Teaching According to Talents!
Instruction Tuning LLMs with Competence-Aware Curriculum Learning

Yangning Li[*], Tingwei Lu[*], Yinghui Li, Yankai Chen, Wei-Chieh Huang,
Wenhao Jiang, Hui Wang[‡], Hai-Tao Zheng[‡], Philip S. Yu
[1]Shenzhen International Graduate School, Tsinghua University, [2]Peng Cheng Laboratory
[3]University of Illinois Chicago, [4]Guangming Laboratory

Abstract

Efficient instruction tuning aims to enhance the ultimate performance of large language models (LLMs) trained on a given instruction dataset. Curriculum learning as a typical data organization strategy has shown preliminary effectiveness in instruction tuning. However, current curriculum tuning methods suffer from the curriculum rigidity, since they rely solely on static heuristic difficulty metrics. These methods fail to adapt to the evolving capabilities of models during training, resulting in a fixed and potentially sub-optimal learning trajectory. To address the issue, **C**ompetence-**A**ware **M**ulti-**P**erspective c**U**rriculum in**S**truction tuning framework termed **CAMPUS** is proposed. CAMPUS offers several advantages: (1) Dynamic selection for sub-curriculum. (2) Competency-aware adjustment to the curriculum schedule. (3) Multiple difficulty-based scheduling. Extensive experiments prove the superior performance of CAMPUS, compared to other state-of-the-art baselines for efficient instruction tuning.

1 Introduction

Instruction tuning (Zhang et al., 2023; Zhao et al., 2023) aligns large language models (LLMs) (Touvron et al., 2023; Bai et al., 2023) with human preferences, enhancing their effectiveness across real-world tasks. Some studies (Zhou et al., 2024; Wang et al., 2023a) have highlighted the importance of instruction data management, including data quality curation and training strategy. This has fueled interest in *efficient instruction tuning* (Wang et al., 2024, 2023b), which aims **to maximize LLM performance trained on a given instruction dataset**. Among various strategies, optimizing training order has emerged as a key research focus.

Curriculum learning (Bengio et al., 2009), as a typical data order organization strategy that mimics the learning process of human education, has shown preliminary effectiveness in instruction tuning. The essential idea is to train LLMs with instruction data in a progression from easy to difficult, thereby accelerating model convergence and achieving a higher performance upper bound. The key to curriculum learning lies in the design of difficulty metrics for instruction data. For instance, Tree-Instruct (Zhao et al., 2024) treats instruction data as a semantic tree and measures difficulty based on the number of tree nodes. Referring to the expert-designed educational framework, Lee et al. (2024) utilizes ChatGPT to synthesize the instruction dataset CORGI with increasing difficulty, and then train the LLMs sequentially.

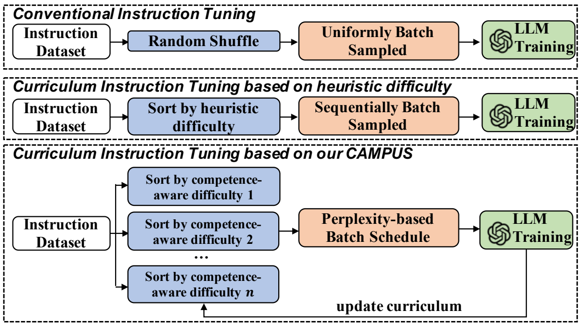

Conventional Instruction Tuning
Instruction Dataset
Random Shuffle
Uniformly Batch Sampled
LLM Training
Curriculum Instruction Tuning based on heuristic difficulty
Instruction Dataset
Sort by heuristic difficulty
Sequentially Batch Sampled
LLM Training
Curriculum Instruction Tuning based on our CAMPUS
Instruction Dataset
Sort by competence-aware difficulty 1
Sort by competence-aware difficulty 2
...
Sort by competence-aware difficulty n
Perplexity-based Batch Schedule
LLM Training
update curriculum

Figure 1: Comparison of different curriculum instruction tuning.

Despite these advancements, **previous difficulty metrics are often static and predefined with human intuition**. These metrics fail to gauge the difficulty variations of the same data for different LLMs or at different learning stages. Consequently, as shown in Figure 1, curriculum instruction tuning based on them lacks the flexibility to adjust curriculum for different LLMs or as LLMs evolve. The rigid curriculum schedule result in a sub-optimization in instruction tuning. As educator John Dewey said, “if we teach today’s students as we taught yesterday’s, we rob them of tomorrow," underscoring the necessity of adaptable learning strategies in human cognition. However, it remains

---

[*] Equal Contribution. ‡ Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 11724–11741
November 4-9, 2025 ©2025 Association for Computational Linguistics

unclear for us how to customize a "suitable" instruction curriculum for a specific LLM.

In light of the above issues, we seek to develop a data order organization strategy that customizes a dynamic curriculum for different LLMs or LLM at different training phases, based on which we can improve the training efficiency of instruction dataset. To this end, a **C**ompetence-**A**ware **M**ulti-**P**erspective c**U**rriculum in**S**truction tuning framework termed **CAMPUS** is proposed. Initially, CAMPUS independently sorts instruction data using multiple difficulty metrics, each representing a curriculum schedule developed from distinct perspectives. At each training step, CAMPUS assesses the LLM competence and inherent data difficulty, selecting the sub-curriculum from the various schedules that poses the minimum perplexity, i.e., sub-curriculum that can be relatively easily comprehended and internalized by current LLM.

CAMPUS is compatible with arbitrary quantities of difficulties. Besides the conventional heuristic difficulty, we introduce competence-aware difficulty based on a lightweight scoring model, which is trained with adversarial learning. It considers both LLM parameters and instruction data as joint inputs to assess the instruction data's difficulty for the current LLM. As the LLM's training progresses, the competence-aware difficulty consistently evolves, allowing the curriculum based on the reward model to dynamically adjust.

Compared to existing curriculum instruction tuning methods, CAMPUS offers the following advantages: (1) **Dynamic Selection for Sub-Curriculum**: It continuously evaluates the learning state and progress of the model, selecting the most appropriate sub-curriculum to match the current learning needs. (2) **Competency-Aware Adjustment to the Curriculum Schedule**: CAMPUS adjusts the curriculum schedule based on real-time assessments of the model's competencies, ensuring that the curriculum aligns with the model's evolving strengths and weaknesses. (3) **Multiple Difficulty-Based Scheduling**: Unlike methods that rely on a single static metric, CAMPUS utilizes multiple difficulty metric, holding a more comprehensive perspective of the data difficulty.

LLMs from different families are equipped with CAMPUS to demonstrate their effectiveness, including BLOOMZ and LLaMA. In experiments, we utilized a mixed instruction dataset derived from Code Alpaca, GSM8K, and ShareGPT as the original training dataset D. Various types of efficient instruction methods, including data selection and training order optimization, are served as baseline models to enhance training efficiency with D. We conduct an extensive evaluation on three benchmarks: GSM8K for mathematical reasoning, HumanEval for coding, and MT-Bench for general language understanding. CAMPUS consistently outperforms state-of-the-art methods by an average of 7.0%. Additionally, as a pluggable training strategy, CAMPUS can be integrated with other data selection methods to further enhance the efficiency of instruction tuning.

2 Related Work

2.1 Efficient Instruction Tuning

Instruction Tuning (Zhang et al., 2023; Zhao et al., 2023) aims to align the pre-trained LLM abilities towards human preference, which enables LLMs to quickly adapt to specific domains or acquire specialized skills. Some recent research (Zhou et al., 2024; Wang et al., 2023a) reveals that only a fairly small amount of high-quality data is needed to align large models well with human preferences, in contrast to traditional task-specific fine-tuning (Kenton and Toutanova, 2019) or pre-training (Brown et al., 2020) where data quantity is crucial. This finding has motivated the AI research community to dedicate on *efficient instruction tuning* (Wang et al., 2024, 2023b), including data and training efficiency. The common objective of various methods is to **enhance the ultimate performance of LLMs trained on a given instruction dataset** D.

The essence of the data-efficient instruction tuning (Li et al., 2024; Dong et al., 2024; Liu et al., 2024) lies in selecting a high-quality subset $D_{sub} \subseteq D$ from original datasets. Previous work filtered data from three dimensions: quality, diversity, and complexity. For quality assessment (Chen et al.), powerful LLMs such as ChatGPT or fine-tuned LLMs are frequently employed as quality evaluators. Diversity (Wu et al., 2023) is often measured using heuristic metrics like ROUGE-L similarity. Additionally, some works also endeavor to quantify and evaluate instruction complexity. For instance, #InsTag (Lu et al.) proposes to quantify instruction complexity using the number of fine-grained tags generated by ChatGPT. DEITA (Liu et al., 2024) introduces a comprehensive method that simultaneously considers all three aspects, enabling automatic data selection.

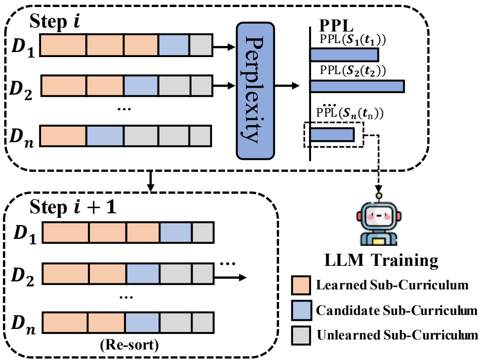


Figure 2: Framework of our CAMPUS to demonstrate how to schedule the sub-curricula.

2.2 Curriculum Instruction Tuning

A representative way for training-efficient instruction tuning is to organize the order of training data. Curriculum learning (Bengio et al., 2009), as a data organization strategy that emulates the human brain's learning process, has demonstrated effectiveness in instruction tuning. Its core principle involves arranging learning data from simple to hard, and the key is the design of difficulty metric. Zhao et al. (2024) and Sun et al. (2024) have employed the number of nodes in semantic trees and evaluations by ChatGPT, respectively, to gauge difficulty and create learning schedules. Lee et al. (2024) leverages expert-designed educational frameworks as basic curriculum outline and then synthesizes easy-to-hard instruction dataset CORGI. However, these rigid heuristic difficulties hindered dynamic adjustment of curriculum schedules.

3 Method

3.1 Overall Framework

Previous curriculum instruction tuning typically relied on static difficulty metrics, which limited their ability to assess the difficulty variations of the same data for different LLMs, and also hindered dynamic adjustment of learning schedules based on LLMs' mastery of the training data during the learning processes. To address these issues, we propose a **C**ompetence-**A**ware **M**ulti-**P**erspective c**U**rriculum in**S**truction tuning framework, termed **CAMPUS**. The overall framework is illustrated in Figure 2. CAMPUS first sorts the instruction dataset individually based on n difficulty metrics, and then dynamically programs the training batch during the training process utilizing a perplexity-based curriculum scheduler. As for the difficulty metrics, in addition to the heuristic metrics as in past work, we introduce novel model competence-aware metrics. Besides the inherent data difficulty, these metrics also account for LLMs' proficiency in data and their learning capacity, enabling flexible adjustments to the curriculum schedule. The details of curriculum scheduler and metrics will be provided in Section 3.2 and 3.3.

Algorithm 1 Overall process of **CAMPUS**.

```
Input: Training instruction dataset $D$, $i \in \{1,2,3,4\}$.
Output: A LLM with CAMPUS learning.
Compute four difficulties $d_i$ for each data sample in $D$;
Initial Sort $D$ based each difficulty of every sample, resulting in $D_i$ (i.e., $D_1$, $D_2$, $D_3$, $D_4$);
for $i = 1,2,3,4$ do
    $t_i = 1$; Initialize the learning scope from $i^{th}$ perspective, $s_i(1)$, by Eq. (1);
    Compute the perplexity (PPL) on top $s_i(1)$ portion, PPL($S_i(1)$), by Eq. (2);
end for
repeat
    $j = \arg\max_i(\mathrm{PPL}(S_i(t_i)))$;
    Train the LLM with the $S_j(t_j)$;
    if $j^{th}$ difficulty is competence-aware then
        Re-sort the curriculum order after $s_j(t_j)$;
    end if
    $t_j \leftarrow t_j + 1$;
    Update the learning scope from $j^{th}$ perspective, $S_j(t_j)$, by Eq. (1); Update the candidate training batch from the scope $S_j(t_j)$ portions of $D_j$;
    Compute the PPL of LLM on $S_j(t_j)$ portion , PPL($S_j(t_j)$), by Eq. (2);
until LLM converges.
```

3.2 Dynamic Curriculum Scheduler

For the given n difficulties, CAMPUS separately organizes the original instruction dataset into n easy-to-hard curriculum schedules $\{D_1, D_2, \ldots, D_n\}$, which can be considered as designed from distinct pedagogical perspectives. The primary objective is to intricately devise a sub-curriculum program strategy across these schedules to enhance the training of LLM, tailored to the LLM capabilities and sub-curriculum complexity. A straightforward method might involve selecting sub-curricula in a sequential format (i.e., $1 \to 2 \to \ldots \to n \to 1$), with the expectation that the overall curriculum order would still adhere to a progressive difficulty gradient. However, this method proves ineffective as it does not account for the incomparability of difficulty across heterogeneous measurement scales of different schedules. Hence, a dynamic curriculum scheduler that leverages perplexity metrics is proposed. First, we define the learning scope $s(t)$ for a particular curriculum schedule at training step t as follows:

$$s(t) = \min(1, \sqrt[p]{t\frac{1-s(1)^p}{T} + s(1)^p}), t = 2, 3, ..., T \quad (1)$$

where $s(1) = 0.01$ represents the initial learning scope, $p = 2$ controls the curriculum progression rate, and T represents the total training steps. Each curriculum D_i is segmented into T sub-curricula. The sub-curriculum range $S_i(t_j)$ at step t_j is denoted as by $[s(t_{j-1}), s(t_j)]$, which is utilized as a training batch. Given that $s(t)$ is an exponential function proportional to $\sqrt[p]{\frac{t}{T}}$, its growth decelerates with increasing t. In other words, the learning granularity becomes finer as difficulty of curriculum escalates, akin to human educational practices. For each candidate training batch (sub-curriculum) derived from these schedules, the scheduler calculates the perplexity (PPL), defined as:

$$\mathrm{PPL}(S_i(t_i)) = \frac{\sum_{X \in S_i(t_i)} \sqrt[N]{\prod_{m=1}^{N} \frac{1}{P(w_m|w_1,w_2,\ldots,w_{m-1})}}}{|S_i(t_i)|} \tag{2}$$

where $X = \{w_1, w_2, ..., w_N\}$ represents a data sample in $S_i(t_i)$. The PPL measures the model's confusion with real data samples. A training sample with a higher PPL indicates that the LLM has not mastered the basic ability to comprehend it and needs deferred learning. Consequently, the scheduler dynamically selects the training batch with the minimum PPL among the candidates, maintaining an orderly progression of overall difficulty. After that, the curriculum order is updated if the selected metric is competence-aware, and the process iteratively proceeds to the next sub-curriculum selection. Algorithm 1 presents the entire process.

3.3 Competence-Aware Difficulty Metrics

In our experiments, we employed four difficulty metrics (i.e., n=4), of which loss and reward score are competence-aware. These metrics not only assess the inherent difficulty of the instruction data but also adjust according to the LLM's proficiency with the skills associated with the instruction data. In other words, they serve as difficulty indicator functions $d_i = f_i\left(\mathrm{inst}, \theta_{\mathrm{LLM}}\right)$ with data and model parameter as inputs. Additionally, we introduced heuristic difficulty metrics data length and textual lexical diversity, considering their empirical effectiveness. **Each metric is viewed as capturing the curriculum from a different perspective, thereby reducing the potential bias that could arise from relying on a single metric.** More meaningful metrics can be integrated into CAMPUS in the future.

Data Length d_1 The data length is well-suited as a heuristic difficulty metric, aligning with intuitive human assessments: shorter instruction data are typically easier for LLMs to comprehend and learn. Consequently, we employed the total length of the instruction and output tokens, denoted as $\mathrm{len}(x + y)$, as d_1.

Measure of textual lexical diversity (MTLD) d_2 Lexical diversity refers to the variety of different words employed in a text, which serves as a crucial factor in studies of linguistic complexity. MTLD (McCarthy and Jarvis, 2010) is a widely used quantitative measure for assessing lexical diversity. It is calculated as the mean length of sequential token lists in a text that maintains a given type-token ratio threshold. We selected MTLD as a metric due to its length-insensitivity, since text length does not always accurately reflect the data complexity, e.g., redundancies. MTLD can be considered as a complementary perspective on instruction complexity that is distinct from data length.

Data Loss d_3 The data difficulty as perceived by LLMs depends not only on the data's inherent complexity, but also on the LLMs' comprehension abilities. Therefore, in addition to the two heuristics mentioned above, we introduce a simple, model competence-aware difficulty metric, data loss, as d_3. Data loss indirectly quantifies problem-solving capacity of LLM by measuring the difference between the predicted content and the ground-truth value, which is defined as:

$$d_3 = L(x, y, \theta_{LLM}) = -\sum_{t=1}^{N} logp_{\theta_{LLM}}(y_t|x, y_{<t})$$

where N is the length of the output text sequence and θ_{LLM} is the LLM parameter. A higher loss suggests that LLM struggles with tasks related to specific instruction data.

Competence-aware Data Score d_4 When assessing the data difficulty under specific model, data loss typically focuses solely on the difference between the model's output and the actual results, often neglecting the inherent data complexity and inner state of model. Consequently, we propose a model competent-aware scoring model R that takes both the data and model state as joint input, and outputs the data difficulty, denoted as d_4. To train the model R, we assign labels to each piece of instruction data. Empirically, models typically perceive data they have been trained on as straightforward, labeling these as 0; conversely, data they have not encountered are considered challenging, thus labeled as 1.

We commence by randomly shuffling the dataset

D (We used the same training data in Appendix B.1 to train the scoring model) and equally dividing it into n portions, $D_1, D_2, \dots, D_n$. At the ith epoch of training R, we choose the dataset pair (D_i, D_{i+1}) to process. We update the LLM parameters from the last epoch using dataset D_i:

$$\theta_{LLM}^{(i)} = SFT_{x \in D_i}(\theta_{LLM}^{(i-1)}), i = 1, 2, \dots, n \qquad (3)$$

where $\theta_{LLM}^{(0)} = \theta_{LLM}$. At the ith epoch, each data in D_i forms a training instance with $\theta_{LLM}^{(i)}$ for model R, and is assigned with a difficulty score label of 0 since $\theta_{LLM}^{(i)}$ has mastered it. Correspondingly, data in D_{i+1} is given a label of 1.

We intend to employ model R to leverage both data and LLM's model feature. For input data $x \in [D_i, D_{i+1}]$, the initial model parameter θ_{LLM} is utilized to obtain data feature z_1. To acquire model features, we apply a trainable embedding matrix with Kaiming initialization, serving as an initial semantic segment. By processing the semantics through $\theta_{LLM}^{(i)}$, model feature z_2 is obtained. Then, two features are concatenated to form a hidden feature $z = \mathrm{concat}(z_1, z_2)$. This process is succinctly expressed as $H(x, \theta) = z$. Subsequently, we use another MLP structure to transform z into a quantifiable difficulty score, that is, $R(x, \theta) = \mathrm{MLP}(z) = \mathrm{MLP}(H(x, \theta))$. The corresponding cross-entropy loss is calculated as:

$$\min_R L_R = E_{x \in D_i}[logR(x, \theta)] + E_{x \in D_{i+1}}[log(1 - R(x, \theta))] \qquad (4)$$

Additionally, we introduce an adversarial training mechanism to further enhance generalization of model R. In this process, discriminator D engages in a rivalry with model R, progressively refining the capability of model R in assessing data difficulty. Specifically, model R tries to fool the discriminator to correctly predict whether the instance is easy. While the discriminator D with a two-layer MLP structure is adversarially trained to accurately approximate the true difficulty label. Therefore, the training objectives for model R and the discriminator D are symmetrically formulated as:

$$\max_R L_R = E_{x \in D_i}[logR(x, \theta)] + E_{x \in D_{i+1}}[logR(x, \theta)] \qquad (5)$$

$$\begin{aligned} \min_D L_D = {} & E_{x \in D_i}[logD(H(x, \theta)] \\ & + E_{x \in D_{i+1}}[log(1 - D(H(x, \theta))] \end{aligned} \qquad (6)$$

4 Experiments

4.1 Experiment Setting

Training Dataset To intimate the educational scenario in which students simultaneously engage with multiple science curricula, we constructed a comprehensive instruction dataset for training, including mathematical reasoning, code generation, and general language understanding. They are sourced from 3 instruction datasets: training set of **GSM8K** (Cobbe et al., 2021), **Code Alpaca** (Chaudhary, 2023), and **ShareGPT**[1] (Chiang et al., 2023).

Evaluation Benchmark and Metrics We assess the aforementioned three capabilities using the following benchmarks and metrics: evaluation set of **GSM8K**, **HumanEval** (Chen et al., 2021), and **MT-Bench** (Zheng et al., 2023). The details about evaluation metrics are in Appendix B.

Baselines We compare three categories of baseline methods, and all baselines used the same training dataset. The first category is efficient tuning methods based on ***data selection***, including **IFD** (Li et al., 2024) and **DEITA** (Liu et al., 2024), in which DEITA represents the state-of-the-art methods. The second category comprises efficient tuning methods based on ***training order optimization***. In addition to **random shuffle tuning** and **sequential tuning** on a dataset-by-dataset basis, **Tree-Instruct** (Zhao et al., 2024) and **Conifer** (Sun et al., 2024)[2] are introduced as representatives of heuristic curriculum instruction tuning. **DMT** (Dong et al., 2024) is also a data order optimization baseline.

We also incorporated a third category of methods, which ***individually train LLMs*** on three separate training datasets. This approach is designed to explore the impact of capability conflicts (Dong et al., 2024) embodied in various curricula (datasets). Additionally, we aim to assess whether CAMPUS can potentially mitigate these conflicts by adjusting curricula with the awareness of model competence changes, compared to other baselines.

4.2 Main Experiments

We employ LLMs from the LLaMA (Touvron et al.) and BLOOMZ (Muennighoff et al., 2023) families as backbone models. The main results are shown in Table 1 and 2. To validate the robustness of our results, we include the standard deviation of the average performance. We can observe that:

(1) Our CAMPUS significantly outperforms other baselines, including efficient instruction tuning methods based on data selection and training order optimization. Taking LLaMA as an instance,

---

[1] `https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered`

[2] We only used the metrics in the two papers to measure the data difficulty in our experiments.

Backbone LLM	**LLaMA-7B**			**LLaMA-13B**			**Avg**
Dataset	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	
Individual Training							
Math only	35.56	-	-	**41.85**	-	-	-
Code only		11.59	-	-	12.20	-	-
General only	9.55	7.93	5.83	12.28	9.15	5.98	$26.17_{\pm 0.13}$
Data Selection							
IFD (Li et al., 2024)	33.43	13.40	5.69	37.62	15.24	5.78	$35.73_{\pm 0.07}$
DEITA (Liu et al., 2024)	33.21	14.02	5.73	37.83	15.24	5.88	$36.07_{\pm 0.15}$
Training Order Optimization							
Random Shuffle Tuning	35.03	14.63	5.63	40.11	15.24	5.76	$36.49_{\pm 0.11}$
Sequential Tuning	32.45	13.41	5.65	36.62	15.85	5.71	$35.32_{\pm 0.14}$
DMT (Dong et al., 2024)	33.81	14.02	5.75	38.14	15.85	5.83	$36.27_{\pm 0.19}$
Tree-Instruct (Zhao et al., 2024)	33.02	13.62	5.51	37.91	15.72	5.36	$34.83_{\pm 0.08}$
Conifer (Sun et al., 2024)	34.70	14.00	5.74	38.37	16.46	5.85	$36.57_{\pm 0.11}$
CAMPUS (ours)	**35.86**	**15.24**	**5.95**	40.56	**17.68**	**6.01**	**38.16**$_{\pm 0.09}$

Table 1: The average results of three times for LLaMA-7B, 13B on three benchmarks. Since MT-Bench scores are in tenths, we multiplied them by 10 when calculating overall performance averages in the last column (the same below). See Appendix D.4 for performance on other models such as LLaMA-33B.

Backbone LLM	**BLOOMZ-560M**			**BLOOMZ-1B7**			**BLOOMZ-3B**			**Avg**
Dataset	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	
Individual Training										
Math only	**7.81**	-	-	**12.66**	-	-	**15.24**	-	-	-
Code only	-	4.88	-	-	7.93	-	-	9.76	-	-
General only	0.68	3.05	3.26	1.36	3.66	3.78	2.33	6.10	4.11	$14.30_{\pm 0.17}$
Data Selection										
IFD (Li et al., 2024)	6.22	4.27	3.28	9.37	7.93	3.6	13.56	10.97	4.05	$17.96_{\pm 0.11}$
DEITA (Liu et al., 2024)	6.60	4.88	3.24	10.24	9.76	3.71	13.65	11.59	4.13	$18.61_{\pm 0.09}$
Training Order Optimization										
Random Shuffle Tuning	7.43	4.88	3.14	11.37	7.93	3.61	14.94	11.59	3.94	$18.34_{\pm 0.11}$
Sequential Tuning	6.37	4.88	3.15	9.55	8.54	3.65	13.42	11.59	4.09	$18.14_{\pm 0.11}$
DMT (Dong et al., 2024)	6.98	5.49	3.28	9.86	8.54	3.81	13.95	12.80	4.25	$19.00_{\pm 0.09}$
Tree-Instruct (Zhao et al., 2024)	6.19	4.39	3.14	9.41	8.23	3.74	14.03	11.49	4.09	$18.16_{\pm 0.13}$
Conifer (Sun et al., 2024)	6.79	4.88	3.28	9.89	8.54	3.8	13.83	11.59	4.16	$18.66_{\pm 0.10}$
CAMPUS (ours)	7.06	**6.10**	**3.43**	10.84	**10.37**	**4.13**	15.09	**13.41**	**4.46**	**20.34**$_{\pm 0.08}$

Table 2: The results of BLOOMZ-560M, 1B7, 3B on three benchmarks.

CAMPUS achieves an average gain of 7.0% over other curriculum instruction tuning baselines Tree-Instruct and Conifer, which rely on a single heuristic difficulty. The improvement can be attributed to two aspects: on the one hand, CAMPUS dynamically selects the most suitable sub-curriculum during the training process based on LLM capability and data difficulty. On the other hand, its difficulty metrics are also model competence-aware, enabling flexible adjustments to the whole curriculum schedule.

(2) It is noted that existing curriculum instruction tuning did not even outperform random shuffle tuning on partial benchmarks, despite performing well under specific conditions reported in their original studies. This underperformance demonstrates their limited generalization, primarily due to their reliance on static metrics. For example, Tree-Instruct utilized the number of nodes in semantic tree formed by instruction as the difficulty metric, which may not be applicable to code instruction data, as its difficulty is not necessarily related to the code length.

(3) We find that some baselines trained on the full training set exhibit varying degrees of "catastrophic forgetting" compared to individual training, which is caused by the capability conflicting of the different training sets. Especially in mathematical reasoning, since both training and test sets are derived from the same GSM8K benchmark, the mixture of multiple datasets results in a significant degradation in mathematical capability. In contrast, CAMPUS shows marginal performance degradation, and in some cases, surpasses the performance of training solely on math data. This demonstrates that CAMPUS is capable of mitigating such conflict by perceiving the model capability changes on the fly and dynamically planning the most "suitable" sub-curriculum for the current LLMs.

(4) Efficient instruction tuning based on data selection are originally intended to enhance the LLM performance by filtering out harmful data. However, these methods also employ heuristic algorithms, which rendering them less generalizable.

Backbone LLM	**LLaMA-7B**			**Avg**
Metrics	GSM8K	HumanEval	MT-Bench	
Random Shuffle Tuning	35.03	14.63	5.63	35.32
Length d_1	33.36	14.63	5.74	35.12
MTLD d_2	33.43	15.85	5.73	35.51
Loss d_3	34.72	14.02	5.69	35.21
Scoring Model d_4	35.48	15.85	5.91	36.82
CAMPUS	35.86	15.24	5.95	36.87

Table 3: Ablation experiments on difficulty metrics.

Backbone LLM	**LLaMA-7B**			**Avg**
Scheduler	GSM8K	HumanEval	MT-Bench	
Random	34.23	14.29	5.59	34.81
Sequential	34.26	14.37	5.54	34.68
PPL max	34.41	14.63	5.61	35.05
PPL min	35.86	15.24	5.95	36.87

Table 4: Comparison experiments on different sub-curriculum scheduling strategies.

In the experiments, they also did not outperform random shuffle tuning. Instead of improving the performance of the model with reduced training data, it turns into a sacrifice of performance for time-efficiency.

(5) CAMPUS leads to a more pronounced performance rise on larger LLMs compared to smaller ones. This also emphasizes the importance of efficient instruction tuning for nowadays LLMs with increasingly larger parameter sizes.

4.3 Ablation Experiments on CAMPUS

4.3.1 Ablation on Difficulty Metrics

Table 3 illustrates the results using a single difficulty metric, which is one of the difficulty metrics for the CAMPUS. The benefits from curriculum tuning based on single difficulties are relatively modest compared to the multiple difficulty-based CAMPUS. Static metrics, such as data length d_1, even have negative impacts on GSM8K and HumanEval, highlighting the limited generalization of conventional heuristic difficulties. This also confirms that data length is not a suitable metric for assessing the complexity of mathematical and coded data. Additionally, the scoring model we developed demonstrates promising performance, which is close to CAMPUS overall.

4.3.2 Comparison on Different Schedulers

We compare different sub-curriculum scheduling strategies in Table 4, including random selection, sequential selection from 1 to n, and both maximum and minimum PPL (used in CAMPUS). It is evident that selecting the sub-curriculum with the minimum PPL is the optimal scheduling strategy, which is in line with the curriculum learning principles and human intuition. Sub-curriculum with minimal perplexity implies that the LLM is primed to comprehend it, and thus the associated competencies of them can be more easily internalized by LLM. In contrast, sequential selection of sub-curriculum disrupts the ordered progression of the curriculum, as the difficulty of sub-curricula across different schedules cannot be directly compared. Consequently, the performance of sequential selection strategy is inferior even to random selection.

Backbone LLM	**LLaMA-7B**			**Avg**
Dataset	GSM8K	HumanEval	MT-Bench	
Full & Random	35.03	14.63	5.63	35.32
1/3 Easy	32.30	13.41	5.89	34.86
1/3 Medium	33.21	13.41	5.83	34.96
1/3 Hard	34.34	14.02	5.76	35.33
Full & CAMPUS	35.86	15.24	5.95	36.87

Table 5: Data difficulty level experiments.

4.3.3 Comparison on Data with Different Difficulty

Table 5 illustrates the impact of data with different difficulty levels. Specifically, we evenly divided the instruction data programmed by CAMPUS into three portions, and categorized them as easy, medium, and hard, according to their precedence order. The results indicate that no single data segment provides more significant gains to LLMs than CAMPUS trained with the complete data, underscoring that each portion contributes to the overall performance. Similar experiments are conducted in Tree-Instruct (Zhao et al., 2024), and its experiments show that the gain brought by the hard sub-dataset exceeds that of curriculum learning with the complete dataset. This contrary experimental conclusion is attributed to the sub-optimization of Tree-Instruct, which does not fully stimulate the potential of curriculum learning. Unlike Tree-Instruct, CAMPUS employs dynamic, multi-perspective difficulty metrics that customize the curriculum specifically to the competence and learning needs of the particular LLM. Of course, referring to the middle portion of Table 5, we can similarly conclude that harder data is more critical for LLMs, which aligns with the view of previous work (Xu et al., 2023; Lu et al.).

4.4 Generalization of CAMPUS

4.4.1 Compatibility with Other Methods

Our CAMPUS framework is designed to be compatible with various data selection-based efficient

Backbone LLM	LLaMA-7B				LLaMA-13B				Avg
Dataset	GSM8K	HumanEval	MT-Bench	Training Time (h)	GSM8K	HumanEval	MT-Bench	Training Time (h)	
Random Shuffle Tuning	35.03	14.63	5.63	47	40.11	15.24	5.76	53	36.49
IFD	33.43	13.40	5.69	37 + 8	37.62	15.24	5.78	37 + 10	35.73
+ CAMPUS	34.77	14.63	5.79	17	38.49	15.85	5.85	20	36.69
DEITA	33.21	14.02	5.73	41 + 4	37.83	15.24	5.88	41 + 5	36.07
+ CAMPUS	34.93	14.63	5.83	9	39.21	16.46	5.94	11	37.16
CAMPUS	35.86	15.24	5.95	54	40.56	17.68	6.01	64	38.16

Table 6: Experiments on our CAMPUS combined with other data selection-based efficient tuning methods. $x + y$ in training time for data selection methods means data selecting and LLM training time, respectively.

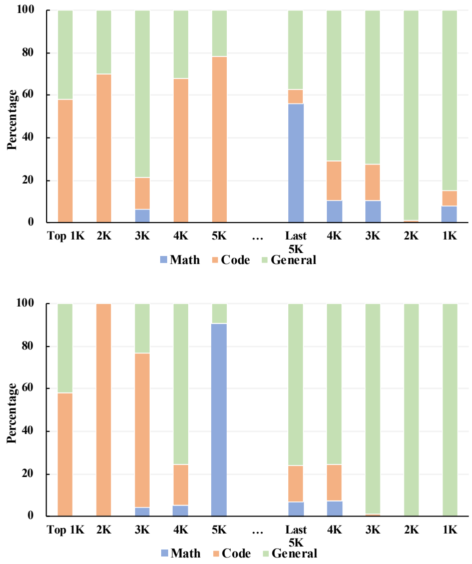


Figure 3: Data composition of first/last 5K data by CAMPUS. Top: LLaMA-7B. Bottom: LLaMA-13B.

tuning methods, further enhancing the performance of LLMs. To verify the effectiveness of CAMPUS combined with them, we first filter the original training set leveraging the data selection-based method, and then apply our CAMPUS to the refined dataset. The results in Table 6 demonstrate that CAMPUS further boosts the performance of both IFD and DEITA, confirming its excellent adaptability. Moreover, while the selection-based method reduces the training time for LLMs, it simultaneously requires considerable time for selecting. Interestingly, the performance gain from CAMPUS is more pronounced with DEITA, which retains more data compared to IFD. This observation suggests that DEITA and IFD may inadvertently discard potentially useful data. Inspired by this, we believe that future work could combine the ideas of several types of efficient tuning methods in a unified way, and propose a more comprehensive data management strategy, considering more aspects such as data order, ratio, and the criteria for data exclusion, etc. This could potentially maximize the synergistic effects of combining different tuning methods and further enhance LLM training efficacy.

4.5 Training Process of CAMPUS

4.5.1 Data Composition at Different Stages

Figure 3 reports the data type composition of the first and last 5K data scheduled by CAMPUS for different LLMs. Variations in data composition across different LLMs and training phases can be clearly observed. During the early stages of training, the code data proportion is relatively high, which may arise from the fact that LLMs require the underlying reasoning capabilities embodied in the code tasks, which is also indicated in some works (Bi et al., 2024; Cheng et al.). In contrast, the later stages of training witness an increase in math and general instruction data. Further case analysis in Table 16 and 17 in Appendix reveals that these later stages involve a significant number of mathematical tasks that require complex logical reasoning and intricate multi-round conversation data which proves that CAMPUS behaves in line with our intuition. As for the different LLMs, it is notable that math data are introduced earlier in the training process for larger LLMs (LLaMA-13B). Since larger LLMs possess more advanced foundational capabilities, allowing them to tackle complex mathematical tasks more readily and reducing their perceived difficulty more quickly. This phenomenon again confirms the CAMPUS's ability to perceive the model's competence and tailor the LLM-specific curriculum dynamically.

5 Conclusion

In this paper, to address the rigidity issue of existing curriculum instruction tuning methods, we propose a **C**ompetence-**A**ware **M**ulti-**P**erspective c**U**rriculum in**S**truction tuning framework termed **CAMPUS**. CAMPUS offers the following advantages: (1) dynamic selection for sub-curriculum. (2) competency-aware adjustment to the curriculum schedule. (3) multiple difficulty-based curriculum schedule. Extensive experiments prove the superior performance of CAMPUS, compared to state-of-the-art efficient instruction tuning methods.

Limitations

In this paper, we discuss efficient instruction tuning methods based on training order optimization, aiming to enable models to achieve higher performance using the same training data. The potential for combining with other types of efficient instruction tuning methods was primarily explored in our experiments. Specifically, we integrated our CAMPUS with the data selection based methods DEITA and IFD. The results demonstrate the feasibility of this idea, and future work could further explore how to generalize the ideas of different efficient instruction tuning methods in a unified way. For example, by drawing on the idea of reinforcement learning to view a data-wise management strategy as an action of agent.

Acknowledgment

This research is supported by National Natural Science Foundation of China(Grant No.62276154), Research Center for ComputerNetwork (Shenzhen) Ministry of Education, the Natural Science Foundation of Guangdong Province(Grant No.2023A1515012914 and 440300241033100801770), Basic Research Fund of Shenzhen City (Grant No.JCYJ20210324120012033, JCYJ20240813112009013 and GJHZ20240218113603006), the Major Key Projectof PCL for Experiments and Applications (PCL2023A09). This work is also supported in part by NSF under grants III-2106758, and POSE-2346158.

References

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

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Zhen Bi, Ningyu Zhang, Yinuo Jiang, Shumin Deng, Guozhou Zheng, and Huajun Chen. 2024. When do program-of-thought works for reasoning? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17691–17699.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. *GitHub repository*.

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 *The Twelfth International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, et al. Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

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

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 177–198, Bangkok, Thailand. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

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. *arXiv preprint arXiv:2103.03874*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota.

Bruce W Lee, Hyunsoo Cho, and Kang Min Yoo. 2024. Instruction tuning with human curriculum. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1281–1309.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024. 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 7595–7628.

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

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 *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Philip M McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, et al. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15991–16111.

Haoran Sun, Lixin Liu, Junjie Li, Fengyu Wang, Baohua Dong, Ran Lin, and Ruohui Huang. 2024. Conifer: Improving complex constrained instruction-following ability of large language models. *arXiv preprint arXiv:2404.02823*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. 2024. A survey on data selection for llm instruction tuning. *arXiv preprint arXiv:2402.05123*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023a. 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.

Zige Wang, Wanjun Zhong, Yufei Wang, Qi Zhu, Fei Mi, Baojun Wang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023b. Data management for large language models: A survey. *arXiv preprint arXiv:2312.01700*.

Shengguang Wu, Keming Lu, Benfeng Xu, Junyang Lin, Qi Su, and Chang Zhou. 2023. Self-evolved diverse data sampling for efficient instruction tuning. *arXiv preprint arXiv:2311.08182*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yingxiu Zhao, Bowen Yu, Binyuan Hui, Haiyang Yu, Minghao Li, Fei Huang, Nevin L Zhang, and Yongbin Li. 2024. Tree-instruct: A preliminary study of the intrinsic relationship between complexity and alignment. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language*

Resources and Evaluation (LREC-COLING 2024), pages 16776–16789.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

A Appendix for CAMPUS Method

A.1 Supplementary Description of Difficulty Metrics for CAMPUS

In our experiments, we employed four difficulty metrics (i.e., n=4), of which loss and reward score are competence-aware. These metrics not only assess the inherent difficulty of the instruction data but also adjust according to the LLM's proficiency with the skills associated with the instruction data. In other words, they serve as difficulty indicator functions $d_i = f_i\left(\text{inst}, \theta_{\text{LLM}}\right)$ with data and model parameter as inputs. Additionally, we introduced heuristic difficulty metrics data length and textual lexical diversity, considering their empirical effectiveness.

Data Length d_1 The data length is well-suited as a heuristic difficulty metric, aligning with intuitive human assessments: shorter instruction data are typically easier for LLMs to comprehend and learn. Conversely, longer instruction data often occurs in multi-round conversations, indicating that the instructions contain more diverse information and complex interactions requiring extended processing, thus making them more challenging to learn. Consequently, we employed the total length of the instruction and output tokens, denoted as $\text{len}(x + y)$, as d_1. From the perspective of data length, it is natural for LLMs to first master basic skills from simple single-round conversations, and then learn to comprehensively utilize these skills in complex multi-round conversations.

Measure of textual lexical diversity (MTLD) d_2 Lexical diversity refers to the variety of different words employed in a text, which serves as a crucial factor in studies of linguistic complexity. MTLD (McCarthy and Jarvis, 2010) is a widely used quantitative measure for assessing lexical diversity. It is calculated as the mean length of sequential token lists in a text that maintains a given type-token ratio (TTR) threshold (0.72 in our paper). The TTR is defined as $\frac{l}{m}$, where l and m are the token length and the number of unique token types, respectively. A higher MTLD indicates greater vocabulary diversity in instruction data, which may contain more advanced vocabulary, or varied word combinations to convey the same meanings, which is more difficult to be learned by the LLMs. We selected MTLD as a metric due to its length-insensitivity compared to traditional lexical diversity metrics such as TTR. Text length does not always accurately reflect the data complexity. For instance, in programming code, lengthy text may include blocks with duplicated functionality, leading to an artificially inflated complexity solely based on length. Therefore, MTLD can be considered as a complementary perspective on instruction complexity that is distinct from data length.

Data Loss d_3 The data difficulty as perceived by LLMs depends not only on the data's inherent complexity, but also on the LLMs' comprehension abilities. Therefore, in addition to the two heuristics mentioned above, we introduce a simple, model competence-aware difficulty metric, data loss, as d_3. Data loss indirectly quantifies problem-solving capacity of LLM by measuring the difference between the predicted content and the ground-truth value, which is defined as:

$$d_3 = L(x, y, \theta_{LLm}) = -\sum_{t=1}^{N} logp_{\theta_{LLM}}(y_t|x, y_{<t})$$

where N is the length of the output text sequence and θ_{LLm} is the LLM parameter. Predicting the next token y_t requires utilizing the input x and the preceding output sequence $y_{<t}$. A higher loss suggests that LLM struggles with tasks related to specific instruction data. The difficulty stems from either the intricate knowledge inherent in the data or the models' limited comprehension and reasoning capabilities. By monitoring the loss, we can orderly manage the instructional data.

B Appendix for Experiment Setting

B.1 Training Dataset

To imitate the educational scenario in which students simultaneously engage with multiple science curricula, we constructed a comprehensive instruction dataset for training, including mathematical reasoning, code generation, and general language understanding. They are sourced from the following instruction datasets, respectively:

- **GSM8K** (Cobbe et al., 2021) is a mathematical dataset consisting of ~8.5K high-quality grade school math problems that require multi-step reasoning. The dataset is divided into 6K training and 2K test instances.

- **Code Alpaca** (Chaudhary, 2023) aims to build an instruction-following LLaMA model for code generation, which contains 20K instruction-following data generated by the techniques in the Self-Instruct.

- **ShareGPT**[3] (Chiang et al., 2023) is a multi-round human-machine dialog dataset comprising 90K human queries and responses from ChatGPT or other chatbots. We cleaned it to reduce the size to 53K.

B.2 Evaluation Benchmark and Metrics

We assess the aforementioned three capabilities of LLMs using the following benchmarks and metrics:

- The test set of **GSM8K** is utilized to evaluate the mathematical reasoning ability of LLMs, which comprises 3K data. Following Yuan et al. (2023), we use the accuracy of answers generated by greedy decoding (i.e., maj@1) as our metrics.

- **HumanEval** (Chen et al., 2021) is the benchmark for code-writing with 164 handwritten programming problems, with an average of 7.7 tests per problem. Similar to (Chen et al., 2021), we measure pass@k on the HumanEval, where correctness is defined by an unbiased estimate of passing a set of unit tests within k samples.

- **MT-Bench** (Zheng et al., 2023) is a challenging benchmark widely adopted to assess the general instruction-following ability. It features multi-turn conversations across 8 domains. GPT-4 is employed as the judge to score model responses on a ten-point scale.

B.3 Baselines

We compare three categories of baseline methods. The first category encompasses efficient tuning methods based on ***data selection***. These methods primarily employ metrics to assess the benefit of raw instruction data for LLMs from perspectives of quality, diversity, and complexity, and then discarding harmful data. **IFD** (Li et al., 2024) and **DEITA** (Liu et al., 2024) are included, where DEITA represents the state-of-the-art methods. The second category comprises efficient tuning methods based on ***training order optimization***, which enhance the efficiency by adjusting the training order of original instruction data. In addition to **random shuffle tuning** and **sequential tuning** on a dataset-by-dataset basis, **Tree-Instruct** (Zhao et al., 2024) and **Conifer** (Sun et al., 2024)[4] are introduced as representatives of heuristic curriculum instruction tuning. **DMT** (Dong et al., 2024) is also a data order optimization baseline.

We also incorporated a third category of methods, which ***individually train LLMs*** on three separate training datasets. This approach is designed to explore the impact of capability conflicts (Dong et al., 2024) embodied in various curricula (datasets). Additionally, we aim to assess whether CAMPUS can potentially mitigate these conflicts by adjusting curricula with the awareness of model competence changes, compared to other baselines.

B.4 Training Details

The CAMPUS is implemented by LLaMA-Factory framework[5] and fine-tuned with full parameters. The temperature is configured to 0.95, and the top-k is set to 50, indicating that each generation step involves sampling from 50 candidate tokens. The learning rate is managed with a 0.03 warm-up ratio, ramping up linearly to 2e-5 during the initial 3% of the training data. Furthermore, we implement a cosine annealing strategy to adjust the learning rate in a cyclical manner, helping the model converge on the optimal solution.

When training the reward model R and the discriminator D, we choose a batch size of 4 and a learning rate of 1e-5. Both models utilize identical MLP structures to map the hidden feature z into a score, with a hidden dimension of 256. Employing the Kaiming initialization technique, they undergo continuous training through stochastic gradient descent (SGD), operating iteratively without parameter sharing. To accelerate convergence, we iterate twice for the training of R and D after each fine-tuning of the LLM.

C Supplementary Experiments on Generalization of CAMPUS

C.1 Performance Differences on Different Domains of MT-Bench

In Figure 4, we compare the performance of different models across the eight domains of MT-Bench, as represented in a radar chart. The chart demonstrates that CAMPUS covers a relatively larger area, indicating a more comprehensive and stable performance across all domains. While DEITA and random shuffle tuning exhibit superior performance in specific domains, CAMPUS consistently maintains robust performance without significant drawbacks in any particular area. In contrast, the other two

---

[3] `https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered`

[4] We only used the metrics in the two papers to measure the data difficulty in our experiments.

[5] `https://github.com/hiyouga/LLaMA-Factory`

Backbone LLM→ Reward Model Trained on ↓	LLaMA-7B			LLaMA-13B		
	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench
LLaMA-7B	35.48	15.85	5.91	37.86	16.46	5.96
LLaMA-13B	38.61	16.89	5.95	40.56	17.68	6.01

Table 7: Experiments on the generalization of scoring models.

Backbone LLM	**LLaMA-7B**			**LLaMA-13B**			**Avg**
Method	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	
CAMPUS	35.86	15.24	5.95	40.56	17.68	6.01	**38.16**
+ Label Smoothing & Upsampling	36.39	15.93	5.99	41.22	18.03	6.09	**38.73**

Table 8: The performance of CAMPUS with label smoothing and upsampling on LLaMA-7B and LLaMA-13B.

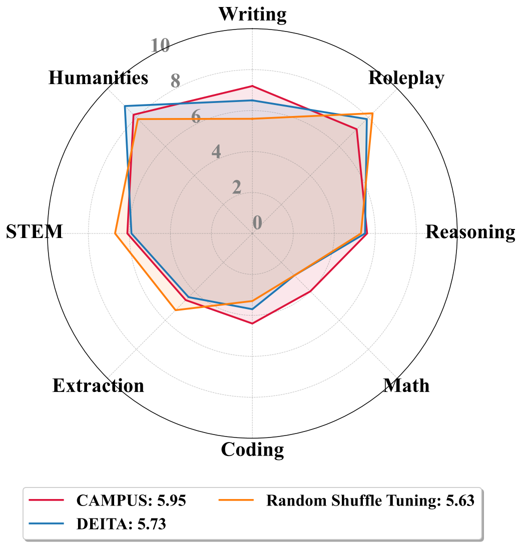


Figure 4: Model performance comparison on different domains of MT-Bench.

baseline methods display pronounced deficiencies in challenging domains such as math, writing, and code. We attribute the stability of CAMPUS to its ability to perceive whether the abilities corresponding to different training data can be internalized by the current LLM. This capability allows CAMPUS to select the most appropriate training data tailored to the needs of the LLM, rather than employing a blind selection. This strategic selection process ensures that CAMPUS enhances LLM capabilities uniformly across diverse domains, preventing significant performance drops in any single area and promoting a well-rounded skill set in the model.

C.2 Generalization of Scoring Model

In this section, we explore the generalization capabilities of the scoring model R when trained with different LLMs. Table 7 assesses the curriculum scheduling ability of scoring models obtained by training with LLaMA-7B and 13B, respectively. The results, particularly evident from the first row, reveal that the scoring model trained with the smaller model (LLaMA-7B) can also effectively aid in the training of the larger model (LLaMA-13B). This transfer ability is highly desired in industrial scenarios. When computational resources are tight, instead of developing a new scoring model from scratch to perfectly align with the target LLM, organizations can employ an existing scoring model as an off-the-shelf dynamic curriculum schedule model directly for optimizing the training strategy of the target LLM.

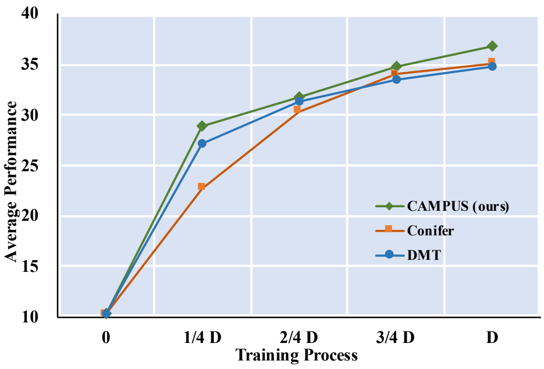
Besides, we introduce two improvements to the scoring model to enhance its effectiveness: (1) applying label smoothing to the loss in Equation 4 to overcome overfitting, and (2) performing upsampling to mitigate label imbalance. Table 8 shows that these refinements to scoring model lead to further performance gains.

C.3 Generalization on other benchmarks

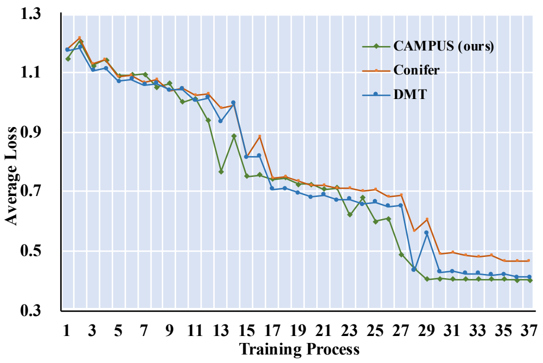
To further demonstrate the generalization of CAMPUS, we extend our evaluation to three additional widely used benchmarks: MATH (Hendrycks et al., 2021) for mathematics reasoning, MBPP (Austin et al., 2021) for code generation, and MMLU (Hendrycks et al.) for general instruction following. As shown in Table 9, CAMPUS consistently outperforms other state-of-the-art methods on these benchmarks, illustrating strong generalization ability.

Backbone LLM	**LLaMA-7B**			**LLaMA-13B**			**Avg**
Method	MATH	MBPP	MMLU	MATH	MBPP	MMLU	
DEITA	14.71	20.29	40.52	17.02	25.78	48.29	27.77
Random Shuffle Tuning	15.33	20.45	38.23	18.67	26.25	47.13	27.68
DMT	15.52	20.87	42.41	18.34	26.87	49.52	28.92
Conifer	15.28	20.62	41.57	18.59	25.96	49.27	28.55
CAMPUS(ours)	**16.24**	**21.71**	**44.18**	**19.31**	**27.33**	**52.41**	**30.20**

Table 9: The performance of CAMPUS on three benchmarks.



(a) Average performance curve of LLaMA-7B model during training process.



(b) Loss curve of LLaMA-7B model during training process.

Figure 5: Visualization of performance and loss.

D Supplementary Experiments on Training Process of CAMPUS

D.1 Model Convergence During Training

We visualize the training process for CAMPUS and the other two baselines in Figure 5. Figure 5(a) illustrates the average performance of the model after training with the corresponding amount of data, where D represents the full dataset. Figure 5(b) records the average loss values at the corresponding step. We can observe that CAMPUS achieves faster loss convergence compared to the baseline model. Additionally, throughout the training process, CAMPUS consistently outperforms the baseline models and reaches higher performance upper bounds. This performance trend effectively demonstrates that CAMPUS aligns with the ultimate goal of curriculum instruction tuning, optimizing overall model performance, meanwhile facilitating learning efficiency.

Method	**GSM8K**	**HumanEval**	**MT-Bench**	**Avg**
Random Shuffle Tuning	35.03	14.63	5.63	35.32
CAMPUS	35.86	15.24	5.95	36.87
w/o Length d_1	36.39	14.98	5.83	36.56
w/o MTLD d_2	36.41	14.93	5.68	36.05
w/o Loss d_3	36.02	15.31	5.76	36.31
w/o Scoring Model d_4	34.81	14.69	5.74	35.63

Table 10: Supplementary ablation experiments on difficulty metrics.

	Training Time (Scoring Model)	**Training Time (LLMs)**
BLOOMZ 3B	15 h (75%)	20 h
LLaMA 7B	41 h (76%)	54 h

Table 11: Training time comparison for both the scoring model and the LLMs.

D.2 Supplementary Ablation on Difficulty Metrics

The ablation experiments on difficult metrics in Table 10 are complementary to those in Table 3. In Table 10, we removed the individual metrics of CAMPUS to provide a different perspective on understanding the different metrics. The trends observed are consistent with Table 3. For example, removing the length metric improved GSM8K performance slightly, but overall, all metrics contributed to the performance gains of CAMPUS. These results support the complementary nature of the difficulty metrics.

D.3 Training Time Analysis

We include a detailed analysis of training times for both the scoring model and the LLMs, as shown in Table 11:

Backbone LLM	**LLaMA-3.2 1B**			**LLaMA-3.1 8B**			**Deepseek-V2-Lite 16B**			**Avg**
Method	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	GSM8K	HumanEval	MT-Bench	
DEITA	33.17	13.52	5.59	46.35	20.14	6.57	53.17	28.12	6.68	42.54
Random Shuffle Tuning	34.87	14.72	5.37	49.17	21.72	6.44	54.85	28.91	6.46	42.99
DMT	34.23	14.57	5.51	48.29	21.96	6.61	53.71	29.27	6.68	43.34
Conifer	34.49	14.36	5.63	49.02	22.13	6.55	54.66	28.83	6.72	43.61
CAMPUS(ours)	35.29	15.13	5.85	50.21	23.27	6.86	56.27	30.52	6.93	45.23

Table 12: We use LLaMA and more recent LLMs as our backbone following some important baselines.

Method	**GSM8K**	**HumanEval**	**MT-Bench**	**Avg**
DEITA	47.43	21.44	6.22	43.69
Random Shuffle Tuning	46.23	17.32	5.93	40.95
DMT	49.51	22.62	6.45	45.54
Conifer	48.74	23.15	6.37	45.20
CAMPUS (ours)	**51.38**	**25.84**	**6.79**	**48.37**

Table 13: The performance of CAMPUS and other baselines on LLaMA-33B.

p	**GSM8K**	**HumanEval**	**MT-Bench**	**Avg**
1.5	36.01	15.21	5.99	37.04
2.0	35.86	15.24	5.95	36.87
2.5	35.83	15.19	5.89	36.64
3.0	35.88	15.21	5.92	36.76

Table 14: The performance of CAMPUS on different hyperparameter p.

All baselines used the same training data. Hence, training time for training order optimization-based methods remains consistent. Data selection-based methods reduce time but sacrifice performance due to data over-deletion.

D.4 CAMPUS Performance on Other LLMs

To verify the performance of CAMPUS on larger or more recent LLMs, we conduct additional experiments on LLaMA-33B in Table 13. The results of CAMPUS on more recent LLaMA-3.2 1B, LLaMA-3.1 8B, and Deepseek-V2-Lite 16B are also supplemented in Table 12. As expected, CAMPUS demonstrated stable and superior performance compared to baselines, showcasing excellent scalability.

s(1)	**GSM8K**	**HumanEval**	**MT-Bench**	**Avg**
0.005	35.63	15.11	5.97	36.81
0.01	35.86	15.24	5.95	36.87
0.1	35.92	15.22	5.93	36.81
0.2	35.89	15.29	5.91	36.76

Table 15: The performance of CAMPUS on different hyperparameter s(1).

D.5 Hyperparameter Sensitivity Analysis

We add experiments to analyze the influence of two curriculum parameters: p (curriculum expansion progression rate) in Table 14 and s(1) (initial learning scope) in Table 15. The results suggest that CAMPUS is robust across a range of reasonable values. Besides, a smaller p (slower curriculum progression) tends to encourage more fine-grained learning and sometimes leads to slightly better results.

Data Stage	Data Type	Instruction Data
First 1K Data	Math	### Instruction: Output the 5th Fibonacci number. ### Response: 5.

Code

```
### Instruction:
Create an HTML page with a text box that stores entered data in the local storage.

### Response:
<html>
<head>
<title>Text Box</title>
</head>

<body>
<input type="text" id="text_box" />
</body>

<script>
// Store data in localStorage
function storeData() {
    let inputValue = document.getElementById("text_box").value;
    localStorage.setItem("inputValue", inputValue);
}

// Set the text box value from localStorage
window.onload = () => {
    if(localStorage.getItem("inputValue"))
    document.getElementById("text_box").value = localStorage.getItem("inputValue");
}

// Set up the event listener
document.getElementById('text_box').addEventListener('input', storeData);
</script>

</html>
```

Data Stage	Data Type	Instruction Data
	General	### Instruction: Hello. ### Response: Hello! How can I help you today?

Table 16: The data cases sampled from the first 1K data.

Data Stage	Data Type	Instruction Data
Last 1K Data	Math	### Instruction: John buys a chair. He then buys a table that is 3 times the price of the chair. Then, he buys a couch that is 5 times the price of the table. If John paid $380 for all these items, what is the price of the couch? ### Response: Let x be the price of the chair. The table costs 3*x. The couch costs 3*x*5=15*x. x+3*x+15*x=380 19*x=380 x=«20=20»20 The couch costs 20*15=«20*15=300»300 dollars.
	Code	### Instruction: Generate a function in python that takes two lists and returns the product of their longest common substrings. ### Input: list1 = [1, 2, 3, 4, 5] list2 = [2, 3, 5, 7] ### Response:

```
def longest_common_substring(list1, list2):
    longest_substring = 0
    for i in range(len(list1)):
        for j in range(len(list2)):
            if list1[i] == list2[j]:
                current_substring = 1
                k, l = i+1, j+1
                while k < len(list1) and l < len(list2) and list1[k] == list2[l]:
                    current_substring += 1
                    k += 1
                    l += 1
                longest_substring = max(longest_substring, current_substring)

    if longest_substring == 0:
        return 0
    else:
        return list1[0] * list2[0] * longest_substring
```

Data Stage	Data Type	Instruction Data
	General	### Instruction: 5.0 Varying Magical Theories 5.1 Miseishun Theory; Spiritual Belief Method The Miseishun hubworld has a comparatively spiritual take on arcanum, known as the Spiritual Belief Method (SBM), opposed to the theoretical views practiced in TOR, Theoretical Magical Formulae (TMF)... ### Response: It seems that the "Looped Filter Theory" is a theory about... ### Instruction: "6.0 Duinn This section was written in coalition with TOR Combined Holy Clergy (TORCHC). Despite the focus of this document being Arcanum, a necessary distinction to make in this report is that Duinn is not classified as magic... ### Response: It seems that in this fictional universe, there are multiple types of magic and magical abilities that people can possess and use... .... More than 7 round conversations.

Table 17: The data cases that sampled from the last 1K data.