Think Natively: Unlocking Multilingual Reasoning with Consistency-Enhanced Reinforcement Learning

Xue Zhang[1,2*], **Yunlong Liang**[3], **Fandong Meng**[3], **Songming Zhang**[1,2], **Kaiyu Huang**[1,2], **Yufeng Chen**[1,2†], **Jinan Xu**[1,2], **Jie Zhou**[3]
[1]Key Laboratory of Big Data & Artificial Intelligence in Transportation, Beijing Jiaotong University
[2]School of Computer Science and Technology, Beijing Jiaotong University, Beijing, China
[3]WeChat AI, Tencent Inc, China
{zhang_xue,smzhang22,chenyf,jaxu}@bjtu.edu.cn

Abstract

Large Reasoning Models (LRMs) have achieved remarkable performance on complex reasoning tasks by adopting the "think-then-answer" paradigm, which enhances both accuracy and interpretability. However, current LRMs exhibit two critical limitations when processing non-English languages: (1) They often struggle to maintain input-output language consistency; (2) They generally perform poorly with wrong reasoning paths and lower answer accuracy compared to English. These limitations significantly compromise the interpretability of reasoning processes and degrade the user experience for non-English speakers, hindering the global deployment of LRMs. To address these limitations, we propose **M-Thinker**, which is trained by the GRPO algorithm that involves a Language Consistency (LC) reward and a novel Cross-lingual Thinking Alignment (CTA) reward. Specifically, the LC reward defines a strict constraint on the language consistency between the input, thought, and answer. Besides, the CTA reward compares the model's non-English reasoning paths with its English reasoning path to transfer its own reasoning capability from English to non-English languages. Through an iterative RL procedure, our M-Thinker-1.5B/4B/7B models not only achieve nearly 100% language consistency and superior performance on two multilingual benchmarks (MMATH and PolyMath), but also exhibit excellent generalization on out-of-domain languages.

1 Introduction

Large reasoning models (LRMs), such as DeepSeek-R1 (DeepSeek-AI, 2025), OpenAI-o3 (OpenAI, 2025), and Qwen3 (Yang et al., 2025a), have achieved impressive performance across a variety of complex reasoning tasks, such as mathematical problem solving, code generation, and logical deduction. A key advantage of these models lies in their response pattern: They first generate an explicit chain of reasoning (Tam et al., 2025) that may include problem decomposition, solution planning, and intermediate verification, and then offer an answer summary. This "think-then-answer" paradigm not only enhances performance but also significantly improves transparency and interpretability of answers (Wang et al., 2025c), making the decision-making process more accessible and trustworthy for users.

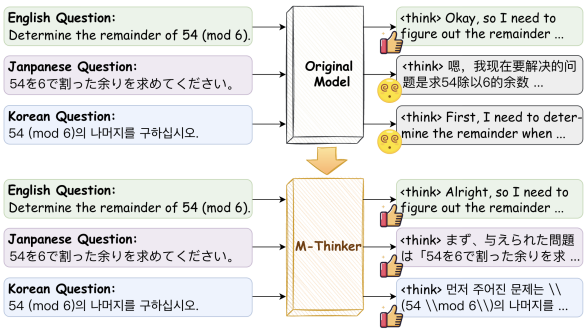

Figure 1: Existing LRMs struggle to maintain input-output language consistency and probably offer us the wrong answer when processing non-English inputs, while our M-Thinker can respond in the input language with the correct answer.

However, current LRMs generally suffer from two major issues under multilingual scenarios. First, they often suffer from **input-output language inconsistency** (Wang et al., 2025d; Tam et al., 2025), *i.e.*, they frequently default to thinking and answering in English (or other unintended languages) rather than the input language (please refer to Figure 1). Second, they present **inferior performance** for other languages compared to English (Luo et al., 2025; Wang et al., 2025d; Weihua et al., 2025, 2026). These issues significantly reduce the readability and explainability of the rea-

[*] This work was done during the internship at Pattern Recognition Center, WeChat AI, Tencent Inc, China.
[†] Yufeng Chen is the corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 20766–20783

soning processes and degrade the user experience of LRMs in multilingual environments. To mitigate these issues, current solutions include language control instructions (Tam et al., 2025), supervised fine-tuning (SFT) with specific language data (Luo et al., 2025; Fan et al., 2025a), and GRPO (Shao et al., 2024) with a soft language reward (Park et al., 2025; Mistral-AI, 2025; Hwang et al., 2025). However, these solutions still face notable limitations: Prompt-based methods struggle to enforce output language consistency with the input; SFT generally entails a trade-off between answer accuracy and language consistency; Soft consistency rewards in GRPO can only impose weak constraints on maintaining language consistency. Therefore, there still remains a clear need for a solution to effectively enhance both language consistency and multilingual reasoning capability of LRMs.

To this end, we propose M-Thinker, a real multilingual reasoning model trained by the GRPO algorithm that includes a Language Consistency (LC) reward and a novel Cross-lingual Thinking Alignment (CTA) reward. Specifically, the LC reward strictly constrains the language consistency between the input, thought, and answer, encouraging the model to generate language-consistent responses. Additionally, given that LRMs often exhibit stronger reasoning proficiency in English compared to other languages (Huang et al., 2025; Zhang et al., 2025b), we regard the English reasoning paths of the model itself as the teacher and design the CTA reward for cross-lingual reasoning alignment. The CTA reward is computed by comparing the model's reasoning paths in English and other languages via LLM-as-a-Judge (Gu et al., 2025; Wang et al., 2025a), which encourages the model to transfer its reasoning capability from English to non-English languages. On this basis, our M-Thinker is trained with a systematic training procedure incorporating cold-start SFT, rejection sampling, and iterative RL training.

Experimental results on two publicly-used multilingual benchmarks (MMATH and PolyMath) show that our M-Thinker-1.5B/4B/7B models not only achieve nearly 100% language consistency and substantial performance improvement, but also demonstrate remarkable generalization on out-of-domain languages. In summary, the major contributions of this paper are as follows[1]:

- We propose M-Thinker, which both achieves the input-output language consistency with a Language Consistency reward and enhances the multilingual reasoning performance with a Cross-lingual Thinking Alignment reward.
- Experimental results of our M-Thinker-1.5B/4B/7B models on MMATH and PolyMath benchmarks demonstrate superior performance on both language consistency and answer accuracy for multiple languages.
- We also conduct an analysis on the generalization of M-Thinker to out-of-domain languages, which reveals that the models typically generalize better to languages within the same or similar language families.

2 Related Work

The multilingual reasoning capabilities of current LRMs have recently drawn increasing research interest (Fan et al., 2025b; Tan et al., 2026; Zheng et al., 2025). Luo et al. (2025) point that DeepSeek-R1 exhibits substantial performance disparities across languages and suffers from a critical off-target issue, *i.e.*, generating responses in unintended languages. Wang et al. (2025d) also show that reasoning models exhibit lower input-output language consistency, particularly in their thinking processes. Additionally, when constrained to reason in the same language as the input, the model's performance declines, especially for low-resource languages (Tam et al., 2025). Furthermore, Wang et al. (2025c) investigate that the language-mixing phenomenon may affect the final performance, which may hinder the readability and usability of outputs in multilingual contexts.

In addition, some concurrent works have already conducted preliminary studies based on GRPO in multilingual scenarios. Park et al. (2025) find that GRPO rapidly amplifies pre-training language imbalances within just a few hundred updates, resulting in the cross-lingual collapse, and language consistency reward mitigates this drift with a large drop in accuracy. Hwang et al. (2025) combine SFT and multilingual GRPO with a language-consistency reward to enhance multilingual reasoning fidelity on a geography-based multilingual factual reasoning benchmark. Lee et al. (2025) only employ a customized GRPO to improve the reasoning performance on Korean. Differently, we use the strict LC reward to achieve better input-output language consistency and design a novel CTA reward that

[1] https://github.com/XZhang00/M-Thinker

transfers reasoning capability from English to other languages to improve the multilingual reasoning performance.

3 Methodology

In this section, we first briefly introduce the GRPO algorithm (§3.1), and then present our designed rewards (§3.2), which quantify the language consistency and alignment ratio to the English thinking sequence, besides format and answer accuracy. Finally, we introduce our training procedure (§3.3).

3.1 Background: GRPO

Recently, GRPO (Shao et al., 2024) has been widely utilized for enhancing the performance of language models (DeepSeek-AI, 2025; Mistral-AI, 2025; Wang et al., 2025a,b). GRPO discards the critic model and estimates the baseline from group scores instead to largely save the training costs. Specifically, for each question q in the question set Q, GRPO first utilizes the old policy model $\pi_{\theta_{old}}$ to samples a group of outputs $\{o_1, o_2, \cdots, o_N\}$ and then optimizes the policy model π_θ by maximizing the following objective:

$$\begin{aligned}\mathcal{J}_{\text{GRPO}}(\theta) = & \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{N} \sim \pi_{\theta_{old}}(O|q)] \\ & \frac{1}{N}\sum_{i=1}^{N}(\min(\frac{\pi_\theta(o_i\,|\,q)}{\pi_{\theta_{\text{old}}}(o_i\,|\,q)}A_i, \\ & \text{clip}(\frac{\pi_\theta(o_i\,|\,q)}{\pi_{\theta_{\text{old}}}(o_i\,|\,q)}, 1-\epsilon, 1+\epsilon)A_i) - \beta\mathbb{D}_{\text{KL}}),\end{aligned} \tag{1}$$

where ϵ and β are hyper-parameters, and A_i is the advantage computed using a group of rewards $\{r_1, r_2, \ldots, r_N\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_N\})}{\text{std}(\{r_1, r_2, \cdots, r_N\})}, \tag{2}$$

where $r_i = R(o_i)$ is calculated by the reward function $R(o)$.

3.2 Reward Modeling

To make LRMs generate correct thinking processes and answer sequences in the input language when processing non-English inputs, we employ the following four reward modeling functions.

Language Consistency Reward. To improve the input-output language consistency, we design the LC reward to judge whether the thinking sequence o_t and the answer sequence o_a of the output o are generated with the input language ℓ. First, we identify the involved language(s) of one sequence x using the `langdetect`[2] library following Wang et al. (2025d). Formally, we define the detected language(s) set in the sequence x as $\phi(x)$, and x is language-consistent with ℓ when only one language is detected and the language is equal to ℓ:

$$\text{LC}(x) = (|\phi(x)| = 1) \wedge (\ell \in \phi(x)), \tag{3}$$

where $|\cdot|$ is the number of detected language(s) set and $\text{LC}(x)$ is True or False.

Based on $\text{LC}(x)$, the LC reward $R_{\text{lc}}(o)$ is defined as 0 when o_t and o_a are all language-consistent with ℓ, and -1 otherwise:

$$R_{\text{lc}}(o) = \begin{cases} 0, & \text{if LC}(o_t) \wedge \text{LC}(o_a), \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

The LC reward strictly ensures that the model can generate the thinking and answering sequence in the input language ℓ by punishing the inconsistency phenomenon.

Cross-lingual Thinking Alignment Reward. Existing LRMs generally exhibit better performance on English compared to other languages (Huang et al., 2025; Yang et al., 2025b; Zhang et al., 2025b), which motivates us to align the multilingual reasoning capacity to the English reasoning ability to further improve the answer correctness of multilingual responses. Therefore, we design the CTA reward $R_{\text{cta}}(o)$, which represents the alignment ratio between the English thinking sequence o_t^{en} and the current thinking sequence o_t^{ℓ}:

$$R_{\text{cta}}(o) = \text{LLMJudge}(o_t^{\ell}, o_t^{en}) \in [0, 1]. \tag{5}$$

Specifically, we carefully design the judge instruction and request `DeepSeek-v3-0324` to evaluate the alignment ratio according to the overlap between intermediate results of o_t^{en} and o_t^{ℓ}. Please refer to Appendix A for the specific judge instruction. The CTA reward utilizes the English thinking sequence as a reliable teacher to advance the cross-lingual alignment, further improving the correctness of the multilingual reasoning process.

Format Reward. This reward is commonly used (DeepSeek-AI, 2025; Wang et al., 2025a; Mistral-AI, 2025) to ensure the format correctness of the

[2] https://pypi.org/project/langdetect/

Algorithm 1 Iterative Training Procedure for M-Thinker

Input: Cold-started model π_{θ_0}; Multilingual questions $\mathcal{Q}_\ell$; Parallel English questions $\mathcal{Q}_{en}$; Reward functions R_{format}, R_{acc}, R_{lc}, and R_{all}; Hyperparameters: outer iterations I, sampling candidates N

Let $\mathbb{I}(\cdot)$ be an indicator function that returns 1 if the condition is true, and 0 otherwise
for iteration $i = 1, \ldots, I$ **do**
  {**Phase A: Data Construction with Rejection Sampling**}
  Set reference model for this iteration: $\pi_{\text{ref}} \leftarrow \pi_{\theta_{i-1}}$
  Initialize RL training dataset $\mathcal{D}_{\text{RL}}^{(i)} \leftarrow \emptyset$
  for each question $q_\ell \in \mathcal{Q}_\ell$ with its parallel English question $q_{en} \in \mathcal{Q}_{en}$ **do**
    Generate N candidate outputs $\{o_k^\ell\}_{k=1}^N \sim \pi_{\text{ref}}(\cdot|q_\ell)$
    Define $\mathcal{O}_{\text{correct}}^\ell = \{o_k^\ell \mid \mathbb{I}(R_{\text{format}}(o_k^\ell) = 0 \wedge R_{\text{lc}}(o_k^\ell) = 0 \wedge R_{\text{acc}}(o_k^\ell) = 1) = 1\}$
    Generate N English candidate outputs $\{o_k^{en}\}_{k=1}^N \sim \pi_{\text{ref}}(\cdot|q_{en})$
    Define $\mathcal{O}_{\text{correct}}^{en} = \{o_k^{en} \mid \mathbb{I}(R_{\text{format}}(o_k^{en}) = 0 \wedge R_{\text{lc}}(o_k^{en}) = 0 \wedge R_{\text{acc}}(o_k^{en}) = 1) = 1\}$
    if $0 < |\mathcal{O}_{\text{correct}}^\ell| < N$ **then**
      Randomly select one correct English output as the thinking reference: $o^{en*} \leftarrow \text{RandomSample}(\mathcal{O}_{\text{correct}}^{en})$
      Add the multilingual question to the training set: $\mathcal{D}_{\text{RL}}^{(i)} \leftarrow \mathcal{D}_{\text{RL}}^{(i)} \cup \{(q_\ell, o^{en*})\}$
    end if
  end for
  {**Phase B: GRPO Training**}
  Train with GRPO (using R_{all}) on $\mathcal{D}_{\text{RL}}^{(i)}$ following Eq.(1) and update $\pi_{\theta_i} \leftarrow \pi_{\theta_{i-1}}$
end for

Output: The final trained model π_{θ_I}.

generated outputs. Given a question q_ℓ in language ℓ, the output o generated by the old policy model $\pi_{\theta_{old}}$ must conform to the response pattern "<think>o_t</think>o_a", where "<think>" and "</think>" are two special tokens to split the thinking sequence (o_t) and the answer sequence (o_a). Based on the strict pattern, we utilize the regular expression to verify the pattern correctness of o and define the format reward as:

$$R_{\text{format}}(o) = \begin{cases} 0, & \text{if format is correct,} \\ -1, & \text{if format is incorrect.} \end{cases} \tag{6}$$

Accuracy Reward. For mathematical questions, the accuracy reward $R_{\text{acc}}(o)$ is widely utilized to verify the correctness of o:

$$R_{\text{acc}}(o) = \begin{cases} 1, & \text{if answer is correct,} \\ 0, & \text{if answer is incorrect.} \end{cases} \tag{7}$$

Specifically, the final answer is extracted from inside the last "`\boxed{}`" in o and compared against the ground truth using a rule-based verifier (Sheng et al., 2024).

Overall Reward. Based on the above four rewards, we design the overall reward $R_{\text{all}}(o)$ as follows:

$$R_{\text{all}}(o) = \begin{cases} -1, \text{ if } R_{\text{format}}(o) = -1 \vee R_{\text{lc}}(o) = -1, \\ R_{\text{acc}}(o) \cdot (1 + R_{\text{cta}}(o)), \text{ otherwise.} \end{cases} \tag{8}$$

Particularly, only when $R_{\text{format}}(o) = 0$ and $R_{\text{lc}}(o) = 0$, we then calculate the reward following $R_{\text{acc}}(o) \cdot (1 + R_{\text{cta}}(o))$.

3.3 Training Procedure

We present our training procedure in Algorithm 1, incorporating cold-start SFT (Wang et al., 2025a), rejection sampling (Liu et al., 2024), and iterative RL training (Yang et al., 2025b). Specifically, given the model π_θ, we first conduct the cold-start SFT to ensure that the initial model π_{θ_0} can generate valid samples during the GRPO training process, which is a prerequisite for effective training. Subsequently, the model enters an iterative RL training loop.

In each iteration i, we first construct the training data. Using the previous model $\pi_{\theta_{i-1}}$, we apply a rejection sampling strategy to select "hard" but solvable problems. Specifically, a multilingual question q_ℓ is selected if the model generates both correct and incorrect answers for it (i.e., $0 < |\mathcal{O}_{\text{correct}}^\ell| < N$). For each selected question, we also select a high-quality English output, o^{en*}, by randomly sampling from the correct outputs for its parallel English question q_{en}. The thinking sequence o_t^{en} of o^{en*} is used for R_{cta}. The reason why we utilize the self-generated English thinking as the reference of R_{cta} is that they not only do not request other models but also may have a smaller gap between the ability of non-English languages and English compared to external models. These selected questions and their corresponding English answers form the training data $\mathcal{D}_{\text{RL}}^{(i)}$ for the current iteration. Next, we perform GRPO training with our designed reward $R_{\text{all}}(o)$. The model $\pi_{\theta_{i-1}}$ is updated to π_{θ_i} by optimizing the GRPO objective

following Eq.(1) on $\mathcal{D}_{\text{RL}}^{(i)}$. And we utilize our designed reward $R_{\text{all}}(o)$ to calculate the rewards in Eq.(2). The iterative cycle of data construction and policy optimization enables the model to progressively master complex multilingual reasoning.

4 Experiments

4.1 Experimental Setups

Backbones and Languages. We select three commonly-used reasoning models with different sizes as our backbones: DeepSeek-R1-Distill-Qwen-1.5/7B (DeepSeek-AI, 2025) and Qwen3-4B-Thinking-2507 (Yang et al., 2025a). The three models exhibit imbalanced reasoning performance in different languages, showing better ability in English compared to other languages. Based on the imbalanced ability and the included languages of the MMATH (Luo et al., 2025) benchmark, we select Japanese (*ja*), Korean (*ko*), French (*fr*), Portuguese (*pt*), and Thai (*th*) as the training (in-domain, ID) languages and English (*en*), Spanish (*es*), Arabic (*ar*), Vietnamese (*vi*), and Chinese (*zh*) as out-of-domain (OOD) languages to observe the generalization[3] of each method. The details for each language are introduced in Table 8 of Appendix B.1.

Benchmarks and Metrics. In this paper, we focus on the math reasoning task, which has sufficient multilingual benchmarks. We mainly evaluate the multilingual reasoning ability on the MMATH (Luo et al., 2025) benchmark, which comprises 374 mixed-difficulty math problems sourced from AIME24/25, CNMO, and MATH-500 (Lightman et al., 2023), and covers the above mentioned ten languages (*ja*/*ko*/*fr*/*pt*/*th*/*en*/*es*/*ar*/*vi*/*zh*). Following Luo et al. (2025), we conduct each evaluation four times and report the average result across all runs. Specifically, for each individual evaluation, we compute the macro-average metric rather than the micro-average to account for the varying difficulty levels across subsets in MMATH.

To evaluate both the language consistency and answer accuracy of model responses, we adopt three metrics: Language Consistency (LC), Accuracy (Acc), and Language Consistency & Accuracy (LC&Acc). LC assesses whether the language used throughout the response (including both the thinking and answer sequences) matches the language of the input question, referring to Eq.(3). Acc measures the correctness of the final extracted answer[4], regardless of the language in which the response is generated. LC&Acc evaluates answer correctness only when the response o is fully in the input language, *i.e.*, $R_{\text{lc}}(o) = 0 \wedge R_{\text{acc}}(o) = 1$, which combines both language consistency and answer accuracy as our main evaluation metric. Furthermore, we also evaluate our model on the PolyMath (Wang et al., 2025d) benchmark for additional validation. The evaluation details on PolyMath are present in Appendix B.2.

Data. We conduct our experiments based on the Light-R1-SFTData[5] dataset (Wen et al., 2025), which contains about 76K carefully selected data samples, *i.e.*, each English question with the accurate response generated from DeepSeek-R1 (DeepSeek-AI, 2025). To obtain the multilingual questions, we deploy the `DeepSeek-V3-0324` model (DeepSeek-AI, 2024) to translate[6] the English questions to *ja*/*ko*/*fr*/*pt*/*th*. For the cold-start SFT, we randomly sample 7.5K questions for each language and deploy the `DeepSeek-R1-0528` model (DeepSeek-AI, 2025) to generate responses in the input language. We then filter these samples based on their LC&Acc scores (retaining only those responses that are both language consistent with the input and answer correct) to construct the training dataset for the cold-start SFT, which comprises approximately 20K samples across all five ID languages. For each iteration of RL training, we apply rejection sampling on the remaining data from Light-R1-SFTData. And we set the sampling candidates N is 8. From the filtered RL dataset, we randomly select 3K samples per ID language for RL training.

Implementation Details. We set the iterations for RL training I is 2. The detailed training settings of cold-start SFT and iterative RL training, and generation configs are listed in Appendix B.3.

4.2 Baselines

Prompt-Control. Following Wang et al. (2025d), we concatenate the language control instructions after the input prompts to make the model generate

[3] Since the original model performs well on *en*, we actually want to observe the catastrophic forgetting phenomenon for *en*. To simplify writing, we refer to it as generalization here.

[4] We directly utilize the extraction and verification tool of MMATH (Luo et al., 2025).

[5] https://huggingface.co/datasets/qihoo360/Light-R1-SFTData

[6] The translation prompt follows Wang et al. (2024) and Zhang et al. (2025b).

responses using the same language as the query. Please refer to Figure 2 of Appendix B.4 for the detailed language control instructions of each language.

DIT. Discourse-Initiated Thinking (Luo et al., 2025) appends the most popular beginning discourse markers in each language after the "<think>" token, encouraging models to initiate their reasoning using multilingual discourse cues as entry points into the thinking process. The used multilingual discourse marks are shown in Figure 3 of Appendix B.4.

QRT. Question-Restatement Thinking (Luo et al., 2025) restates the question in the target language at the beginning of the thinking process, which encourages the model to think in the target language. The restatement instructions for each language are listed in Figure 4 of Appendix B.4.

Cold-Start SFT. We conduct the cold-start SFT training on the constructed training dataset.

Naive-RL. We equip the GRPO algorithm only with the accuracy reward to conduct the RL training based on the same cold-started SFT model. The training dataset is the same as our first training iteration (Iter-1).

SLC-RL. We equip the GRPO algorithm with the accuracy reward and a soft language consistency reward (Mistral-AI, 2025) to conduct the RL training, i.e., $R(o) = R_{\text{format}}(o) * (R_{\text{acc}}(o) + R_{\text{slc}}(o))$. When the format is correct: $R_{\text{format}}(o) = 1$, when the answer is correct: $R_{\text{acc}}(o) = 0.9$, and when the language is consistent with the input language: $R_{\text{slc}}(o) = 0.1$, otherwise, $R_{\text{format}}(o) = R_{\text{acc}}(o) = R_{\text{slc}}(o) = 0$. The initial policy model (after cold-start SFT) and training dataset are the same as our first training iteration (Iter-1).

Other Related but Non-comparable Methods. Regarding the judge metrics for R_{cta}, several common approaches measure the alignment between English and multilingual sentences, notably translation-consistency (She et al., 2024) and embedding-similarity methods (Hwang et al., 2025; Faisal et al., 2025). For instance, MAPO (She et al., 2024) employs the NLLB model to compute translation probabilities between English and multilingual responses, using them as a selection criterion to construct preference data for DPO training. However, we omit direct comparisons with these methods due to their suboptimal performance, as evidenced in Table 4 of CM-Align (Zhang et al., 2025b). Further discussions are provided in Appendix B.5.

Additionally, translating and summarizing English answers into other languages is a standard industry practice for multilingual questions. However, these cascaded systems require extra models and computational overhead. In contrast, M-Thinker is an end-to-end solution that aims to improve intrinsic multilingual reasoning rather than engineering a complex pipeline. To this end, we do not include the translated performance.

4.3 Main Results

Performance of our M-Thinker. We report the evaluation results on MMATH of the three backbones in Table 1, Table 2, and Table 9 (Appendix C.1). The results demonstrate that our M-Thinker-1.5B/4B/7B achieves excellent improvement on LC, Acc, and the combined metric (LC&Acc). On the main evaluation metric (LC&Acc), our M-Thinker-1.5B/4B/7B (Iter-1) drastically outperforms all baselines, which highlights the effectiveness of our designed rewards in simultaneously optimizing for correctness and language fidelity. Furthermore, our M-Thinker-1.5B/7B (Iter-2) achieves further improvement than Iter-1, which proves that our iterative training procedure can progressively enhance the model's capabilities. And the performance on LC&Acc of our M-Thinker-1.5B/7B (Iter-2) has surpassed the performance on Acc of the backbones DeepSeek-R1-Distill-Qwen-1.5B/7B, which means that responding in the input language can exceed the performance of responding in English or other default languages. This superior performance indicates that our method mitigates the trade-off between language consistency and answer accuracy, achieving powerful multilingual reasoning ability.

Performance of baselines. No training baselines have a minor improvement on LC&Acc, and QRT outperforms DIT and Prompt-Control. The performance of these prompt-based methods heavily depends on the original instruction-following ability of backbones, *i.e.*, the larger improvement on 7B than 1.5B. Additionally, the improvement on LC and the decrease on Acc also reflect the trade-off between the language consistency and answer accuracy. Naive-RL (GRPO only with the accuracy reward) shows the best results on Acc but the lowest LC (0.0) since the responses generated in

Methods	In-Domain Languages						Out-of-Domain Languages						
	ja	ko	fr	pt	th	*ID-AVG*	en	es	ar	vi	zh	*OOD-AVG*	*ALL-AVG*
***Metric:* Language Consistency (LC, %)**													
DeepSeek-R1-Distill-Qwen-7B	9.49	2.47	16.56	10.88	2.19	8.32	96.35	15.61	7.70	23.35	71.23	42.85	25.58
Prompt-Control (No Training)	29.63	2.99	26.08	33.77	9.93	20.48	95.47	43.15	8.92	44.92	73.58	53.21	36.84
DIT (No Training)	68.99	2.85	78.28	66.39	15.66	46.43	95.93	66.78	6.22	65.79	71.14	61.17	53.80
QRT (No Training)	29.77	4.21	85.00	67.72	37.26	44.79	95.38	69.07	9.26	62.30	77.02	62.61	53.70
Cold-Start SFT	13.69	0.64	30.59	21.47	4.13	14.10	98.09	28.51	2.03	29.81	84.87	48.66	31.38
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	96.29	0.00	0.00	0.00	85.86	36.43	18.22
SLC-RL	91.20	0.00	99.54	99.09	90.18	76.00	99.77	99.15	1.61	81.84	88.82	74.24	75.12
M-Thinker-7B ⇒ Iter-1 (Ours)	**98.32**	98.74	**99.96**	**99.88**	**99.27**	**99.23**	**100.00**	**99.80**	**84.68**	99.56	89.17	**94.64**	**96.94**
M-Thinker-7B ⇒ Iter-2 (Ours)	97.86	**99.37**	99.50	99.05	95.68	98.29	98.00	99.44	75.12	**100.00**	**90.97**	92.70	95.50
***Metric:* Accuracy (Acc, %)**													
DeepSeek-R1-Distill-Qwen-7B	53.44	61.61	64.47	62.67	50.71	58.58	65.20	61.31	55.28	58.10	52.99	58.58	58.58
Prompt-Control (No Training)	40.63	60.18	60.92	58.43	49.66	53.96	62.18	57.64	52.24	50.80	57.69	56.11	55.04
DIT (No Training)	21.36	40.86	47.35	55.72	41.60	41.38	64.51	51.37	50.78	38.39	56.98	52.41	46.89
QRT (No Training)	30.88	42.52	53.92	52.80	32.27	42.48	63.36	54.73	51.47	44.79	56.18	54.11	48.29
Cold-Start SFT	48.15	55.40	60.78	61.16	49.15	54.93	63.62	61.21	52.69	51.76	58.20	57.50	56.21
Naive-RL	**66.11**	65.18	65.71	66.81	**65.82**	**65.93**	69.21	64.16	**63.29**	**64.42**	63.60	64.94	**65.43**
SLC-RL	47.00	**66.86**	57.91	61.48	49.96	56.64	67.62	61.86	60.99	51.09	61.17	60.55	58.59
M-Thinker-7B ⇒ Iter-1 (Ours)	53.92	52.24	60.56	64.46	54.71	57.18	67.94	60.76	54.79	55.40	63.97	60.57	58.87
M-Thinker-7B ⇒ Iter-2 (Ours)	58.23	60.56	**68.58**	**66.99**	63.98	63.66	**71.75**	**68.34**	63.00	61.72	**67.25**	**66.41**	65.04
***Metric:* Language Consistency & Accuracy (LC&Acc, %)**													
DeepSeek-R1-Distill-Qwen-7B	6.73	2.11	13.99	9.93	1.67	6.89	65.14	14.16	5.47	15.69	45.00	29.09	17.99
Prompt-Control (No Training)	14.62	2.67	20.36	26.75	7.47	14.37	61.81	33.95	6.79	24.64	46.95	34.83	24.60
DIT (No Training)	17.99	2.07	43.76	44.94	12.55	24.26	64.45	45.90	4.15	35.13	48.24	39.57	31.92
QRT (No Training)	18.51	3.82	52.17	44.66	18.02	27.44	63.26	48.69	6.98	39.82	51.30	42.01	34.72
Cold-Start SFT	8.58	0.44	23.64	18.51	2.13	10.66	63.58	25.22	1.41	20.03	50.50	32.15	21.40
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	68.48	0.00	0.00	0.00	54.11	24.52	12.26
SLC-RL	46.52	0.00	57.87	61.42	49.90	43.14	67.60	61.70	1.57	49.57	53.96	46.88	45.01
M-Thinker-7B ⇒ Iter-1 (Ours)	53.30	52.12	60.54	64.34	54.71	57.00	67.94	60.58	52.14	55.38	56.21	58.45	57.73
M-Thinker-7B ⇒ Iter-2 (Ours)	**57.50**	**60.26**	**68.52**	**66.87**	**63.44**	**63.32**	**71.71**	**68.22**	**53.70**	**61.72**	**60.58**	**63.18**	**63.25**

Table 1: The LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the DeepSeek-R1-Distill-Qwen-7B backbone. "***ID-avg*/*OOD-avg***" is the average result of five In-Domain/Out-of-Domain languages and "***ALL-AVG***" is the average result of all ten languages. The result in **bold** means the best result, and the underlined result means the second-best result in each setting. "**Iter-1/2**" means the training iteration 1/2.

English can obtain a higher reward score during RL training, so that the trained model is most likely to think and answer in English, which is contrary to the goal of a multilingual reasoning model. Although SLC-RL is trained with a soft language consistency reward, the models still struggle to maintain language consistency, particularly for the 1.5B backbone. By contrast, our method with the strict LC reward can promote the input-output language consistency while having no degradation[7] on Acc compared to SLC-RL[8] (58.87 vs. 58.59).

OOD generalization. Refer to the "*OOD-avg*", our M-Thinker also significantly surpasses other baselines, which indicates that the reasoning patterns learned through our rewards and training procedure are not confined to the training languages but are successfully transferred to unseen languages. The evaluation results on PolyMath (as shown in Table 14 in Appendix C.5) also present similar trends, which further prove the superiority of our method.

In summary, these results demonstrate that our M-Thinker effectively improves both the language consistency and answer accuracy in multilingual reasoning scenarios.

5 Analysis

5.1 Ablation Study

We conduct an ablation study to verify the effectiveness of our designed reward functions and involved training strategies. The ablation results listed in Table 3 show that the LC&Acc performance degrades in both ID and OOD languages without R_{cta}. For the setting "*w/o* R_{lc}", although the Acc improves over M-Thinker-1.5B, the model responds to all questions in English, resulting in the lowest language consistency. "*w/o* (R_{cta} & R_{lc})" present the lowest performance. These results prove the effectiveness of our designed reward functions. Additionally, "*w/o* Cold-Start SFT" and "*w/o* Rejection Sampling" also have a performance decline, which demonstrates the necessity of these strategies. Furthermore, directly using English responses from

[7]More detailed analyses about hard/soft LC reward are listed in Appendix C.3.

[8]We also conduct SLC-RL with the same reward magnitude as ours and present it in Table 13 of Appendix C.4.

Methods	LC			Acc			LC&Acc		
	ID-AVG	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*
DeepSeek-R1-Distill-Qwen-1.5B	5.98	36.11	21.04	34.81	39.83	37.32	3.87	19.19	11.53
Prompt-Control (No Training)	12.64	48.52	30.58	31.95	33.55	32.75	5.65	22.32	13.99
DIT (No Training)	22.48	43.30	32.89	23.10	28.50	25.80	11.56	23.68	17.62
QRT (No Training)	23.47	45.46	34.46	19.26	28.11	23.69	11.68	23.87	17.78
Cold-Start SFT	23.73	47.75	35.74	19.18	27.79	23.49	7.39	21.73	14.56
Naive-RL	0.00	30.97	15.48	**49.99**	**50.08**	**50.04**	0.00	16.16	8.08
SLC-RL	0.00	37.16	18.58	46.80	49.16	47.98	0.00	19.47	9.74
M-Thinker-1.5B ⇒ Iter-1 (Ours)	99.19	**84.48**	**91.83**	35.59	44.22	39.90	35.39	38.37	36.88
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**99.49**	79.51	89.50	42.72	46.53	44.62	**42.47**	**38.83**	**40.65**

Table 2: The LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the DeepSeek-R1-Distill-Qwen-1.5B backbone. The detailed results for each language are list in Table 10 of Appendix C.2.

Settings	LC			Acc			LC&Acc		
	ID-AVG	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*
M-Thinker-1.5B ⇒ Iter-1 (Ours)	99.19	84.48	91.83	35.59	44.22	39.90	35.39	38.37	36.88
w/o R_{cta}	99.16	92.44	95.80	31.72	39.85	35.78	31.68	37.18	34.43
w/o R_{lc}	0.00	35.61	17.80	50.22	50.83	50.52	0.00	18.66	9.33
w/o (R_{cta} **&** R_{lc})	0.00	30.97	15.48	49.99	50.08	50.04	0.00	16.16	8.08
***w/o* Cold-Start SFT**	99.19	84.33	91.76	33.60	42.83	38.22	33.35	36.91	35.13
***w/o* Rejection Sampling**	99.71	85.31	92.51	33.87	41.24	37.55	33.73	35.48	34.60
w/ o_t^{en} **from Light-R1 for** R_{cta}	99.76	88.27	94.01	33.71	41.87	37.79	33.67	37.65	35.66

Table 3: The ablation results of the MMATH benchmark based on our M-Thinker-1.5B (Iter-1). "*w/o*" means without one setting and "*w/*" means with one setting.

Judge Models	*ID-AVG*	*OOD-AVG*	*ALL-AVG*	*GAP*↓
w/o R_{cta}	31.68	37.18	34.43	13.47
DeepSeek-V3-0324	**35.39**	**38.37**	**36.88**	**9.24**
Qwen3-4B-Instruct-2507	32.48	38.51	35.49	12.56
Qwen3-30B-A3B-Instruct-2507	33.12	37.08	35.10	11.75
Qwen2.5-7B-Instruct	31.69	34.13	32.91	13.65

Table 4: The LC&Acc results of different judge models for R_{cta} based on our M-Thinker-1.5B (Iter-1). "*GAP*" denotes the accuracy gap between English and other languages.

Data	ja	ko	fr	pt	th	en	es	ar	vi	zh
1.5B	0.22	**0.02**	7.05	11.92	0.12	46.56	13.38	0.16	3.56	32.30
fr	2.73	0.00	**37.12**	**34.72**	7.20	**52.27**	**37.21**	3.54	20.92	38.45
ja	**26.76**	0.00	21.91	32.23	**8.97**	49.55	35.74	**3.79**	**23.17**	**39.69**

Table 5: The LC&Acc generalization results on OOD languages when only using *fr*/*ja* as training data for DeepSeek-R1-Distill-Qwen-1.5B. The blue results mean the performance on the training language. The results in **bold** represent the best result in each language.

the Light-R1-SFT dataset (which is generated by DeepSeek-R1) for R_{cta} also underperforms our M-Thinker (using generated English responses from the model itself), since the latter may have a smaller gap between the abilities of non-English languages and English. Detailed results of each ablation setting are listed in Table 15 of Appendix D.1.

5.2 Effects of Different Judge Models for R_{cta}

In this section, we analyze the effects of different judge models for calculating R_{cta} on performance and report the results in Table 4.

Findings 1: Frontier small LLMs can also provide reliable rewards. Beyond DeepSeek-V3, we also test two smaller models as the judge model, i.e., Qwen3-30B-A3B and Qwen3-4B. Although smaller, these two frontier models still deliver notable performance gains while being more cost-efficient than DeepSeek-V3-0324. Besides, we also try another small model, Qwen2.5-7B-Instruct, that is relatively outdated compared to other models. We find that it decreases the overall performance (32.91%) compared to "*w/o* R_{cta}" due to the limited multilingual capability, demonstrating that the multilingual capability of the judge model is crucial for the effectiveness of the R_{cta} reward.

Findings 2: R_{cta} achieves cross-lingual transfer of the reasoning capability from English to other languages. As shown in Table 4, we also find that R_{cta} significantly brings the accuracy gap between English and other languages with multiple judge models according to the "*GAP*" values. This further proves the effectiveness of R_{cta} for bridging the multilingual reasoning gap in existing models.

5.3 Generalization Study

In this section, we investigate the generalization to non-training (OOD) languages when training on different languages. Specifically, we separately use *fr* and *ja* to train the model and observe the performance of the other nine languages (as shown

Language	from Human	from LLM	Pearson Correlation
ja	0.72	0.69	0.87
ko	0.68	0.64	0.86
fr	0.85	0.81	0.93
pt	0.82	0.78	0.91
th	0.63	0.57	0.83
AVG	0.74	0.70	0.88

Table 6: The Pearson correlation between human-annotated ratios and the judge model's CTA scores (DeepSeek-V3-0324).

in Table 5). We find that if training on *fr*, the performance of *pt*, *es*, and *en* is better than training on *ja* since *pt*/*es*/*en* and *fr* all belong to the Indo-European language family (as introduced in Table 8 of Appendix B.1). By contrast, training on *ja* shows better generalization to *zh*/*vi*. We guess that although *ja* generally is regarded as an Isolate language, some scripts are sourced from Chinese, and a few scripts of Vietnamese also source from Chinese. Additionally, since *ko* is an isolate language with a writing system distinct from those of *ja* and *fr*, it achieves the lowest generalization (0.0). Overall, these results indicate that if you want to improve the performance of one language, the similar or same-language-family languages must be added to the training dataset.

5.4 The Reliability of LLM-as-a-Judge

To empirically validate the reliability of our utilized judge model (DeepSeek-V3-0324) for R_{cta}, we conduct a human evaluation on a randomly sampled subset of the training data (30 samples per language). We collaborate with a professional data annotation service to recruit linguistic experts proficient in the target languages (*ja*/*ko*/*fr*/*pt*/*th*). We request that they return the alignment ratios, and the detailed annotation guidelines are introduced in Appendix D.2. We then calculate the Pearson correlation between these human-annotated ratios and the judge model's CTA scores. The results in Table 6 demonstrate a positive Pearson correlation coefficient across different languages, confirming that the judge model aligns well with human judgment and proving the reliability of the CTA reward.

5.5 The Robustness of `langdetect`

To verify the robustness of `langdetect` on code-switching scenarios, we randomly sample 20 questions per language from MMATH and generate responses using our 1.5B/7B-Iter2 models. We then conduct a strict human evaluation to determine whether code-switching occurs in a given response. The results are listed in Table 7, which shows that the 1.5B model exhibits a minor code-switching phenomenon (29%), while the 7B model nearly does not exhibit this behavior. This suggests that while smaller models struggle slightly in some languages, the LC reward signal by `langdetect` is generally robust enough to guide models effectively without significant reward hacking.

(%)	ja	ko	fr	pt	th	en	es	ar	vi	zh	AVG
1.5B	30	45	15	10	55	0	10	80	45	0	29
7B	0	5	0	0	25	0	0	35	5	0	7

Table 7: The code-switching ratio of model responses judged by human evaluation.

6 Conclusion

In this paper, we design a Language Consistency reward to strictly enforce input-output language consistency and a Cross-lingual Thinking Alignment reward to further improve the accuracy of multilingual answers. Additionally, we train M-Thinker-1.5B/4B/7B models with a systematic training procedure incorporating cold-start SFT, rejection sampling, and iterative RL training. Experimental results on the MMATH and PolyMath show that our M-Thinker models exhibit excellent multilingual reasoning performance. In summary, our work offers an effective method and valuable empirical insights for the community to enhance the intrinsic multilingual capabilities of LRMs.

Limitations

In this paper, we only conduct experiments on five languages (3K samples for each language) and set the RL training iterations to 2 due to time and resource limitations. We believe that more languages, more training samples, and more RL training iterations will achieve better performance. Additionally, we utilize the `langdetect` library to detect involved languages in one sequence for the LC Reward following Wang et al. (2025d). However, there are some other language detection tools or models that we do not test, such as xlm-roberta-base-language-detection (Conneau et al., 2020), `Cld3`[9], and `FastText`[10]. We will try to investigate a more robust and faster language detection method in the future.

[9] https://pypi.org/project/pycld3/

[10] https://pypi.org/project/fasttext-langdetect/

As for extremely low-resource languages where the Judge model completely fails to comprehend the input, the CTA reward would indeed be unreliable. In such cases, we suggest falling back to only Format+Acc+LC reward or replacing it with rule-based CTA rewards. Considering our paper mainly focuses on language consistency and the effectiveness of the CTA reward on multilingual reasoning, we leave the study of extremely low-resource languages for further study.

Acknowledgments

The research work described in this paper has been supported by the National Natural Science Foundation of China (No. 62476023, 61976016, 62376019, 61976015), and the authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve this paper.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. *Preprint*, arXiv:1911.02116.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Fahim Faisal, Kaiqiang Song, Song Wang, Simin Ma, Shujian Liu, Haoyun Deng, and Sathish Reddy Indurthi. 2025. Aligning multilingual reasoning with verifiable semantics from a high-resource expert model. *Preprint*, arXiv:2509.25543.

Yuchun Fan, Yongyu Mu, YiLin Wang, Lei Huang, Junhao Ruan, Bei Li, Tong Xiao, Shujian Huang, Xiaocheng Feng, and Jingbo Zhu. 2025a. SLAM: Towards efficient multilingual reasoning via selective language alignment. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9499–9515, Abu Dhabi, UAE. Association for Computational Linguistics.

Yuchun Fan, Yilin Wang, Yongyu Mu, Lei Huang, Bei Li, Xiaocheng Feng, Tong Xiao, and Jingbo Zhu. 2025b. Language-specific layer matters: Efficient multilingual enhancement for large vision-language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 12473–12500.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Kaiyu Huang, Fengran Mo, Xinyu Zhang, Hongliang Li, You Li, Yuanchi Zhang, Weijian Yi, Yulong Mao, Jinchen Liu, Yuzhuang Xu, Jinan Xu, Jian-Yun Nie, and Yang Liu. 2025. A survey on large language models with multilingualism: Recent advances and new frontiers. *Preprint*, arXiv:2405.10936.

Jaedong Hwang, Kumar Tanmay, Seok-Jin Lee, Ayush Agrawal, Hamid Palangi, Kumar Ayush, Ila Fiete, and Paul Pu Liang. 2025. Learn globally, speak locally: Bridging the gaps in multilingual reasoning. *Preprint*, arXiv:2507.05418.

Jungyup Lee, Jemin Kim, Sang Park, and SeungJae Lee. 2025. Making qwen3 think in korean with reinforcement learning. *Preprint*, arXiv:2508.10355.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *Preprint*, arXiv:2305.20050.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J. Liu, and Jialu Liu. 2024. Statistical rejection sampling improves preference optimization. *Preprint*, arXiv:2309.06657.

Wenyang Luo, Wayne Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. 2025. Mmath: A multilingual benchmark for mathematical reasoning. *Preprint*, arXiv:2505.19126.

Mistral-AI. 2025. Magistral. *arXiv e-prints*, arXiv:2506.10910.

OpenAI. 2025. Competitive programming with large reasoning models. *Preprint*, arXiv:2502.06807.

Cheonbok Park, Jeonghoon Kim, Joosung Lee, Sanghwan Bae, Jaegul Choo, and Kang Min Yoo. 2025. Cross-lingual collapse: How language-centric foundation models shape reasoning in large language models. *Preprint*, arXiv:2506.05850.

Rui Qi, Zhibo Man, Yufeng Chen, Fengran Mo, Jinan Xu, and Kaiyu Huang. 2025. SoT: Structured-of-thought prompting guides multilingual reasoning in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 11024–11039, Suzhou, China. Association for Computational Linguistics.

Rui Qi, Fengran Mo, Yufeng Chen, Xue Zhang, Shuo Wang, Hongliang Li, Jinan Xu, Meng Jiang, Jian-Yun Nie, and Kaiyu Huang. 2026. Language-coupled reinforcement learning for multilingual retrieval-augmented generation. *Preprint*, arXiv:2601.14896.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

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.

Shuaijie She, Wei Zou, Shujian Huang, Wenhao Zhu, Xiang Liu, Xiang Geng, and Jiajun Chen. 2024. Mapo: Advancing multilingual reasoning through multilingual alignment-as-preference optimization. *Preprint*, arXiv:2401.06838.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Zhi Rui Tam, Cheng-Kuang Wu, Yu Ying Chiu, Chieh-Yen Lin, Yun-Nung Chen, and Hung yi Lee. 2025. Language matters: How do multilingual input and reasoning paths affect large reasoning models? *Preprint*, arXiv:2505.17407.

Bryan Chen Zhengyu Tan, Weihua Zheng, Zhengyuan Liu, Nancy F. Chen, Hwaran Lee, Kenny Tsu Wei Choo, and Roy Ka-Wei Lee. 2026. BLEnD-vis: Benchmarking multimodal cultural understanding in vision language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4647–4669, Rabat, Morocco. Association for Computational Linguistics.

Haoyu Wang, Shuo Wang, Yukun Yan, Xujia Wang, Zhiyu Yang, Yuzhuang Xu, Zhenghao Liu, Ning Ding, Xu Han, Zhiyuan Liu, and Maosong Sun. 2024. Ultralink: An open-source knowledge-enhanced multilingual supervised fine-tuning dataset. *Preprint*, arXiv:2402.04588.

Jiaan Wang, Fandong Meng, and Jie Zhou. 2025a. Deeptrans: Deep reasoning translation via reinforcement learning. *Preprint*, arXiv:2504.10187.

Jiaan Wang, Fandong Meng, and Jie Zhou. 2025b. Extrans: Multilingual deep reasoning translation via exemplar-enhanced reinforcement learning. *Preprint*, arXiv:2505.12996.

Mingyang Wang, Lukas Lange, Heike Adel, Yunpu Ma, Jannik Strötgen, and Hinrich Schütze. 2025c. Language mixing in reasoning language models: Patterns, impact, and internal causes. *Preprint*, arXiv:2505.14815.

Yiming Wang, Pei Zhang, Jialong Tang, Haoran Wei, Baosong Yang, Rui Wang, Chenshu Sun, Feitong Sun, Jiran Zhang, Junxuan Wu, Qiqian Cang, Yichang Zhang, Fei Huang, Junyang Lin, Fei Huang, and Jingren Zhou. 2025d. Polymath: Evaluating mathematical reasoning in multilingual contexts. *arXiv preprint arXiv:2504.18428*.

Zheng Weihua, Xin Huang, Zhengyuan Liu, Tarun Kumar Vangani, Bowei Zou, Xiyan Tao, Yuhao Wu, AiTi Aw, Nancy F Chen, and Roy Ka-Wei Lee. 2026. Adamcot: Rethinking cross-lingual factual reasoning through adaptive multilingual chain-of-thought. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33863–33871.

Zheng Weihua, Roy Ka-Wei Lee, Zhengyuan Liu, Wu Kui, AiTi Aw, and Bowei Zou. 2025. CCL-XCoT: An efficient cross-lingual knowledge transfer method for mitigating hallucination generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1768–1788, Suzhou, China. Association for Computational Linguistics.

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.

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. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Wen Yang, Junhong Wu, Chen Wang, Chengqing Zong, and Jiajun Zhang. 2025b. Language imbalance driven rewarding for multilingual self-improving. *Preprint*, arXiv:2410.08964.

Songming Zhang, Xue Zhang, Tong Zhang, Bojie Hu, Yufeng Chen, and Jinan Xu. 2025a. Alignditil: Token-level language model alignment as adaptive policy distillation. *Preprint*, arXiv:2503.02832.

Xue Zhang, Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2024. Multilingual knowledge editing with language-agnostic factual neurons. *Preprint*, arXiv:2406.16416.

Xue Zhang, Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025b. Cm-align: Consistency-based multilingual alignment for large language models. *Preprint*, arXiv:2509.08541.

Xue Zhang, Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025c. Less, but better: Efficient multilingual expansion for LLMs via layer-wise mixture-of-experts. In *Proceedings of the 63rd Annual Meeting of the Association*

for Computational Linguistics (Volume 1: Long Papers), pages 17948–17963, Vienna, Austria. Association for Computational Linguistics.

Xue Zhang, Songming Zhang, Yunlong Liang, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025d. A dual-space framework for general knowledge distillation of large language models. *Preprint*, arXiv:2504.11426.

Weihua Zheng, Zhengyuan Liu, Tanmoy Chakraborty, Weiwen Xu, Xiaoxue Gao, Bryan Chen Zhengyu Tan, Bowei Zou, Chang Liu, Yujia Hu, Xing Xie, Xiaoyuan Yi, Jing Yao, Chaojun Wang, Long Li, Rui Liu, Huiyao Liu, Koji Inoue, Ryuichi Sumida, Tatsuya Kawahara, and 16 others. 2025. Mma-asia: A multilingual and multimodal alignment framework for culturally-grounded evaluation. *Preprint*, arXiv:2510.08608.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

A Instruction for Cross-lingual Thinking Alignment Reward

The designed judge instruction for requesting `DeepSeek-V3-0324` to evaluate the alignment ratio is as follows:

> # Task
> Analyze and quantify the consistency of **key intermediate results** between an English and a [target] thought process for a given math problem.
>
> # Inputs
> I will provide you with three items: [English Math Problem]: The original problem in English.
> [English Thought Process]: The step-by-step reasoning for solving the problem in English.
> [[target] Thought Process]: The step-by-step reasoning for solving the problem in [target].
>
> # Instructions
> You must perform the following analysis internally:
> Identify **all key intermediate results** from the [English Thought Process]. Key results include variable definitions, equations, critical calculation values, and the final answer.
> For each key result identified in English, find its **mathematical equivalent** in the [[target] Thought Process].
> Calculate the **consistency score** using the following formula: **Score = (Number of matched, mathematically equivalent pairs) / (Total number of key results identified in the English process)**
>
> # Output Format
> Your final output MUST BE **a single decimal number between 0 and 1**. And the number should be wrapped by <score> and </score>. Do NOT include any text, explanation, titles, analysis, or any other characters. The response must only be the number itself wrapped by <score> and </score>.
>
> Example of a valid response:
> <score>0.925</score>
> ———
> [English Math Problem]: [en-question]
> [English Thought Process]: [en-think]
> [[target] Thought Process]: [x-think]

B Experimental Details

B.1 Introduction of Different Languages

The language families and writing systems (Zhang et al., 2025c) of all ID/OOD languages are listed in Table 8. Specifically, *fr*, *pt*, and *es* all belong to the Romance branch of the Indo-European family, *ja*

Languages	Language Family	Writing System
English (*en*)	Indo-European (Germanic)	Latin alphabet (26 letters)
French (*fr*)	Indo-European (Romance)	Latin alphabet (26 letters)
Portuguese (*pt*)	Indo-European (Romance)	Latin alphabet (26 letters + diacritics)
Spanish (*es*)	Indo-European (Romance)	Latin alphabet (27 letters, including ñ)
Japanese (*ja*)	Japonic (Isolate Language)	Japanese script (Kanji + Hiragana + Katakana)
Korean (*ko*)	Koreanic (Isolate Language)	Hangul (24 basic letters, often syllabically grouped)
Thai (*th*)	Kra–Dai (Tai)	Thai script (44 consonants + vowel symbols, abugida)
Arabic (*ar*)	Afro-Asiatic (Semitic)	Arabic script (28 letters, right-to-left)
Vietnamese (*vi*)	Austroasiatic (Vietic)	Latin alphabet (Vietnamese variant) with diacritics (29 letters)
Chinese (*zh*)	Sino-Tibetan (Sinitic)	Chinese characters

Table 8: The detailed language families and writing systems for all ID/OOD languages.

is often considered as the Isolate language, through its writing system incorporates Kanji, which originated from *zh*.

B.2 Evaluation Details for PolyMath

PolyMath (Wang et al., 2025d) is a multilingual mathematical reasoning benchmark covering 18 languages and 4 easy-to-hard difficulty levels. In our experiments, we only test 10 languages overlapped with MMATH. For PolyMath, we also conduct each evaluation four times and report the average result across all runs. Differently, we report the Difficulty-Weighted Accuracy (DW-ACC) (Wang et al., 2025d), which assigns level-specific weights w_1, w_2, w_3, w_4 to each problem from the low/medium/high/top level, respectively. Specifically, the weights double at each ascending level: $w_1 = 1$, $w_2 = 2$, $w_3 = 4$, and $w_4 = 8$, which provides a more reliable measure of performance by minimizing the impact of success on easier problems and placing greater emphasis on correct answers at higher difficulty levels. Given the accuracy at each level a_1, a_2, a_3, a_4, DW-ACC is defined as:

$$\text{DW-ACC} = \frac{\sum_{i=1}^{4} w_i a_i}{\sum_{i=1}^{4} w_i} = \sum_{i=1}^{4} \left(\frac{2^{i-1}}{15} a_i \right). \quad (9)$$

Based on DW-ACC, we also calculate and report the LC&DW-ACC.

B.3 Implementation Details

Cold-Start SFT. We use the Llama-Factory[11] framework (Zheng et al., 2024) for the cold-start SFT (Zhang et al., 2025d,a, 2024; Qi et al., 2025, 2026). For DeepSeek-R1-Distill-Qwen-1.5B, we set the learning rate to 1e-6, the batch size to 256, and the training epoch to 1. For DeepSeek-R1-Distill-Qwen-7B, we set the learning rate to 5e-7, the batch size to 256, and the training epoch to 1. All SFT experiments are conducted on 1×NVIDIA H20 GPUs (96G). DeepSpeed ZeRO-2/ZeRO-3 optimization (Rasley et al., 2020) during SFT is adopted for DeepSeek-R1-Distill-Qwen-1.5B/7B, respectively. Additionally, we deploy `DeepSeek-V3-0324` and `DeepSeek-R1-0528` on 2×NVIDIA H20 GPU (96G) during the construction of the training dataset for the cold-start SFT.

RL Training. Following previous work (DeepSeek-AI, 2025; Wang et al., 2025a,b), We use GRPO algorithm implemented by verl[12] (Sheng et al., 2024). We conduct all RL training experiments on 8×8 H20 GPUs, and we use another 2×8 H20 GPUs to deploy `DeepSeek-V3-0324` to calculate the CTA reward. For DeepSeek-R1-Distill-Qwen-1.5B/7B, we set the batch size to 512, the learning rate to 5e-6/3e-6, the rollout number to 8 and the rollout temperature to 0.9, and the KL loss coefficient to 0.0. The number of training epochs is set to 15. For Iter-1 and Iter-2, we set the max sequence length to 16384 and 24000, respectively.

Generation Details. During evaluation, we use the vLLM toolkit[13] to accelerate the model generation process. For the original backbone and no-training baselines, we use the recommended sampling decoding strategy (DeepSeek-AI, 2025) with 0.6 temperature and 0.95 top-p value. For other training baselines, we set the sampling decoding strategy with 0.9 temperature and 0.95 top-p value for the best performance. During the RL training, we test the checkpoints from step-320 to step-435 (per 5 steps) for the best performance.

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

[12] https://github.com/volcengine/verl

[13] https://github.com/vllm-project/vllm

B.4 Instructions of No-Training Baselines

We show the detailed instructions for Prompt-Control, DIT, and QRT in Figure 2, 3, and 4, respectively.

```
'en': "Use English to think and answer.",
'es': "Usa español para pensar y responder.",
'fr': "Utilisez le français pour penser et répondre.",
'zh': "使用中文进行思考和回答。",
'ja': "日本語を使って考え、回答してください。",
'th': "ใช้ภาษาไทยในการคิดและตอบคำถาม.",
'ko': "한국어로 생각하고 답변하세요.",
'pt': "Use português para pensar e responder.",
'vi': "Sử dụng tiếng Việt để suy nghĩ và trả lời.",
'ar': "استخدم العربية للتفكير والإجابة.",
```

Figure 2: The language control instructions (Wang et al., 2025d) of the Prompt-Control baseline.

```
'en': "Alright, Okay",
'es': "Buneo",
'fr': "Bon",
'zh': "嗯, 好",
'ja': "まず",
'th': "โอเค",
'ko': "좋아",
'pt': "Ok, Bem",
'vi': "Được rồi, Đầu tiên",
'ar': " حسنًا "
```

Figure 3: The multilingual discourse marks for each language (Luo et al., 2025) of the DIT baseline.

B.5 Other Alternative Judge Metrics of the CTA Reward

While this translation-consistency approach can be adapted as a reward function for cross-lingual alignment, we suspect that it may not be an effective additional reward signal from two primary concerns: (1) Translation models often degrade when processing long chain-of-thought (CoT) reasoning heavily interspersed with mathematical formulas (LaTeX). This may result in erroneous reward signals and lead to unstable RL training. As evidenced in Table 4 of CM-Align (Zhang et al., 2025b), MAPO struggles to maintain high reward accuracy across different models. (2) Fundamentally, a translation-based reward encourages the model to generate literal translations of the English reasoning path. This forces the model into "translationese" rather than allowing it to "think natively" in the target language. Embedding-similarity-based methods also suffer from the same limitations. In contrast, our CTA reward checks for the logical equivalence of key intermediate steps. This allows the model to employ native reasoning patterns and syntactic structures as long as the key steps remain correct.

C Additional Results

C.1 Results of Qwen3-4B-Thinking-2507

We list the LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the Qwen3-4B-Thinking-2507 backbone in Table 9. The results demonstrate the superiority of our method on both language consistency and answer accuracy over other baselines. Due to the time and resource limitations, we only conduct GRPO training for one iteration. The effectiveness of the iterative training strategy has been proven in the other backbones.

Particularly, the performance drop of Acc compared to the original Qwen3-4B-Thinking-2507 is likely because this model has undergone excellent post-training. That is, when answering multilingual questions, it first translates the question into English, then thinks in English, and finally answers in the target language. However, our method forces it to switch back to native thinking, disrupting its learned patterns, which may lead to a decrease in Acc.

Additionally, our M-Thinker framework (LC + CTA rewards + training strategies) is architecture-agnostic. The consistent improvements across 1.5B, 4B, and 7B models suggest strong potential for generalization to other LLMs.

C.2 Detailed Results of DeepSeek-R1-Distill-Qwen-1.5B

We report the LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the DeepSeek-R1-Distill-Qwen-1.5B backbone for each language in Table 10.

C.3 Hard vs. Soft Language Consistency Reward

To compare the performance of hard/soft LC reward, we conduct GRPO training from the same cold-start SFT model and report the results in Table 11. The results reveal distinct behaviors across model sizes. For the 1.5B Model, the hard LC constraint ensures higher language consistency but leads to a drop in answer accuracy (Acc) compared to the soft LC reward, as the smaller model struggles to satisfy strict language constraints while reasoning correctly. For the 7B Model, the accuracy gap becomes negligible (the ALL-AVG "58.83" of Hard-LC even surpasses the one "58.59" of Soft-LC). Additionally, the Hard-LC reward significantly outperforms the Soft-LC reward in maintaining language consistency ("95.8" vs. "75.12").

```
'en': "OK, so the problem is {question}. Let me think in English. First",
'es': "Bien, el problema es {question}. Déjame pensar en español. Primero",
'fr': "D\'accord, donc le problème est {question}. Laissez-moi réfléchir en français. D\'abord",
'zh': "好的，问题是{question}。让我用中文思考一下。首先",
'ja': "わかりました。問題は{question}です。日本語で考えさせてください。まず",
'th': "ตกลง ดังนั้นปัญหาคือ{question} ให้ฉันคิดเป็นภาษาไทยก่อนอื่น",
'ko': "좋습니다. 문제는 {question}입니다. 한국어로 생각해 보겠습니다. 먼저",
'pt': "Ok, então o problema é {question}. Deixe-me pensar em português. Primeiro",
'vi': "Được rồi, vấn đề là {question}. Hãy để tôi nghĩ bằng tiếng Việt. Đầu tiên",
'ar': "حسنًا، المشكلة هي {question}. دعني أفكر باللغة العربية. أولاً"
```

Figure 4: The restatement instructions (Luo et al., 2025) of the QRT baseline.

Methods	In-Domain Languages						Out-of-Domain Languages						
	ja	ko	fr	pt	th	*ID-AVG*	en	es	ar	vi	zh	*OOD-AVG*	*ALL-AVG*
***Metric:* Language Consistency (LC, %)**													
Qwen3-4B-Thinking-2507	0.00	0.00	0.00	0.00	0.00	0.00	99.79	0.00	0.00	0.00	65.52	33.06	16.53
Prompt-Control (No Training)	0.00	0.00	0.00	0.00	0.00	0.00	99.94	0.00	0.02	0.02	68.50	33.70	16.85
DIT (No Training)	95.13	16.66	2.37	94.31	9.44	43.58	99.94	97.71	64.13	98.72	82.61	88.62	66.10
QRT (No Training)	99.13	97.17	96.87	92.35	96.31	96.37	**99.96**	**99.32**	96.83	**99.35**	**84.21**	**95.94**	96.15
Cold-Start SFT	68.73	54.72	16.64	3.97	17.64	32.34	89.50	9.89	1.66	24.18	64.67	37.98	35.16
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	96.01	0.00	0.00	0.00	10.87	21.38	10.69
SLC-RL	76.00	65.02	78.94	71.72	80.71	74.48	87.99	67.70	61.75	69.34	68.20	71.00	72.74
M-Thinker-4B ⇒ Iter-1 (Ours)	**99.61**	**99.86**	**99.46**	**99.44**	**99.90**	**99.66**	99.42	98.53	**99.61**	99.33	82.52	95.88	**97.77**
***Metric:* Accuracy (Acc, %)**													
Qwen3-4B-Thinking-2507	**76.15**	**77.30**	**78.70**	79.17	**76.06**	**77.48**	84.36	80.15	74.51	77.81	75.42	**78.45**	**77.96**
Prompt-Control (No Training)	75.72	76.85	78.10	79.70	75.37	77.15	**85.10**	77.95	73.66	77.35	74.30	77.67	77.41
DIT (No Training)	70.63	72.07	77.88	77.20	75.44	74.64	80.55	78.72	70.74	73.01	78.20	76.24	75.44
QRT (No Training)	69.16	67.14	75.78	78.14	66.80	71.40	78.12	**80.65**	66.30	74.80	76.86	75.34	73.37
Cold-Start SFT	64.41	68.72	78.55	**80.24**	75.77	73.54	83.05	77.95	76.09	77.23	76.90	78.24	75.89
Naive-RL	76.07	77.13	77.57	78.29	75.82	76.98	77.52	78.11	**76.32**	**79.03**	77.60	77.72	77.35
SLC-RL	58.78	58.47	71.82	75.60	60.89	65.11	81.11	75.37	52.83	68.25	75.51	70.61	67.86
M-Thinker-4B ⇒ Iter-1 (Ours)	71.41	68.11	76.38	74.64	67.23	71.56	82.24	78.63	71.23	73.27	**78.57**	76.79	74.17
***Metric:* Language Consistency & Accuracy (LC&Acc, %)**													
Qwen3-4B-Thinking-2507	0.00	0.00	0.00	0.00	0.00	0.00	84.36	0.00	0.00	0.00	47.40	26.35	13.18
Prompt-Control (No Training)	0.00	0.00	0.00	0.00	0.00	0.00	**85.03**	0.00	0.02	0.02	49.59	26.93	13.47
DIT (No Training)	66.67	10.31	2.37	72.97	8.84	32.23	80.51	76.86	44.89	72.35	63.33	67.59	49.91
QRT (No Training)	68.96	65.32	73.68	71.99	64.08	68.81	78.10	80.39	65.73	**74.59**	63.91	72.54	70.68
Cold-Start SFT	50.88	41.79	12.94	3.20	13.12	24.39	82.99	8.50	1.62	17.40	50.51	32.21	28.30
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	74.20	0.00	0.00	0.00	10.46	16.93	8.47
SLC-RL	56.25	48.44	66.37	62.33	59.68	58.61	81.07	59.93	36.46	51.86	52.14	56.29	57.45
M-Thinker-4B ⇒ Iter-1 (Ours)	**71.03**	**67.96**	**75.84**	**74.11**	**67.17**	**71.22**	81.87	**77.51**	**70.85**	73.23	**65.36**	**73.76**	**72.49**

Table 9: The LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the Qwen3-4B-Thinking-2507 backbone. "***ID-avg*/*OOD-avg***" is the average result of five In-Domain/Out-of-Domain languages and "***ALL-AVG***" is the average result of all ten languages. The result in **bold** means the best result.

In summary, the performance achieved with either hard-LC or soft-LC ultimately depends on the model's inherent capability: the stronger the model, the better hard-LC can simultaneously ensure linguistic consistency and answer accuracy.

C.4 SLC-RL with Different Reward Magnitudes

For a clear comparison, we also conduct SLC-RL with the same reward magnitude as ours. First, we list the detailed reward scores of different methods in Table 12. Specifically, "SLC-RL-s" utilizes the same LC-reward magnitude as our method; however, it still underperforms our method as shown in Table 13. The results demonstrate that the hard LC reward facilitates higher language consistency, and the effectiveness of our M-Thinker stems from the strict LC reward and the CTA reward rather than merely the LC reward scale.

C.5 Results of PolyMath

We report the LC, DW-ACC, and LC&DW-ACC (%) evaluation results on the PolyMath benchmark of the DeepSeek-R1-Distill-Qwen-1.5B/7B backbones in Table 14. These results also demonstrate the superiority of our M-Thinker-1.5B/7B.

D Additional Analysis

D.1 Detailed Ablation Results

We list the detailed ablation results of the MMATH benchmark based on our M-Thinker-1.5B (Iter-1) in Table 15.

Methods	In-Domain Languages						Out-of-Domain Languages						
	ja	ko	fr	pt	th	*ID-AVG*	en	es	ar	vi	zh	*OOD-AVG*	*ALL-AVG*
***Metric:* Language Consistency (LC, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	0.70	0.25	10.90	17.48	0.54	5.98	91.01	17.68	0.62	8.24	63.00	36.11	21.04
Prompt-Control (No Training)	4.41	0.04	20.35	35.90	2.49	12.64	92.63	40.93	3.97	39.89	65.19	48.52	30.58
DIT (No Training)	15.34	0.29	48.41	44.85	3.54	22.48	90.25	32.91	4.18	27.64	61.50	43.30	32.89
QRT (No Training)	12.21	0.08	52.72	41.60	10.71	23.47	90.97	33.39	9.34	26.41	67.19	45.46	34.46
Cold-Start SFT	1.81	0.00	49.82	54.34	12.68	23.73	90.39	42.53	2.01	26.06	77.77	47.75	35.74
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	99.61	0.00	0.00	0.00	55.23	30.97	15.48
SLC-RL	0.00	0.00	0.00	0.00	0.00	0.00	**100.00**	0.00	0.00	0.00	85.79	37.16	18.58
M-Thinker-1.5B ⇒ Iter-1 (Ours)	98.68	98.17	99.54	99.70	99.84	99.19	98.44	**99.38**	**33.31**	**99.40**	91.88	**84.48**	**91.83**
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**99.76**	**98.23**	**99.73**	**99.84**	**99.88**	**99.49**	96.31	98.30	11.03	99.06	**92.86**	79.51	89.50
***Metric:* Accuracy (Acc, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	34.28	32.48	36.91	39.22	31.17	34.81	47.47	40.37	37.07	36.45	37.77	39.83	37.32
Prompt-Control (No Training)	30.15	31.34	39.81	32.74	25.71	31.95	47.31	32.83	29.26	20.24	38.11	33.55	32.75
DIT (No Training)	19.39	17.41	31.51	28.66	18.53	23.10	47.52	27.10	11.49	17.31	39.09	28.50	25.80
QRT (No Training)	14.89	16.51	28.16	30.06	6.68	19.26	45.55	26.10	10.25	16.67	42.01	28.11	23.69
Cold-Start SFT	24.59	16.45	24.42	20.60	9.86	19.18	46.29	23.48	16.67	12.78	39.74	27.79	23.49
Naive-RL	**51.12**	**50.15**	**54.52**	**52.58**	**41.58**	**49.99**	55.36	**53.83**	**45.09**	**47.70**	48.45	**50.08**	**50.04**
SLC-RL	46.69	43.80	54.23	49.69	39.57	46.80	56.37	53.51	42.95	46.11	46.86	49.16	47.98
M-Thinker-1.5B ⇒ Iter-1 (Ours)	34.37	24.90	43.76	46.02	28.88	35.59	54.97	49.37	31.33	36.26	49.15	44.22	39.90
M-Thinker-1.5B ⇒ Iter-2 (Ours)	45.72	33.40	50.02	51.63	32.80	42.72	**56.51**	49.42	37.14	37.73	**51.85**	46.53	44.62
***Metric:* Language Consistency & Accuracy (LC&Acc, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	0.22	0.02	7.05	11.92	0.12	3.87	46.56	13.38	0.16	3.56	32.30	19.19	11.53
Prompt-Control (No Training)	0.98	0.02	9.69	17.34	0.22	5.65	46.42	19.65	0.62	13.52	31.40	22.32	13.99
DIT (No Training)	7.83	0.06	25.99	23.36	0.55	11.56	47.10	22.38	1.93	13.74	33.23	23.68	17.62
QRT (No Training)	6.10	0.06	25.22	25.17	1.86	11.68	45.45	21.61	2.48	13.66	36.17	23.87	17.78
Cold-Start SFT	1.11	0.00	17.29	16.99	1.56	7.39	45.84	20.54	0.52	7.25	34.51	21.73	14.56
Naive-RL	0.00	0.00	0.00	0.00	0.00	0.00	55.31	0.00	0.00	0.00	25.47	16.16	8.08
SLC-RL	0.00	0.00	0.00	0.00	0.00	0.00	56.37	0.00	0.00	0.00	40.99	19.47	9.74
M-Thinker-1.5B ⇒ Iter-1 (Ours)	34.25	24.48	43.72	45.78	28.72	35.39	54.89	49.19	**6.39**	35.76	45.60	38.37	36.88
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**45.54**	**32.86**	**49.75**	**51.47**	**32.72**	**42.47**	**56.41**	**49.20**	2.80	**37.55**	**48.20**	**38.83**	**40.65**

Table 10: The LC, Acc, and LC&Acc (%) results on the MMATH benchmark of the DeepSeek-R1-Distill-Qwen-1.5B backbone. "***ID-avg/OOD-avg***" is the average result of five In-Domain/Out-of-Domain languages and "***ALL-AVG***" is the average result of all ten languages. The result in **bold** means the best result, and the underlined result means the second-best result in each setting. "**Iter-1/2**" means the training iteration 1/2.

Settings	LC			Acc			LC&Acc		
	ID-AVG	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*	*ID-AVG*	*OOD-AVG*	*ALL-AVG*
w/ Hard-LC (1.5B)	**99.16**	**92.44**	**95.80**	31.72	39.85	35.78	**31.68**	**37.18**	**34.43**
w/ Soft-LC (1.5B)	0.00	37.16	18.58	**46.80**	**49.16**	**47.98**	0.00	19.47	9.74
w/ Hard-LC (7B)	**99.34**	**92.27**	**95.80**	55.56	**62.09**	**58.83**	**55.47**	**58.16**	**56.81**
w/ Soft-LC (7B)	76.00	74.24	75.12	**56.64**	60.55	58.59	43.14	46.88	45.01

Table 11: The detailed comparison between the "Hard-LC" (with LC/Format/Acc rewards) and "Soft-LC" (with SLC/Format/Acc rewards).

D.2 Annotation Guidelines of Human Evaluation

The annotation follows a rigorous two-step procedure:

- **Reference Extraction:** First, experts proficient in English extract key intermediate reasoning steps from the reference English thinking paths.
- **Alignment Verification:** Experts proficient in the target languages are then provided with the non-English thinking paths and the extracted English key steps. They independently identify key intermediate steps in the target language and calculate the alignment ratio with the English counterparts.

Format	Language Consistency	Accuracy	SLC-RL	SLC-RL-s	Ours (*w/o* R_{cta})	Ours
✗	/	/	0	-1	-1	-1
✓	✗	✗	1*(0+0)=0	1*(-1+0)=-1	-1	-1
✓	✗	✓	1*(0+0.9)=0.9	1*(-1+1)=0	-1	-1
✓	✓	✗	1*(0.1+0)=0.1	1*(0+0)=0	0*1=0	0*(1+R_{cta})=0
✓	✓	✓	1*(0.1+0.9)=1	1*(0+1)=1	1*1=1	1*(1+R_{cta})=1+R_{cta}

Table 12: The detailed reward scores of different methods on different scenarios. "SLC-RL-s" means the same scale of soft-LC reward with our R_{lc}. "Ours" represents M-thinker-1.5B (Iter-1).

		LC			Acc			LC&Acc		
Methods	**LC/Acc Reward**	***ID-AVG***	***OOD-AVG***	***ALL-AVG***	***ID-AVG***	***OOD-AVG***	***ALL-AVG***	***ID-AVG***	***OOD-AVG***	***ALL-AVG***
SLC-RL	[0,0.1]+[0,0.9]	0.00	37.16	18.58	46.80	49.16	47.98	0.00	19.47	9.74
SLC-RL-s	[-1,0]+[0,1]	39.34	57.46	48.40	44.66	48.78	46.72	16.95	28.96	22.96
Ours (*w/o* R_{cta})	[-1,0]&[0,1]	99.16	92.44	95.80	31.72	39.85	35.78	31.68	37.18	34.43
Ours	[-1,0]&[0,1](+R_{cta})	99.19	84.48	91.83	35.59	44.22	39.90	35.39	38.37	36.88

Table 13: The results of SLC-RL with different reward magnitudes. "SLC-RL-s" means the same scale of soft-LC reward with our R_{lc}. "Ours" represents M-thinker-1.5B (Iter-1).

	In-Domain Languages						Out-of-Domain Languages						
Methods	**ja**	**ko**	**fr**	**pt**	**th**	***ID-AVG***	**en**	**es**	**ar**	**vi**	**zh**	***OOD-AVG***	***ALL-AVG***
***Metric:* Language Consistency (LC, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	7.30	0.15	25.65	25.80	8.45	13.47	91.30	26.55	9.05	22.90	63.35	42.63	28.05
M-Thinker-1.5B ⇒ Iter-1 (Ours)	98.25	96.40	**99.85**	99.00	99.70	98.64	97.40	**99.40**	**40.40**	97.50	88.60	**84.66**	**91.65**
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**99.40**	**98.65**	99.80	99.00	**99.85**	**99.34**	**97.50**	98.90	19.65	**99.25**	**90.10**	81.08	90.21
DeepSeek-R1-Distill-Qwen-7B	20.85	11.35	26.80	24.10	14.85	19.59	96.05	26.20	14.90	26.30	67.70	46.23	32.91
M-Thinker-7B ⇒ Iter-1 (Ours)	**99.05**	**97.65**	**99.85**	**99.25**	**98.40**	**98.84**	**99.80**	**99.65**	**83.75**	99.80	**89.70**	**94.54**	**96.69**
M-Thinker-7B ⇒ Iter-2 (Ours)	98.75	95.30	99.65	99.00	94.65	97.47	97.55	98.65	64.80	**100.00**	89.25	90.05	93.76
***Metric:* Difficulty-Weighted Accuracy (DW-ACC, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	13.60	15.77	18.62	18.73	11.25	15.59	21.23	19.47	14.36	18.25	20.00	18.66	17.13
M-Thinker-1.5B ⇒ Iter-1 (Ours)	16.64	12.33	23.03	23.40	12.77	17.63	30.13	23.23	15.90	17.42	25.08	22.35	19.99
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**19.65**	**17.39**	**24.87**	**24.76**	**16.84**	**20.70**	**32.41**	**25.63**	**19.11**	**20.83**	**27.98**	**25.19**	**22.95**
DeepSeek-R1-Distill-Qwen-7B	28.45	31.72	35.41	33.12	27.72	31.28	36.93	33.51	28.93	31.96	30.67	32.40	31.84
M-Thinker-7B ⇒ Iter-1 (Ours)	29.99	28.59	35.02	35.30	29.35	31.65	40.80	34.72	29.83	31.99	34.80	34.43	33.04
M-Thinker-7B ⇒ Iter-2 (Ours)	**35.24**	**33.92**	**40.02**	**39.73**	**34.40**	**36.66**	**42.48**	**38.33**	**37.08**	**37.19**	**42.73**	**39.56**	**38.11**
***Metric:* Language Consistency & Difficulty-Weighted Accuracy (LC&DW-ACC, %)**													
DeepSeek-R1-Distill-Qwen-1.5B	0.67	0.00	3.03	3.36	0.16	1.44	21.15	3.71	0.57	2.40	16.09	8.78	5.11
M-Thinker-1.5B ⇒ Iter-1 (Ours)	16.43	12.07	23.03	23.27	12.75	17.51	29.89	23.18	**4.55**	16.96	22.87	19.49	18.50
M-Thinker-1.5B ⇒ Iter-2 (Ours)	**19.61**	**17.04**	**24.86**	**24.51**	**16.80**	**20.56**	**32.36**	**25.44**	1.55	**20.59**	**24.62**	**20.91**	**20.74**
DeepSeek-R1-Distill-Qwen-7B	3.05	1.89	5.89	4.44	2.39	3.53	36.70	5.74	2.45	4.89	25.27	15.01	9.27
M-Thinker-7B ⇒ Iter-1 (Ours)	29.81	28.55	34.97	35.11	29.11	31.51	40.67	34.61	**27.18**	31.97	31.73	33.23	32.37
M-Thinker-7B ⇒ Iter-2 (Ours)	**35.13**	**33.13**	**39.96**	**39.36**	**33.79**	**36.27**	**42.48**	**37.96**	24.18	**37.19**	**39.24**	**36.21**	**36.24**

Table 14: The LC, DW-ACC, and LC&DW-ACC (%) results on the PolyMath benchmark of the DeepSeek-R1-Distill-Qwen-1.5B/7B backbones. The result in **bold** means the best result in each backbone.

Methods	In-Domain Languages						Out-of-Domain Languages						
	ja	ko	fr	pt	th	*ID-AVG*	en	es	ar	vi	zh	*OOD-AVG*	*ALL-AVG*
***Metric:* Language Consistency (LC, %)**													
M-Thinker-1.5B ⇒ Iter-1 (Ours)	98.68	98.17	99.54	99.70	99.84	99.19	98.44	99.38	33.31	99.40	91.88	84.48	91.83
w/o R_{cta}	99.40	97.59	99.16	99.67	99.98	99.16	98.61	98.91	73.88	99.96	90.84	92.44	95.80
w/o R_{lc}	0.00	0.00	0.00	0.00	0.00	0.00	99.88	0.00	0.00	0.00	78.17	35.61	17.80
w/o (R_{cta} & R_{lc})	0.00	0.00	0.00	0.00	0.00	0.00	99.61	0.00	0.00	0.00	55.23	30.97	15.48
***w/o* Cold-Start SFT**	99.23	99.02	98.37	99.42	99.90	99.19	95.37	99.59	35.93	99.63	91.14	84.33	91.76
***w/o* Rejection Sampling**	99.24	99.68	99.98	99.80	99.86	99.71	99.31	98.55	40.24	97.32	91.12	85.31	92.51
w/ o_t^{en} **from Light-R1 for** R_{cta}	99.98	99.82	99.52	99.46	100.00	99.76	99.73	99.61	49.19	99.55	93.26	88.27	94.01
***Metric:* Accuracy (Acc, %)**													
M-Thinker-1.5B ⇒ Iter-1 (Ours)	34.37	24.90	43.76	46.02	28.88	35.59	54.97	49.37	31.33	36.26	49.15	44.22	39.90
w/o R_{cta}	30.48	23.75	39.45	41.34	23.59	31.72	51.87	43.26	27.99	31.11	45.01	39.85	35.78
w/o R_{lc}	49.53	47.06	56.48	53.18	44.84	50.22	57.40	55.27	43.55	50.37	47.54	50.83	50.52
w/o (R_{cta} & R_{lc})	51.12	50.15	54.52	52.58	41.58	49.99	55.36	53.83	45.09	47.70	48.45	50.08	50.04
***w/o* Cold-Start SFT**	31.18	22.15	42.44	45.25	26.99	33.60	52.68	45.66	31.31	34.15	50.35	42.83	38.22
***w/o* Rejection Sampling**	34.40	19.17	45.75	44.63	25.41	33.87	54.55	43.41	28.16	33.42	46.66	41.24	37.55
w/ o_t^{en} **from Light-R1 for** R_{cta}	31.37	24.88	42.02	43.18	27.10	33.71	54.43	46.96	28.53	33.06	46.37	41.87	37.79
***Metric:* Language Consistency & Accuracy (LC&Acc, %)**													
M-Thinker-1.5B ⇒ Iter-1 (Ours)	34.25	24.48	43.72	45.78	28.72	35.39	54.89	49.19	6.39	35.76	45.60	38.37	36.88
w/o R_{cta}	30.46	23.73	39.41	41.24	23.57	31.68	51.77	42.67	19.76	31.09	40.63	37.18	34.43
w/o R_{lc}	0.00	0.00	0.00	0.00	0.00	0.00	57.28	0.00	0.00	0.00	36.03	18.66	9.33
w/o (R_{cta} & R_{lc})	0.00	0.00	0.00	0.00	0.00	0.00	55.31	0.00	0.00	0.00	25.47	16.16	8.08
***w/o* Cold-Start SFT**	31.00	21.77	42.19	44.88	26.89	33.35	50.60	45.50	8.93	34.09	45.43	36.91	35.13
***w/o* Rejection Sampling**	34.20	19.01	45.73	44.45	25.27	33.73	53.86	42.37	5.81	32.65	42.68	35.48	34.60
w/ o_t^{en} **from Light-R1 for** R_{cta}	31.35	24.80	42.00	43.10	27.10	33.67	54.39	46.63	10.90	32.63	43.68	37.65	35.66

Table 15: The detailed ablation results of the MMATH benchmark based on our M-Thinker-1.5B (Iter-1).