Learning from Evolving Training Dynamics: An Entropy-Maximizing Data Curation Strategy for LLM Supervised Post-Training

Mengxiang Zhang
The University of Hong Kong
mxzhang6@connect.hku.hk

Lingyuan Liu
Independent Researcher
ly.liu@my.cityu.edu.hk

Abstract

Supervised post-training is essential for refining Large Language Models (LLMs), yet its effectiveness relies heavily on strategic data curation. Traditional Curriculum Learning (CL) strategies often fail to account for the evolving proficiency of the learner, relying instead on static, single dimensional metrics. We propose **EVO-Curate**, a dynamic data curation framework that synchronizes sample complexity with the maturing capacity of the LLM. EVO-Curate employs an *Adaptive Dynamics Measurer* to synthesize instantaneous difficulty and historical variability into a multidimensional utility score. To maintain representational diversity, we introduce an *Evolutionary Sampling Scheduler* based on an entropy maximizing mechanism. Empirical evaluations across instruction following, mathematical reasoning, and code generation demonstrate that EVO-Curate consistently outperforms standard training baselines and traditional CL methods across various architectures and scales. Specifically, our framework achieves relative performance gains of up to about 10% while maintaining manageable computational overhead. These results establish EVO-Curate as a scalable and model agnostic solution for enhancing the efficiency of modern LLM training pipelines.

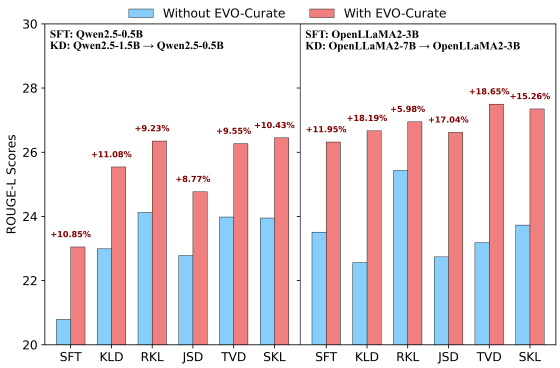

Figure 1: Effectiveness of the EVO-Curate framework on five instruction-following datasets. This figure compares SFT and KD (KLD, RKL, JSD, TVD, SKL) with and without EVO-Curate using ROUGE-L scores. We employ Qwen2.5-0.5B and OpenLLaMA2-3B as target models for SFT. For KD scenarios, we utilize teacher-student pairs consisting of Qwen2.5-1.5B → 0.5B and OpenLLaMA2-7B → 3B. The results demonstrate that EVO-Curate consistently yields performance improvements over all standard training baselines

1 Introduction

Large Language Models (LLMs) have achieved unprecedented milestones in text generation, linguistic comprehension, and logical reasoning by scaling parameters and utilizing high quality training data (Ouyang et al., 2022). Supervised post training, encompassing supervised finetuning (SFT) and knowledge distillation (KD) (Hinton et al., 2015), is essential for refining model capabilities and ensuring generalizability. These techniques have become foundational to the development of state of the art models such as DeepSeek-v3 (Liu et al., 2024), Qwen3 (Yang et al., 2025), and MiMo-V2-Flash (Xiao et al., 2026), allowing them to align with human intent across various specialized domains.

The effectiveness of supervised post training depends heavily on the quality and strategic selection of training data (Zhou et al., 2023). In SFT, models often struggle when the distribution of the finetuning dataset significantly deviates from the pretraining distribution (Yang et al., 2024b). Similarly, KD faces a training inference mismatch caused by the inherent capacity gap between the teacher and student models (Shing et al., 2025). To mitigate these issues, Curriculum Learning (CL) has emerged as a prominent data curation strategy that introduces samples in a progression from simple to complex (Wang et al., 2021). By smoothing the optimization landscape, CL enhances both convergence stability and final performance in various machine learning paradigms (Zeng et al., 2025; Liu and Zhang, 2025).

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 11029–11045

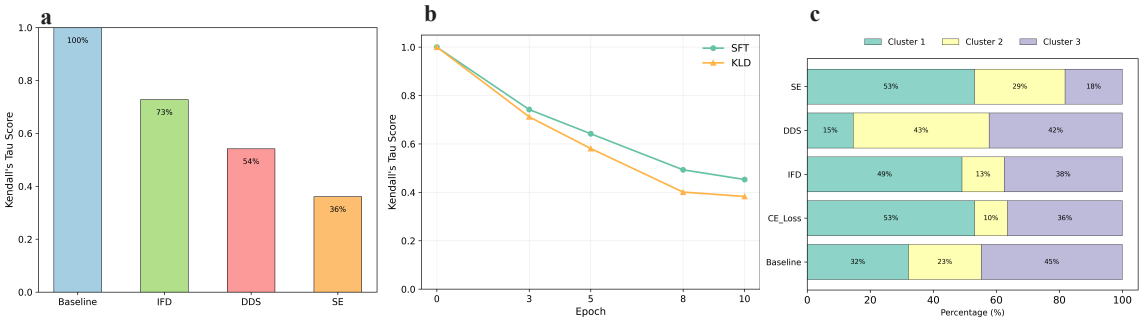

Figure 2: (a) Kendall's Tau correlation of sample rankings on `dolly-15k` across four metrics: Perplexity, Instruction Following Difficulty (IFD), Distillation Difficulty Score (DDS), and Sentence Entropy (SE). Low correlation suggests that difficulty is metric dependent. (b) Kendall's Tau correlation of rankings using Perplexity as the model matures during supervised post training, relative to an untrained baseline. The declining correlation indicates that sample difficulty is model state dependent. (c) Semantic cluster distribution of selected instances for traditional difficulty-based CL versus standard training. Traditional CL results in significant cluster collapse, demonstrating that static curation compromises dataset diversity.

However, maximizing data efficiency through CL in the context of LLM supervised post training remains a formidable challenge due to three primary limitations. First, traditional frameworks often rely on isolated, single dimensional metrics such as Perplexity (Paul et al., 2021) or specific distillation scores (He et al., 2025) to rank instances. Such linear rankings oversimplify the complexity of the dataset and fail to distinguish between highly informative ambiguous samples and detrimental noise (Fig. 2 a). Second, conventional difficulty measures are typically static and model independent (Fig. 2 b), which prevents them from tracking the evolving proficiency of the learner as training progresses (Zhu et al., 2021; Li et al., 2024). Finally, the fixed exposure sequences utilized in most CL implementations can compromise dataset diversity (Fig. 2 c). By repeatedly prioritizing easy samples, these methods may lead the model to overfit on a narrow subset while neglecting the diverse information required for robust generalization (Soviany, 2020).

To address these challenges, we introduce **EVO-Curate**, a modular framework for evolution based dynamic data curation. While grounded in the core principles of CL, EVO-Curate is specifically engineered to overcome the constraints of static curricula through an adaptive process synchronized with the evolving capacity of the LLM. Our framework utilizes an *Adaptive Dynamics Measurer* that integrates instantaneous loss with historical variability to provide a multidimensional assessment of sample utility. Furthermore, we address the limitations of static metrics by re-evaluating the entire dataset at discrete training stages using the current model state. To preserve representational diversity, we implement an *Evolutionary Sampling Scheduler* based on an entropy maximizing mechanism. This probabilistic approach ensures that while high utility samples are prioritized, the model maintains exposure to a diverse data distribution, thereby enhancing both robustness and generalizability. While individual components such as loss-based difficulty scoring and diversity-preserving sampling are established concepts, their synergistic integration into a unified, self-adaptive framework is non-trivial. A naive recombination fails because static difficulty rankings cannot track evolving sample utility, and fixed ordering inevitably collapses data diversity. EVO-Curate resolves these issues by treating data selection as a closed-loop process where measurement, utility computation, and entropy-maximizing selection co-evolve with the model, yielding gains that exceed the sum of individual components as demonstrated in our ablation analysis.

We evaluate the effectiveness of EVO-Curate across a rigorous suite of supervised post training tasks, including instruction following, mathematical reasoning, and code generation. To demonstrate the versatility of our framework, we evaluate EVO-Curate across a range of challenging scenarios that encompass significant teacher-student capacity discrepancies and constrained low resource environments. Our empirical results consistently show that EVO-Curate outperforms both standard training baselines and traditional CL strategies across multiple model architectures and scales (See Fig. 1). Crucially, these performance gains are achieved with manageable additional computational costs, making EVO-Curate a practical and scalable solution for modern LLM training pipelines.

2 Preliminaries

2.1 Problem Formulation

LLM SFT Objective. Let q_θ denote a transformer based language model parameterized by θ. In the context of SFT, we adapt a pretrained LLM using a dataset $D = \{(x, y)_i\}_{i=1}^{N}$ consisting of N pairs of prompts x and corresponding ground truth responses y. For a given prompt $x = (x_1, \ldots, x_R)$, the model generates a response $y = (y_1, \ldots, y_L)$. The primary objective of supervised finetuning is to minimize the expected cross entropy loss, defined as:

$$\mathcal{L}_{ce} = -\sum_{i}^{|y|} \log q_\theta(y_i|x, y_{<i}), \tag{1}$$

where $q_\theta(y_i|x, y_{<i})$ represents the conditional probability assigned by the model to the i-th token given the prompt and preceding response tokens.

LLM KD Objective. In the KD setting, a student model q_θ learns from a fixed teacher model p. The training objective incorporates two distinct components: (1) the standard cross entropy loss $\mathcal{L}_{ce}$ relative to the ground truth labels, and (2) a distillation loss $\mathcal{L}_{kd}$ that minimizes the divergence between the token level probability distributions of the teacher and the student. The total loss $\mathcal{L}_s$ is formulated as:

$$\mathcal{L}_s = \alpha \cdot \mathcal{L}_{ce} + (1 - \alpha) \cdot \mathcal{L}_{kd}, \tag{2}$$

where $\alpha \in [0, 1]$ is a balancing coefficient. The term $\mathcal{L}_{kd}$ typically utilizes measures such as Kullback Leibler Divergence (KLD) (Hinton et al., 2015) or Jensen Shannon Divergence (JSD) (Agarwal et al., 2024), often smoothed by a temperature hyperparameter. When $\alpha = 1$, the objective simplifies to standard supervised finetuning. The detailed description of KD objective is shown in appendix A.1.

Data Curation Objective. Given the fixed dataset D, the goal of data curation is to identify an optimal sequence of training subsets $D_{sub} \subseteq D$. For each epoch or training stage, we aim to select samples that maximize the generalization performance of the target model q_θ while adhering to the constraints of the training budget and the capacity of the model.

2.2 Limitations of Traditional CL

Single-dimensional difficulty metrics lack the comprehensiveness required to evaluate sample quality accurately. Traditional CL frameworks often rely on isolated scores such as Perplexity (Paul et al., 2021) or specific distillation difficulty metrics (He et al., 2025) to rank instances. However, as demonstrated by Swayamdipta et al. (2020), data quality is better characterized by multi-dimensional attributes including confidence and variability. Relying on a linear ranking based on a single metric oversimplifies the complexity of the dataset, as it fails to distinguish between informative ambiguous samples and detrimental noisy samples (Fig. 2 a). Integrating these multi-dimensional training dynamics into a cohesive CL framework remains a significant challenge that current methods have yet to fully address.

Static metrics fail to account for the evolving capacity of the model throughout the post-training process. Conventional difficulty measures, such as sentence entropy (Zhu et al., 2021) or instruction-following difficulty (Li et al., 2024), are typically computed once via an untrained model or are entirely model-independent (Fig. 2 b). While these metrics provide an initial ranking, they cannot track the training dynamics as the model matures. Consequently, a fixed curriculum ignores the shifting relationship between the model and the data. Although some approaches attempt to mitigate this by employing smaller proxy models to estimate dynamic features (Yang et al., 2024b), such methods introduce significant computational overhead and may suffer from a distribution mismatch between the proxy and the target model, thereby sacrificing accuracy and efficiency.

Pre-determined training orders frequently compromise dataset diversity and hinder the model's generalizability. In most CL implementations, the sequence of exposure is fixed prior to training, where easy samples are introduced early and subsequently repeated across multiple stages. This imbalance often results in the model over-optimizing on a specific subset of the data while neglecting the diverse information contained in harder, later-stage examples (Soviany, 2020) (Fig. 2 c). By prioritizing a rigid order, CL may inadvertently collapse the representational diversity of the dataset, leading to diminished performance on out-of-distribution tasks and a reduction in the overall robustness of the LLM during supervised post-training.

3 Methodology

We propose **EVO-Curate**, an evolution-based dynamic data curation framework designed to maximize the efficacy of CL for LLM supervised post-training. While grounded in the CL philosophy of staged learning, EVO-Curate introduces an evolutionary mechanism to address three critical limitations of traditional approaches: (1) the reliance on *single-dimensional metrics* is resolved through a multi-faceted utility score; (2) the failure of *static rankings* is mitigated by re-evaluating data dynamics in tandem with the model's evolving capacity; and (3) the loss of *dataset diversity* caused by rigid ordering is prevented via entropy-maximizing probabilistic sampling. By treating data selection as an adaptive process rather than a pre-determined sequence, EVO-Curate synchronizes the complexity of the training subset with the current maturity of the model, thereby ensuring stable convergence and enhanced generalizability.

The architecture of EVO-Curate is organized into two primary interactive modules that operate across M discrete training stages. In each stage $m \in \{1, \ldots, M\}$, the *Adaptive Dynamics Measurer* evaluates the utility of every sample by synthesizing the current model state with the accumulated training history. Subsequently, the *Evolutionary Sampling Scheduler* employs an entropy-maximizing mechanism to construct a non-repeated training subset D_{sub} of increasing size. The model is then optimized on this curated subset for k local epochs, evolving its parameters to a higher level of maturity before the next iteration of measurement and selection. This cyclical process, detailed in Algorithm 1 in Appendix A.1, ensures that the dataset complexity grows in tandem with the model's capabilities.

3.1 Adaptive Dynamics Measurer

The data selection logic of EVO-Curate is grounded in the taxonomy of training dynamics proposed by Swayamdipta et al. (2020), which categorizes samples based on their contribution to model optimization. Following these insights, we recognize three distinct archetypes: (1) *easy-to-learn* samples, characterized by low difficulty and low variability, which provide the stable gradients necessary to prevent early training divergence; (2) *ambiguous* samples, exhibiting high variability, which reside near the decision boundary and offer the most informative signals for promoting generalization; and (3) *hard-to-learn* samples, defined by high difficulty and low variability, which often represent noise or misaligned data that can negatively impact performance. Importantly, while prior work has used loss statistics in isolation, EVO-Curate uniquely synthesizes these three axes—difficulty, variability, and diversity—within a single coherent mechanism that dynamically drives data selection as the model matures. To operationalize this taxonomy within a dynamic framework, we define two stage-dependent metrics that quantify these attributes as the model evolves.

Instantaneous Difficulty (a_i^m): We utilize the cross entropy loss of the model q_θ^{m-1} as a proxy for the difficulty of instance $(x, y)_i$ at the current stage:

$$a_i^m = -\log q_\theta^{m-1}(y_i|x_i). \tag{3}$$

A low value of a_i^m signifies a high confidence sample that the model has already begun to master, whereas a high value indicates significant representational difficulty.

Dynamic Amplitude (b_i^m): To measure the informativeness of an instance, we calculate the weighted variability of the loss across previous stages:

$$b_i^m = \sum_{j=2}^{m} 2^{j-m} \cdot |a_i^j - a_i^{j-1}|, \tag{4}$$

where the term 2^{j-m} serves as a temporal decay factor that prioritizes recent dynamics. This amplitude reflects how much the model's understanding of a sample fluctuates during training. High variability typically identifies ambiguous samples that reside near the decision boundary, offering the greatest potential for learning. Note that for the first stage ($m = 1$), b_i^m is initialized to zero.

3.2 Evolutionary Sampling Scheduler

The scheduler translates the measured dynamics into a robust sampling strategy that favors high utility data while maintaining necessary dataset diversity.

Net Utility Formulation: We define the Net Utility U_i^m for each sample as the difference between its positive attribute of variability and its negative attribute of difficulty:

$$U_i^m = b_i^m - a_i^m. \tag{5}$$

The term $-a_i^m$ encourages the selection of easier samples to ensure stable gradient updates during the initial phases of training. Simultaneously, the term $+b_i^m$ promotes the inclusion of ambiguous

Table 1: Performance comparison of instruction following for Qwen2.5-0.5B. We compare standard SFT and various KD objectives with and without the EVO-Curate framework. **Avg.** represents the mean ROUGE-L score across the five benchmarks. Results for GPT2-0.1B and OpenLLaMA2-3B are in Tab. 7.

	SFT: Qwen2.5-0.5B; KD: Qwen2.5-1.5B → Qwen2.5-0.5B												
	DollyEval		**SelfInst**		**VicunaEval**		**S-NI**		**UnNI**		**Avg.**		
Methods	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	**Improvement**
SFT	24.86	49.64	14.85	47.75	16.10	50.18	23.23	43.42	24.91	44.86	20.79	47.17	-
+EVO-Curate	**27.25**	**52.30**	15.94	49.18	**17.50**	**53.03**	**26.74**	44.38	**27.80**	45.78	**23.05**	48.94	10.85%
SeqKD	24.95	49.81	15.14	49.31	16.05	49.76	23.31	44.19	25.24	46.39	20.94	47.89	-
+EVO-Curate	26.84	51.31	**16.88**	**51.51**	17.12	51.32	25.27	**46.01**	26.97	**47.94**	22.62	**49.62**	8.01%
KLD	26.71	54.82	15.72	51.35	17.32	54.88	27.21	50.25	27.98	50.19	22.99	52.30	-
+EVO-Curate	**29.36**	**57.70**	17.81	53.35	19.36	57.64	31.26	52.82	29.88	51.52	25.54	54.61	11.08%
RKL	27.13	53.04	16.62	51.97	17.94	55.02	29.61	51.51	29.32	50.98	24.12	52.51	-
+EVO-Curate	28.87	54.62	18.44	53.75	19.02	**57.78**	**32.81**	54.46	**32.61**	53.62	26.35	**54.85**	9.23%
JSD	26.84	53.46	15.42	50.28	16.97	53.11	27.41	50.48	27.24	48.48	22.78	51.16	-
+EVO-Curate	28.66	56.51	16.84	52.96	18.78	54.98	29.77	51.75	29.82	50.05	24.77	53.25	8.77%
TVD	26.76	52.94	16.08	51.35	17.15	53.69	30.04	52.60	29.87	51.67	23.98	52.45	-
+EVO-Curate	28.69	54.74	18.81	53.58	19.72	55.98	32.48	**54.85**	31.65	**53.80**	26.27	54.59	9.55%
SKL	27.02	53.63	17.04	53.12	17.78	54.92	29.11	52.65	28.82	50.64	23.95	52.99	-
+EVO-Curate	28.78	55.13	**19.74**	**55.35**	**20.78**	56.19	31.18	54.69	31.80	51.99	**26.45**	54.67	10.43%

Note: *R-L* and *WR* denote ROUGE-L and Winning Rate respectively. Winning Rate is determined via LLM as a Judge using DeepSeek-V3-0324. **Avg.** represents the mean ROUGE-L score across the five benchmarks. **Improvement** denotes the relative gain in average ROUGE-L. **Bold** values indicate the best performance within each category. All results are averaged over five independent runs with different random seeds.

samples that provide rich information for the model to generalize. By combining these terms, the model focuses on the most productive samples relative to its current maturity.

Entropy-Maximizing Selection: We map the Net Utility to a probability distribution P_i^m using the Boltzmann distribution:

$$P_i^m = \frac{\exp(U_i^m)}{\sum_{j=1}^{N} \exp(U_j^m)}. \tag{6}$$

This approach is grounded in the *Principle of Maximum Entropy* (Guiasu and Shenitzer, 1985), which posits that the exponential distribution is the most unbiased representation of our knowledge given a linear utility constraint. By employing a probabilistic Softmax rather than a hard ranking, we ensure that the selection process remains diverse. This prevents the model from overfitting to a narrow range of easy examples and allows for the eventual inclusion of more complex data as the training stages progress.

Non-Repeated Multi-Stage Sampling: In each stage m, we sample $n = \frac{m}{M} \cdot N$ unique instances. As the stage index m increases, the probability mass shifts according to the updated model feedback, and the volume of data grows. At the final stage M, the entire dataset D is utilized to ensure the model benefits from the full range of available information. This progression simulates an evolutionary trajectory where the model is gradually exposed to increasing complexity without sacrificing the stability of the learning process.

4 Experiments

4.1 Experimental Setup

We evaluate the effectiveness, scalability, and generalizability of the EVO-Curate framework across three distinct tasks: instruction following, mathematical reasoning, and code generation. For each task, we conduct a comparative analysis by applying SFT and KD to target models both with and without the EVO-Curate strategy (Gu et al., 2023). Comprehensive implementation details are provided in Appendix A.2.

Base Models. To ensure a robust evaluation across varying scales, we employ Qwen2.5-0.5B (Team, 2024), GPT2-0.1B (Radford et al., 2019), and OpenLLaMA2-3B (Touvron et al., 2023) as target models for instruction following. In KD experiments, we utilize their larger counterparts, Qwen2.5-1.5B, GPT2-1.5B, and OpenLLaMA2-7B, as teacher models. For specialized tasks, we select Qwen2.5-Math-1.5B and Qwen2.5-Coder-1.5B as targets, distilling knowledge from their respective 7B instruction tuned variants.

Datasets and Training. Our experiments utilize `databricks-dolly-15K` (Conover et al., 2023) for instruction following. For mathematical reasoning, we sample 20K training and 2K validation instances from `MetaMathQA` (Yu et al., 2023). Code generation is evaluated using 20K training and 2K validation samples from `Evol-Instruct-Code-80k-v1` (Luo et al., 2023), which is enhanced via the Evol-Instruct method (Xu et al., 2024).

In the EVO-Curate configuration, we set the total training stages to $M = 4$. To ensure a fair com-

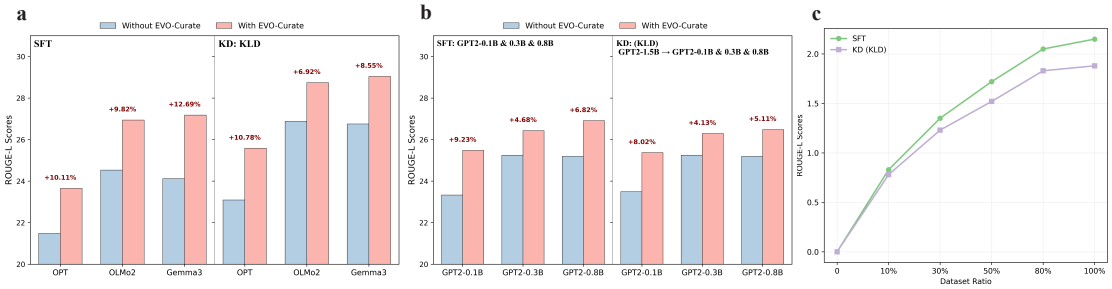

Figure 3: Comparative analysis of EVO-Curate in instruction following tasks. (a) ROUGE-L scores across diverse model configurations (OPT-0.3B, OLMo2-1B, and Gemma3-1B) for both SFT and KLD. (b) Performance across varying capacity gaps using GPT2 variants (0.1B, 0.3B, 0.8B) distilled from a GPT2-1.5B teacher. (c) Net ROUGE-L gains relative to baselines under low resource settings using the GPT2-0.1B model across varying training data percentages.

parison with baseline methods, we equalize the total training iterations by adjusting the number of epochs per stage. For a baseline of 10 epochs, this results in $k = 4$ epochs per stage, yielding a total exposure of $0.25{*}4{+}0.5{*}4{+}0.75{*}4{+}1.0{*}4 = 10$ epochs. For all KD objectives, the balancing coefficient is set to $\alpha = 0.5$.

Evaluation Metrics. Instruction following performance is measured on DollyEval, SelfInst, VicunaEval, S-NI, and UnNI (Conover et al., 2023; Wang et al., 2022a; Chiang et al., 2023; Wang et al., 2022b; Honovich et al., 2022). We report ROUGE-L scores across five random seeds and complement these with winning rates (WR) derived from LLM-as-a-Judge (Zheng et al., 2023) using DeepSeek-V3-0324 (Liu et al., 2024). Mathematical reasoning and code generation are assessed using Pass@1 accuracy on GSM8K (Cobbe et al., 2021) and MBPP (Austin et al., 2021), respectively. All scores are averaged over five independent runs with a generation temperature of 1.

Baselines. We compare our approach against standard training paradigms, including:

- **SFT and SeqKD**: Standard supervised fine tuning on ground truth data and teacher generated outputs (Lin et al., 2020).
- **KD**: A diverse set of divergence serve as KD loss objectives, including Forward KLD (Hinton et al., 2015), Reverse KLD (RKL) (Gu et al., 2023), JSD (Agarwal et al., 2024), Total Variation Distance (TVD) (Wen et al., 2023), and Skew KLD (SKL) (Ko et al., 2024).

4.2 Results

4.2.1 Instruction-Following

Overall Performance. The results in Tab. 1 and 7 demonstrate that EVO-Curate consistently and significantly improves performance across three distinct model families in both SFT and KD paradigms. The framework yields robust gains in both text generation quality (average ROUGE-L) and instruction-following alignment (Winning Rate). This consistent uplift confirms the efficacy of our framework in dynamically prioritizing high-utility training data, thereby enhancing supervised post-training outcomes. These improvements are further validated using an alternative LLM judge (Qwen3-Max), which yields consistent results across all benchmarks (see Appendix A.4 for details).

Table 2: Performance comparison of EVO-Curate in mathematical reasoning and code generation.

SFT Model	Qwen2.5-Math-1.5B	Qwen2.5-Coder-1.5B
KD Model	Qwen2.5-Math-7B → Qwen2.5-Math-1.5B	Qwen2.5-Coder-7B → Qwen2.5-Coder-1.5B
Methods	GSM8K *Pass@1*	MBPP *Pass@1*
SFT	76.20	60.70
+EVO-Curate	**79.91**	**63.36**
KLD	77.90	61.30
+EVO-Curate	80.59	64.29
RKL	78.40	61.40
+EVO-Curate	81.85	63.84
JSD	78.90	61.00
+EVO-Curate	80.81	62.29
TVD	78.20	61.70
+EVO-Curate	80.75	64.51
SKL	79.40	61.80
+EVO-Curate	**83.17**	**64.77**

Note: Best student results are **bold**.

Evaluation across Diverse Architectures. We validate the framework's scalability by evaluating it on diverse model families: OPT-0.3B (Zhang et al., 2022), OLMo2-1B (OLMo et al., 2024), and Gemma3-1B (Team et al., 2025). As shown in Fig. 3 (a), EVO-Curate provides a consistent performance improvement across all models. This demonstrates that the our framework is model-agnostic and generalizes effectively across different

pretraining distributions and architectural priors.

Robustness to Capacity Gaps. We investigate the relationship between model size and curation efficacy by evaluating three GPT2 variants (0.1B, 0.3B, and 0.8B) under both fine-tuning and distillation from a 1.5B teacher. Fig. 3 (b) shows that EVO-Curate maintains a consistent performance advantage across different model sizes. This indicates that our framework is robust to varying capacity gaps and effectively mitigates the training-inference mismatch commonly observed when KD into significantly smaller student models.

Performance in Low Resource Settings. We simulate data scarcity by sampling subsets of the `dolly-15k` dataset ranging from 10% to 80%(Fig. 3 c). EVO-Curate remains effective even with 10% of data. The increasing utility with data volume makes it valuable for specialized domains where high quality instruction data is difficult to acquire.

Generalization to Diverse Instruction Mixtures. For out-of-domain generalization assessment, we randomly draw 20k prompts from UltraChat200k (Ding et al., 2023) and generate corresponding responses using the Qwen2-7B teacher model. Using this dataset, we then train the Qwen2-1.5B student model via both SFT and KD. Tab. 3 reports winning rates evaluated via LLM-as-a-Judge (DeepSeek-V3-0324) on Evol-Instruct (Xu et al., 2024) and UltraFeedback (Cui et al., 2023). EVO-Curate consistently outperforms standard SFT and KD across both benchmarks, confirming that the framework's benefits extend to unseen instruction mixtures.

Table 3: Generalization to diverse instruction mixtures. Winning rates (%) on Evol-Instruct and UltraFeedback using Qwen2 (7B teacher, 1.5B student).

Methods	**Evol-Inst WR (%)**	**UltraFeed WR (%)**
SFT	27.84	34.74
+ EVO-Curate	29.87	37.43
KD	34.31	37.53
+ EVO-Curate	38.14	40.02

Note: Evaluation via LLM-as-a-Judge using DeepSeek-V3-0324. Training data: 20k prompts sampled from UltraChat200k.

4.2.2 Mathematical Reasoning and Code Generation

Tab. 2 demonstrates cross domain efficacy in mathematical reasoning and code generation. On GSM8K, EVO-Curate improves Qwen2.5-Math-1.5B *Pass@1* by 3.77% absolute over standard KD (SKL). Similarly, MBPP scores for Qwen2.5-Coder-1.5B rise from 60.70% to 63.36% in SFT. These consistent gains validate our method as a robust, task agnostic optimizer for technical domains.

Table 4: Ablation analysis of EVO-Curate components using GPT2-0.1B.

Methods	**SFT** *Rouge-L*	**KD** *Rouge-L*	**Preparation Time** *Mins*
(a) Baseline Comparison			
Standard (no data curation)	23.33	23.49	0
Uniform Sampling	23.35	23.46	0
EVO-Curate	25.48	25.37	54
(b) Difficulty Metrics			
EVO-Curate (CE-Loss)	25.48	25.37	54
EVO-Curate (SE)	24.01	23.86	18
EVO-Curate (IFD)	25.51	25.43	91
(c) Dual Metric Roles			
EVO-Curate (a_i^m & b_i^m)	25.48	25.37	-
EVO-Curate (only a_i^m)	25.14	24.87	-
EVO-Curate (only b_i^m)	24.58	24.74	-
(d) Evaluation dynamics			
EVO-Curate (dynamic)	25.48	25.37	-
EVO-Curate (static)	24.53	24.34	-
(e) Sampling Strategies			
EVO-Curate (NRS)	25.48	25.37	-
EVO-Curate (RS)	25.01	24.93	-
EVO-Curate (NS)	24.53	24.38	-

Note: Preparation time denotes the overhead for difficulty measurement and subset generation. ROUGE-L scores are averaged across five seeds. CE-Loss: Cross Entropy; SE: Sentence Entropy; IFD: Instruction Following Difficulty. NRS: Non Repeated Sampling; RS: Repeated Sampling; NS: No Sampling.

5 Analysis and Discussion

5.1 Training Stability and Convergence

To evaluate the impact of EVO-Curate on optimization efficiency, we analyze the validation dynamics across the training process. Validation dynamics (Fig. 4 a) show that EVO-Curate accelerates convergence and enhances stability. In early SFT, it achieves a ROUGE-L of 19.10 at the first validation step, substantially outperforming the baseline. Our framework dynamically prioritizes high-utility samples, enabling faster convergence to better optima with stable learning.

5.2 Ablation Analysis

We conduct comprehensive ablation studies using the GPT2 model family on the `dolly-15k` dataset to isolate the contributions of key EVO-Curate components. For SFT experiments, we utilize GPT2-0.1B as the target model. For KD experiments, GPT2-1.5B serves as the teacher to distill the GPT2-0.1B student using a KLD loss.

Baseline Comparison. Tab. 4 (a) compares standard training without data curation, uniform random sampling, and the full EVO-Curate framework. Uniform Sampling performs on par with the

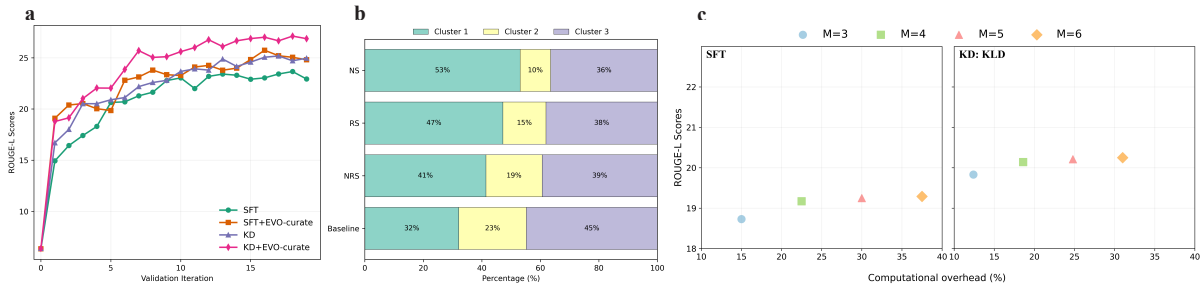

Figure 4: a) Validation ROUGE-L scores versus iterations for SFT and KLD, comparing standard baselines with **EVO Curate**. b) Impact of sampling strategies on data diversity. c) Hyperparameter sensitivity analysis of the curriculum stages M.

standard baseline, confirming that merely subsetting the data without principled selection provides no meaningful benefit. In contrast, EVO-Curate achieves a substantial improvement, demonstrating that the performance gains stem from the principled data selection mechanism rather than from data subsetting alone.

Selection of Difficulty Metric. We evaluate the impact of different difficulty metrics, comparing Cross Entropy Loss (CE-Loss), Sentence Entropy (SE) (Zhu et al., 2021), and Instruction Following Difficulty (IFD) (Li et al., 2024). Comparing CE-Loss, SE, and IFD (Tab. 4 b) reveals that while IFD offers slightly higher scores, it requires significantly more preparation time. We select CE-Loss as the default for its optimal balance of accuracy and efficiency.

Role of Dual Difficulty Metrics. Based on an analysis of the synergy between Instantaneous Difficulty (a_i^m) and Dynamic Amplitude (b_i^m), we find that utilizing both metrics yields the best performance (Tab. 4 c). This result confirms that monitoring the rate of change of training difficulty is as critical as measuring its absolute level, which aligns with observations from dataset cartography (Swayamdipta et al., 2020).

Effect of Dynamic Evaluation. The necessity of updating difficulty scores as model capacity evolves is examined by comparing our dynamic approach with a static evaluation (measuring difficulty once using the untrained model). Tab. 4 (d) shows that dynamic evaluation significantly outperforms static approaches, highlighting that data selection must adapt to model maturation.

Effect of Instance Sampling. We further compare our default Non-Repeated Sampling (NRS) strategy against Repeated Sampling (RS) and No Sampling (NS). The results in Tab. 4 (e) show NRS achieves the highest performance, which we attribute to its preservation of superior data diversity. This is corroborated by the cluster distribution analysis in Fig. 4 (b).

5.3 Computational Cost Analysis

The EVO-Curate framework introduces a manageable computational overhead, limited to the initial difficulty measurement and sampling phases of each stage. This requires one forward pass per stage over the dataset to compute the difficulty metrics and perform non-repeated sampling. Importantly, it introduces no runtime overhead during backpropagation. To quantify this overhead, we evaluate our setup using four A100 80GB GPUs and an Intel Xeon Platinum 8350C CPU. For the `dolly-15k` dataset ($N = 11,435$) and a GPT2-0.1B model, the Difficulty Measurer Module requires approximately 18 minutes for a single measurement and sampling cycle. Across the $M = 4$ stages (where only the first three stages require measurement), the cumulative overhead is 54 minutes. Compared to standard training durations, this represents an additional 22.5% overhead for SFT (totaling $\sim$240 minutes) and 18.6% for KD (totaling $\sim$290 minutes). These results demonstrate that EVO-Curate provides significant performance gains while maintaining a computational cost that remains well within the practical limits of large scale supervised post training.

Performance vs. Wall-Clock Time. To explicitly evaluate whether EVO-Curate's overhead translates into meaningful efficiency gains, we compare standard SFT/KD against their EVO-Curate–augmented counterparts under equal numbers of update iterations. Tab. 5 presents ROUGE-L scores and wall-clock times on Dolly-15k with GPT2-0.1B across four training checkpoints.

This analysis reveals a critical insight: to achieve a target ROUGE-L score of approximately 24.8, SFT with EVO-Curate requires only 156 minutes, while standard SFT never reaches this level even

Table 5: Performance (ROUGE-L) vs. Wall-Clock Time (minutes) on Dolly-15k with GPT2-0.1B.

	Iter 1	Iter 2	Iter 3	Iter 4
SFT				
Time (min)	60	120	180	240
Performance	20.64	23.04	22.91	22.93
SFT + EVO-Curate				
Time (min)	78	156	234	312
Performance	20.85	23.27	24.83	24.81
KD				
Time (min)	72.5	145	217.5	290
Performance	20.90	23.67	24.58	24.96
KD + EVO-Curate				
Time (min)	90.5	181	271.5	362
Performance	22.04	25.60	26.88	26.87

Note: ROUGE-L scores are averaged across five seeds. Each iteration corresponds to one training stage checkpoint.

Table 6: Comparative analysis of EVO-Curate against TCL and DA-KD.

Methods	OpenLLaMA2 (7B/3B) **Avg.**	**Gains (%)**	GPT2 (1.5B/0.1B) **Avg.**	**Gains (%)**
SFT	23.51	-	16.80	-
+EVO-Curate	26.63	11.95%	19.17	14.11%
+TCL	25.36	7.87%	18.32	9.05%
KD (KLD)	22.56	-	18.11	-
+EVO-Curate	26.67	18.19%	20.14	11.26%
+TCL	26.04	15.43%	19.65	8.50%
KD (DA-KD)	25.73	-	21.78	-
+EVO-Curate	26.85	4.35%	22.53	3.44%

Note: SFT Model: OpenLLaMA2-3B and GPT2-0.1B; KD Model: OpenLLaMA2-7B → OpenLLaMA2-3B and GPT2-1.5B → GPT2-0.1B. Performance is evaluated using mean ROUGE-L across five instruction following benchmarks. Gains (%) represent the relative improvement over the respective non curriculum baselines. TCL denotes traditional static curriculum learning. DA-KD refers to the difficulty aware distillation method.

after 240 minutes. Similarly, KD with EVO-Curate achieves a ROUGE-L of 26.88 at 271.5 minutes, substantially exceeding standard KD's best result of 24.96 at 290 minutes. These results demonstrate that EVO-Curate's performance-per-iteration efficiency translates into superior performance-per-wall-clock-time, making the overhead a worthwhile trade-off.

5.4 Sensitivity Analysis on M

We analyze the core hyperparameter of EVO-Curate, the number of curriculum stages M, to find the optimal balance between performance and computational cost. Using GPT2-0.1B, we evaluate $M \in 3, 4, 5, 6$. As shown in Fig. 4 (c), performance (ROUGE-L) generally improves with more stages, but at a linear increase in cost. The most significant performance gain occurs when moving from $M = 3$ to $M = 4$. Further increases yield only marginal improvements while disproportionately increasing overhead. Consequently, $M = 4$ represents the Pareto-optimal choice, offering substantial gains with a manageable computational budget for practical post-training.

5.5 Comparison with Traditional Curriculum Learning

We compare EVO-Curate against two key baselines: Traditional Curriculum Learning (TCL) (Liu and Zhang, 2025) and Difficulty Aware KD (DA-KD) (He et al., 2025). Unlike our dynamic framework, these methods rely on static difficulty estimation performed only once prior to training. EVO-Curate consistently outperforms them across both SFT and KD settings (Tab. 6). This performance advantage highlights the necessity of dynamic data evaluation; a static difficulty ranking cannot capture the evolving utility of samples as the model's capacity changes during training. Our framework adapts to these dynamics, ensuring the model is consistently trained on the most informative data.

6 Related Work

We discuss related work in Appendix A.5.

7 Conclusion

In the paper, we introduced **EVO-Curate**, a dynamic data curation framework designed to enhance LLM supervised post training by aligning data complexity with evolving model maturity. Our approach addresses the fundamental limitations of static CL by utilizing an *Adaptive Dynamics Measurer* to synthesize instantaneous difficulty and historical variability into a multidimensional utility score. Furthermore, the *Evolutionary Sampling Scheduler* employs an entropy maximizing mechanism to maintain representational diversity, effectively preventing the model from over optimizing on narrow data subsets. Empirical results across instruction following, mathematical reasoning, and code generation demonstrate that EVO-Curate consistently outperforms standard baselines and traditional curriculum strategies. By providing significant performance gains with manageable computational overhead, our framework offers a practical and scalable solution for modern training pipelines. Future work will explore multimodal extensions to enhance EVO-Curate's versatility.

8 Limitations

While EVO-Curate consistently enhances performance across diverse benchmarks, several limita-

tions warrant further investigation. First, the framework exhibits more modest absolute improvements in specialized technical domains such as mathematical reasoning and code generation compared to general instruction following. In these tasks, where correctness is highly sensitive to minor token variations, the current reliance on token-level cross-entropy loss as a difficulty proxy may prioritize superficially similar but functionally incorrect trajectories. This suggests that integrating execution-based validation or semantic equivalence scoring could further refine the *Adaptive Dynamics Measurer* for rigorous reasoning tasks. Importantly, our framework is flexible and not tied to a single proxy signal; as demonstrated in our ablation analysis (Section 5.2), the dynamics measurer can incorporate alternative metrics, making the investigation of task-specific proxies a promising direction.

Second, the multi-stage measurement process requires additional forward passes over the full dataset. While this overhead is manageable (15–20% of training time) and pays off in both per-iteration and per-wall-clock-time efficiency (Section 5.3), it may pose scalability challenges for ultra-large-scale training sets on the order of millions of examples. Addressing these constraints through proxy-based scoring on representative subsets and staged measurement on data partitions remains a key direction for future work.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

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*.

Andrew Bai, Chih-Kuan Yeh, Cho-Jui Hsieh, and Ankur Taly. 2025. An efficient rehearsal scheme for catastrophic forgetting mitigation during multi-stage fine-tuning. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2557–2569.

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.

Wei-Lin Chiang, Zhuohan Li, Ziqing 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*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. 2023. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

Shen Gao, Zhengliang Shi, Minghang Zhu, Bowen Fang, Xin Xin, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. 2024. Confucius: Iterative tool learning from introspection feedback by easy-to-difficult curriculum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18030–18038.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Silviu Guiasu and Abe Shenitzer. 1985. The principle of maximum entropy. *The mathematical intelligencer*, 7(1):42–48.

Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. 2025. Da-kd: Difficulty-aware knowledge distillation for efficient large language models. In *Forty-second International Conference on Machine Learning*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*.

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

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. 2025. Distillm-2: A contrastive approach boosts the distillation of llms. *arXiv preprint arXiv:2503.07067*.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

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 7602–7635.

Zheng Li, Xiang Li, Lingfeng Yang, Borui Zhao, Renjie Song, Lei Luo, Jun Li, and Jian Yang. 2023. Curriculum temperature for knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1504–1512.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. 2020. Autoregressive knowledge distillation through imitation learning. *arXiv preprint arXiv:2009.07253*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36:21558–21572.

Lingyuan Liu and Mengxiang Zhang. 2025. Being strong progressively! enhancing knowledge distillation of large language models through a curriculum learning framework. *arXiv preprint arXiv:2506.05695*.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2024. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*.

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

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Ayan Sengupta, Shantanu Dixit, Md Shad Akhtar, and Tanmoy Chakraborty. 2023. A good learner can teach better: Teacher-student collaborative knowledge distillation. In *Proceedings of the The Twelfth International Conference on Learning Representations, Virtual Event*, pages 25–29.

Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. 2025. Taid: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. *arXiv preprint arXiv:2501.16937*.

Petru Soviany. 2020. Curriculum learning with diversity for supervised computer vision tasks. *arXiv preprint arXiv:2009.10625*.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. *arXiv preprint arXiv:2009.10795*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Qwen Team. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2407.10671*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022a. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*, 2:2.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. F-divergence minimization for sequence-level knowledge distillation. *arXiv preprint arXiv:2307.15190*.

Liuyu Xiang, Guiguang Ding, and Jungong Han. 2020. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 247–263. Springer.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. 2026. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. 2024a. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Yu Yang, Siddhartha Mishra, Jeffrey Chiang, and Baharan Mirzasoleiman. 2024b. Smalltolarge (s2l): Scalable data selection for fine-tuning large language models by summarizing training trajectories of small models. *Advances in Neural Information Processing Systems*, 37:83465–83496.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

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. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021.

Qingqing Zhu, Xiuying Chen, Pengfei Wu, JunFei Liu, and Dongyan Zhao. 2021. Combining curriculum learning and knowledge distillation for dialogue generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1284–1295.

A Technical Appendices

A.1 EVO-Curate Training Algorithm

Algorithm 1 provides a detailed description of the EVO-Curate training procedure, including the Adaptive Dynamics Measurer and Evolutionary Sampling Scheduler.

Algorithm 1 EVO-Curate: Evolution Based Dynamic Data Curation

Input: Training dataset $D = \{(x, y)_i\}_{i=1}^{N}$, initial model q_θ^0, total stages M, epochs per stage k;
Output: Final post trained model q_θ^M.
Initialize dynamic amplitude $b_i^1 = 0$ for all $i \in \{1, \ldots, N\}$;
for $m = 1, \ldots, M$ **do**
1. **Measure Difficulty:** Compute $a_i^m = -\log q_\theta^{m-1}(y_i|x_i)$ for all $(x, y)_i \in D$;
2. **Update Dynamics:** Update $b_i^m \leftarrow \frac{1}{2} b_i^{m-1} + |a_i^m - a_i^{m-1}|$ (active for $m > 1$);
if $m = M$ **then**
Set $D_{sub} \leftarrow D$; ▷ Use full dataset in final stage
else
3. **Calculate Utility:** Compute Net Utility $U_i^m = b_i^m - a_i^m$;
4. **Compute Probabilities:** Determine $P_i^m = \frac{\exp(U_i^m)}{\sum_{j=1}^{N} \exp(U_j^m)}$;
5. **Sample Subset:** Sample $n = \frac{m}{M} \cdot N$ unique instances $D_{sub} \subset D$ using P_i^m;
end if
6. **Evolve Model:** Train q_θ^{m-1} on D_{sub} for k epochs to obtain q_θ^m;
end for

A.2 Experimental Setup

This appendix provides detailed information on the base models (Section A.2.1), training procedures (Section A.2.2), evaluation protocols (Section A.2.3), and baseline methods (Section A.2.4).

A.2.1 Base Models Description

We conduct experiments across three distinct tasks—instruction following, mathematical reasoning, and code generation—using representative open-source model families at varying scales to ensure broad validity of our findings. The selected models and their specific roles are as follows:

- **Qwen2.5** (Team, 2024): We leverage the Qwen2.5 suite of models for all three experimental domains. For instruction following, Qwen2.5-0.5B serves as the target model for SFT and as the student for KD, with Qwen2.5-1.5B as the teacher. For mathematical reasoning, Qwen2.5-Math-1.5B is the target for SFT and the KD student, taught by Qwen2.5-Math-7B-Inst. For code generation, Qwen2.5-Coder-1.5B is used analogously as the SFT target and KD student, with Qwen2.5-Coder-7B-Inst as the teacher.

- **GPT2** (Radford et al., 2019): We employ the widely recognized GPT2 family to validate our method on a classic autoregressive architecture. GPT2-0.1B acts as the target model for SFT and the student for KD, while GPT2-1.5B serves as the teacher model.

- **OpenLLaMA2** (Touvron et al., 2023): To include a LLaMA-architecture variant, we utilize OpenLLaMA2. The 3B-parameter model is the SFT target for instruction following. In the corresponding KD experiment, OpenLLaMA2-3B and OpenLLaMA2-7B serve as the student and teacher, respectively.

This selection provides a diverse testbed encompassing different model architectures, licensing frameworks, and parameter scales, facilitating reproducible and comparative analysis of our proposed data curation strategy.

A.2.2 Training Details

General Instruction-Following Experiment. All experiments are conducted on four A100 80GB GPUs and an Intel(R) Xeon(R) Platinum 8350C CPU. We use the `databricks-dolly-15K` dataset (Conover et al., 2023), comprising 15,000 human-generated instruction-response pairs. Samples exceeding the maximum context length are truncated to fit model input constraints. The dataset is randomly divided into 12.5K training, 1K validation, and 0.5K test instances.

Hyperparameters—learning rates from {5e-4, 1e-4, 5e-5} and batch sizes {8, 16}—are chosen based on validation set performance. To ensure fair comparison, total training steps are matched between baseline models and our framework. Specifically, while baselines are trained for 20 epochs on the full dataset, our method divides training into 4 stages—each lasting 8 epochs—on progressively larger subsets: 25%, 50%, 75%, and 100% of the training data, resulting in matched total training steps. The optimal checkpoint is selected based on Rouge-L scores on the validation set, consistent with prior work showing its alignment with human evaluation (Agarwal et al., 2024).

Task-Specific Experiments. For mathematical reasoning and code generation, we start from models fine-tuned on `MetaMathQA` (Yu et al., 2023) for 100K steps. Step-by-step reasoning is elicited via chain-of-thought prompting (Wei et al., 2022) using the prompt: *"Please reason step by step, and put your final answer within* `\boxed{}`*."* Baseline models are trained for 2 epochs with a fixed learning rate of 5×10^{-5}. Across all tasks, we maximize the per-GPU batch size under the 4-A100 constraint, adjusting gradient accumulation steps to maintain an effective batch size of 128.

We employ LoRA (Low-Rank Adaptation) (Hu et al., 2022) for parameter-efficient fine-tuning of OpenLLaMA2, Qwen2.5-Math, and Qwen2.5-Coder in their respective tasks, reducing computational cost and accelerating training.

Hyperparameter Settings. The hyperparameters for the EVO-Curate framework are configured as follows:

- **Curriculum Stages**: The number of curriculum stages is set to $M = 4$.

- **Stage Training**: Each stage is trained for 40% of the baseline epoch count to maintain parity in the total number of training steps.

- **Checkpoint Initialization**: Training for each subsequent stage initializes from the final checkpoint of its preceding stage.

- **Knowledge Distillation**: For KD experiments, we set the balancing coefficient $\alpha = 0.5$ and the temperature $\tau = 1$.

A.2.3 Evaluation Details

All evaluations are conducted on a single NVIDIA A100 80GB GPU, adhering to the evaluation protocol outlined in Gu et al. (2023).

Instruction-following. A fixed prompt template is used during evaluation, as shown in Fig. 5, to ensure consistent input formatting across all models. During inference, responses are generated with a temperature of 1.0 and a maximum output length of 512 tokens. To improve estimate reliability, five responses per prompt are sampled using distinct random seeds ({10, 20, 30, 40, 50}), and results are averaged across these runs.

```
Below is an instruction that describes a task.
Write a response that appropriately completes
the request.

### Instruction:
{instruction}

### Input:
{input}

### Response:
```

Figure 5: Prompt template used for training and evaluation in instruction-following experiments, adapted from (Gu et al., 2023)

For the LLM-as-a-Judge evaluation (Zheng et al., 2023), we perform pairwise comparisons using the prompt template in Fig. 6, with the judging LLM's temperature set to 0.7. For each model family, the outputs of a model trained with our strategy are compared against the outputs of a fixed baseline from the same family: Qwen2.5-1.5B for the Qwen2.5 family, GPT2-1.5B for the GPT2 family, and OpenLLaMA2-7B for the OpenLLaMA2 family.

Mathematical Reasoning and Code Generation. We utilize task-specific prompt templates during inference: Fig. 7 for mathematical reasoning and Fig. 8 for code generation. Following Yang et al. (2024a), we adopt an 8-shot prompt for GSM8K and a 4-shot prompt for MATH to elicit chain-of-thought reasoning, with all in-context examples sourced from their work. Responses are produced via greedy decoding with a maximum generation length of 1024 tokens to accommodate extended reasoning traces. For code generation, we adopt the EvalPlus framework (Liu et al., 2023),

```
### System:
Please act as an impartial judge and evaluate the quality of
the responses provided by two AI assistants to the user
question displayed below. You should choose the assistant
that follows the user's instructions and answers the user's
question better. Your evaluation should consider factors such
as the helpfulness, relevance, accuracy, depth, creativity,
and level of detail of their responses. Begin your evaluation
by comparing the two responses and provide a short
explanation. Avoid any position biases and ensure that the
order in which the responses were presented does not
influence your decision. Do not allow the length of the
responses to influence your evaluation. Do not favor certain
names of the assistants. Be as objective as possible. After
providing your explanation, output your final verdict by
strictly following this format: "[[A]]" if assistant A is
better, "[[B]]" if assistant B is better, and "[[C]]" for
a tie.

### User Question:
{question}

[The Start of Assistant A's Answer]
{answer a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer b}
[The End of Assistant B's Answer]
```

Figure 6: Pairwise comparison prompt used in LLM-as-a-Judge evaluation, adapted from (Zheng et al., 2023)

which extends the HumanEval benchmark by enriching test cases and applying automated validation, thereby offering a more comprehensive and robust evaluation of functional correctness.

```
<8 examples> # for GSM8K
<4 examples> # for MATH

### Problem:
{instruction}
Please reason step by step, and put your final answer within
\boxed{}.

### Solution:
```

Figure 7: The prompt used in mathematical reasoning experiments, adapted from (Yang et al., 2024a)

```
Please provide a self-contained Python script that solves the
following problem in a markdown code block:

### Problem:
{instruction}

### Response:
Below is a Python script with a self-contained function that
solves the problem and passes corresponding tests:
```

Figure 8: The prompt used in code generation experiments, adapted from (Hui et al., 2024)

Below, we describe the datasets used for training and evaluation:

- `databricks-dolly-15K` (Conover et al., 2023): An open-source dataset containing 15,000 human-written instruction-response pairs across diverse categories such as brainstorming, classification, closed QA, genera-

tion, information extraction, open QA, and summarization (Ouyang et al., 2022).

- `self-instruct-eval` (Wang et al., 2022a): A framework that leverages model-generated outputs to synthesize instructional data. It includes 52,000 instructions and 82,000 input-output pairs for training, along with 252 expert-written tasks and an additional 50,000 public examples for evaluation.

- `vicuna-eval` (Chiang et al., 2023): A benchmark consisting of 80 challenging questions originally used to evaluate Vicuna, designed to test complex reasoning and instruction-following capabilities.

- `supernatural-instructions` (Wang et al., 2022b): A comprehensive benchmark comprising 1,616 distinct NLP tasks with expert-written instructions, spanning 76 task types. The test set contains approximately 9,000 samples drawn from 119 tasks.

- `unnatural-instructions-core` (Honovich et al., 2022): A large-scale dataset of 240,000 machine-generated instructions, demonstrating that high-quality synthetic data can rival human-written data for training language models. The core subset consists of 66,000 samples.

- `GSM8K` (Cobbe et al., 2021): A dataset of 8,500 linguistically diverse grade school math word problems, each requiring multi-step arithmetic reasoning. Its emphasis on structured problem decomposition makes it a standard benchmark for assessing reasoning coherence in distilled models.

- `MetaMathQA` (Yu et al., 2023): A reasoning-oriented dataset constructed via question bootstrapping, where source math problems are expanded through forward and backward reasoning, along with paraphrasing. It includes 39,500 training examples and is tailored to improve generalization in mathematical reasoning tasks.

- `WizardCoder` (Luo et al., 2023): A code generation dataset derived from Code Alpaca using the Evol-Instruct framework. It applies iterative complexity-enhancing transformations—such as deepening logical reasoning, adding constraints, and injecting code errors—to produce approximately 78,000 high-quality programming prompts, supporting rigorous training for code synthesis.

- `MBPP` (Austin et al., 2021): A benchmark of approximately 1,000 Python programming tasks created by crowdworkers, aimed at evaluating beginner-level coding proficiency. Each task includes a natural language description, reference solution, and three test cases. A manually verified subset ensures high accuracy and reliability for evaluation purposes.

A.2.4 Baseline Description

This section provides formal definitions of the KD loss functions used in our experiments. As described in Eq. (2), the KD loss measures the divergence between the teacher model distribution p and the student model distribution q_θ using a distance function $D(\cdot||\cdot)$. Below are the specific formulations considered in this study.

The Kullback–Leibler Divergence (KLD) (Hinton et al., 2015) is defined as:

$$D_{KLD}(p \parallel q_\theta) = \mathbb{E}_{y\sim p}\left[\log \frac{p(y)}{q_\theta(y)}\right], \tag{7}$$

where y is sampled from the teacher distribution p.

The Reverse KLD (RKL) is given by:

$$D_{RKL}(q_\theta \parallel p) = \mathbb{E}_{y\sim p}\left[\log \frac{q_\theta(y)}{p(y)}\right]. \tag{8}$$

The Jensen–Shannon Divergence (JSD) (Agarwal et al., 2024) is formulated as:

$$D_{JSD} = \frac{1}{2}\mathbb{E}_{y\sim p}\left[\log \frac{p(y)}{m(y)}\right] + \frac{1}{2}\mathbb{E}_{y\sim p}\left[\log \frac{q_\theta(y)}{m(y)}\right], \tag{9}$$

where $m(\cdot) = \frac{1}{2}p(\cdot) + \frac{1}{2}q_\theta(\cdot)$ is the average of the two distributions.

The Total Variation Distance (TVD) (Wen et al., 2023) is defined as:

$$D_{TVD} = \frac{1}{2}\sum_y |q_\theta(y) - p(y)|. \tag{10}$$

The Forward Skew KLD (SKL) (Ko et al., 2024) is expressed as:

$$D_{SKL}(p \parallel q_\theta) = D_{KLD}\left(p \parallel \beta \cdot p + (1-\beta)\cdot q_\theta\right), \tag{11}$$

where $\beta \in [0, 1]$ controls the mixing ratio between the teacher and student distributions. These loss functions represent commonly used strategies for aligning student and teacher distributions in KD.

Table 7: Performance comparison of instruction following for GPT 2 0.1B and OpenLLaMA2 3B. The table evaluates the generalizability of EVO Curate across smaller and mid scale architectures under SFT and KD paradigms.

	SFT: GPT2-0.1B; KD: GPT2-1.5B → GPT2-0.1B												
	DollyEval		**SelfInst**		**VicunaEval**		**S-NI**		**UnNI**		**Avg.**		
Methods	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	*R-L*	*WR*	**Improvement**
SFT	23.33	47.04	10.01	42.28	14.72	48.31	16.38	36.32	19.57	39.37	16.80	42.67	-
+EVO-Curate	**25.48**	50.39	11.15	45.17	**16.69**	**51.59**	18.57	39.15	**23.97**	41.92	**19.17**	45.64	14.11%
SeqKD	23.72	48.21	11.23	43.92	14.31	48.05	16.48	37.81	19.81	43.59	17.11	44.32	-
+EVO-Curate	24.10	**51.33**	**12.88**	**46.98**	15.47	50.15	**18.75**	**39.59**	21.47	**46.84**	18.53	**46.98**	8.32%
KLD	23.49	47.30	10.33	42.78	14.96	49.11	19.71	40.80	22.01	42.04	18.10	44.41	-
+EVO-Curate	25.37	49.47	12.45	44.47	16.51	52.26	22.56	43.21	23.79	45.32	20.14	46.94	11.26%
RKL	23.79	47.87	12.13	46.96	14.94	48.67	23.81	45.38	22.52	42.63	19.44	46.30	-
+EVO-Curate	25.19	**51.58**	14.13	49.52	16.88	51.72	25.97	47.25	25.15	45.08	21.47	49.03	10.43%
JSD	24.07	48.66	11.38	45.59	15.87	50.79	22.84	44.20	23.06	43.30	19.44	46.51	-
+EVO-Curate	**26.97**	50.21	13.06	47.97	**17.99**	52.72	25.82	47.19	25.19	44.90	21.81	48.60	12.15%
TVD	24.32	48.25	11.09	45.01	15.51	49.70	25.93	47.66	26.55	46.96	20.68	47.52	-
+EVO-Curate	26.61	50.76	13.72	47.81	17.79	51.49	**29.29**	**49.70**	28.57	**48.72**	**23.19**	**49.70**	12.16%
SKL	24.24	48.64	12.27	47.23	15.71	50.15	23.33	45.06	24.02	44.25	19.91	47.07	-
+EVO-Curate	26.48	51.03	**14.61**	**50.33**	17.63	**52.91**	26.70	47.55	**28.89**	46.15	22.86	49.59	14.79%
	SFT: OpenLLaMA2-3B; KD: OpenLLaMA2-7B → OpenLLaMA2-3B												
SFT	24.87	49.20	18.78	50.29	16.50	48.99	28.65	47.87	28.73	47.61	23.51	48.79	-
+EVO-Curate	**27.14**	**51.84**	**20.14**	**52.48**	17.45	49.31	**32.87**	**48.31**	**33.98**	**49.86**	**26.32**	**50.36**	11.95%
SeqKD	23.11	46.31	17.14	47.18	17.06	48.29	28.43	47.93	28.22	46.38	22.79	47.22	-
+EVO-Curate	26.43	48.29	19.78	49.41	**18.72**	**50.07**	30.88	48.29	31.74	48.52	25.51	48.92	11.93%
KLD	23.76	49.68	17.48	48.05	15.68	47.97	28.42	48.10	27.48	46.75	22.56	48.11	-
+EVO-Curate	**27.31**	51.47	18.42	49.80	18.03	49.03	35.12	50.71	34.46	48.31	26.67	49.86	18.19%
RKL	26.67	50.78	18.78	49.71	18.43	53.67	31.27	50.42	32.01	52.18	25.43	51.35	-
+EVO-Curate	27.05	50.94	19.66	50.32	18.35	**54.37**	33.98	51.98	**35.72**	**53.69**	26.95	**52.26**	5.98%
JSD	25.64	50.34	17.18	47.54	15.56	47.58	27.93	49.29	27.39	46.48	22.74	48.25	-
+EVO-Curate	26.91	51.49	18.84	48.53	17.62	49.04	34.70	50.71	35.01	47.53	26.62	49.46	17.04%
TVD	24.62	48.57	18.32	48.90	15.85	48.49	29.01	48.54	28.08	47.54	23.18	48.41	-
+EVO-Curate	27.03	49.39	20.52	49.72	**18.43**	50.14	**35.88**	49.89	35.63	49.28	**27.50**	49.68	18.65%
SKL	24.99	49.08	18.77	49.54	16.69	49.92	29.71	50.86	28.49	48.02	23.73	49.48	-
+EVO-Curate	27.05	**51.73**	**20.99**	**50.86**	18.41	52.31	34.73	**53.81**	35.58	50.83	27.35	51.91	15.26%

Note: Benchmark abbreviations follow Table 1. Average (Avg.) ROUGE L is calculated across all five evaluation datasets. **Improvement** indicates the relative performance increase provided by the **EVO Curate** framework over the respective baseline. Results are averaged over 5 seeds. **Bold** indicate the best performance within each category.

A.3 Additional Results

In Tab. 7, we present evaluation results showing ROUGE-L scores across five random seeds for GPT2-0.1B and OpenLLaMA2-3B as the target model, to show the effectiveness and generalizability of EVO-Curate framework on SFT and KD across different LLMs.

Table 8: Multi-judge validation. Winning rates (%) on instruction-following benchmarks evaluated by Qwen3-Max (Qwen2.5-0.5B student; 1.5B teacher for KD).

Methods	DollyEval	SelfInst	VicunaEval	S-NI	UnNI
SFT	48.32	48.01	50.13	41.35	45.03
+ EVO-Curate	51.91	49.52	52.24	45.67	46.37
KD	53.83	50.72	54.55	51.31	50.01
+ EVO-Curate	57.63	54.13	58.12	53.72	52.84

A.4 Multi-Judge Validation

To validate the reliability of our findings beyond a single LLM judge, we conduct additional evaluations using Qwen3-Max as an alternative judge on five instruction-following benchmarks. Tab. 8 presents winning rates for Qwen2.5-0.5B under SFT and KD. The results are consistent with our primary evaluations using DeepSeek-V3-0324 (Tab. 1), confirming the robustness of EVO-Curate's improvements across different evaluation frameworks.

A.5 Related Work

Curriculum Learning in LLM Post-Training CL structures the sequencing of training samples to enhance optimization, primarily by progressing from simpler to more complex instances (Bengio et al., 2009). While widely explored in computer vision (Xiang et al., 2020; Li et al., 2023), its application to LLM post-training remains an emerging field. In the context of KD, recent approaches like MPDistil (Sengupta et al., 2023) and Confucius (Gao et al., 2024) utilize reinforcement learning or task-specific designs to schedule curricula. More recently, POCL (Liu and Zhang, 2025) and DA-KD (He et al., 2025) have introduced plug and play frameworks to adjust training sets based on perceived sample difficulty. However, existing methods often rely on static difficulty metrics or auxiliary models that incur high computational overhead. Most importantly, they frequently neglect the dynamic interplay between evolving student capacity and the necessity of maintaining dataset diversity. EVO-Curate addresses these gaps by implementing a student-aware, entropy-maximizing strategy that

adapts to the model's maturing state in real-time.

Theoretical Foundations of Curriculum Strategies From an optimization perspective, CL functions as a continuation method for non-convex objectives. By initially focusing on high-utility, "easy" instances, CL smooths the loss landscape, allowing the model to capture the core task distribution before refining on complex edge cases (Wang et al., 2021). This staged refinement mitigates gradient instability and catastrophic forgetting, which are common when shifting from massive-scale pretraining to specialized supervised datasets (Bai et al., 2025). In KD specifically, CL facilitates smoother alignment between teacher and student distributions by addressing the capacity gap. Early exposure to samples where the student's distribution q_θ is closer to the teacher's p provides stable gradient signals (Ko et al., 2025). As the student evolves, incorporating harder samples approximates the benefits of on-policy KD strategies (Agarwal et al., 2024) without the prohibitive cost of iterative rollouts. EVO-Curate builds upon these foundations by dynamically recalibrating difficulty and sampling probabilities, ensuring that the training trajectory remains optimally aligned with the student's current optimization state.