SPARD: Self-Paced Curriculum for RL Alignment via Integrating Reward Dynamics and Data Utility

Xuyang Zhi[1], Peilun Zhou[2], Chengqiang Lu[2], Hang Lv[1], Yiwei Liang[1], Rongyang Zhang[1], Yan Gao[2], Yi Wu[2], Yao Hu[2], Hongchao Gu[1], Defu Lian[1], Hao Wang[1]*, Enhong Chen[1]*

[1]University of Science and Technology of China, [2]Xiaohongshu Inc.

{zxy_zds,lvhang1001,ywliang,zhangry13,hcgu}@mail.ustc.edu.cn

{zhoupeilun,lusuo,wanjianyi,luyun2,xiahou}@xiaohongshu.com

{liandefu,wanghao3,cheneh}@ustc.edu.cn

Abstract

The evolution of Large Language Models (LLMs) is shifting the focus from single, verifiable tasks toward complex, open-ended real-world scenarios, imposing significant challenges on the post-training phase. In these settings, the scale and complexity of reward systems have grown significantly, transitioning toward multi-objective formulations that encompass a comprehensive spectrum of model capabilities and application contexts. However, traditional methods typically rely on fixed reward weights, ignoring non-stationary learning dynamics and struggling with data heterogeneity across dimensions. To address these issues, we propose SPARD, a framework that establishes an automated, self-paced curriculum by perceiving learning progress to dynamically adjust multi-objective reward weights and data importance, thereby synchronizing learning intent with data utility for optimal performance. Extensive experiments across multiple benchmarks demonstrate that SPARD significantly enhances model capabilities across all domains. Our code is publicly available at https://github.com/USTC-StarTeam/SPARD.

1 Introduction

Recently, Large Language Models (LLMs) have seamlessly integrated into people's daily lives and professional workflows. As application scenarios become increasingly diverse and complex, the capability evolution of LLMs is accelerating from single verifiable tasks such as mathematical reasoning and code generation (DeepSeek-AI et al., 2025; Lambert et al., 2025; Zeng et al., 2025) toward open-ended real-world scenes like general dialogue and deepresearch (Shao et al., 2025; Huang et al., 2024; Yu et al., 2025a; Liang et al., 2025; Zhang et al., 2025; Shen et al., 2025). This paradigm shift imposes significantly higher demands on the post-training phase, not only requiring the model to uphold objective factual accuracy but also demanding it to cater to subjective perceptual preferences.

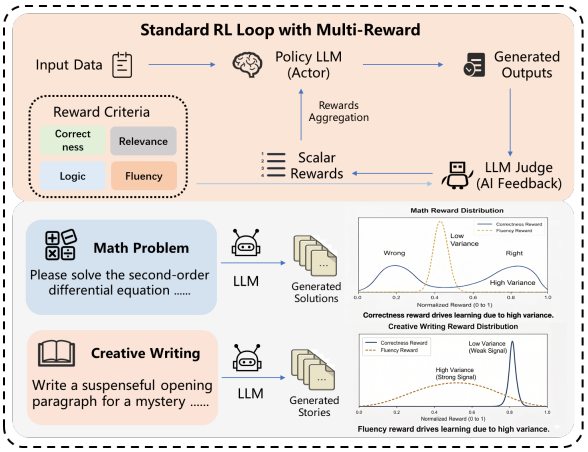

Figure 1: Illustration of the standard **Multi-Reward** RL loop and examples of training characteristics across diverse data types. The upper panel depicts the workflow of generating multi-reward via an LLM judge and aggregating them for policy updates. The lower panel highlights **data heterogeneity**, demonstrating that different types of input data differentially impact specific reward dimensions during training.

In these scenarios, the definition of rewards has evolved into multi-objective frameworks covering diverse criteria like correctness and fluency (Huang et al., 2025b; Wu et al., 2026; Gu et al., 2025), as shown in Figure 1. However, effectively leveraging these multi-dimensional signals remains a significant challenge. Prevailing methods typically aggregate signals using fixed weights, ignoring non-stationary learning dynamics. Consequently, static strategies risk over-optimizing dimensions with diminishing returns while neglecting bottlenecks (Chen et al., 2025a; Min et al., 2024). This issue is further exacerbated by data heterogeneity, where a training example that is highly informative for one criterion (e.g., correctness) may be suboptimal for another (e.g., fluency). As a result, static paradigms lack the flexibility to adapt to evolving bottlenecks and varying data utility.

*Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 47402–47422

To address these challenges, methods such as RaR (Gunjal et al., 2025) and MPO (Kim et al., 2025) implicitly synthesize multi-criterion into a single reward signal by incorporating multiple dimensions into a prompt for judge models to output a holistic score, which obscures the granularity of supervision and hinders the model from localizing specific optimization directions. Alternatively, dynamic strategies like DRBO (Chen et al., 2025a) and MDO (Ryu et al., 2024) attempt to mitigate weaknesses by prioritizing objectives with lower scores. However, these approaches overlook data heterogeneity and risk leveraging inappropriate data samples for the targeted capabilities, leading to inefficient optimization and inter-objective interference. Conversely, Omni-Thinker (Li et al., 2025) and Rubicon (Huang et al., 2025b) address data variance through curriculum-style schedules that transition from strongly constrained tasks to weakly constrained generation, yet rely on static progression plans that lack the flexibility to adapt to the real-time evolution of model capabilities.

To overcome these limitations, we propose SPARD, a framework that establishes an automated, **S**elf-**P**aced curriculum for RL **A**lignment by perceiving learning progress to synchronize **R**eward Dynamics with **D**ata utility. Specifically, we treat learning progress as a signal to dynamically adjust reward weights, directing the model's attention toward dimensions with significant remaining improvement potential. In parallel, SPARD implements adaptive data prioritization by upweighting sample categories that are highly aligned with stage-specific objectives and yield the largest marginal gains. This integrated mechanism ensures that as the model evolves, limited training compute remains precisely focused on the most promising objectives and the most informative data samples.

To sum up, our contributions are threefold:

- We propose SPARD, an automated curriculum framework that leverages real-time learning progress to enable self-paced learning for complex, open-ended generation tasks, dynamically guiding capability acquisition through increasingly challenging stages.
- We introduce a unified optimization framework that couples reward weight adjustment with adaptive data importance weighting. This closed-loop system synchronizes learning intent with data utility, overcoming the limitations of single-sided curriculum strategies.
- Extensive experiments across multiple benchmarks demonstrate that SPARD consistently enhances model capabilities across diverse dimensions. Further analysis validates the framework's advantages in learning efficiency and stability, validating the effectiveness of our proposed framework.

2 Related Works

Reinforcement Learning Alignment via Feedback To navigate increasingly sophisticated LLM scenarios, hybrid reward strategies integrate multidimensional feedback signals to satisfy fine-grained quality benchmarks across open-ended tasks (Liao et al., 2025; Liu et al., 2025a,b). For example, Writing-Zero (Jia et al., 2025b) uses a Pairwise Generative Reward Model to convert self-critique into verifiable feedback for creative writing. QA-LIGN (Dineen et al., 2025) and RLCF (Viswanathan et al., 2025) further decompose evaluation into explicit principles, delivering fine-grained feedback that targets specific issues in logic or style. This idea has also been extended to multimodal reasoning, where process-level feedback guides step-by-step reasoning alongside outcome rewards (Jia et al., 2025a). Despite these advances, optimizing multiple forms of feedback remains challenging: most methods adopt static aggregation, which is often unable to adapt to shifting training dynamics, limiting performance gains.

Curriculum Learning for Reinforcement learning Curriculum learning structures training by progressing from easier to harder examples and is widely used in reinforcement learning to stabilize optimization (Team et al., 2025; Wen et al., 2025). Rubicon (Gunjal et al., 2025) and Omni-Thinker (Li et al., 2025) follow a similar two-stage scheme, first training on strongly constrained tasks and then fine-tuning on more open-ended questions. However, these curricula are typically static and fail to adapt to the model's evolving competence. Beyond static schedules, some methods (Chen et al., 2025b; Wang et al., 2025) estimate difficulty from model-based priors and cast data reweighting as a multi-armed bandit problem to adjust sampling weights online, but they still rely on heuristic difficulty signals or annotations, limiting applicability when difficulty is ambiguous. In contrast, we propose a method that adaptively schedules both reward objectives and data importance based on online learning progress, allowing the curriculum to

emerge from feedback rather than a fixed syllabus.

3 Method

3.1 Preliminaries

Task Formulation An LLM π_θ (with parameters θ) defines a probability distribution over response sequences y given a query $x \sim \mathcal{D}$. To align LLMs with desired behaviors, we formulate language generation as a reinforcement learning (RL) problem. The policy π_θ receives a scalar reward $r(x, y) \in \mathbb{R}$ that reflects the quality of the generation. The training objective is to optimize the policy parameters θ to maximize the expected reward over the dataset:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}[r(x, y)]. \tag{1}$$

Group Relative Policy Optimization (GRPO) To optimize the policy efficiently without an additional value network, we employ GRPO algorithm (Shao et al., 2024). For each query x, the algorithm samples a group of G outputs $\{y_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\text{old}}}$. The policy π_θ is updated by maximizing the following surrogate objective:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\Bigg\{\min\Big(\rho_{i,t}\hat{A}_i, \text{clip}\big(\rho_{i,t}, 1-\epsilon, 1+\epsilon\big)\hat{A}_i\Big) - \beta D_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\Bigg\}. \tag{2}$$

where $\rho_{i,t} = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})}$ is the importance ratio, ϵ is the clipping parameter, and β controls the KL-divergence regularization. Crucially, GRPO estimates the baseline directly from group statistics. The advantage $\hat{A}_i$ for the i-th response is computed by standardizing the rewards within the group:

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_1, \ldots, r_G\})}{\text{std}(\{r_1, \ldots, r_G\})}. \tag{3}$$

Here, r_i denotes the scalar reward for response y_i. Consequently, the effectiveness of the optimization hinges heavily on the design and construction of this scalar signal r_i.

Multi-Reward Aggregation While scalar rewards suffice for tasks with objective ground truth, open-ended generation necessitates evaluating a diverse array of quality dimensions. We formalize this evaluation using a set of scoring criteria $\mathcal{P} = \{p_k\}_{k=1}^{N}$ to capture a comprehensive spectrum of model capabilities, where an $\text{LLM}_{\text{judge}}$ maps a response y to a multi-dimensional reward vector $\mathbf{r}(y)$ such that $r_k(y) = \text{LLM}_{\text{judge}}(y, p_k)$.

To facilitate RL optimization, existing approaches typically employ *linear scalarization* to derive a unified learning signal:

$$r(x, y) = \sum_{k=1}^{N} w_k^r \cdot r_k(y), \quad \text{s.t.} \sum_{k=1}^{N} w_k^r = 1, \tag{4}$$

where $\{w_k^r\}$ are the static weight hyperparameters.

Although this simplifies optimization, it ignores the *non-stationary learning dynamics* inherent in scaling reward dimensions. In practice, model capabilities exhibit *asynchronous convergence*: different dimensions plateau at varying rates as training progresses. Enforcing fixed weights $\{w_k^r\}$ fails to adapt to this evolution, leading to inefficient gradient allocation and hindering the model's ability to achieve balanced proficiency across the entire objective space.

3.2 Methodology

In this section, we present SPARD, an RL framework that orchestrates an automated, self-paced curriculum. Diverging from fixed weighting schemes that overlook training dynamics, SPARD dynamically aligns the optimization trajectory with the model's evolving proficiency. At its core, the framework leverages Progress-Aware Weight Adaptation 3.2.1 to identify and prioritize capabilities within their prime learning phase. Concurrently, Reward-Attributed Data Rebalancing 3.2.2 assigns adaptive importance weights to training samples, ensuring that the gradient updates are primarily driven by data that yields the highest marginal gains for these targeted objectives. The complete training process is presented in Algorithm 1.

3.2.1 Progress-Aware Weight Adaptation

This module focuses on the dynamic evolution of the reward weight vector $\mathbf{w^r}$ during training. We formulate this process as a dynamic resource allocation problem, where the objective is to direct the limited optimization budget toward dimensions that exhibit the highest learning potential. Static weighting schemes often fail to distinguish between *stagnant dimensions* (where the model has reached a performance plateau) and *active frontiers* (where capabilities are rapidly emerging). To bridge this gap, we treat the stable rate of improvement as a proxy for learnability, identifying dimensions where parameter updates yield the most significant and robust gains.

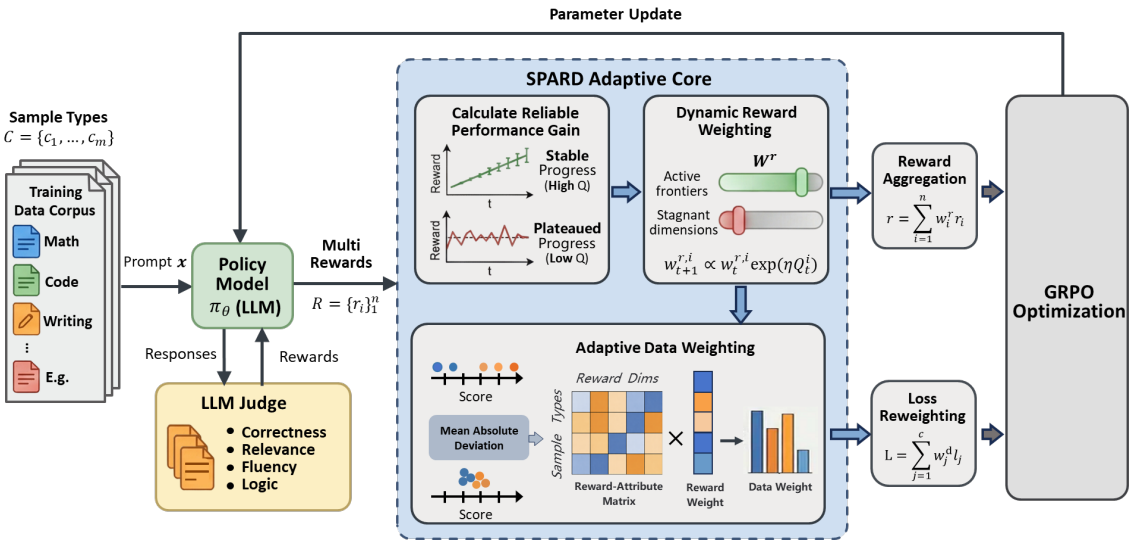

Figure 2: The framework of SPARD, which consists of two main synergistic mechanisms: (1) **Progress-Aware Weight Adaptation** dynamically adjusts reward weights ($\mathbf{w}^r$) based on the reliability of performance gains, and (2) **Reward-Attributed Data Rebalancing** computes data weights ($\mathbf{w}^d$) by aggregating reward importance via a reward-attribute matrix derived from score dispersion. These components jointly guide the optimization to prioritize current learning objectives and leverage the most efficient data.

To capture these stable gains while filtering out transient noise, we draw on the Lower Confidence Bound (LCB) principle. We define the *Reliable Performance Gain* Q_t^i for the i-th reward as:

$$Q_t^i = (\mu_t^i - \beta\sigma_t^i) - (\mu_{t-1}^i - \beta\sigma_{t-1}^i), \quad (5)$$

where μ_t^i and σ_t^i are the Exponential Moving Average (EMA) mean and standard deviation of the i-th reward component, and β is a coefficient penalizing uncertainty. This formulation ensures that Q_t^i is positive only when the mean improvement outweighs the variability, signaling robust acquisition of the corresponding capability. These reward statistics are updated as follows:

$$\begin{aligned} \mu_t^i &= \alpha \cdot r_t^i + (1-\alpha) \cdot \mu_{t-1}^i, \\ \sigma_t^i &= \alpha \cdot \mathrm{std}_t^i + (1-\alpha) \cdot \sigma_{t-1}^i. \end{aligned} \quad (6)$$

To translate these progress signals into updated weights, we aim to maximize alignment with high-growth dimensions while preventing catastrophic forgetting or training instability caused by abrupt weight shifts. We formulate this as a KL-regularized Online Mirror Descent problem:

$$\mathbf{w}_{t+1}^r = \arg\max_{\mathbf{w} \in \Delta_{n-1}} \left(\mathbf{Q}_t^\top \mathbf{w} - \frac{1}{\eta} \mathrm{KL}(\mathbf{w} \parallel \mathbf{w}_t^r) \right). \quad (7)$$

The first term encourages the model to prioritize dimensions with the highest reliable gains, while the KL divergence serves as a proximal constraint to maintain a smooth optimization trajectory. The detailed derivation is provided in Appendix. A.1. The closed-form solution yields an exponentiated gradient update:

$$w_{t+1}^{r,i} = \frac{w_t^{r,i} \exp(\eta Q_t^i)}{\sum_{j=1}^n w_t^{r,j} \exp(\eta Q_t^j)}, \quad (8)$$

where η is the learning rate for weight adaptation, controlling the sensitivity of the curriculum to recent progress. This mechanism naturally amplifies focus on fast-improving capabilities, synchronizing the optimization focus with the model's evolving proficiency frontier.

3.2.2 Reward-Attributed Data Rebalancing

While the evolution of $\mathbf{w}_t^r$ determines the optimization *direction*, the efficiency of this trajectory depends heavily on the underlying data utility. To synchronize data provision with the model's evolving proficiency, we propose a reward-attributed mechanism that realigns the importance of data categories based on their responsiveness to the identified growth areas. This process follows a structured pipeline consisting of reward-data attribution, weight aggregation, and loss reweighting.

We first quantify the sensitivity of each data category $C = \{c_j\}_{j=1}^m$ to different reward dimensions. Intuitively, a data category is most informative for a specific reward i if the candidate responses for its prompts exhibit high score dispersion, providing a

Algorithm 1 SPARD Training Process

Require: Policy π_θ, N criteria $\{p_k\}_{k=1}^N$, M data categories, interval k , η, α, β, μ

Init: $\mathbf{w}^r, \mathbf{w}^d \leftarrow$ Uniform; Stats $\mu, \sigma \leftarrow 0$
for step $t = 1, \ldots, T$ **do**
  Sample batch $\mathcal{B} = \bigcup_{j=1}^M \mathcal{B}_j$ from dataset
  Generate responses $\{y_i\}_{i=1}^G$ and evaluate reward vectors $\{\mathbf{r}_i\}_{i=1}^G$ using $\{p_k\}_{k=1}^N$
  Update statistics μ_t, σ_t based on current rewards {Eq. 6}
  if $t\%k == 0$ **then**
    Compute reliable gain $\mathbf{Q}_t$ and evolve reward weights $\mathbf{w}_t^r$ {Eq. 5, 8}
    Construct reward-data attribution distribution matrix $\tilde{F} \in \mathbb{R}^{N\times M}$ {Eq. 9,10}
    Derive target data importance $\mathbf{u}$ from $\tilde{F}$ and $\mathbf{w}_t^r$ {Eq. 11}
    Update data weights $\mathbf{w}_t^d$ {Eq. 12}
  else
    $\mathbf{w}_t^r, \mathbf{w}_t^d \leftarrow \mathbf{w}_{t-1}^r, \mathbf{w}_{t-1}^d$
  end if
  Aggregate reward $r_i \leftarrow \sum_{k=1}^N w_{t,k}^r r_{i,k}$ for advantage $\hat{A}_i$ {Eq. 3}
  $\theta \leftarrow \theta - \nabla_\theta \sum_j w_{t,j}^d \mathcal{L}_{\text{GRPO}}^{(j)}(\theta)$ {Eq. 13}
end for

clear contrastive signal for the model to distinguish superior behaviors (Shao et al., 2025; Yu et al., 2025b). To formalize this, we construct an *attribution matrix* $F \in \mathbb{R}^{n\times m}$, where F_{ij} measures the utility of candidates in category c_j along reward dimension i. For each prompt b in a recent buffer B_j, we generate a set of G candidate responses $\mathcal{G}_b$ and calculate the score separation using the mean absolute deviation (MAD):

$$F_{ij} = \frac{1}{|B_j|} \sum_{b\in B_j} \frac{1}{G} \sum_{x\in\mathcal{G}_b} \left| r_i(x) - \bar{r}_i^{(b)} \right|, \quad (9)$$

where $\bar{r}_i^{(b)}$ denotes the group mean. Effectively, a larger F_{ij} implies that category c_j yields high-contrast supervision for reward i, while a small F_{ij} suggests that reward signals are clustered for this category, leading to a weak gradient signal.

To translate these raw attribution scores into actionable importance weights, we first normalize F such that each reward dimension i induces a proper distribution over data categories. We apply a temperature-controlled Boltzmann normalization

$$\tilde{F}_{ij} = \frac{\exp(F_{ij}/\mu)}{\sum_{k=1}^m \exp(F_{ik}/\mu)}, \quad (10)$$

where μ controls the sharpness of the mapping. We then define the **target data importance vector** $\mathbf{u} \in \mathbb{R}^m$ by aggregating these normalized attributions with the current reward importance $\mathbf{w}^r$ via a matrix product:

$$u_j = \sum_{i=1}^n w_i^r \tilde{F}_{ij}. \quad (11)$$

Conceptually, u_j is high when category c_j is strongly attributed to reward dimensions that currently exhibit high learning potential. To ensure training stability, the global data weights $\mathbf{w}_t^d$ are updated via an EMA:

$$\mathbf{w}_t^d = \alpha \cdot \mathbf{u} + (1-\alpha) \cdot \mathbf{w}_{t-1}^d. \quad (12)$$

Finally, $\mathbf{w}_t^d$ is used to reweight the training losses across categories. For a minibatch $\mathcal{B} = \bigcup_{j=1}^m \mathcal{B}_j$, the overall objective is formulated as:

$$\mathcal{L}(\theta) = \sum_{j=1}^m w_j^d \cdot \frac{1}{|\mathcal{B}_j|} \sum_{x\in\mathcal{B}_j} \ell(x;\theta). \quad (13)$$

By assigning higher weights to categories that are most conducive to the current optimization priorities, this mechanism ensures that the gradient updates are primarily driven by data samples that maximize cumulative optimization efficiency.

4 Experiments

4.1 Experimental Setting

Dataset We construct our dataset by selecting 5760 prompts from the **WildChat-IF subset** (Zhao et al., 2024). It is sampled from WildChat's conversational prompts, covering a broad range of user queries that closely reflect real-world scenarios. To improve optimization efficiency when training on this heterogeneous instruction collection, we annotate each prompt with a category label using an LLM-based classifier. We partition the dataset into four different categories: **Code**, **Knowledge QA**, **Text Transformation**, and **Creative Writing**. These category tags enable us to analyze the contribution of different data types during training. For more detailed information on data classification, please refer to Appendix A.2.

Baselines To systematically evaluate the effectiveness of our proposed method, we compare it against several representative benchmarks. For direct alignment strategies, we include SFT and DPO, which utilize preferred responses and annotated

Methods	General Capability			Creative Writing		Chat		AVG
	IFEval	GPQA	LCB	Arena-Hard	CW	MT-Bench	WildBench	
Qwen2.5-7B-Instruct								
Base	70.79	33.84	39.75	10.70	48.09	77.93	41.72	46.12
+ SFT	59.70	32.83	35.00	12.50	41.85	77.56	24.05	40.50
+ DPO	74.67	33.54	39.50	12.70	50.49	78.37	43.60	47.55
+ GRPO_{rm}	73.56	34.85	39.75	11.20	50.17	78.12	41.77	47.06
+ GRPO_{imp}	66.91	32.83	40.00	12.40	45.88	78.62	42.47	45.59
+ GRPO_{avg}	73.75	35.35	40.00	14.40	50.89	79.75	**45.08**	48.46
+ *Ours*	**75.78**	**38.38**	**41.75**	**15.60**	**52.49**	**81.38**	44.85	**50.03**
Qwen3-8B								
Base	86.32	42.93	52.50	42.00	69.90	75.06	55.21	60.56
+ SFT	84.73	33.33	45.25	32.70	57.50	71.62	22.09	49.60
+ DPO	85.39	43.94	50.50	43.10	73.15	77.62	54.73	61.20
+ GRPO_{rm}	86.32	43.94	52.50	40.60	72.75	76.81	55.10	61.15
+ GRPO_{imp}	85.95	45.96	51.75	43.10	72.45	77.25	55.73	61.74
+ GRPO_{avg}	86.32	46.97	52.75	44.30	72.16	76.81	**55.89**	62.17
+ *Ours*	**88.17**	**49.49**	**54.75**	**45.90**	**73.95**	**78.31**	55.29	**63.69**

Table 1: Overall performance comparison on multiple benchmarks. The **bold** font indicates the best results and an underline indicates the second-best results.

preference pairs directly from the dataset. Regarding reward-based reinforcement learning methods, we evaluate the following approaches:

- GRPO_{rm}: A standard baseline that optimizes the policy using scalar signals derived from a learned reward model (Bhaskar et al., 2025).
- GRPO_{avg}: A baseline where the LLM judge generates individual rewards for each specific criterion independently. These partial rewards are then aggregated via static, uniform weighting to compute the final signal.
- GRPO_{imp}: An approach consolidating all criteria into a single prompt, delegating the implicit aggregation to the LLM judge to directly yield a final unified score (Gunjal et al., 2025).

Benchmarks We evaluate our models on a comprehensive suite of benchmarks spanning **General Capability**, **Creative Writing**, and **Chat**. Under General Capability, we examine fundamental reasoning and constraint adherence by employing **IFEval** (Zhou et al., 2023) using loose prompt-level accuracy for verifiable instruction following, **LiveCodeBench (LCB)** (Jain et al., 2024) for code generation, and **GPQA-Diamond** (Rein et al., 2024) to probe PhD-level scientific reasoning and domain-specific knowledge. For Creative Writing, we employ **CreativeWritingV3 (CW)** (Paech, 2024) and **Arena-Hard (AH)** (Li et al., 2024a,b) to test the model's generative flexibility and capacity for handling writing tasks. Finally, in the Chat domain, we focus on real-world interaction quality, adopting **WildBench (WB)** (Lin et al., 2024) to assess alignment with human intent and **MT-Bench (MT)** (Zheng et al., 2023) for multiturn dialogue scenarios. Further details can be found in Appendix A.3.

Implementation Details We evaluate our proposed method primarily using **Qwen2.5-7B-Instruct** (Qwen et al., 2025) and **Qwen3-8B** (Yang et al., 2025). Notably, for Qwen3-8B, we explicitly suppress the internal reasoning process during both the training and inference stages. We design eight individual reward metrics covering aspects such as instruction-following, correctness and fluency (He et al., 2025; Liu et al., 2025b; Chen et al., 2025c) , which are then combined to form the final reward function. For our method, we use `DeepSeek-R1` (DeepSeek-AI et al., 2025) as the reward model, and set its sampling temperature to 0.3 to reduce scoring variance and enhance evaluation stability. The full judging prompts are provided in Appendix A.7. For GRPO_{rm}, we adopt `Skywork-v1-Llama-3.1-8B-v0.2` as the reward model (Liu et al., 2024). Regarding the implementation details, we adopt a learning rate of 1×10^{-6}, a prompt batch size of 32, and a group size of $G = 8$. For our proposed SPARD, the hyperparameters are configured as follows: $\alpha = 0.5$, $\beta = 0.1$,

$\mu = 0.1$, $\eta = 3$ and interval $k = 4$. A comprehensive list of all hyperparameters is provided in Appendix A.4.2.

4.2 Overall Performance Evaluation

SPARD improves model performance across all domains Table 1 presents the results across different methods. From the table we observe that SPARD achieves the highest overall average performance for different model backbones, ranking first in the majority of individual domains. This demonstrates our approach's capacity to comprehensively bolster model capabilities while ensuring harmonious improvements across diverse domains. In contrast, standard SFT is prone to distribution shift (Huang et al., 2025a) which can lead to a noticeable degradation of general capabilities despite minor gains in chat performance. Other baseline methods such as DPO and various GRPO implementations enhance chat and writing they sometimes struggle to improve or even maintain performance in rigorous tasks such as coding and instruction following. These results suggest that models might overfit to stylistic rewards which potentially erodes core reasoning abilities.

Notably, the results show that GRPO_{avg} consistently outperforms both GRPO_{rm} and GRPO_{imp}. This performance gap suggests that fine-grained reward signals facilitate superior optimization outcomes. Specifically, explicit aggregation provides transparent guidance by decomposing optimization targets (Viswanathan et al., 2025; Liu et al., 2025a). In contrast, implicit approaches often conflate individual criteria within a monolithic score and consequently obscure specific learning signals. However, despite its competitive performance, GRPO_{avg} remains inferior to SPARD in harmonizing multi-task capabilities. This limitation indicates that static aggregation acts as a performance bottleneck. Its rigid and fixed-weighting scheme lacks the sensitivity to prioritize optimization focus based on the real-time progress of different objectives during training. Conversely, SPARD adaptively re-weights optimization targets by monitoring learning dynamics across training stages, ultimately fostering a synergistic improvement across diverse capabilities.

SPARD achieves faster and more stable reward improvement. Figure 3 illustrates the training trajectories of the **mean reward** and **standard deviation** for Qwen2.5-7B-Instruct under different training methods. As shown in the figure,

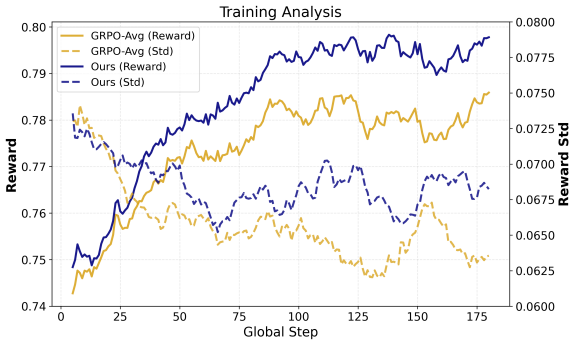

Figure 3: Training trajectories for Qwen2.5-7B-Instruct. To facilitate a clearer comparison of long-term trends and to reduce the visual impact of short-term fluctuations, all curves are smoothed using an exponential moving average (EMA).

SPARD consistently achieves a higher average reward throughout training and exhibits a smaller variance relative to competing approaches, indicating not only stronger overall performance but also improved optimization stability and reduced sensitivity to stochasticity during training.

Detailed trajectories for each individual reward are provided in Figure 5 and Figure 6. As illustrated in these figures, we observe pronounced gains in rewards related to creative writing and chat. Meanwhile, performance on other metrics remains on par with the GRPO_{avg} baseline. This discrepancy is likely attributable to the intrinsic subjectivity and open-ended nature of these tasks, which demand imagination and creativity rather than deterministic precision. Such capabilities can be easily disproportionately affected when training data or reward signals impose overly rigid constraints. Consequently, the rigidity of static weighting makes it difficult to stimulate the full potential of the model in these domains. Overall, SPARD improves both reward maximization efficiency and training robustness, consistently delivering benefits without sacrificing the model's broader applicability across a wide range of tasks.

4.3 Ablation and Further Analysis

Ablation Studies We conduct an ablation study to validate the contributions of the key components in our framework. Specifically, we examined the impact of removing either Progress-Aware Weight Adaptation (PAWA) and Reward-Attributed Data Rebalancing (RADR), which constitute the core mechanisms of SPARD. The results are summarized in Table 2. The experimental results demonstrate the effectiveness of both components. Re-

Method	IF	GPQA	LCB	CW	MT
Qwen2.5-7B-Instruct					
SPARD	**75.78**	**38.38**	**41.75**	**52.49**	**81.38**
w/o PAWA	74.86	37.88	41.75	51.24	80.25
w/o RADR	73.56	36.36	40.00	51.87	80.93
Qwen3-8B					
SPARD	**88.17**	49.49	**54.75**	**73.95**	**78.31**
w/o PAWA	87.98	**51.01**	54.50	72.41	77.06
w/o RADR	86.50	47.47	52.25	73.15	77.68

Table 2: Ablation study on the core mechanisms in SPARD. The **bold** font indicates the best results and an underline indicates the second-best results.

	Method	IF	GPQA	LCB	CW	MT
	Base	60.07	27.78	27.00	40.17	73.37
3b	+Avg	64.51	30.81	26.75	43.41	73.68
	+Ours	**65.24**	**31.31**	**28.50**	**44.16**	**74.25**
	Base	78.03	**46.97**	46.50	55.80	79.84
14b	+Avg	79.48	45.45	**47.50**	60.02	82.25
	+ Ours	**80.96**	**46.97**	47.25	**60.63**	**84.88**
	Base	80.22	47.47	55.25	56.07	84.68
32b	+Avg	**81.70**	48.99	**56.25**	59.35	85.31
	+Ours	80.59	**49.49**	55.75	**60.81**	**86.00**

Table 3: Model Performance Across Different Sizes. Avg refers to the method GRPO_{avg}, which averages rewards across dimensions.

moving PAWA leads to a significant performance drop in open-ended generation tasks, such as Creative Writing and Chat. This indicates that the dynamic weight adjustment mechanism effectively alleviates the rigid constraints imposed by static weighting strategies, thereby preserving the stylistic diversity and flexibility required for subjective tasks. Conversely, relying solely on PAWA inhibits the improvement of general capabilities such as instruction following and coding. This stagnation arises because, without the reward-attribution mechanism provided by RADR, the model struggles to effectively utilize high-utility training samples (e.g., code data) that align with current optimization objectives (e.g., code generation) during the process of improving general capabilities, consequently hindering their further enhancement. Notably, the performance using either of these methods exceeds the current baseline GRPO_{avg}, which indicates that SPARD possesses strong robustness.

We further analyze the sensitivity of SPARD to hyperparameter choices. Compared with standard multi-reward GRPO, SPARD introduces four additional scalar hyperparameters. In all experiments, we use a single fixed configuration across all backbones and benchmarks, without task-specific tuning. As shown in Table 8, varying one hyperparameter at a time results in only mild performance changes on representative benchmarks, indicating that SPARD is robust to reasonable hyperparameter choices in practice. Since these hyperparameters control the magnitude and pace of weight updates, they should not be set excessively large or small: overly large values may make adaptation too aggressive and harm training stability, while overly small values may weaken the effect of dynamic adjustment. Overall, SPARD does not exhibit pronounced sensitivity within a reasonable range, further supporting its practical applicability.

SPARD is effective for different size of models We conduct experiments on Qwen2.5 Instruct models at multiple scales, and the results are reported in Table 3. SPARD demonstrates strong scalability across different model sizes and consistently outperforms both the base model and the static aggregation baseline GRPO_{avg} in overall evaluations. For smaller models (e.g., 3B and 7B), SPARD achieves broad and stable improvements. This suggests that multiple capabilities can benefit simultaneously during the training of smaller-scale models. For larger models, the gains become more selective and primarily manifest in challenging domains such as scientific reasoning and multi-turn dialogue. Meanwhile, performance on relatively saturated capabilities, including instruction following and code generation remains comparable to GRPO_{avg}. These results indicate that as model capacity increases, SPARD adaptively allocates optimization focus according to learning progress to maintain robust training benefits across scales.

Learning dynamics Detailed changes in reward weights and data importance are documented in Appendix A.6.2. As illustrated in Figures, reward weights undergo continuous adjustments throughout the training process to adaptively balance different objectives based on real-time feedback. From a data perspective, figure 7 shows that text transformation tasks receive the highest initial weight and yield immediate gains, reflecting the rapid acquisition of instruction-following abilities. In contrast, the weight for code-related data peaks early and then declines, suggesting that the optimization focus shifts once the model achieves proficiency in code reasoning. As training progresses into the middle and late stages, knowledge QA and creative

writing exhibit an upward trend in weight to occupy a larger proportion of the optimization budget. This pattern confirms that different capability dimensions exhibit non-stationary dynamics. These findings align with recent studies (Gunjal et al., 2025; Huang et al., 2025b) suggesting that verifiable tasks like coding and constrained tasks are learned earlier. Subjective tasks such as long-form QA and creative writing require sustained optimization due to their inherent flexibility.

5 Conclusion

In this work, we proposed SPARD, a self-paced RL framework that orchestrates a dynamic alignment curriculum. By coupling reward dynamics with adaptive data rebalancing, SPARD resolves the inefficiencies inherent in static multi-objective optimization. Our extensive evaluation shows that SPARD consistently enhances model performance across diverse tasks while ensuring training stability. These findings underscore the necessity of progress-aware scheduling in complex alignment scenarios. For future work, we aim to generalize this framework to multimodal domains, further exploring the potential of automated curriculum learning in scaling post-training.

Limitations

While SPARD establishes a robust RL framework for dynamic alignment in open-ended scenes, two limitations warrant consideration. First, the framework relies on high-capability LLMs as reward judges. While this ensures alignment with complex human preferences, it introduces significant inference latency and computational overhead during the online RL loop, potentially constraining scalability and training throughput. Secondly, the current reward aggregation remains a linear approximation. This formulation may oversimplify the optimization landscape, failing to capture the intricate, nonlinear interdependencies among conflicting objectives. Future research should investigate more expressive, non-linear aggregation mechanisms to better navigate these complex relationships.

Ethical Considerations

This work introduces a curriculum-based RL alignment framework that relies on LLM-as-a-judge signals to optimize open-ended generation. A central ethical concern is that judge models may encode subjective preferences, annotation artifacts, and broader social or cultural biases, which can be propagated or even amplified through dynamic reward weighting and data rebalancing. In particular, when judge outputs are used as optimization targets, systematic biases in reward estimation may encourage the policy to overfit to preferred styles, tones, or value-laden behaviors, rather than genuinely improving fairness or overall alignment quality. Our method does not eliminate these risks; rather, it improves the efficiency of learning from such signals. Therefore, models trained with SPARD should be accompanied by careful bias analysis, reward robustness evaluation, and task-specific safety audits before deployment. All experiments in this work are conducted on public datasets and benchmarks, and no private user data or human-subject data are involved.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 62441239, U23A20319 , 62441227, and 62472394), as well as the Anhui Province Science and Technology Inno vation Project (Nos. 202423k09020010 and 202423k09020011), and we sincerely thank the ACL Rolling Review reviewers for their valuable suggestions.

References

Adithya Bhaskar, Xi Ye, and Danqi Chen. 2025. Language models that think, chat better. *Preprint*, arXiv:2509.20357.

Nuo Chen, Yufei Gao, Yongnan Jin, Yan Hu, Anningzhe Gao, Lingyong Yan, and Benyou Wang. 2025a. DRBO: Mitigating the bottleneck effect via dynamic reward balancing in multi-reward LLM optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 8817–8841, Suzhou, China. Association for Computational Linguistics.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. 2025b. Self-evolving curriculum for llm reasoning. *Preprint*, arXiv:2505.14970.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, Hanghang Tong, and Heng Ji. 2025c. Rm-r1: Reward modeling as reasoning. *Preprint*, arXiv:2505.02387.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu,

Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and Ben Zhou. 2025. Qa-lign: Aligning llms through constitutionally decomposed qa. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, page 20619–20642. Association for Computational Linguistics.

Hongchao Gu, Dexun Li, Kuicai Dong, Hao Zhang, Hang Lv, Hao Wang, Defu Lian, Yong Liu, and Enhong Chen. 2025. Rapid: Efficient retrieval-augmented long text generation with writing planning and information discovery. *arXiv preprint arXiv:2503.00751*.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *Preprint*, arXiv:2507.17746.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, Shengjie Bi, Shishir G. Patil, Qi Qi, Shengyu Feng, Julian Katz-Samuels, Richard Yuanzhe Pang, Sujan Gonugondla, Hunter Lang, Yue Yu, and 6 others. 2025. Advancedif: Rubric-based benchmarking and reinforcement learning for advancing llm instruction following. *Preprint*, arXiv:2511.10507.

Yuqing Huang, Rongyang Zhang, Xuesong He, Xuyang Zhi, Hao Wang, Xin Li, Feiyang Xu, Deguang Liu, Huadong Liang, Yi Li, Jian Cui, Zimu Liu, Shijin Wang, Guoping Hu, Guiquan Liu, Qi Liu, Defu Lian, and Enhong Chen. 2024. Chemeval: A comprehensive multi-level chemical evaluation for large language models. *Preprint*, arXiv:2409.13989.

Yuqing Huang, Rongyang Zhang, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, Yao Hu, Xuyang Zhi, Guiquan Liu, Xin Li, Hao Wang, and Enhong Chen. 2025a. Selfaug: Mitigating catastrophic forgetting in retrieval-augmented generation via distribution self-alignment. *Preprint*, arXiv:2509.03934.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, Xijun Gu, Peiyi Tu, Jiaxin Liu, Wenyu Chen, Yuzhuo Fu, Zhiting Fan, Yanmei Gu, Yuanyuan Wang, Zhengkai Yang, and 2 others. 2025b. Reinforcement learning with rubric anchors. *Preprint*, arXiv:2508.12790.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Mengzhao Jia, Zhihan Zhang, Ignacio Cases, Zheyuan Liu, Meng Jiang, and Peng Qi. 2025a. Autorubric-r1v: Rubric-based generative rewards for faithful multimodal reasoning. *Preprint*, arXiv:2510.14738.

Ruipeng Jia, Yunyi Yang, Yongbo Gai, Kai Luo, Shihao Huang, Jianhe Lin, Xiaoxi Jiang, and Guanjun Jiang. 2025b. Writing-zero: Bridge the gap between non-verifiable tasks and verifiable rewards. *Preprint*, arXiv:2506.00103.

Zae Myung Kim, Chanwoo Park, Vipul Raheja, Suin Kim, and Dongyeop Kang. 2025. Toward evaluative thinking: Meta policy optimization with evolving reward models. *Preprint*, arXiv:2504.20157.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. *Preprint*, arXiv:2411.15124.

Derek Li, Jiaming Zhou, Leo Maxime Brunswic, Abbas Ghaddar, Qianyi Sun, Liheng Ma, Yu Luo, Dong Li, Mark Coates, Jianye Hao, and Yingxue Zhang. 2025. Omni-thinker: Scaling multi-task rl in llms with hybrid reward and task scheduling. *Preprint*, arXiv:2507.14783.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2024a. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *Preprint*, arXiv:2406.11939.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2024b. From live data to high-quality benchmarks: The arena-hard pipeline. LMSYS Blog.

Sheng Liang, Hang Lv, Zhihao Wen, Yaxiong Wu, Yongyue Zhang, Hao Wang, and Yong Liu. 2025. Adaptive schema-aware event extraction with retrieval-augmented generation. *arXiv e-prints*, pages arXiv–2505.

Jianxing Liao, Tian Zhang, Xiao Feng, Yusong Zhang, Rui Yang, Haorui Wang, Bosi Wen, Ziying Wang, and Runzhi Shi. 2025. Rlmr: Reinforcement learning with mixed rewards for creative writing. *Preprint*, arXiv:2508.18642.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2024. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. *Preprint*, arXiv:2406.04770.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024. Skywork-reward: Bag of tricks for reward modeling in llms. *Preprint*, arXiv:2410.18451.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2025a. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *Preprint*, arXiv:2510.07743.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. 2025b. Inference-time scaling for generalist reward modeling. *Preprint*, arXiv:2504.02495.

Do June Min, Veronica Perez-Rosas, Kenneth Resnicow, and Rada Mihalcea. 2024. Dynamic reward adjustment in multi-reward reinforcement learning for counselor reflection generation. *Preprint*, arXiv:2403.13578.

Samuel J. Paech. 2024. Eq-bench: An emotional intelligence benchmark for large language models. *Preprint*, arXiv:2312.06281.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Sangwon Ryu, Heejin Do, Yunsu Kim, Gary Lee, and Jungseul Ok. 2024. Multi-dimensional optimization for text summarization via reinforcement learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5858–5871, Bangkok, Thailand. Association for Computational Linguistics.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G. Finlayson, David Sontag, Tyler Murray, Sewon Min, Pradeep Dasigi, Luca Soldaini, Faeze Brahman, Wen tau Yih, Tongshuang Wu, Luke Zettlemoyer, Yoon Kim, and 2 others. 2025. Dr tulu: Reinforcement learning with evolving rubrics for deep research. *Preprint*, arXiv:2511.19399.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Tingjia Shen, Hao Wang, Chuan Qin, Ruijun Sun, Yang Song, Defu Lian, Hengshu Zhu, and Enhong Chen. 2025. Prompting is not enough: Exploring knowledge integration and controllable generation on large language models. *Big Data Mining and Analytics*.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 77 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. 2025. Checklists are better than reward models for aligning language models. *Preprint*, arXiv:2507.18624.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. 2025. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *Preprint*, arXiv:2504.09710.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. 2025. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *Preprint*, arXiv:2503.10460.

Wei Wu, Peilun Zhou, Liyi Chen, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, Yao Hu, and Hui Xiong. 2026. Aligning large language models with searcher preferences. *Preprint*, arXiv:2603.10473.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Haocheng Yu, Yaxiong Wu, Hao Wang, Wei Guo, Yong Liu, Yawen Li, Yuyang Ye, Junping Du, and Enhong Chen. 2025a. Thought-augmented planning for llm-powered interactive recommender agent. *arXiv preprint arXiv:2506.23485*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025b. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *Preprint*, arXiv:2503.18892.

Rongyang Zhang, Yuqing Huang, Chengqiang Lu, Qimeng Wang, Yan Gao, Yi Wu, Yao Hu, Yin Xu, Wei Wang, Hao Wang, and 1 others. 2025. Rag-igbench: Innovative evaluation for rag-based interleaved generation in open-domain question answering. *arXiv e-prints*, pages arXiv–2512.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. 2024. Wildchat: 1m chatgpt interaction logs in the wild. *Preprint*, arXiv:2405.01470.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Hong Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2025. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

A Appendix

A.1 Proofs

Proposition A.1 (Optimal Reward Weight Update)**.** *Given the reliable performance gain vector $Q_t \in \mathbb{R}^n$ and the current weight distribution $w_t^r \in \Delta_{n-1}$, the closed-form solution to the regularization-constrained optimization problem defined in Eq.(7):*

$$w_{t+1}^r = \arg\max_{w \in \Delta_{n-1}} \left(Q_t^\top w - \frac{1}{\eta} D_{KL}(w \parallel w_t^r) \right) \tag{14}$$

is given by the exponentiated gradient update rule:

$$w_{t+1}^{r,i} = \frac{w_{t,i}^r \exp(\eta Q_t^i)}{\sum_{j=1}^n w_{t,j}^r \exp(\eta Q_t^j)} \tag{15}$$

Proof. Let $J(w)$ denote the objective function. Expanding the KL-divergence term, the objective is formulated as:

$$J(w) = \sum_{i=1}^n Q_t^i w_i - \frac{1}{\eta} \sum_{i=1}^n w_i \ln \left(\frac{w_i}{w_{t,i}^r} \right) \tag{16}$$

The negative relative entropy term is strictly concave with respect to w. Consequently, the optimization problem is strictly concave over the probability simplex Δ_{n-1}, guaranteeing the existence of a unique global maximum.

To derive the optimal solution, we construct the Lagrangian $\mathcal{L}(w, \lambda)$ to enforce the simplex constraint $\sum_{i=1}^n w_i = 1$. The non-negativity constraints $w_i > 0$ are implicitly satisfied by the domain of the logarithmic term (acting as a barrier function). The Lagrangian is given by:

$$\begin{aligned} \mathcal{L}(w, \lambda) = &\sum_{i=1}^n Q_t^i w_i - \frac{1}{\eta} \sum_{i=1}^n \left(w_i \ln w_i - w_i \ln w_{t,i}^r \right) \\ &+ \lambda \left(\sum_{i=1}^n w_i - 1 \right) \end{aligned} \tag{17}$$

Where $\lambda \in \mathbb{R}$ is the Lagrange multiplier associated with the equality constraint.

Taking the partial derivative with respect to w_i:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w_i} &= Q_t^i - \frac{1}{\eta} \left(\ln w_i + 1 - \ln w_{t,i}^r \right) + \lambda \\ &= Q_t^i - \frac{1}{\eta} \ln \left(\frac{w_i}{w_{t,i}^r} \right) - \frac{1}{\eta} + \lambda \end{aligned} \tag{18}$$

Rearranging the terms to isolate $\ln w_i$, we obtain:

$$\begin{aligned} \frac{1}{\eta} \ln \left(\frac{w_i}{w_{t,i}^r} \right) &= Q_t^i + \lambda - \frac{1}{\eta} \\ \ln \left(\frac{w_i}{w_{t,i}^r} \right) &= \eta Q_t^i + (\eta\lambda - 1) \end{aligned} \tag{19}$$

Exponentiating both sides yields the functional form of the optimal weights:

$$w_i = w_{t,i}^r \exp(\eta Q_t^i) \cdot \exp(\eta\lambda - 1) \tag{20}$$

Let $Z = \exp(\eta\lambda - 1)$ denote the normalization constant, which is independent of the index i. To determine Z, we enforce the probability constraint $\sum_{j=1}^n w_j = 1$:

$$\sum_{j=1}^n w_j = Z \sum_{j=1}^n w_{t,j}^r \exp(\eta Q_t^j) = 1 \tag{21}$$

Solving for Z, we find:

$$Z = \frac{1}{\sum_{j=1}^n w_{t,j}^r \exp(\eta Q_t^j)} \tag{22}$$

Substituting Z back into Eq. (20), we arrive at the closed-form update rule:

$$w_{t+1}^{r,i} = \frac{w_{t,i}^r \exp(\eta Q_t^i)}{\sum_{j=1}^n w_{t,j}^r \exp(\eta Q_t^j)} \tag{23}$$

□

A.2 Dataset Description

A.2.1 Source and Composition

The WildChat-IF dataset utilized in this work is derived from the WildChat corpus. The original WildChat corpus comprises approximately 1 million user-chatbot conversations consisting of over 2.5 million interaction turns, collected from a publicly available service powered by GPT-3.5 and GPT-4 APIs. We filtered and categorized the samples into four distinct types: Creative Writing (CW), Text Transformation (Text), Code Generation (Code), and Knowledge QA (QA). The final dataset comprises a total of 5,760 samples. As illustrated in Figure 4.

A.2.2 Data Construction

We employed DeepSeek-R1 (DeepSeek-AI et al., 2025), a state-of-the-art language model, to process the raw data. Specifically, DeepSeek-R1 was utilized to categorize the samples based on their semantic intent. The specific prompts designed for this curation and classification process are detailed in Prompt 1

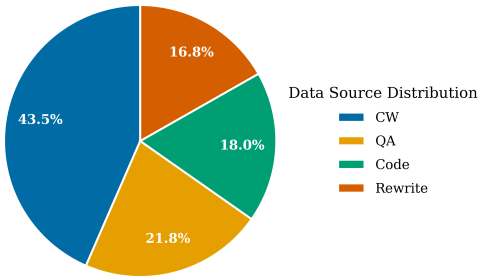

Figure 4: Distribution of selected data sources in the WildChat-IF dataset.

A.3 Evaluation Details

We evaluate SPARD using a suite of seven benchmarks organized into three distinct domains: General Capability, Creative Writing, and Chat.

A.3.1 General Capability

This category targets reasoning and constraint adherence, encompassing verifiable instruction following, code generation, and domain-specific scientific knowledge. We use OpenCompass in GPQA and LCB.

- Instruction-Following Evaluation (IFEval) (Zhou et al., 2023): IFEval assesses the objective ability of models to adhere to strict execution constraints. The dataset comprises approximately 500 prompts covering 25 types of verifiable instructions, such as word count limits and formatting requirements. Unlike model-based judges, IFEval employs programmatic metrics to calculate deterministic constraint satisfaction. We use the **prompt-level loose accuracy** metric for our result.

- Graduate-Level Google-Proof Q&A (GPQA) (Rein et al., 2024): GPQA contains 448 high-difficulty multiple-choice questions spanning biology, physics, and chemistry. Authored by PhD-level domain experts, these questions are designed to be "Google-proof" to resist simple retrieval. It serves as a rigorous test for expert-level reasoning and deep domain knowledge. We use GPT-4o (version: 2024-06-01) as the judge model to calculate the **accuracy**.

- LiveCodeBench (LCB) (Jain et al., 2024): LiveCodeBench evaluates code generation on contest problems published after the model's

training cutoff. The benchmark measures performance via functional correctness (Pass@1) on hidden test cases, ensuring the model generalizes to novel algorithmic problems rather than recalling memorized solutions. We report results on the **Code Generation scenario**.

A.3.2 Creative Writing

This domain stresses the model's generative flexibility, evaluating its capacity to handle complex, open-ended tasks that demand stylistic nuance and high-entropy output

- Creative Writing v3 (CW) (Paech, 2024): CW consists of 32 open-ended prompts designed to elicit nuanced literary output. Responses are evaluated by a strong judge model using a criterion focused on narrative flow and emotional depth. The scoring mechanism is specifically calibrated to minimize length bias and assess subjective quality. We use Claude-3.7 as the judge model to calculate the **judge score**.

- Arena-Hard V2.0(AH) (Li et al., 2024a,b): AH utilizes 500 challenging prompts curated from the Chatbot Arena, selected for their high separability. The evaluation employs a pairwise comparison mechanism where a judge model compares the target against a baseline. The resulting win-rates correlate highly (98.6%) with human preference rankings, serving as a proxy for performance on complex queries. We use GPT-4o as the judge model to calculate the **win rate** in the **creative writing** subset and the baseline model is GPT-o1.

A.3.3 Chat

This category focuses on real-world interaction quality, assessing robustness against diverse, noisy user intents and the maintenance of coherence across multi-turn dialogues.

- MT-Bench (MT) (Zheng et al., 2023): MT-Bench assesses conversational flow and instruction following through 80 high-quality multi-turn questions across eight domains. Each task involves a two-turn dialogue to test context retention. A **judge grades** responses on a scale of 1 to 10 based on helpfulness, relevance, and accuracy. We use GPT-4o as the judge model.

- WildBench (WB) (Lin et al., 2024): Derived from the WildChat corpus, WildBench evaluates models on 1,024 real-world tasks that reflect diverse and noisy user interactions. It uses fine-grained, checklist-based pairwise comparisons (WB-Reward/Score) to assess practical utility across use cases like debugging and information seeking. We use GPT-4o as the judge model and use the **WB-Reward** as our metric.

A.4 More Details of Experiments

A.4.1 Training Reward Definition

To comprehensively evaluate the quality of generated content during the training process and to provide stable learning signals, we define eight reward functions to assess different dimensions of quality, including **correctness**, **detail**, **tune**, **logic**, **relevance**, **instruction following**, and **structure**. Specifically, we employ the Deepseek-R1 model as our judge model, utilizing carefully designed scoring prompts to enable the model to evaluate responses across these various dimensions. The detailed prompts are provided in Appendix A.7.

A.4.2 Hyperparameters

We provide the corresponding hyperparameters for SFT, DPO and GRPO in Table 4, 5, 6. All the training is conducted based on the Ms-Swift (Zhao et al., 2025) framework.

Hyperparameter	Value
Batch size	32
Epochs	1
Learning rate	5e-6
Warmup ratio	0.05
Weight decay	0.1
Adam betas	(0.9, 0.95)
LR scheduler	cosine

Table 4: Supervised fine-tuning (SFT) hyperparameters used in our experiments.

A.5 Additional Results

A.5.1 Results on Llama Backbone

To further evaluate the generality of our method, we additionally conduct experiments on the Llama backbone. Table 7 reports the performance of different training strategies on five representative benchmarks, including instruction following (IFEval), scientific reasoning (GPQA), code generation

Hyperparameter	Value
Batch size	32
Epochs	1
Learning rate	1e-6
DPO β	0.1
Warmup ratio	0.05
Weight decay	0.1
Adam betas	(0.9, 0.95)
LR scheduler	cosine

Table 5: Direct Preference Optimization (DPO).

Hyperparameter	Value
Batch size	32
Epochs	1
Learning rate	1e-6
Warmup ratio	0.05
Weight decay	0.1
Adam betas	(0.9, 0.95)
LR scheduler	cosine
Group size	8
KL coefficient	0.04
Generation temperature	0.7
Judge model	Deepseek r1
Judge temperature	0.3
SPARD α	0.5
SPARD β	0.1
SPARD μ	0.1
SPARD η	3
SPARD interval k	4

Table 6: hyperparameters of GRPO and our method

(LCB), creative writing (CW), and multi-turn dialogue (MT).

As shown in Table 7, our method achieves the best performance across all evaluated benchmarks, demonstrating that the effectiveness of SPARD is not limited to the Qwen series and can generalize to other model backbones. Compared with the base Llama model, our method yields consistent gains on reasoning, coding, creative writing, and dialogue benchmarks, while also slightly improving instruction-following performance.

We also observe that SFT leads to noticeable degradation on several benchmarks, especially IFEval, CW, and MT, suggesting that naive supervised fine-tuning may cause distribution shift and harm the model's original capabilities in open-ended settings. In contrast, GRPOavg consistently improves

Method	**IFEval**	**GPQA**	**LCB**	**CW**	**MT**
Base	75.60	20.20	20.23	49.33	70.31
+ SFT	62.64	24.24	18.23	40.26	68.12
+ GRPO_{avg}	72.64	26.77	24.86	51.81	74.44
+ Ours	**75.78**	**28.28**	**26.38**	**53.25**	**76.63**

Table 7: Performance comparison on the Llama backbone.

over SFT and the base model on most tasks, which further confirms the benefit of explicit multi-reward optimization.

Nevertheless, our method still outperforms GRPOavg on all five benchmarks. In particular, the gains on GPQA, LCB, and MT indicate that dynamically coordinating reward weighting and data rebalancing helps the model better allocate optimization focus across heterogeneous capabilities. These results provide additional evidence that SPARD can serve as a robust and transferable alignment framework beyond the main backbones studied in the paper.

A.5.2 Hyperparameter Sensitivity

To further assess the practical usability of SPARD, we analyze its sensitivity to the additional hyperparameters introduced by our method. Compared with standard multi-reward GRPO, SPARD only introduces four scalar hyperparameters, namely α, β, μ, and η. In all main experiments, we use one fixed configuration across all backbones and benchmarks, i.e., $\alpha = 0.5$, $\beta = 0.1$, $\mu = 0.1$, and $\eta = 3$, without per-task tuning.

These hyperparameters mainly control the update scale and smoothness of reward adaptation and data rebalancing. Intuitively, overly large values may cause unstable updates, while overly small values may slow down adaptation. To examine whether the method is overly sensitive in practice, we vary one hyperparameter at a time while keeping the others fixed, and evaluate the resulting model on five representative benchmarks: IFEval, GPQA, LiveCodeBench (LCB), MTBench (MT), and Creative Writing V3 (CW).

The results are reported in Table 8. Overall, the performance changes remain relatively mild across different settings. Although certain hyperparameter choices lead to small fluctuations on individual benchmarks, the overall trend is stable, suggesting that SPARD does not rely on fragile hyperparameter tuning. These results support the practical robustness of our method and indicate that a single shared hyperparameter setting is sufficient for

Setting	IFEval	GPQA	LCB	MT	CW
$\alpha = 0.1$	75.78	38.38	41.75	51.05	81.19
$\beta = 0.3$	75.04	37.37	40.25	51.90	81.43
$\mu = 15$	74.86	39.39	40.25	52.26	81.38
$\mu = 5$	74.86	38.85	39.75	51.58	80.73
$\eta = 1$	74.30	37.37	39.75	50.60	80.58
$\eta = 5$	76.70	38.38	41.25	51.78	80.93

Table 8: Hyperparameter sensitivity analysis of SPARD. We vary one hyperparameter at a time while fixing the others.

strong performance across tasks.

A.5.3 Rewards Stability and Human Alignment

Given that our framework relies on LLM-as-a-Judge to provide multi-dimensional reward signals, we further examine whether the judge is sufficiently stable and reliable under our reward setting. In particular, prior work has pointed out that LLM-based evaluators may suffer from inconsistency in numerical cardinal scoring. To address this concern, we conduct additional analyses from two aspects: stability across repeated scoring and alignment with human preference.

We first evaluate the stability and consistency of judge scores under repeated trials. Specifically, we randomly sample 1,000 responses and ask the judge model (DeepSeek-R1) to score each response 10 times under the same rubric. During scoring, we use a low decoding temperature to reduce stochasticity. For each reward dimension, we compute the variance of repeated scores for every sampled response, and then aggregate the results across all samples. As shown in Table 9, the score variance remains consistently small across all evaluated dimensions. This indicates that, under our reward design and scoring setup, the judge outputs are stable and repeatable. Therefore, the reward signals used in SPARD are unlikely to be dominated by random fluctuations in the judge's numerical scoring behavior.

We further assess whether these judge scores are aligned with human preference. To this end, we randomly sample 100 responses and manually inspect the judge's reasoning traces together with the final assigned scores. In the vast majority of cases, the judge's rationale and scores are consistent with human intuition, capturing common preference signals such as instruction adherence, relevance, and helpfulness. This suggests that the reward values used in training are not only stable, but also reliable proxies for human preferences in practice.

Metric	Variance
Correctness	0.5960
Detail	0.6350
Fluency	0.6766
Tune	0.6240
Logic	0.6309
Relevance	0.6250
Structure	0.6420
Instruction-following	0.5844

Table 9: Judgement Variance.

Overall, these analyses support that the LLM-as-Judge reward signals in our framework are both stable and reliable under our reward setting.

A.6 More Discussions

A.6.1 Training Reward Analysis

We analyze the training dynamics of Qwen2.5-7B-instruct across eight specific reward dimensions. Figure 5 and Figure 6 present the training curves, detailing the mean reward and its corresponding standard deviation (Std.) throughout the optimization process.

Overall, our method consistently outperforms the GRPO-Avg baseline across all evaluated metrics, demonstrating both higher reward acquisition and improved stability. Specifically:

- **Convergence and Performance:** As shown in Figures 5 and 6, our method exhibits faster convergence rates. It achieves higher mean scores during the training process, particularly in the *Structure*, *Tune*, and *Fluency* dimensions, suggesting a more effective alignment with the reward objectives.

- **Training Stability:** The standard deviation plots (right columns) reveal that our method generally maintains lower or comparable variance throughout the training process. Notably, in both figures, the reduction in std implies that our policy optimization is less prone to mode collapse or instability, leading to more consistent generation quality.

A.6.2 Training dynamics During Training

We analyzed the training dynamics during the training process, including the variation trends of reward weights and data weights. The figure 7, 8

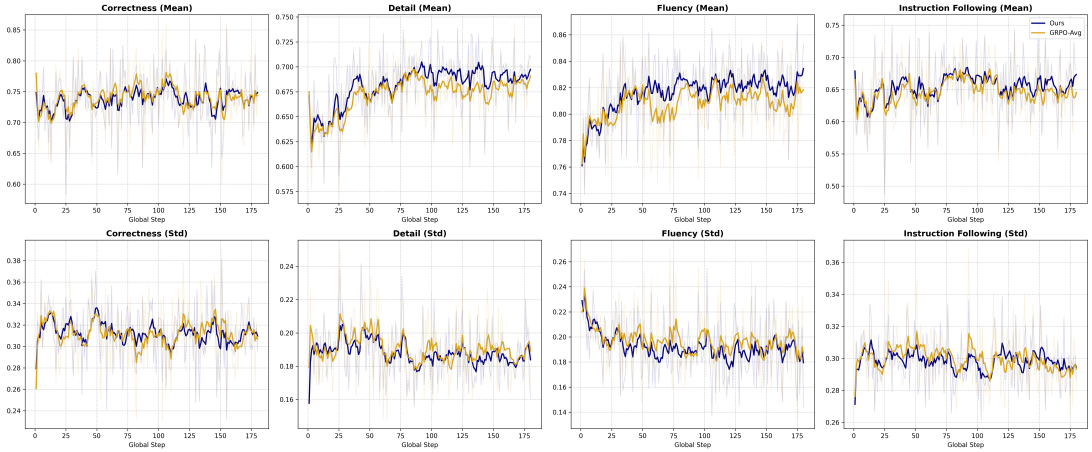

Figure 5: Training dynamics for Correctness, Detail, Fluent, Instruction Following.

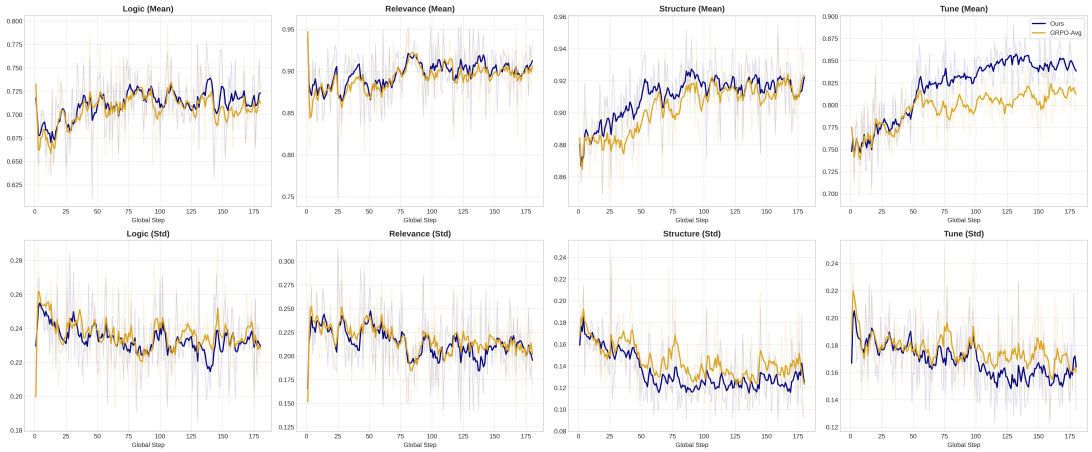

Figure 6: Training dynamics for Logic, Relevant, Structure, and Tune.

below shows the training curves, where the changing trends of both reward weights and data weights throughout training can be observed. This demonstrates that our approach is capable of capturing the model's continuously evolving learning progress.

A.6.3 Computational Cost Analysis

SPARD operates within the standard GRPO framework, introducing dynamic scheduling for reward and data weights. The core mechanisms, Progress-Aware Weight Adaptation (PAWA) and Reward-Attributed Data Rebalancing (RADR), rely solely on statistical aggregates (e.g., EMA and MAD) of the generated rewards. These operations are computationally negligible compared to the policy model's forward and backward passes. Unlike curriculum learning approaches that require external difficulty annotators or separate pre-sorting stages, SPARD adapts online without requiring additional inference calls. Consequently, the computational cost of SPARD remains strictly equivalent to standard Multi-Reward GRPO, while significantly improving optimization efficiency.

A.7 Prompts

We provide a comprehensive breakdown of the prompt protocols utilized for both data construction and evaluation to ensure full reproducibility.

To facilitate granular analysis, Prompt 1 stratifies user queries into four distinct taxonomies: Creative Writing, Question Answering, Code Generation, and Rewriting.

For performance assessment, Prompt 3 are employed to quantify response quality comprehensively(GRPO_{imp}). In this unified prompt structure, we have consolidated the detailed grading rubrics into a single variable placeholder: `{{all criteria}}`. When the model executes this prompt, it will reference the full set of injected criteria to perform a holistic evaluation and generate a comprehensive score based on the combined weights of these standards.

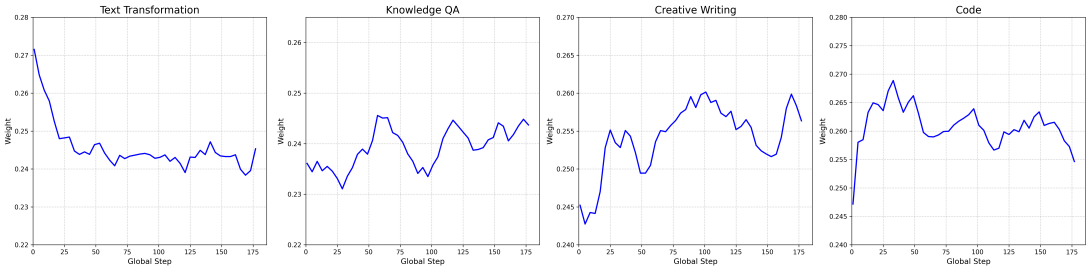

Figure 7: Data Weight Change During Training Process

Figure 8: Reward Weight Change During Training Process

We consolidated eight detailed grading rubrics (including Correctness, Relevance, Level of Detail, Fluency, Logical Flow, Instruction Adherence, Structure, and Tone). Additionally, we provide a representative example, Prompt 4, alongside a unified template, Prompt 2, for other unlisted tasks. You can replace {{DIMENSION}} (evaluation dimensions), {{FOCUS_AREA}} (areas of focus), and {{LEVEL_...}} (specific scoring criteria) in the following template with the actual task content.

Prompt 1: User Query Classification System

[Task Description]
You are an AI assistant responsible for analyzing user queries. Your objective is to classify the user's input into one of four distinct categories based on the content and intent.

[Classification Criteria]

- **0. Code & Software Engineering:**
  - Writing code in specific languages (Python, Java, Kotlin, C++, JavaScript, SQL, HTML, CSS, etc.).
  - Refactoring, compressing, or optimizing existing code.
  - Creating programming exercises or coding problems.
  - Explaining code logic, debugging, performance tuning, or API design.
- **1. Knowledge QA & Problem Solving:**
  - STEM questions (Math, Statistics, Physics, Chemistry, Biology calculations).
  - Exam creation/solving: MCQs, True/False, practice problems; providing answers or validating options.
  - Analyzing, summarizing, or explaining the content of books or articles (e.g., literary genre analysis).
  - General knowledge retrieval: listing examples, defining concepts, comparing product specifications or data.
- **2. Language & Text Rewriting:**
  - Translation, paraphrasing, polishing, or grammar correction.
  - Simplification, summarization, expansion, or shortening of text.
  - Sentence construction based on specific grammar structures.
  - Adjusting writing style, tone, or register.
- **3. Creative Writing & Application Scenarios:**
  - Creative fiction: stories, novel chapters, character design, world-building, fan fiction, scripts, humor.
  - Content creation: articles, speeches, movie reviews, reflections, blog posts, magazine entries.
  - Marketing: Copywriting, headlines, slogans, product descriptions, social media posts.
  - Generating prompts for image generation models (e.g., Midjourney).
  - Role-play, simulated dialogues, or scenario writing.
  - Brainstorming, ideation, and list generation requests.

[User Query]
`{{prompt}}`

[Output Format]
Based on the criteria above, determine the type of the User Query and explain your reasoning. The output must be a JSON object containing the evaluation result and should not contain any other text. The `category` field must be an integer (0 for Code, 1 for Knowledge QA, 2 for Language/Text, 3 for Creative/Scenarios).

```
{
"reason": <string>,   # The reasoning behind your classification
"category": <int>     # The classification result (0, 1, 2, or 3)
}
```

Prompt 2: User Prompt Template: Evaluation of {{DIMENSION}}

[Task Description]
You are to evaluate the {{DIMENSION}} of the response to the user query. Focus exclusively on {{FOCUS_AREA}} using the specified criteria.

[Criteria]

- **Exceptionally Good {{DIMENSION}} (5 points):** {{LEVEL_5_DEFINITION}}
- **Very Good{{DIMENSION}} (4 points):** {{LEVEL_4_DEFINITION}}
- **Adequately Good {{DIMENSION}} (3 points):** {{LEVEL_3_DEFINITION}}
- **Partially Good {{DIMENSION}} (2 points):** {{LEVEL_2_DEFINITION}}
- **Poorly Good{{DIMENSION}} (1 point):** {{LEVEL_1_DEFINITION}}
- **Not Good At All{{DIMENSION}} (0 points):** {{LEVEL_0_DEFINITION}}

[Scoring Rules]

- Provide reasons for each score by indicating specific strengths or deficiencies within the Response. Reference exact text passages to justify the score, ensuring that each reason is concrete and aligns with the criteria requirements.
- Be very STRICT and do not be misled by format or length.
- Scoring Range: Assign an integer score between 0 to 5.

[Input Data]
User Query: `{{prompt}}`
Response: `{{response}}`

[Output Format]
The output should be a JSON object containing the evaluation results for the criterion, and should not contain any other text.

```
{
"reason": <string>, # A detailed rationale substantiating
"score": <integer> # The assigned score
}
```

Prompt 3: Evaluation Prompt: Comprehensive Quality

[Task Description]
You are an expert evaluator. Given a user query and a generated response, please rate the overall quality of the generated response on a scale of 0 to 5 based on how well the response is.

[Scoring Rules]

- Provide reasons for each score by indicating specific strengths or deficiencies within the Response. Reference exact text passages to justify the score.
- Be very STRICT and do not be misled by format or length.
- Based on the general evaluation criteria, state the weights of different criteria, and then provide an overall comprehensive score based on them.
- Consider all criteria holistically when determining your score.

[Criteria]
`{{all criteria}}`

[User Query]
`{{prompt}}`

[Response]
`{{response}}`

[Output Format]
The output should be a JSON object containing the evaluation results for the criterion, and should not contain any other text.

```
{
  "reason": <string>,  # A detailed rationale substantiating the judgment
  "score": <integer>   # The assigned score (0-5)
}
```

Prompt 4: Evaluation Prompt: Level of Detail

[Task Description]
You are to evaluate the level of detail of the response to the user query. Focus exclusively on assessing coverage, specificity, necessary context, and actionable specifics using the criteria below.

[Criteria]

- **Exceptional Detailed (5 points):** Complete coverage of all relevant aspects; highly specific; includes necessary context, assumptions, edge cases, and clear, actionable steps/examples.
- **Very Detailed (4 points):** Strong coverage with minor omissions or shallow spots; mostly specific and actionable with adequate context.
- **Adequately Detailed (3 points):** Covers the main aspects but leaves notable gaps; some specifics/actionable elements present, limited context or edge cases.
- **Partially Detailed (2 points):** Uneven coverage with significant gaps; description is general/vague in many places; lacks sufficient context or actionability.
- **Poorly Detailed (1 point):** Minimal coverage; mostly generic statements; little to no actionable or contextual information.
- **Not Detailed (0 points):** Very brief, off-topic, or lacks assessable detail.

[Scoring Rules]

- Provide reasons for each score by indicating specific strengths or deficiencies within the Response. Reference exact text passages to justify the score, ensuring that each reason is concrete and aligns with the criteria requirements while highlighting key gaps from the ideal answer.
- Be very STRICT and do not be misled by format or length; ensure that the Response is thoroughly evaluated beyond superficial appearances.
- Scoring Range: Assign an integer score between 0 to 5

[User Query]
`{{prompt}}`

[Response]
`{{response}}`

[Output Format]
The output should be a JSON object containing the evaluation results for the criterion, and should not contain any other text.

```
{
"reason": <string>,  # A detailed rationale substantiating the score
"score": <integer>   # An integer from 0 to 5
}
```