **general tasks** that prioritize conversational quality through heuristics such as influence functions, quality metrics, or uncertainty (Xia et al., 2024a; Li et al., 2024b; Liu et al., 2024), and methods for **reasoning tasks** that prioritize verifiable correctness through difficulty-based filtering, Best-of-N sampling, or pass@k optimization (Lightman et al., 2023; Liu et al., 2025; Lyu et al., 2025; Walder and Karkhanis, 2025). While these domain-specific strategies have demonstrated effectiveness within their respective domains, their applicability across varied tasks remains somewhat constrained. This raises a fundamental question: *Is it possible to devise a unified, domain-adaptive method or metric for data selection that automatically adjusts to the characteristics of different tasks and domains?*

To address this challenge, we re-examine fine-tuning through an information-theoretic lens and uncover an empirical pattern: when comparing the

[*] Equal Contribution.
[†] Corresponding Authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10630–10648

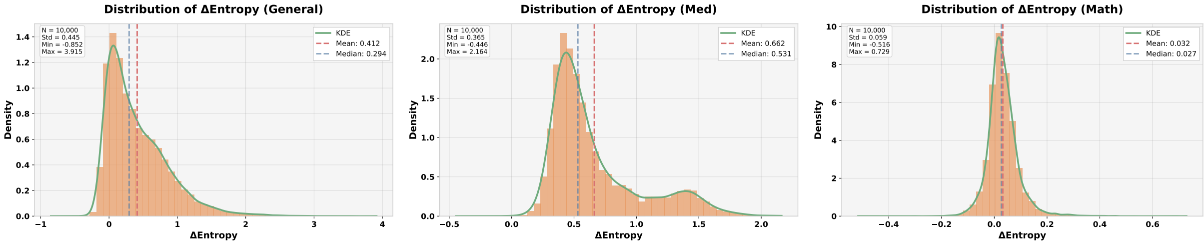

Figure 1: **Domain-adaptive entropy dynamics reveal distinct learning patterns.** We visualize the contrastive entropy ($\Delta H = H_{\text{base}} - H_{\text{inst}}$) distributions across three domains. For **general instruction-following** and **medical QA**, almost all samples show $H_{\text{base}} > H_{\text{inst}}$, i.e., instruction-tuning compresses uncertainty (**entropy decrease**, cognitive compression). In contrast, for **mathematical reasoning**, about half of the samples show $H_{\text{inst}} > H_{\text{base}}$, i.e., instruction-tuning *increases* entropy (cognitive expansion). This contrast supports our unified but domain-adaptive selection principle based on contrastive entropy (ΔH).

base model with a *calibrated model* (obtained by fine-tuning on a small random subset), we find that selecting data with the *lowest contrastive entropy* consistently yields optimal results across domains (see Figure 3 for systematic range analysis). However, how this *Lowest Contrastive Entropy* principle appears depends fundamentally on the task structure (Figure 1). For **reasoning tasks**, the lowest contrastive entropy corresponds to *negative* values, meaning that calibration *increases* entropy. We term this pattern **cognitive expansion** and it reflects that effective reasoning requires preserving solution path diversity. In contrast, for **general tasks** such as instruction-following or medical tasks, the lowest contrastive entropy corresponds to *small positive* values, where calibration *decreases* entropy. This is called **cognitive compression** and fine-tuning in this case concentrates diffuse priors onto canonical behaviors. Despite these differences, both regimes follow the same principle: selecting samples with the *lowest* ΔH. For reasoning, the lowest ΔH manifests as entropy expansion (negative values), while for general tasks it appears as compression (small positive values).

Building on this insight, we introduce **InstructDiff**, a model-aware data selection framework that leverages model-state difference metrics as a unified, domain-adaptive criterion for supervised fine-tuning (SFT), with contrastive entropy (ΔH) as the core selection signal. InstructDiff is designed as a two-stage process. In the first stage, *Warmup Calibration*, we fine-tune the base model on a small random subset to obtain a calibration model that serves as an instruction-tuned reference. In the second stage, *Distribution-Aware Selection*, we compute, for each sample, both the negative log-likelihood difference (ΔNLL) and contrastive entropy (ΔH) between the base and calibration models. We apply bi-directional NLL filtering to remove both redundant and incomprehensible samples, then rank the remaining data by ΔH and select those with the lowest values from the learnable range. This approach allows a single criterion to naturally adapt. For reasoning tasks, the method favors samples with entropy increase (cognitive expansion), and for general instruction-following, it favors entropy decrease (cognitive compression). The entire procedure requires no explicit domain heuristics. Furthermore, we support an efficient *iterative selection* strategy. After training on the initially selected data, the workflow can be repeated with the updated model to achieve further performance gains. Extensive experiments show that InstructDiff achieves 17% relative improvement over full data training on mathematical reasoning and 52% for general instruction-following, outperforming prior baselines while using only 10% of the data.[1]

2 Related Work

2.1 Data Selection for Supervised Fine-Tuning

Efficient data selection is central to LLM adaptation, but most existing approaches, whether task-specific or model-agnostic, struggle to generalize across domains (Wettig et al., 2024). Early studies (Zhou et al., 2023; Ye et al., 2025) demonstrated that carefully chosen small subsets, sometimes as few as 1,000 examples, can match or surpass full-data fine-tuning, underscoring the value

[1]Our code is available at `https://github.com/zhuchichi56/Instruct-diff`.

of data quality over sheer quantity. Representative methods include **LLM-as-judge** scoring (Chen et al., 2023; Liu et al., 2023), which leverages advanced models to rate data but is costly and decoupled from the specific target model. Other approaches are **difficulty- or uncertainty-based** (Li et al., 2024b,a; Zhu et al., 2025a; Lu et al., 2023), using single-model metrics like loss, perplexity, or instruction complexity based on tag counts; however, these often struggle to adapt across models or domains. **Gradient/influence-based** techniques (Xia et al., 2024a; Li et al., 2024c, 2023a) estimate training influence but require significant computation. Heuristic strategies are also common, such as ranking by length (Zhu et al., 2024; Xia et al., 2024b). To address these challenges, we propose quantifying *distributional change*, specifically the negative log-likelihood and contrastive entropy (ΔNLL, ΔH) between the base model and a lightly instruction-tuned reference, as a unified and adaptive data selection criterion that is both scalable and robust across diverse SFT scenarios.

2.2 Entropy in Language Model Training

Entropy quantifies output uncertainty in LLMs and plays dual roles. ZIP (Yin et al., 2024) exploits an entropy-based compression ratio to select low-redundancy, high-diversity data, improving training efficiency. In RL, entropy regularization promotes exploration, and recent work exploits semantic- and token-level entropy to improve reasoning LLMs via adaptive entropy bonuses or high-entropy decision weighting (Vanlioglu, 2025; Wang et al., 2025). In inference, entropy has been used to characterize uncertainty at the sequence level and to detect hallucinated outputs through semantic uncertainty estimation (Farquhar et al., 2024; Zhu et al., 2025b). Critically, prior work examines entropy within a *single model state*. Our work pioneers **contrastive entropy** (ΔH) across model states, revealing domain-specific patterns: negative ΔH (entropy increase) for reasoning versus positive ΔH (decrease) for general tasks. This shift from absolute entropy to *cross-state contrastive entropy dynamics* enables automatic domain adaptation.

3 Methodology

In this section, we formalize the data selection problem and describe our two-stage selection framework with iterative refinement. Figure 2 illustrates the complete pipeline.

3.1 Problem Formulation and Motivation

Formulation. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a supervised fine-tuning dataset, where x_i represents the instruction and y_i the target response. Given a pre-trained base model π_{base} and computational budget $k \ll N$, our objective is to select a subset $\mathcal{D}' \subset \mathcal{D}$ with $|\mathcal{D}'| = k$ that maximizes downstream performance while mitigating emergent misalignment (Betley et al., 2025).

Motivation. Our empirical investigation (Figure 1) reveals that comparing π_{base} with a minimally instruction-tuned model π_{inst} uncovers domain-adaptive entropy patterns. For reasoning tasks, optimal samples exhibit entropy *increase* (cognitive expansion); for general tasks, entropy *decrease* (cognitive compression). Critically, both regimes converge on selecting samples with the *lowest* contrastive entropy ΔH. This observation motivates operationalizing ΔH between π_{base} and π_{inst} as a unified, domain-adaptive selection criterion.

3.2 Model-State Difference Metrics

To operationalize our selection principle, we define two complementary metrics that capture distributional changes between π_{base} and π_{inst}. For each sample (x_i, y_i), the negative log-likelihood (NLL) difference quantifies learning signal strength:

$$\Delta\text{NLL}_i = \mathcal{L}_{\text{inst}}(x_i, y_i) - \mathcal{L}_{\text{base}}(x_i, y_i) \quad (1)$$

where $\mathcal{L}_\pi(x, y) = -\frac{1}{|y|}\sum_{t=1}^{|y|} \log \pi_\theta(y_t \mid x, y_{<t})$ denotes length-normalized NLL. The contrastive entropy reveals the mode of learning:

$$\Delta H_i = H_{\text{base}}(x_i, y_i) - H_{\text{inst}}(x_i, y_i) \quad (2)$$

where per-token entropy is:

$$H_\pi(x, y) = -\frac{1}{|y|}\sum_{t=1}^{|y|}\sum_{v\in\mathcal{V}} \pi_\theta(v \mid x, y_{<t}) \log \pi_\theta(v \mid x, y_{<t}) \quad (3)$$

Positive ΔH indicates uncertainty compression during fine-tuning, while negative ΔH suggests solution diversity expansion. Together, ΔNLL identifies the learnable range, and ΔH enables domain-adaptive selection within that range.

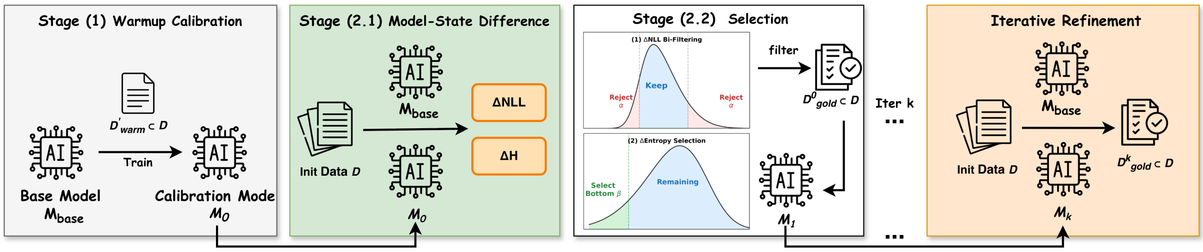

Figure 2: **The `InstructDiff` two-stage selection pipeline. Stage 1: Warmup Calibration.** Randomly sample a small warmup subset to lightly instruction-tune the base model, producing a calibration model as reference. **Stage 2: Distribution-Aware Selection.** For each candidate sample, compute negative log-likelihood difference (ΔNLL) and contrastive entropy (ΔH) between the base and calibration models. Filter out samples with extreme ΔNLL, then select the lowest ΔH samples from the learnable range, in a domain-adaptive way. This process can optionally be repeated with the updated model for further gains.

3.3 Two-Stage Selection Framework

INSTRUCTDIFF implements our selection principle through two stages (Figure 2).

Stage (1) Warmup Calibration: To obtain the reference distribution π_{inst}, we randomly sample a warmup subset $\mathcal{D}_{\text{warmup}}$ of size $\alpha \cdot N$ (typically $\alpha = 0.1$) and fine-tune:

$$\pi_{\text{inst}}^{(0)} = \arg\min_{\pi} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{warmup}}} [-\log \pi_\theta(y \mid x)] \quad (4)$$

Recent work demonstrates that small subsets suffice to activate latent instruction-following capabilities (Zhou et al., 2023), making this lightweight calibration both feasible and effective.

Stage (2) Distribution-Aware Selection: We compute ΔNLL_i and ΔH_i for all samples using Equations (1) and (2), then apply bi-directional NLL filtering followed by entropy-based ranking.

Bi-Directional NLL Filtering. We exclude pathological extremes through symmetric rejection: the bottom γ percentile (typically $\gamma = 0.1$) contains near-duplicates providing negligible gradient signal, while the top γ percentile contains incomprehensible patterns that may induce catastrophic forgetting (Kirkpatrick et al., 2017). This retains the learnable middle range:

$$\mathcal{D}_{\text{filtered}} = \{(x_i, y_i) \in \mathcal{D} : q_\gamma \leq \Delta\text{NLL}_i \leq q_{1-\gamma}\} \quad (5)$$

Entropy-Based Selection: From $\mathcal{D}_{\text{filtered}}$, we select $\beta \cdot N$ samples (typically $\beta = 0.1$) by ranking according to ΔH. Empirical validation (Section 5.1) confirms that selecting samples with the *lowest contrastive entropy* (ΔH) consistently yields superior performance. This unified criterion automatically adapts: negative ΔH (cognitive expansion) for reasoning tasks, and low positive ΔH (cognitive compression) for general tasks, without requiring manual domain specification.

3.4 Iterative Refinement

After fine-tuning on the initially selected subset $\mathcal{D}'^{(1)}$, we obtain an improved calibration model:

$$\pi_{\text{inst}}^{(t)} = \arg\min_{\pi} \mathbb{E}_{(x,y)\sim\mathcal{D}'^{(t)}} [-\log \pi_\theta(y \mid x)] \quad (6)$$

This refined model serves as the measuring instrument for iteration $t+1$, enabling progressive identification of samples at the model's evolving learnable frontier. As shown in Section 5.6 and Figure 5, performance gains are largest at the second iteration, with diminishing returns thereafter. The complete algorithm is detailed in Algorithm 1 (Appendix I).

4 Experiments

4.1 Experimental Setup

Datasets and Models. We evaluate `InstructDiff` across four domains: **mathematics** using NuminaMath (LI et al., 2024) (10k samples) with Qwen2.5-7B (Qwen et al., 2025), **general instruction-following** using Alpaca (Li et al., 2023b) (10k samples) with LLaMA3-8B (Dubey et al., 2024), **medical QA** using MedCAQA (Pal et al., 2022) (10k samples) with LLaMA3-8B, and **code generation** using BigCode (Cassano et al., 2024) (10k samples) with LLaMA3-8B. Detailed dataset characteristics are in Appendix A.

Baselines. We compare against: (1) **Random Sampling**; (2) **PPL-based** (PPL_{Min}, PPL_{Mid}, PPL_{Max}); (3) **Entropy-based** ($\text{Entropy}_{\text{Min}}$, $\text{Entropy}_{\text{Mid}}$, $\text{Entropy}_{\text{Max}}$); (4) **Length-based** (Resp Len_{Max}, Inst Len_{Max}, $\text{Inst/Resp}_{\text{Max}}$, Inst/Resp$_{\text{Min}}$); (5) **IFD** (Li et al., 2024b), **Superfiltering** (Li et al., 2024a), **SelectIT** (Liu et al., 2024), and **ZIP** (Yin et al., 2024); (6) **Full Training** on complete 10k data. Implementation details are in Appendix B.

Training and Evaluation. All models use full-parameter supervised fine-tuning. For `InstructDiff`, key hyperparameters include warmup ratio $\alpha = 0.1$, NLL rejection ratio $\gamma = 0.1$, and selection ratio $\beta = 0.1$ (code: $\beta = 0.2$). Evaluation is conducted on standard benchmark splits in each domain. More details on training settings, evaluation protocols, and benchmark breakdowns are provided in Appendix D.

4.2 Main Results

Tables 1 and 2 present our main results across four domains. `InstructDiff` consistently achieves the best performance using only 10% of training data (20% for code), outperforming all baselines including full-data training. `InstructDiff` is the *only* method that consistently surpasses full training performance while using merely 10-20% of data, achieving average relative improvements of +17% (math), +6.2% (medical), +52% (general), and +4.9% (code) over full training. On mathematics (Qwen2.5-7B), `InstructDiff` achieves 31.63 average score versus 27.05 for full training, with particularly strong gains on AIME 2024 (7.71 vs. 5.00, +54%). For medical QA (LLaMA3-8B), `InstructDiff` reaches 56.42 average accuracy compared to 53.14 for full training (+6.2%). On general instruction-following (LLaMA3-8B), `InstructDiff` achieves 12.09% LC win rate versus 8.15% for full training (+48%). For code generation (LLaMA3-8B, 20% data), `InstructDiff` scores 45.1 versus 43.0 for full training (+4.9%). These results validate that entropy-guided selection identifies samples at the model's learnable frontier, enabling more efficient and effective fine-tuning than volume-based approaches.

4.3 Ablation Studies

We conduct comprehensive ablation studies to validate each component of `InstructDiff`, covering: (1) bi-directional NLL filtering, (2) entropy selection range, and (3) warmup data size. We additionally decouple the contrastive entropy metric from the iterative refinement mechanism in Appendix E.

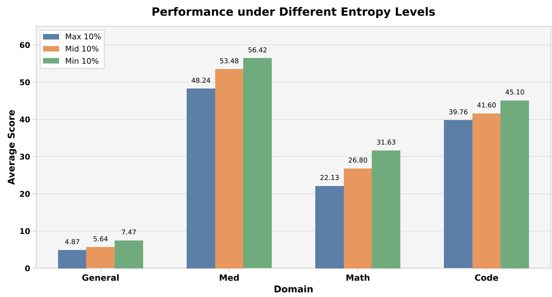

Figure 3: **Contrastive Entropy Selection Range Analysis.** Bottom 10% (lowest ΔH) consistently outperforms across all domains, validating the domain-adaptive contrastive entropy principle.

Bi-Directional NLL Filtering. Table 3 demonstrates that removing bi-directional filtering causes substantial performance drops: 2.59 points on mathematics (31.69 → 29.10), 1.66 points on general instruction-following (12.09 → 10.43), and 5.56 points on medical QA (56.42 → 50.86). This validates that extreme ΔNLL samples, representing either redundant knowledge (bottom percentile) or incomprehensible patterns (top percentile), introduce training noise. The filtering mechanism effectively identifies the learnable frontier where samples provide meaningful gradient signals without catastrophic forgetting risks.

Entropy Selection Range. We systematically evaluate three entropy selection strategies: top 10% (Max 10%, highest ΔH), middle 10% (Mid 10%), and bottom 10% (Min 10%, lowest ΔH). Figure 3 shows that bottom 10% (low ΔH) consistently outperforms across all domains. For mathematics, Min 10% achieves 31.63 versus 22.13 for Max 10% (+43% relative gain). For general instruction-following, Min 10% reaches 7.47 versus 4.87 for Max 10% (+53%). Medical and code domains show similar patterns: Min 10% achieves 56.42 (vs. 48.24 Max) and 45.19 (vs. 39.76 Max) respectively. Middle-range selections consistently underperform across all domains. These results validate our core principle: selecting samples with the lowest contrastive entropy consistently yields superior performance.

Warmup Data Size. Table 4 analyzes warmup set size trade-offs. We evaluate four ratios: 1%, 10%, 20%, and 50%. Results show 10% (1k samples) provides optimal balance, achieving 31.63 average score versus 25.03 for 1% and 29.59 for 50%. The 1% degradation stems from insufficient

Method	Math Domain (*Qwen2.5-7B / select 10%*)						General (*LLaMA3-8B / select 10%*)		
	AIME24	Math-OAI	Minerva	Olympiad	ACM23	Avg	AlpacaEval (%)	Arena-Hard (%)	Avg
Base Model	1.65	28.79	9.26	7.69	15.65	12.61	1.36	0.26	0.81
ALL	5.00 (+3.35)	57.93 (+29.14)	17.83 (+8.57)	22.30 (+14.61)	32.19 (+16.54)	27.05 (+14.44)	8.15 (+6.79)	1.70 (+1.44)	4.93 (+4.12)
PPL$_{\text{Min}}$	5.63 (+3.98)	57.80 (+29.01)	16.35 (+7.09)	23.68 (+13.99)	31.72 (+16.07)	27.04 (+14.43)	9.20 (+7.84)	2.12 (+1.86)	5.66 (+4.85)
PPL$_{\text{Mid}}$	6.05 (+4.40)	54.76 (+25.97)	13.61 (+4.35)	21.01 (+13.32)	32.81 (+17.16)	25.65 (+13.04)	7.25 (+5.89)	2.51 (+2.25)	4.88 (+4.07)
PPL$_{\text{Max}}$	3.95 (+2.30)	50.56 (+21.77)	12.77 (+3.51)	18.57 (+10.88)	28.28 (+12.63)	22.83 (+10.22)	9.72 (+8.36)	1.54 (+1.28)	5.63 (+4.82)
Entropy$_{\text{Min}}$	5.83 (+4.18)	60.74 (+31.95)	18.59 (+9.33)	27.13 (+19.44)	36.72 (+21.07)	29.80 (+17.19)	10.45 (+9.09)	**3.51 (+3.25)**	6.98 (+6.17)
Entropy$_{\text{Mid}}$	3.54 (+1.89)	54.50 (+25.71)	12.27 (+3.01)	20.12 (+12.43)	29.38 (+13.73)	23.96 (+11.35)	7.44 (+6.08)	2.36 (+2.10)	4.90 (+4.09)
Entropy$_{\text{Max}}$	3.12 (+1.47)	50.50 (+21.71)	12.41 (+3.15)	18.54 (+10.85)	23.59 (+7.94)	21.63 (+9.02)	7.50 (+6.14)	1.53 (+1.27)	4.52 (+3.71)
Resp Len$_{\text{Max}}$	2.28 (+0.63)	44.65 (+15.86)	10.02 (+0.76)	15.38 (+7.69)	21.56 (+5.91)	18.78 (+6.17)	11.04 (+9.68)	2.91 (+2.65)	6.98 (+6.17)
Inst Len$_{\text{Max}}$	4.79 (+3.14)	54.98 (+26.19)	12.27 (+3.01)	20.99 (+13.30)	29.53 (+13.88)	24.51 (+11.90)	5.47 (+4.11)	2.02 (+1.76)	3.75 (+2.94)
Inst/Resp$_{\text{Max}}$	3.33 (+1.68)	50.33 (+21.54)	12.08 (+2.82)	16.96 (+9.27)	28.28 (+12.63)	22.20 (+9.59)	4.60 (+3.24)	3.20 (+1.26)	3.90 (+3.09)
Inst/Resp$_{\text{Min}}$	3.74 (+2.09)	52.50 (+23.71)	12.27 (+3.01)	20.14 (+12.45)	27.97 (+12.32)	23.32 (+10.71)	9.44 (+8.08)	2.74 (+2.38)	6.09 (+5.28)
Random	5.43 (+3.78)	55.41 (+26.62)	14.86 (+5.60)	20.18 (+12.49)	33.13 (+17.48)	25.80 (+13.19)	5.44 (+4.08)	2.18 (+1.92)	3.81 (+3.00)
IFD	3.12 (+1.47)	50.36 (+21.57)	13.98 (+4.72)	18.14 (+10.45)	24.38 (+8.73)	21.99 (+9.38)	10.75 (+9.39)	2.29 (+2.03)	6.52 (+5.71)
SelectIT	4.15 (+2.50)	52.86 (+24.07)	15.06 (+5.80)	20.29 (+12.60)	29.38 (+13.73)	24.35 (+11.74)	7.84 (+6.48)	2.34 (+2.08)	5.09 (+4.28)
ZIP	4.15 (+2.50)	50.30 (+21.51)	12.23 (+2.97)	16.69 (+8.99)	30.00 (+14.35)	22.61 (+10.00)	6.90 (+5.54)	0.78 (+0.52)	3.84 (+3.03)
Superfiltering	4.80 (+3.15)	55.55 (+26.76)	14.06 (+4.80)	21.38 (+13.69)	30.78 (+15.13)	25.31 (+12.70)	12.08 (+10.72)	2.61 (+2.35)	7.35 (+6.54)
Ours	**7.71 (+6.06)**	**61.79 (+33.00)**	**21.42 (+12.16)**	**26.94 (+19.25)**	**40.31 (+24.66)**	**31.63 (+19.02)**	**12.09 (+10.73)**	2.84 (+2.58)	**7.47 (+6.66)**

Table 1: Comparison of data selection strategies across Math and General domains. Values are annotated with the change relative to the Base Model (red for decrease, olive for increase). Math benchmarks are evaluated with Qwen2.5-7B, while General benchmarks are evaluated with LLaMA3-8B.

Method	Medical (*LLaMA3-8B / select 10%*)				Code (*LLaMA3-8B / select 20%*)					
	MedQA	MMLU	MedMCQA	Avg	HumanEval	HumanEval+	MBPP	MBPP+	Bigcode	Avg
Base Model	43.60	51.04	48.98	47.87	40.2	33.5	55.3	46.3	22.5	39.6
ALL	49.96 (+6.36)	61.04 (+10.00)	48.41 (-0.57)	53.14 (+5.27)	44.5 (+4.3)	35.4 (+1.9)	59.8 (+4.5)	49.7 (+3.4)	**25.8 (+3.3)**	43.0 (+3.4)
PPL$_{\text{Min}}$	48.23 (+4.63)	57.69 (+6.65)	43.51 (-5.47)	49.81 (+1.94)	41.5 (+1.3)	34.1 (+0.6)	56.1 (+0.8)	46.3 (-0.0)	23.4 (+0.9)	40.3 (+0.7)
PPL$_{\text{Mid}}$	43.75 (-0.85)	48.45 (-2.59)	35.09 (-13.89)	42.43 (-5.44)	44.5 (+4.3)	36.6 (+3.1)	56.9 (+1.6)	46.8 (-0.5)	21.6 (-0.9)	41.3 (+1.7)
PPL$_{\text{Max}}$	47.76 (+4.16)	58.38 (+7.34)	45.37 (-3.61)	50.50 (+2.63)	40.2 (-0.0)	34.1 (+0.6)	55.6 (+0.3)	47.4 (+1.1)	21.0 (-1.5)	39.7 (-0.1)
Entropy$_{\text{Min}}$	36.76 (-6.84)	42.24 (-8.80)	34.23 (-14.75)	37.74 (-10.13)	46.3 (+6.1)	39.0 (+5.5)	58.7 (+3.4)	46.3 (-0.0)	23.4 (+0.9)	42.7 (+3.1)
Entropy$_{\text{Mid}}$	46.43 (+2.83)	53.30 (+2.26)	42.19 (-6.79)	47.31 (-0.56)	41.5 (+1.3)	34.1 (+0.6)	59.3 (+4.0)	48.1 (+1.8)	23.8 (+1.3)	41.4 (+1.8)
Entropy$_{\text{Max}}$	52.40 (+8.80)	**66.52 (+15.48)**	**52.16 (+3.18)**	**57.03 (+9.16)**	40.9 (+0.7)	34.8 (+1.3)	57.9 (+2.6)	48.4 (+2.1)	21.1 (-1.4)	40.6 (+1.0)
Resp Len$_{\text{Max}}$	42.11 (-1.49)	41.99 (-9.05)	32.44 (-16.54)	38.85 (-9.02)	47.0 (+6.8)	37.8 (+4.3)	57.4 (+2.1)	48.4 (+2.1)	24.2 (+1.7)	43.0 (+3.4)
Inst Len$_{\text{Max}}$	49.25 (+5.65)	57.10 (+6.06)	45.52 (-3.46)	50.62 (+2.75)	42.1 (+1.9)	35.4 (+1.9)	57.7 (+2.4)	46.3 (-0.0)	23.6 (+1.1)	41.0 (+1.4)
Inst/Resp$_{\text{Max}}$	50.98 (+7.38)	58.71 (+7.67)	43.92 (-5.06)	51.20 (+3.33)	43.9 (+3.7)	38.4 (+4.9)	59.0 (+3.7)	49.2 (+2.9)	21.9 (-0.6)	42.5 (+2.9)
Inst/Resp$_{\text{Min}}$	35.51 (-8.09)	44.65 (-6.39)	40.16 (-8.82)	40.11 (-7.76)	47.0 (+6.8)	38.4 (+4.9)	56.1 (+0.8)	46.3 (-0.0)	22.8 (+0.3)	42.1 (+2.5)
Random	45.95 (+2.35)	54.22 (+3.18)	40.11 (-8.87)	46.76 (-1.11)	42.1 (+1.9)	33.5 (-0.0)	57.7 (+2.4)	47.9 (+1.6)	22.9 (+0.4)	40.8 (+1.2)
IFD	43.13 (-0.47)	48.27 (-2.77)	42.36 (-6.62)	44.59 (-3.28)	40.9 (+0.7)	34.1 (+0.6)	51.6 (-3.7)	40.7 (-5.6)	18.2 (-4.3)	37.1 (-2.5)
SelectIT	46.11 (+2.51)	46.84 (-4.20)	35.93 (-13.05)	42.96 (-4.91)	48.2 (+8.0)	38.4 (+4.9)	60.6 (+5.3)	**51.9 (+5.6)**	23.0 (+0.5)	44.4 (+4.8)
ZIP	51.53 (+7.93)	62.87 (+11.83)	48.84 (-0.14)	54.41 (+6.54)	38.4 (-1.8)	32.9 (-0.6)	59.3 (+4.0)	50.3 (+4.0)	16.0 (-6.5)	39.4 (-0.2)
Superfiltering	41.63 (-1.97)	41.15 (-9.89)	32.08 (-16.90)	38.29 (-9.58)	43.3 (+3.1)	34.8 (+1.3)	54.2 (-1.1)	44.2 (-2.1)	20.6 (-1.9)	39.4 (-0.2)
Ours	**54.67 (+11.07)**	64.48 (+13.44)	50.11 (+1.13)	56.42 (+8.55)	**48.2 (+8.0)**	**40.9 (+7.4)**	**60.6 (+5.3)**	50.0 (+3.7)	25.7 (+3.2)	**45.1 (+5.5)**

Table 2: Comparison of data selection strategies across Medical and Code domains, with relative changes from Base Model shown in olive (+) and red (-). Both domains use LLaMA3-8B. Avg is the average across the metrics of each domain.

Method	Math	General	Medical	Code
Full pipeline	31.69	12.09	56.42	45.1
w/o NLL filter	29.10	10.43	50.86	40.2
Δ	-2.59	-1.66	-5.56	-4.9

Table 3: **Ablation: Bi-Directional NLL Filtering.** Removing NLL filtering degrades performance across all domains.

Warmup	AIME24	Math-OAI	Minerva	Olympiad	ACM23	Avg
1%	3.32	55.61	14.91	21.13	30.16	25.03
10%	**7.71**	**61.79**	21.42	**26.94**	**40.31**	**31.63**
20%	7.51	61.36	**23.35**	26.98	36.88	31.22
50%	6.25	60.68	21.14	25.81	34.06	29.59

Table 4: **Ablation: Warmup Data Size.** 10% warmup provides optimal balance between calibration quality and computational cost.

calibration—only 100 samples cannot capture representative instruction-tuning dynamics, yielding noisy gap estimates. The 50% decline indicates diminishing returns: larger warmup sets consume excessive resources without improving distributional measurements. The 10% setting achieves strong performance across all benchmarks (AIME 2024: 7.71, Olympiad: 26.94), and even 20% warmup (31.22) performs comparably, suggesting robustness to moderate variations.

5 Analyses

5.1 Understanding Entropy-Guided Selection

Why Low ΔH Works Across Domains. To understand the mechanism behind ΔH's effectiveness, we analyze its correlation with various metrics in Figure 4 (detailed metric definitions in Appendix C). ΔH exhibits moderate correlation with ΔNLL but negligible correlation with length, confirming it captures cognitive alignment rather than superficial statistics. The effectiveness stems from two key insights. First, **model-awareness**: optimal samples must be evaluated relative to the base model's current knowledge state, not in isolation. Second, **domain-adaptive learning trajectories**: the manifestation of lowest contrastive entropy fundamentally differs by task structure. For mathematical reasoning, the lowest ΔH corresponds to *negative* values—entropy increase during calibration, i.e., **cognitive expansion**. These samples encourage the model to preserve path diversity rather than collapsing into a single rigid pattern, as effective problem-solving requires exploring diverse solution paths (Cheng et al., 2025). Selecting the *highest* ΔH for reasoning conversely induces massive uncertainty reduction (**cognitive compression**), discarding path diversity in favor of rote memorization; our ablation empirically confirms this: Min 10% achieves 31.63 versus 22.13 for Max 10%. Conversely, for general instruction-following and medical QA, the lowest ΔH manifests as *small positive* values—entropy decrease. Here, fine-tuning compresses diffuse priors onto canonical behaviors (Yin et al., 2024), representing knowledge activation rather than injection. Weak correlations with complexity proxies further validate that gains arise from aligning with learning dynamics rather than task difficulty alone. Unlike single-model metrics, ΔH directly measures distributional shifts, automatically identifying samples at the model's learnable frontier across domains.

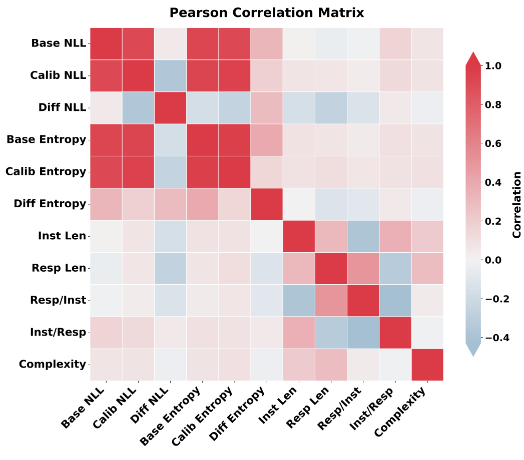

Figure 4: **Correlation Matrix.** ΔH measures cognitive alignment orthogonal to difficulty metrics.

Seed	AIME24	Math-OAI	Minerva	Olympiad	ACM23	Avg
42	7.71	61.79	21.42	26.94	40.31	31.63
43	6.68	62.21	20.62	27.32	34.53	30.27
44	8.13	62.20	21.66	27.54	39.38	31.78
45	7.70	62.72	22.32	27.29	39.06	31.82
46	6.88	61.49	19.92	26.01	34.69	29.80
Mean ± Std	**7.42±0.59**	**62.08±0.42**	**21.19±0.86**	**27.02±0.59**	**37.59±2.42**	**31.06±0.92**

Table 5: **Robustness to warm-up subsample choice (Math domain, Qwen2.5-7B).** Five independent random seeds yield consistent results with low standard deviations, confirming that `InstructDiff`'s performance is stable with respect to the choice of random subsample used during warm-up calibration.

5.2 Robustness to Warm-up Subsample Choice

To assess whether results are sensitive to the specific random subsample used during the initial warm-up fine-tuning stage, we run `InstructDiff` on the Math domain using five different random seeds (42–46). Table 5 reports per-seed results together with mean and standard deviation.

The small standard deviations (e.g., 0.92 on average score) confirm that `InstructDiff` is robust to the specific warm-up subsample. The performance is consistently strong across all seeds, and the average of 31.06 substantially outperforms the full-data baseline (27.05).

> ⭐ **Takeaway 1:**
>
> `InstructDiff` is robust to warm-up subsample choice: across 5 random seeds, the Math average score varies only 31.06 ± 0.92, all well above the full-data baseline of 27.05.

5.3 Scaling to Large-Scale Data Selection

We evaluate `InstructDiff`'s effectiveness on larger datasets by selecting 10k from 100k samples for both mathematics and general instruction-following, comparing against random selection and full training. Table 6 shows `InstructDiff` successfully scales to large pools. On mathematics, selecting 10% from 100k achieves 29.80 average score, outperforming random-10% (27.94, +6.7% relative) and full-100k training (27.16, +9.7% relative). For general instruction-

Domain	Method	Pool	Selected	Avg
Math	Full	100k	100%	27.16
	Random	100k	10%	27.94
	Superfilter	100k	10%	27.51
	Ours	100k	10%	**29.80**
General	Full	52k	10%	7.34
	Random	52k	10%	7.20
	Superfilter	52k	10%	6.61
	Ours	52k	10%	**8.01**

Table 6: **Scaling to Large-Scale Selection.** `InstructDiff` maintains effectiveness at scale, outperforming both random selection and full training.

following, `InstructDiff` achieves 8.01 average score versus 7.20 for random (+11.3%) and 7.34 for full training (+9.1%). These results indicate entropy-guided principles generalize effectively to larger data pools, with relative gains maintained even at scale.

⭐ Takeaway 2:

`InstructDiff` scales to 100k data pools, where selecting 10% outperforms both full training (+9.7%) and random selection (+6.7%), with efficiency gains amplifying at larger scales.

5.4 Weak-to-Strong Calibration for Efficient Selection

We investigate whether smaller models can calibrate data selection for larger models by using Qwen2.5-0.5B and Qwen2.5-1.5B to select data for training Qwen2.5-7B. Table 7 shows weak-to-strong calibration achieves competitive performance. Qwen2.5-7B calibrated using Qwen2.5-0.5B achieves 28.4 average score, +10.1% over random (25.80) and only 10.2% below same-size 7B calibration (31.63). Qwen2.5-1.5B calibration reaches 27.6 average, +7.0% over random, offering a middle ground between efficiency and performance. The computational advantage is substantial: 0.5B or 1.5B calibration reduces warmup computation by up to 14× or 4.7× respectively based on parameter count ratios, while retaining most performance gains, providing an efficient option for compute-constrained practitioners.

Calibration	Target	AIME24	Math-OAI	Minerva	Olympiad	ACM23	Avg
Random	Qwen2.5-7B	5.43	55.41	14.86	20.18	33.13	25.80
Qwen2.5-0.5B	Qwen2.5-7B	6.67	59.35	16.76	23.91	35.31	28.4
Qwen2.5-1.5B	Qwen2.5-7B	6.03	59.5	16.24	23.71	32.5	27.6
Qwen2.5-7B	Qwen2.5-7B	**7.71**	**61.79**	**21.42**	**26.94**	**40.31**	**31.63**

Table 7: **Weak-to-Strong Calibration.** Smaller models effectively calibrate for larger models within the same family, enabling 4.2× computational savings.

⭐ Takeaway 3:

Weak-to-strong calibration using 0.5B model enables 14× computational savings while retaining 89.8% of same-size calibration performance within the same model family.

5.5 Diversity of Selected Data

A potential concern is that entropy-based selection may favor specific data types, reducing instruction diversity. We evaluate the selected subsets using two complementary diversity metrics: **Self-GLEU** (Mutton et al., 2007) (lower is more diverse;) and **Unique V-N Pairs** (Wang et al., 2023) (higher is more diverse; counts unique verb-noun pairs across instructions). Table 8 reports results on the General and Math domains.

Domain	Method	Self-GLEU ↓	NV-Pair ↑
General	`InstructDiff`	0.2371	255
	IFD	**0.2316**	223
	SelectIT	0.2536	**360**
Math	`InstructDiff`	**0.4759**	**406**
	IFD	0.5501	107
	SelectIT	0.5024	357

Table 8: **Diversity of selected data.** On Math, `InstructDiff` achieves substantially higher NV-Pair than IFD (406 vs. 107), indicating much better structural diversity. Self-GLEU remains competitive across methods, confirming no collapse toward repetitive instruction templates.

On the Math domain, `InstructDiff` achieves substantially higher structural diversity than IFD (NV-Pair: 406 vs. 107), demonstrating that entropy-based selection does not overfit to narrow instruction patterns. Self-GLEU remains competitive, confirming no collapse toward repetitive templates. Although SelectIT achieves slightly higher NV-Pair in the General domain, `InstructDiff`'s downstream performance is substantially higher, indicating a better quality–diversity trade-off.

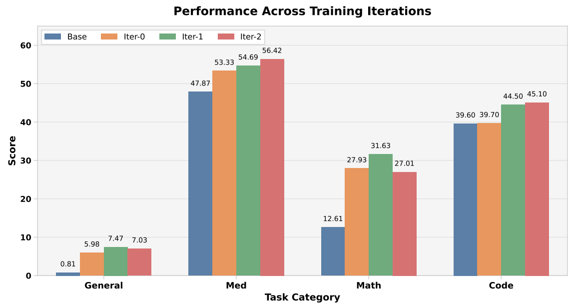

Figure 5: **Iterative Refinement.** Two iterations provide optimal cost-benefit trade-off before diminishing returns.

5.6 Iterative Refinement Analysis

We assess iterative refinement over up to three iterations ($T = 1, 2, 3$) across all four domains, using the previously fine-tuned model for calibration each time (Section 3.4). Figure 5 shows the largest gains at the second iteration—for instance, math improves from 31.63 ($T = 1$) to 32.89 ($T = 2$, +4.0%), with diminishing returns at $T = 3$ (33.24, +1.1%). Similar trends occur in other domains, suggesting $T = 2$ offers the best balance of performance and cost.

Computational Efficiency. `InstructDiff` with $T = 1$ or $T = 2$ requires only 39–64% of full-data training time while substantially outperforming it; using a smaller same-family calibration model (Section 5.4) can further reduce overhead. Full wall-clock time comparisons across iterations are provided in Appendix G.

⭐ Takeaway 4:

Iterative refinement improves selection quality consistently, with 2 iterations optimal before diminishing returns emerge. Full efficiency details and the isolation of ΔH's contribution (vs. NLL filtering alone) are in Appendix G and F.

5.7 Cross-Model Selection Consistency

We investigate cross-model selection consistency via two experiments. First, we use different models (Qwen2.5-7B/14B, LLaMA3-8B/3.1-8B) to independently select data from the same pool using `InstructDiff`, then measure selection overlap (Table 7). Same-family models show 60–64% overlap, compared to just 15–20% for cross-family pairs. Second, we train the target model Qwen2.5-7B on data selected by different calibration models (Table 8). Same-family calibration (Qwen2.5-14B) achieves a 30.84 average score, outperforming cross-family calibration (LLaMA: 25.74–26.31) by 17–20%. These results show that selection consistency is family-dependent: same-family models agree on "good data" and yield better downstream performance.

Model Pair	Overlap	R(A)	R(B)
Qwen2.5-7B / Qwen2.5-14B	545	0.64	0.62
Qwen2.5-7B / LLaMA3-8B	131	0.15	0.15
Qwen2.5-7B / LLaMA3.1-8B	164	0.19	0.15
Qwen2.5-14B / LLaMA3-8B	152	0.17	0.18
Qwen2.5-14B / LLaMA3.1-8B	178	0.20	0.17
LLaMA3-8B / LLaMA3.1-8B	396	0.46	0.37

Table 9: **Cross-Model Selection Overlap.** Same-family models show 3-4× higher overlap than cross-family pairs.

Family	Calibration Model	Size	Avg
Cross-family	LLaMA3-8B	10%	26.31
Cross-family	LLaMA3.1-8B	10%	25.74
Same-family	Qwen2.5-7B	10%	27.93
Same-family	Qwen2.5-14B	10%	**30.84**

Table 10: **Calibration Family Impact.** Target: Qwen2.5-7B. Same-family calibration yields +19.8% gain over cross-family.

⭐ Takeaway 5:

Selection consistency is family-dependent: same-family models share 60-64% selections versus 15-20% cross-family, validating the need for matched calibration.

6 Conclusion

We propose `InstructDiff`, a simple, unified data selection framework based on contrastive entropy (ΔH). Comparing base and calibrated models, we show that selecting examples with the lowest contrastive entropy consistently yields optimal fine-tuning across domains, without expert heuristics. `InstructDiff` achieves up to 52% improvement over full-data training using just 10%–20% of data and scales efficiently to large datasets with fast calibration. These results show that contrastive entropy is a principled, domain-agnostic criterion for efficient, effective LLM adaptation.

Limitations

Hyperparameter Stability. In our main experiments, both the bi-directional reject ratio γ and the selection ratio β are fixed at 0.1 across all domains. Because our method ranks samples by relative entropy shifts, the relative ordering remains stable across domains, and fixing $\beta = 0.1$ performs robustly without domain-specific assumptions. We note in passing that *fine-tuning* γ per domain (e.g., $\gamma = 0.05$ for medical data) may yield marginal additional gains, but this is entirely optional—the framework succeeds without such adjustment. Fully automated hyperparameter selection remains an open direction for future work.

Warmup Dependency. Selection quality depends on the calibration model quality. Very small warmup sets (e.g., 1% or 100 samples) produce noisy gap measurements, as shown in Table 4. We recommend using at least ∼1,000 warmup samples to ensure stable and reliable contrastive entropy estimates.

Model Family Specificity. Cross-model overlap analysis (Table 9) indicates that optimal selected subsets vary significantly across model families (same-family overlap: 60-64% vs. cross-family: 15-20%). This suggests practitioners should ideally compute distributional gaps using their target base model or a same-family calibration model to maximize selection quality and downstream performance.

Future Directions. Understanding the mechanisms behind ΔH—specifically why cognitive expansion (negative ΔH) benefits reasoning while cognitive compression (small positive ΔH) benefits general tasks—remains an open and highly promising direction. Investigating how these opposing entropy dynamics relate to the model's internal representation changes could yield more principled selection criteria beyond ranking by ΔH magnitude, potentially enabling even stronger and more targeted data selection strategies.

Acknowledgements

The work of Guanhua Chen, Junyou Su, and He Zhu is supported by the National Natural Science Foundation of China (No. 62306132), the Guangdong Basic and Applied Basic Research Foundation (No. 2025A1515011564). The work of Yun Chen is supported by the Natural Science Foundation of Shanghai (No. 25ZR1402136).

References

AI Mathematical Olympiad. 2024. Ai mathematical olympiad prize datasets.

American Institute of Mathematics. 2024. Aime 2024 competition mathematical problems.

Anthropic. 2025. Introducing claude 4.5 sonnet. https://www.anthropic.com/news/claude-sonnet-4-5.

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

Jan Betley, Daniel Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martín Soto, Nathan Labenz, and Owain Evans. 2025. Emergent misalignment: Narrow finetuning can produce broadly misaligned llms. *arXiv preprint arXiv:2502.17424*.

Federico Cassano, John Gouwar, Francesca Lucchetti, Claire Schlesinger, Anders Freeman, Carolyn Jane Anderson, Molly Q Feldman, Michael Greenberg, Abhinav Jangda, and Arjun Guha. 2024. Knowledge transfer from high-resource to low-resource programming languages for code llms. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA2):677–708.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, and 1 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. 2025. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. https://huggingface.co/datasets/AI-MO/NuminaMath-CoT.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024a. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. *arXiv preprint arXiv:2402.00530*.

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

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason Weston, and Mike Lewis. 2023a. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Ling-Hao Chen, Junhao Liu, Tongliang Liu, and 1 others. 2024c. 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.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. 2024. Selectit: Selective instruction tuning for llms via uncertainty-aware self-reflection. *Advances in Neural Information Processing Systems*, 37:97800–97825.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*.

Zihan Liu, Yang Chen, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Acemath: Advancing frontier math reasoning with post-training and reward modeling. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3993–4015.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Instag: Instruction tagging for analyzing supervised fine-tuning of large language models. *Preprint*, arXiv:2308.07074.

Zhicun Lyu, Xinye Li, Zheng Xie, and Ming Li. 2025. Top pass: improve code generation by pass@ k-maximized code ranking. *Frontiers of Computer Science*, 19(8):198341.

Mathematical Association of America. 2023. Amc 2023 competition problems.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. GLEU: Automatic evaluation of sentence-level fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, Prague, Czech Republic. Association for Computational Linguistics.

OpenAI. 2025. Openai o3 and o4-mini system card, january 2025. https://openai.com/index/o3-o4-mini-system-card.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Abdullah Vanlioglu. 2025. Entropy-guided sequence weighting for efficient exploration in rl-based llm fine-tuning. *arXiv preprint arXiv:2503.22456*.

Christian Walder and Deep Karkhanis. 2025. Pass@ k policy optimization: Solving harder reinforcement learning problems. *arXiv preprint arXiv:2505.15201*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Yejie Wang, Keqing He, Dayuan Fu, Zhuoma Gongque, Heyang Xu, Yanxu Chen, Zhexu Wang, Yujia Fu, Guanting Dong, Muxi Diao, Jingang Wang, Mengdi Zhang, Xunliang Cai, and Weiran Xu. 2024. How do your code llms perform? empowering code instruction tuning with high-quality data. *Preprint*, arXiv:2409.03810.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

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

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

Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. 2024b. Rethinking data selection at scale: Random selection is almost all you need. *Preprint*, arXiv:2410.09335.

Yang Xiao, Mohan Jiang, Jie Sun, Keyu Li, Jifan Lin, Yumin Zhuang, Ji Zeng, Shijie Xia, Qishuo Hua, Xuefeng Li, and 1 others. 2025. Limi: Less is more for agency. *arXiv preprint arXiv:2509.17567*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*.

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

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

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jianfeng Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023. Lima: Less is more for alignment. In *Advances in Neural Information Processing Systems*, volume 36.

He Zhu, Zhiwen Ruan, Junyou Su, Xingwei He, Yun Chen, Wenjia Zhang, and Guanhua Chen. 2025a. Tag-instruct: Controlled instruction complexity enhancement through structure-based augmentation. *Preprint*, arXiv:2505.18557.

He Zhu, Junyou Su, Tianle Lun, Yicheng Tao, Wenjia Zhang, Zipei Fan, and Guanhua Chen. 2024. Fanno: Augmenting high-quality instruction data with open-sourced llms only. *Preprint*, arXiv:2408.01323.

Yongfu Zhu, Lin Sun, Guangxiang Zhao, Weihong Lin, and Xiangzheng Zhang. 2025b. Uncertainty under the curve: A sequence-level entropy area metric for reasoning llm. *arXiv preprint arXiv:2508.20384*.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, and 1 others. 2024. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. *arXiv preprint arXiv:2406.15877*.

A Dataset Details

Mathematics (NuminaMath). The NuminaMath dataset contains 10,000 mathematical reasoning problems covering arithmetic, algebra, geometry, and competition-level mathematics. Problems range from elementary school to Mathematical Olympiad difficulty. Each sample consists of a problem statement (instruction) and a step-by-step solution (response).

General Instruction-Following (Alpaca). The Alpaca dataset contains 10,000 instruction-following examples spanning diverse tasks including question answering, creative writing, summarization, and brainstorming. Instructions vary in complexity and response length, representing typical user interactions with instruction-tuned models.

Medical QA (MedCAQA). We randomly sample 10,000 training examples from the MedCAQA dataset, which contains multiple-choice questions from Indian medical entrance exams. Questions cover anatomy, physiology, pharmacology, pathology, and clinical scenarios. Each sample includes a medical question and a detailed explanation.

Code Generation (BigCode). We randomly sample 10,000 code generation examples from the BigCode dataset. Samples include function signatures with natural language descriptions and corresponding Python implementations, covering algorithms, data structures, and practical programming tasks.

B Baseline Implementation Details

PPL-based Selection. We compute the per-token perplexity using the base model for all samples, then select the top 10% based on three strategies: PPL_{Min} selects samples with lowest perplexity (easiest for base model), PPL_{Mid} selects samples in the middle 10% range, and PPL_{Max} selects samples with highest perplexity (hardest for base model).

Entropy-based Selection. We compute the average per-token entropy using the base model, then select samples based on three strategies: $\text{Entropy}_{\text{Min}}$ (lowest entropy, highest confidence), $\text{Entropy}_{\text{Mid}}$ (middle range), and $\text{Entropy}_{\text{Max}}$ (highest entropy, lowest confidence).

Length-based Selection. We implement four variants: Resp Len_{Max} selects samples with longest responses, Inst Len_{Max} selects samples with longest instructions, $\text{Inst/Resp}_{\text{Max}}$ selects samples with highest instruction-to-response length ratio, and $\text{Inst/Resp}_{\text{Min}}$ selects samples with lowest ratio.

Advanced Baselines. For IFD, Superfiltering, SelectIT, and ZIP, we follow their original implementations and hyperparameters as described in their respective papers. All methods select 10% of the training data (20% for code domain).

C Correlation Analysis Metrics

We systematically evaluate how our core selection indicator, ΔH, relates to other model and data properties. Figure 4 summarizes all correlation results. All metrics considered are as follows:

Base Model Metrics (computed using the initial model π_{base}):

- **Base NLL** ($\mathcal{L}_{\text{base}}$): average negative log-likelihood per token
- **Base Entropy** (H_{base}): average per-token output entropy (see Section 3.2)

Calibrated Model Metrics (computed after warmup fine-tuning, using π_{inst}):

- **Calib NLL** ($\mathcal{L}_{\text{inst}}$)
- **Calib Entropy** (H_{inst})

Difference Metrics (model-state gap, selection signal):

- **NLL Difference** (ΔNLL): change in NLL (Equation (1))
- **Contrastive Entropy** (ΔH): entropy change between model states (Equation (2))

Auxiliary Statistics:

- **Instruction length** (Inst Len)
- **Response length** (Resp Len)
- **Length ratios**: Resp/Inst, Inst/Resp

Complexity Proxies:

- **Code:** XCoder (Wang et al., 2024) (code complexity/correctness)
- **Math:** pass@k rates (difficulty proxy)
- **General/Medical:** Instagger (Lu et al., 2023) tag counts (instruction complexity)

Across all domains, we observe that the contrastive entropy ΔH is only weakly correlated with the above complexity proxies (see Figure 4). This supports that ΔH mainly captures model alignment and learnability rather than surface-level task difficulty.

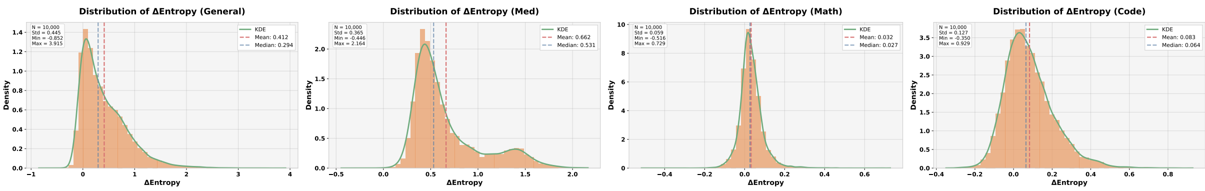

Figure 6: **Domain-adaptive entropy dynamics reveal distinct learning patterns.** We measure contrastive entropy (ΔH) between base and calibrated instruction-tuned models across four domains. General instruction-following and medical QA exhibit entropy *decrease* (**cognitive compression**), while mathematical reasoning and code generation exhibit entropy *increase* (**cognitive expansion**). This domain-dependent pattern motivates our unified selection principle based on contrastive entropy.

D Training Hyperparameters

Training and Evaluation. All models use full-parameter supervised fine-tuning with AdamW optimizer. For `InstructDiff`, we set warmup ratio $\alpha = 0.1$, bi-directional reject ratio $\gamma = 0.1$, and selection ratio $\beta = 0.1$ except for code, where $\beta = 0.2$. We evaluate mathematics on Math500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), OlympiadBench (AI Mathematical Olympiad, 2024), AIME 2024 (American Institute of Mathematics, 2024), and AMC 2023 (Mathematical Association of America, 2023) using accuracy; general instruction-following on Alpaca-Eval (Li et al., 2023b) using length-controlled win rate (LC Win) and raw win rate (WR); medical QA on MMLU-medical (Hendrycks et al., 2020), MedQA (Jin et al., 2021), and MedMCQA (Pal et al., 2022) using accuracy; and code generation on HumanEval/+ (Chen et al., 2021), MBPP/+ (Austin et al., 2021), and BigCodeBench (Zhuo et al., 2024) using pass@k. Training hyperparameters are in Appendix D.

Table 11 summarizes the training hyperparameters used across all domains.

Hyperparameter	Math	General	Medical	Code
Learning Rate	5e-5	2e-5	2e-5	1e-5
Batch Size	256	64	64	256
Epochs	3	10	3	5
Warmup Ratio	0.05	0.05	0.05	0.05
Weight Decay	0.01	0.01	0.01	0.01
Max Seq Length	2048	2048	512	4096

Table 11: Training hyperparameters for different domains.

All experiments use full-parameter fine-tuning (not LoRA) with AdamW optimizer, cosine learning rate decay, and gradient clipping at 1.0. Training is conducted on 8×A100 (80GB) GPUs with mixed precision (bf16).

E Decoupling Contrastive Entropy Metric from Iterative Refinement

To verify that performance gains stem from the proposed contrastive entropy metric rather than merely from iterative wrapping, Table 12 compares IFD and `InstructDiff` under single-pass and iterative settings (10% data budget, General and Math domains).

Method	General (Avg)	Math (Avg)
IFD (Single-pass)	6.52	21.99
IFD + Iter-1	7.23	21.91
IFD + Iter-2	6.55 (↓)	20.68 (↓)
`InstructDiff` (Single-pass)	5.98	27.93
`InstructDiff` (Full/Iterative)	**7.43**	**31.63**

Table 12: **Decoupling contrastive entropy metric from iterative refinement.** Applying iteration to IFD yields unstable results—gains on General tasks disappear at Iter-2, and Math degrades continuously. By contrast, `InstructDiff`'s contrastive entropy metric already outperforms IFD by a large margin at single-pass, and iteration further amplifies these gains. The two components are synergistic, not confounding.

Applying the iterative strategy to IFD yields unstable results: Iter-1 provides a temporary boost on General (7.23), but further iteration causes degradation (Iter-2: 6.55). On Math, iteration causes a continuous drop (21.99 → 21.91 → 20.68). By contrast, `InstructDiff` (Single-pass) already achieves 27.93 on Math, vastly outperforming IFD's 21.99. Coupling with iteration then unlocks its full potential (31.63). This confirms that the contrastive entropy metric and iterative mechanism are deeply synergistic: iteration alone (without the right metric) is harmful, while the metric alone is already highly effective.

F Effect of Selection Ratio and Isolating Contrastive Entropy

Table 13 examines how performance changes when progressively more data is selected, and isolates the contribution of the contrastive entropy criterion by comparing with NLL filtering alone (`InstructDiff` 100%, which applies bi-directional filtering but uses all remaining data without entropy-based bottleneck).

Domain	Method / Ratio	LC Win	Arena	Avg
General	Base (0%)	1.36	0.26	0.81
	ALL (100%)	8.15	1.73	4.94
	`InstructDiff` 10%	**12.09**	**2.76**	**7.42**
	`InstructDiff` 20%	8.95	2.29	5.62
	`InstructDiff` 50%	10.38	2.84	6.61
	`InstructDiff` 100% (NLL-only)	7.27	2.57	4.92
			Avg	
Math	Base (0%)		12.61	
	ALL (100%)		27.05	
	`InstructDiff` 10%		**31.63**	
	`InstructDiff` 20%		25.71	
	`InstructDiff` 50%		26.35	
	`InstructDiff` 100% (NLL-only)		26.39	

Table 13: **Effect of selection ratio and isolation of contrastive entropy contribution.** Performance peaks at 10% selection. The "100% (NLL-only)" row applies bi-directional NLL filtering but skips entropy-based selection; it performs comparably to full-data training, confirming ΔH-based selection as the primary driver of improvement.

Two key findings emerge. First, **10% selection is the optimal ratio**: expanding to 20% or 50% reduces gains; 100% (NLL filtering only) collapses back to the full-data baseline, confirming that gains arise from isolating the learning frontier rather than retaining volume. Second, **NLL filtering alone does not improve over full data**: `InstructDiff` 100% achieves 4.92 (General) and 26.39 (Math), comparable to ALL (100%) baselines of 4.94 and 27.05. The major gains (7.42 and 31.63) emerge exclusively after applying contrastive-entropy-based selection.

G Computational Efficiency

Table 14 reports total wall-clock time (in minutes) for `InstructDiff` versus full-data training at each iteration. The "Ours" time includes *all* overhead: training the calibration model (warm-up) and computing ΔNLL and ΔH for all samples. Math uses 100k samples from NuminaMath; General uses 52k samples from Alpaca.

Under the default setting ($T = 1$ or $T = 2$), `InstructDiff` provides a clear computational advantage over full-data training. Furthermore, using a smaller same-family model for calibration (Section 5.4) can further reduce the time cost by up to 14× based on parameter count ratios.

Iter	Dataset	Ours (min)	Full Data (min)	Ratio
$T = 1$	General	86	202	42.6%
	Math	166	421	39.4%
$T = 2$	General	129	202	63.9%
	Math	249	421	59.1%
$T = 3$	General	172	202	85.1%
	Math	332	421	78.9%

Table 14: **Wall-clock time comparison (minutes).** `InstructDiff` with $T = 1$ or $T = 2$ requires only 39–64% of full-data training time while substantially outperforming it. Time includes all overhead (warm-up training and scoring of all samples).

H Warm-up Size vs. Selection Size Decoupling

To investigate whether the warm-up size and the final selection size are systematically coupled (e.g., both fixed at 10%), we conduct a comprehensive grid search decoupling these two hyperparameters on the Math domain (Qwen2.5-7B, NuminaMath 10k).

Warmup	Select	AIME24	Math-OAI	Minerva	Olympiad	ACM23	Avg
10%	10%	**7.71**	**61.79**	21.42	**26.94**	**40.31**	**31.63**
10%	20%	3.74	57.73	14.20	22.86	30.00	25.71
10%	50%	1.86	58.48	15.48	23.92	32.03	26.35
20%	10%	7.51	61.36	**23.35**	26.98	36.88	31.22
20%	20%	6.68	57.61	16.77	22.59	34.83	27.70
20%	50%	3.76	59.36	15.96	23.62	33.59	27.26
50%	10%	6.25	60.68	21.14	25.81	34.06	29.59
50%	20%	4.79	58.66	17.74	24.08	35.00	28.05
50%	50%	3.95	59.09	17.64	23.78	30.63	27.02

Table 15: **Warm-up Size × Selection Size Decoupling (Math domain).** The two hyperparameters operate largely independently. A 10% selection size consistently yields the highest scores regardless of warm-up size, while a 10% warm-up (≈1k samples) is sufficient for stable calibration.

Three key insights emerge from this grid: (1) **Selection ratio dominates performance.** A 10% selection size consistently achieves the highest average scores across all warm-up sizes, confirming that concentrated low-entropy subsets are intrinsically more valuable. Expanding the selection size dilutes this advantage. (2) **The 10%/10% pairing is empirical, not structural.** The warm-up size and selection size are not inherently coupled. A 10% warm-up (≈1k samples) is sufficient to activate stable instruction-following and produce reliable ΔH measurements; increasing warm-up to 50% yields diminishing returns and slightly degrades peak performance (31.63 → 29.59). (3) **Larger selections benefit modestly from larger warm-ups.** When one must select a larger subset (e.g., 50%), a larger warm-up provides marginally better calibration (50%/50%: 27.02 vs. 10%/50%: 26.35). In practice, we recommend warm-up $\approx$ 1k samples (i.e., 10% of a 10k pool) as the default, independently of the target selection ratio.

I Algorithm Details

Algorithm 1 presents the complete pseudo-code for InstructDiff. The algorithm takes as input the full dataset $\mathcal{D}$, base model π_{base}, three hyperparameter ratios (α for warmup size, β for final selection size, γ for bi-directional filtering), and the number of iterations T. It returns the final selected subset $\mathcal{D}'^{(T)}$ of size $\beta \cdot N$.

The algorithm operates in two main stages. In Stage 1 (lines 1-2), we perform calibration by randomly sampling a warmup subset and fine-tuning the base model to obtain the initial measuring instrument $\pi_{\text{inst}}^{(0)}$. In Stage 2 (lines 3-11), we iterate T times, each iteration computing distributional gaps for all samples (lines 4-6), applying bi-directional NLL filtering to retain the learnable middle range (line 7), selecting samples based on contrastive entropy (line 8), and optionally updating the calibration model for the next iteration (lines 9-11).

Algorithm 1 InstructDiff: Iterative Contrastive-Entropy-Guided Selection

Require: Dataset $\mathcal{D}$, base model π_{base}, warmup ratio α, selection ratio β, bi-reject ratio γ, iterations T
Ensure: Selected subset $\mathcal{D}'$
1: // **Stage 1: Calibration**
2: $\mathcal{D}_{\text{warmup}} \leftarrow \text{RandomSample}(\mathcal{D}, \alpha \cdot N)$
3: $\pi_{\text{inst}}^{(0)} \leftarrow \text{FineTune}(\pi_{\text{base}}, \mathcal{D}_{\text{warmup}})$
4: // **Stage 2: Iterative Selection**
5: **for** $t = 1$ to T **do**
6: **for** $(x_i, y_i) \in \mathcal{D}$ **do**
7: Compute $\Delta\text{NLL}_i \leftarrow \text{NLL}_{\text{inst}}^{(t-1)}(x_i, y_i) - \text{NLL}_{\text{base}}(x_i, y_i)$
8: Compute $\Delta H_i \leftarrow H_{\text{base}}(x_i, y_i) - H_{\text{inst}}^{(t-1)}(x_i, y_i)$
9: **end for**
10: // Bi-directional NLL filtering
11: $\mathcal{D}_{\text{filtered}} \leftarrow \{(x_i, y_i) \in \mathcal{D} : q_\gamma \leq \Delta\text{NLL}_i \leq q_{1-\gamma}\}$
12: // Contrastive-entropy-based selection
13: $\mathcal{D}'^{(t)} \leftarrow \text{SelectLowest}(\mathcal{D}_{\text{filtered}}, \beta \cdot N, \text{by } \Delta H)$
14: // Update calibration model
15: **if** $t < T$ **then**
16: $\pi_{\text{inst}}^{(t)} \leftarrow \text{FineTune}(\pi_{\text{base}}, \mathcal{D}'^{(t)})$
17: **end if**
18: **end for**
19: **return** $\mathcal{D}'^{(T)}$

J Case Studies

We present representative selected and rejected samples in a unified visualization format to qualitatively illustrate our selection criteria. We additionally include reasoning (math) examples from the highest and lowest ΔH percentiles to visualize the distinct cognitive dynamics of expansion versus compression.

This sample has extremely high gaps, indicating it is beyond the model's current capability. Training on such samples introduces harmful noise.

Math Domain: Cognitive Expansion vs. Compression. The following examples contrast reasoning samples from the lowest and highest ΔH percentiles, illustrating why selecting the lowest ΔH (cognitive expansion) is beneficial for mathematical reasoning.

Selected Sample: Low ΔH, Moderate ΔNLL

Instruction
List five potential stakeholders that could be involved in the design of a new regional park.

Response
Some potential stakeholders that could be involved in the design of a new regional park are:
1. Local government agencies: These agencies are responsible for managing and developing public spaces, including parks, within their jurisdiction.
2. Community groups: Representatives from local community groups that can provide insight into the needs and concerns of the residents who will use the park.
3. Environmental organizations: Organizations that focus on environmental conservation and sustainability can provide valuable input on how to design the park in a way that benefits both the community and the environment.
4. Local business owners: Local businesses near the park could benefit from increased foot traffic and may have a vested interest in the park's design and success.
5. Landscape architects / urban planners: Professionals in these fields have expertise in designing functional and aesthetically pleasing public spaces.

Metrics
ΔNLL = 0.06 Δ H = 0.16 ΔPPL = 0.15

Figure 7: Example of a selected training sample with balanced uncertainty and learning signal

Rejected Sample: Excessively High ΔNLL

Instruction
Create an antonym for the word "Early". Output the antonym.

Response
Late.

Metrics
ΔNLL = 4.14 Δ H = 2.19 ΔPPL = 201.24

Figure 8: Rejected example due to excessively high NLL gap and unstable learning signal

Rejected Sample: Degenerate Low ΔNLL

Instruction
Output the longest palindrome in a sentence. This is sentence with a palindrome.

Response
a palindrome

Metrics
ΔNLL = -2.84 Δ H = 1.79 ΔPPL = -552.69

Figure 9: Rejected example with degenerate low NLL gap and limited training value

Math Selected Sample: Low ΔH (Cognitive Expansion, $\Delta H < 0$)

Instruction.
A plane has several seats on it, each with its own price, as shown below (attachment). $2n-2$ passengers wish to take this plane, but none of them wants to sit with any other passenger in the same column or row. The captain realizes that, no matter how he arranges the passengers, the total money he can collect is the same. Prove this fact, and compute how much money the captain can collect.

Response.

1. **Restate the problem in a more abstract form.** We have a grid of seats with prices, and we need to place $2n-2$ passengers such that no two passengers are in the same row or column. We need to prove that the total money collected is the same regardless of the arrangement and compute this total.
2. **Transform the problem.** Consider the grid as an $n \times n$ matrix where each cell (i, j) represents the price of the seat in the i-th row and j-th column. We need to show that the sum of the prices of the seats occupied by the passengers is invariant under any arrangement that satisfies the given constraints.
3. **Diagonal argument.** We can use the concept of the main diagonal of the matrix (from the bottom-left to the top-right). We will show that we can always rearrange the passengers such that they occupy the diagonal without changing the total sum.
4. **Operation definition.** Define an operation that swaps the positions of two passengers in the same column but different rows. Specifically, if we have passengers at (a, b) and (c, d), we can swap them to (a, d) and (c, b).
5. **Induction base case.** For the base case, consider the smallest grid where $n = 2$. The passengers can only occupy the diagonal positions $(1, 1)$ and $(2, 2)$ or $(1, 2)$ and $(2, 1)$. In both cases, the sum of the prices is the same.
6. **Induction hypothesis.** Assume that for an $n \times n$ grid, we can always rearrange the passengers to occupy the diagonal without changing the total sum.
7. **Induction step.** For an $(n+1) \times (n+1)$ grid, assume we have already arranged the first n columns such that the passengers occupy the diagonal. For the $(n+1)$-th column, if the passenger is already on the diagonal, we are done. If not, the passenger must be below the diagonal (since being above would contradict the arrangement of the first n columns). We can then use the defined operation to swap this passenger with another passenger in the same row but different column, ensuring the passenger moves to the diagonal.
8. **Conclusion.** By induction, we can always rearrange the passengers to occupy the diagonal, and since the sum of the prices on the diagonal is invariant, the total sum collected is the same for any valid arrangement.
9. **Compute the total sum.** The sum of the prices on the diagonal of an $n \times n$ grid is given by the sum of the first n natural numbers,
$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}.$$
However, since we have $2n-2$ passengers, we need to adjust this sum accordingly. The total sum collected is
$$\sum_{i=1}^{n-1} i + \sum_{i=1}^{n-1} (n-i) = \sum_{i=1}^{n-1} i + \sum_{i=1}^{n-1} i = 2\sum_{i=1}^{n-1} i = 2 \cdot \frac{(n-1)n}{2} = (n-1)n.$$
Therefore, the final answer is $\boxed{(n-1)n}$.

Metrics.
$\Delta\text{NLL} = -0.04$ $\Delta H = -0.21$ (entropy *increases* after calibration)

Figure 10: Math reasoning sample with low ΔH (cognitive expansion): selected by `InstructDiff` for encouraging solution path diversity.

Math Rejected Sample: High ΔH (Cognitive Compression)

Instruction.
Compute $\sin 30^\circ$.

Response.
Let Q be the point on the unit circle that is 30° counterclockwise from $(1, 0)$. Draw segment QE from Q to the x-axis perpendicular to the axis, forming a right triangle OQE, where O is the origin.

Using the properties of a $30^\circ - 60^\circ - 90^\circ$ triangle, the sides opposite the 30°, 60°, and 90° angles are in the ratio $1 : \sqrt{3} : 2$. Since the hypotenuse OQ is 1 (the unit circle radius), the side opposite the 30° angle, namely OE, must be $\frac{1}{2}$, and the side opposite the 60° angle, namely EQ, must be $\frac{\sqrt{3}}{2}$.
Therefore, since $\sin 30^\circ$ is the y-coordinate of Q, we have

$$\sin 30^\circ = \boxed{\frac{1}{2}}.$$

Metrics.
$\Delta\text{NLL} = 0.57$ $\quad \Delta H = 0.41$ $\quad$ (entropy *massively decreases* after calibration)

Figure 11: Math reasoning sample with high ΔH (cognitive compression): rejected by `InstructDiff` as it promotes memorization over reasoning path diversity.