REPA-PRM: Reference-Evaluated Process Annotation for Process Supervision and Reward Modelling

Anonymous ACL submission

Abstract

Process supervision has played a crucial role in advancing the complex multi-step reasoning capabilities of Large Language Models (LLMs). However, ensuring high-quality and efficient automatic process annotation remains a challenge. To address this, we introduce **R**eference-**E**valuated **P**rocess **A**nnotation (**REPA**), a novel and structured framework that enables per-step annotation in a single stage. REPA evaluates each solution step by referencing one or multiple ground-truth steps with explicit reasoning for assessment. We show that reference-guided step-level evaluation effectively facilitates process supervision. Our results demonstrate that fine-tuning a base-instruct model and training a reward model using REPA annotations improve reasoning performance under both single-greedy decoding and ranking/aggregation of multiple LLM-generated outputs. Notably, we show improvements across four datasets spanning three domains: mathematical reasoning, multi-hop compositional question answering, and spatial reasoning. Our work contributes to reference-guided automatic process supervision which is underexplored and holds potential for enhancing LLM reasoning capabilities.[1]

1 Introduction

Large language models (LLMs), with large scale pre-training and instruction following, have demonstrated remarkable performance on various tasks including reasoning tasks (Wei et al., 2022a,b). However, complex multi-step reasoning still remains a challenge for LLMs even when they are trained and finetuned with ground-truth chains of thoughts (Azerbayev et al., 2024; Yu et al., 2024b). Simple answer consolidation strategies, such as self-consistency (Wang et al., 2023), can improve performance by count-based voting over multiple generations when the final answers are correct in majority of them. To alleviate this, reranking of generated outputs, with reward models trained to assess an output's correctness, has gained popularity. These reward models primarily fall under two categories: Outcome Reward Models (ORMs) (Cobbe et al., 2021; Yu et al., 2024a) are trained using outcome supervision relying on the correctness of the final answer, while Process Reward Models (PRMs) (Uesato et al., 2022; Li et al., 2023; Khalifa et al., 2023; Lightman et al., 2024) are trained using process supervision i.e. relying on the correctness of individual reasoning steps.

While PRMs are better because of targeted step-level feedback, they suffer from expensive and complex annotation requirements. Human-supervised PRMs (Uesato et al., 2022; Lightman et al., 2024) are very demanding in terms of highly skilled human evaluators. This has led to efforts towards automatic process annotations, including (i) Monte Carlo Tree Search (MCTS) based step evaluations (Wang et al., 2024a,c; Luo et al., 2024), which are based on answer correctness from several continuations and hence are computationally expensive, (ii) adaptive models gauging intermediate steps, e.g., relative step confidence thresholding (Lu et al., 2024), which are dependent on and starts from an ORM, or (iii) all-to-all single-step comparisons with reference reasoning traces (Li et al., 2023; Khalifa et al., 2023) that suffer from quadratic step comparisons and potentially erroneous evaluations that require step combinations. Works related to (i) and (ii) disregard valuable step-by-step information from ground-truth reasoning traces that are available and utilized during supervised fine-tuning (SFT), while works pertaining to (iii) utilizes the complete ground truth reasoning traces but in an inefficient, complicated, and fragmented manner.

To address these gaps, we investigate and propose **R**eference-**E**valuated **P**rocess

[1] Codebase provided along with the submission and will be made public.

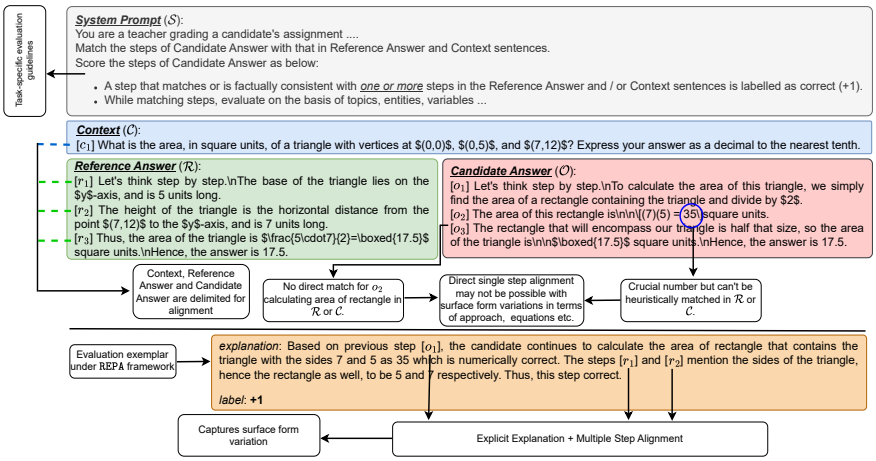

Figure 1: Determining the correctness of a step in a candidate solution against a given context and reference solution presents several challenges, including step alignment, surface-form variations, and heuristic limitations. To address these issues, we propose a unified, single-stage framework: **R**eference-**E**valuated **P**rocess **A**nnotation (**REPA**: $(\mathcal{S}, \mathcal{C}, \mathcal{R}, \mathcal{O}) \rightarrow \mathcal{E}$). REPA produces an explanation-based step-by-step evaluation $\mathcal{E}$ of a candidate model output $\mathcal{O}$, grounded to a given context $\mathcal{C}$, ground-truth reasoning $\mathcal{R}$, and system prompt $\mathcal{S}$ (Section 3.1).

Annotation (**REPA**), a framework (Figure 1) that effectively leverages the complete set of intermediate steps from the ground-truth reasoning traces ($\mathcal{R}$) and question sentences ($\mathcal{C}$) as references to evaluate the correctness of each model output ($\mathcal{O}$) response step, enabling automatic process supervision. We devise a generic and structured evaluation scheme for each step of models' output response $\mathcal{O}$ with a focus on (i) explicit reasoning for the evaluation, and (ii) multi-step comparisons between responses and references. The proposed scheme enables a single-stage evaluation that scales additively with the token length of both the response and the reference.

We demonstrate the effectiveness of REPA annotations in two settings: (i) fine-tuning a model in an offline reinforcement learning (RL) setup that improves performance with greedy decoding, and (ii) training Reward Models (RMs) for ranking and aggregating multiple LLM generations. We conduct extensive experiments on four datasets spread across three domains: two mathematical reasoning datasets, namely, GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021); one multi-hop compositional question answering dataset, namely, ♫ MuSiQue-Ans (Trivedi et al., 2022); and one spatial reasoning dataset, small SpaRP, i.e., SpaRP-S (Rizvi et al., 2024), which in turn is a collection of four textual spatial reasoning dataset covering various spatial characterizations of objects, relations, and contexts. Our approach outperforms strong baselines over majority of the datasets. In summary, our key contributions include:

- We propose a generic, single-stage, and structured reference-guided evaluation framework (REPA) with the focus on annotation reasoning and multi-step alignment. Our framework utilizes the entire ground-truth reasoning trace, generally available for SFT which is often overlooked.
- We utilize REPA annotations to improve reasoning capabilities of LLMs under various scenarios such as finetuning, reward model training, and pairwise *vs.* pointwise training.
- We extensively evaluate our framework on two mathematical datasets (GSM8K and MATH), a question-answering dataset (♫ MuSiQue-Ans), and a spatial reasoning dataset (SpaRP), demonstrating improvements over the baselines. We discuss key factors and insights of our methodology.

2 Related Work

Reasoning abilities of LLMs. Reasoning remains a challenging area for the Large Language Models (LLMs). Various prompting techniques, such as chain-of-thought, few-shot prompting and their variants (Wei et al., 2022b; Kojima et al., 2022; Yao et al., 2023; Hao et al., 2023; Bi et al., 2024) elicited reasoning capabilities in LLMs. Importance of individual steps while prompting (Fu et al., 2023; Zhou et al., 2023) was soon found to be crucial in successfully solving multi-step rea-

soning problems. While prompt-only techniques show promising results, their performances are constrained by and sensitive to prompt design and nature of tasks (Ye and Durrett, 2022). Consequently, explicitly finetuning with high-quality reasoning traces for improving LLM reasoning capabilities has become popular (Yu et al., 2024b; Luo et al., 2025).

Outcome and Process Supervision. Supervised finetuning quickly results in saturation, leading to the search for other advanced techniques and better supervision signals. Outcome supervision (Cobbe et al., 2021; Yu et al., 2024a) relies on signal based on the final answer, and hence, is easier to obtain. Process supervision offers advantages in the form of fine-grained feedback from individual reasoning steps, however, early work (Uesato et al., 2022; Pan et al., 2023; Lightman et al., 2024) relied on time-consuming and costly human annotation. To alleviate this problem, several recent approaches have emerged for automating process supervision. Monte-Carlo Tree Search (MCTS) based approaches (Wang et al., 2024a,c; Luo et al., 2024; Zhang et al., 2024) target obtaining process annotation by several continuations from intermediate steps whose correctness are evaluated based on the final step. In parallel, work that focus on efficiency rely either on adapting outcome-supervision (Lu et al., 2024), or a pipelined multi-stage approach with quadratic step comparison but single step alignment between candidate and reference solutions (Li et al., 2023; Khalifa et al., 2023). Inspired by these approaches, our work introduces a unified, single-stage and structured evaluation scheme for automatic process annotation, with flexible alignment and multi-step comparison with reference solution. We also explore its effectiveness under both finetuning and verification scenarios.

3 Our Approach

3.1 Reference-Evaluated Process Annotation (REPA)

Consider a ground-truth reasoning path $\mathcal{R} = \{r\}_{i=1}^{m}$ consisting of a sequence of m steps, a model generated output $\mathcal{O} = \{o\}_{i=1}^{n}$ consisting of n steps, and a sequence of s sentences as context with question $\mathcal{C} = \{c\}_{i=1}^{s}$. An answer or outcome annotation $y \in \mathbb{R}$ is a score indicating a measure of correctness of the model's output. Most commonly, $y = \mathbb{I}(o_n = r_m)$; i.e., the output's answer matches with the ground-truth reasoning answer. In contrast, a process annotation $\mathcal{Y} = \{y \mid y \in \mathbb{R}\}_{i=1}^{n}$ is a sequence of scalar scores assigned to the corresponding steps $o_i \in \mathcal{O}$.

We propose **R**eference-**E**valuated **P**rocess **A**nnotation (**REPA**: $(\mathcal{S}, \mathcal{C}, \mathcal{R}, \mathcal{O}) \rightarrow \mathcal{E}$) as a unified, single-stage framework that generates a step-by-step evaluation $\mathcal{E} = \{\varepsilon\}_{i=1}^{n}$ of a model output $\mathcal{O}$ with reference to a context $\mathcal{C}$, a ground truth reasoning $\mathcal{R}$, and a system prompt $\mathcal{S}$ that defines evaluation heuristics and guidelines. Each step o_i is evaluated in a structured format $\varepsilon = (e, c^+, o^+, r^+, \epsilon, y_i)$, where e provides an explicit explanation of the evaluation, ϵ is an optional list of error categories, and $y_i \in \{-1, +1\}$ is the assigned evaluation label. In addition to providing justification for the evaluation, the explanation e aligns the step o_i with potentially multiple reference steps ($o_i \mapsto c^+ \cup o^+ \cup r^+$), where $c^+ \subset \mathcal{C}$ represents a subset of relevant context sentences, $o^+ \subset \mathcal{O} \backslash \{o_i\}$ includes other related output steps, and $r^+ \subset \mathcal{R}$ consists of selected ground-truth reasoning steps used for evaluation. This evaluation scheme enables efficient process annotation with an additive token complexity of $O(s + m + n)$, and allows for multiple alignment possibilities, particularly in cases where $m \neq n$:

1. *One-to-one* – Most simple alignment where one output step aligns directly and completely with *at most* one step, making it sufficient for evaluation. The alignment can take one of the forms: (a) a single ground-truth reasoning step ($o_i \mapsto r_j$), (b) a single context sentence ($o_i \mapsto c_j$), (c) follows directly from or complements another output step ($o_i \mapsto o_l$), or (d) no alignment at all ($o_i \mapsto \varnothing$). In this case, $|c^+ \cup o^+ \cup r^+| \leq 1$.
2. *One-to-many* – An output step requires alignment with *at least* two steps for its evaluation. Such an alignment is *necessarily* required:
   i) When the model output step o_i is a composite step, either omitting minor intermediary steps or merging multiple steps into one. Its correctness must be evaluated against multiple reference steps r_j and c_k. In this case, $o^+ = \phi$ and $|c^+ \cup r^+| > 1$. This is likely when $n < m$ or $n < (m + s)$.
   ii) When the model output step o_i is a simple atomic step while the ground-truth steps are composite, its correctness must be evaluated in conjunction with at least one other output

step o_l and at least one reference step r_j or context sentence c_k. In this case, $|o^+| \geq 1$ and $|c^+ \cup r^+| \geq 1$. This is likely when $n > m$ or $n > (m + s)$.

In summary, our **REPA** framework defines step correctness based on its alignment with one or more ground-truth reasoning steps, other output steps or context sentences, assuming the ground-truth reasoning guarantees at least one valid path to the final answer. The multi-step alignment, combined with explicit step evaluation and explanations, accommodates surface form variations such as different topical approaches (Figure 1) and expression formats. This ensures that the steps are properly contextualized within the broader reasoning structure, allowing for their more accurate evaluation.

We implement REPA using LLM-based evaluation. Off-the-shelf LLMs can be directly employed with k-shot exemplars, where a small, manually annotated set of diverse examples captures key evaluation principles of REPA. To generate automatic annotations for model-produced reasoning traces, we use a model—typically from the previous fine-tuning stage in an iterative setup—to generate multiple solutions per problem via non-greedy, temperature-based decoding. Each solution is then decomposed into individual steps, typically delineated by newline characters. Our **REPA** framework subsequently evaluates the correctness of each step, enabling automatic process supervision.

3.2 Training Approach

3.2.1 REPA–based Finetuning (REPA–ORPO)

We propose REPA-based fine-tuning of a model to enhance its reasoning capabilities. The step-by-step process annotations $\mathcal{Y} = \{y_i\}_{i=1}^n$, derived using the REPA framework, can be effectively integrated with both online and offline Reinforcement Learning (RL). For ease of implementation, training stability, and resource efficiency, we employ Odds Ratio Preference Optimization (ORPO) (Hong et al., 2024) for preference training over *chosen* and *rejected* pairs ($\mathcal{O}_w$, $\mathcal{O}_l$).

In REPA-ORPO, we compute a mean aggregation $\bar{y} = \frac{1}{n}\sum y_i$ of the reasoning step annotations to quantify reasoning trace correctness and combine this with the final answer correctness y. The tuple (y, $\bar{y}$) is used as the effective score for preference pair identification, where $y_w = 1$, $y_l = -1$, and $\bar{y}_w > \bar{y}_l$. Thus REPA-ORPO employs a more comprehensive set of preference pairs, where the chosen solution demonstrates superiority over the rejected solution with respect to both reasoning quality and answer accuracy. In contrast, Outcome-ORPO is trained using preference pairs ($\mathcal{O}_w$, $\mathcal{O}_l$) that are determined solely by final answer accuracy, i.e., $y_w = 1$ and $y_l = -1$.

3.2.2 REPA–based Reward Models (REPA–RMs)

To enhance inference-time reasoning performance, we propose training REPA-based Outcome and Process Reward Models.

REPA–ORM. We propose training a REPA-based Outcome Reward Model (REPA-ORM) under a *pairwise* optimization setting between two instances:

$$\mathcal{L}_{ORM}^{pairwise} = -\log\Big(\sigma(r_\theta(\mathcal{C}, \mathcal{O}_w)) - \sigma(r_\theta(\mathcal{C}, \mathcal{O}_l))\Big) \quad (1)$$

where, σ is the sigmoid function, r_θ is the reward model that scores an output in reference to the input, $\mathcal{O}_w$ and $\mathcal{O}_l$ constitute a paired output for a given context-question $\mathcal{C}$, with $\mathcal{O}_w$ representing the chosen output, and $\mathcal{O}_l$ denoting the rejected output. Following our methodology outlined in Section 3.2.1, the REPA-ORM is trained on a superior set of preference pairs. These pairs are identified based on the effective score $(y, \bar{y})$ that combines the final answer with the mean aggregation $\bar{y}$ of the REPA annotations, such that $y_w = 1$, $y_l = -1$, and $\bar{y}_w > \bar{y}_l$.

We also train and compare a *pairwise*-ORM on preference pairs identified solely based on final answer accuracy, i.e., $y_w = 1$ and $y_l = -1$. Additionally, we implement a *pointwise*-ORM under classification setting using a cross-entropy loss:

$$\mathcal{L}_{ORM}^{pointwise} = -\Big(y \ \log \ \sigma(r_\theta(\mathcal{C}, \mathcal{O})) + (1-y) \ \log\left(1 - \sigma(r_\theta(\mathcal{C}, \mathcal{O}))\right)\Big) \quad (2)$$

REPA–PRM. We utilize the step-level evaluations y_i obtained through REPA as direct reward signals to train process reward models. The REPA-PRM is trained in a stepwise classification setting, using the following cross-entropy loss:

$$\mathcal{L}_{PRM} = -\sum_{i=1}^{n}\Big(y_i \ \log \ \sigma(r_\theta(\mathcal{C}, o_{1:i})) + (1-y_i) \ \log\left(1 - \sigma(r_\theta(\mathcal{C}, o_{1:i}))\right)\Big) \quad (3)$$

where $o_{1:i}$ is the sub-sequence of output $\mathcal{O}$ till the i^{th} step. Unlike ORMs which predict a single solution score for $\mathcal{O}$, PRMs generate a probability sequence $\mathcal{P} = \{p_i\}_{i=1}^{n}$ for each step $o_i \in \mathcal{O}$. These step-wise probabilities are aggregated into a final correctness score using functions such as min, prod (or equivalently sum_log)(Lightman et al., 2024; Wang et al., 2024a), last, and max(Wang et al., 2024c).

Ranking and Aggregation. We use reward models to score multiple generations during inference. We then either do rank-and-select (e.g. Best-of-N sampling) or weighted aggregation as:

$$\hat{a} = \underset{a}{\text{argmax}} \sum_{i=1}^{N} \mathbb{I}(a_i = a) \cdot f(\mathcal{C}, \mathcal{O}_i) \quad (4)$$

where each final answer a_i from $\mathcal{O}_i$ is grouped and weighted by a function $f(\cdot)$ over N solutions. In a simple strategy like self-consistency (Wang et al., 2023), the weighting function f accounts for the presence of an answer a_i, meaning all occurrences of an answer are given equal weight., i.e., $f(\mathcal{C}, \mathcal{O}_i) = 1$. Alternatively, the probability scores from Reward Models are used as weights for aggregation, effectively enhancing reasoning accuracy by prioritizing solutions that exhibit both correct final answers and well-structured reasoning steps.

4 Experiment Results

4.1 Experimental Set-up

Datasets. We conduct extensive experiments over a suite of reasoning datasets:

- **Mathematical Reasoning** We use two mathematical datasets, GSM8K (Cobbe et al., 2021), which is a collection of grade school math word problems, and MATH (Hendrycks et al., 2021), which contains high school competition-level math problems across seven diverse topics. For development, we create a train-validation split by dividing the training set of both datasets in a 90:10 ratio.
- **Question-Answering** We use ♫ MuSiQue-Ans dataset (Trivedi et al., 2022), a challenging multi-hop question-answering dataset constructed by composing six diverse reasoning graphs of sub-questions from five different sources. For development, we create a train-validation split by dividing the training set in an 80:20 ratio.
- **Spatial Reasoning** We use the small SpaRP (Rizvi et al., 2024), i.e., SpaRP-S dataset, which comprises four textual spatial reasoning sub-datasets covering various spatial characterizations and including benchmarks such as SpaRTUN (Mirzaee and Kordjamshidi, 2022) and StepGame (Shi et al., 2022). The objective in SpaRP is to infer the spatial relation between two objects when their direct relation is not provided in the text but can be deduced through spatial relation composition. SpaRP has its own validation set.

Models. We aim to demonstrate the use of a single Large Language Model (LLM) throughout the pipeline to improve itself. We, therefore, use the Llama-3 8B Instruct Model (Grattafiori et al., 2024) for all the three tasks:

i) As an *Evaluator* model for reference-guided step-annotations within our REPA framework. Depending on the problem diversity, we manually create structured step-by-step example evaluations per dataset, ranging from 6 for SpaRP to 56 for MATH dataset with 7 topics. Each dataset undergoes 5–shot evaluations using dataset-specific evaluation guidelines as system prompts. See Appendix A for details.[2]

ii) As a *Base-Instruct* model that is fine-tuned to enhance its reasoning performance under greedy decoding. While our framework supports multiple iterations, we limit our study to the first two due to resource constraints. The first iteration involves standard single-epoch Supervised Fine-Tuning (SFT) on the training split. In subsequent iterations, leveraging LLM-generated synthetic data, we start by generating $N = 20$ solutions per problem from the previous iteration's model with temperature 1. We annotate these solutions using final answers, and potentially, using REPA framework as well. Finally, we identify preference pairs for next-iteration training under the ORPO setting. See Appendix B for details.

iii) As *Reward Models (RMs)* used as verifiers for ranking and weighted-aggregation during inference. Similar to model finetuning, we generate $N = 20$ solutions per problem from the base-instruct model with temperature 1. We annotate

[2]Our framework could benefit from specialized evaluator models like Prometheus 2 (Kim et al., 2024), supporting further development in this area. Alternatively, larger and more capable models can also serve as evaluators.

these solutions using final answers for ORM and, additionally, the REPA framework for PRM. We pose the PRM training as a standard-language modelling task, predicting two special tokens for correct and incorrect steps against a special end-of-step ($\mathbb{EOS}$) token. We trained RMs for one epoch under both *pointwise* and *pairwise* loss setting with equivalent number of training examples, i.e., a collection of N positive and N negative examples can either be used as N pairs or $2N$ individual instances. See Appendix C for specific details. While RMs can be trained and updated using generations from each iteration of the fine-tuned base-instruct model, we limit our demonstration to the output of the first iteration due to resource constraints.

While our experiments primarily utilize the Llama-3 8B Instruct model, our methodology is broadly applicable to other models. To demonstrate the generalizability of REPA-based process supervision, we present Reward Model (RM) results using a different model family—Qwen 2.5—across two sizes: 3B and 32B.

Metrics, Baselines and Experimental Setup. We report[3] the accuracy for the GSM8K and MATH datasets, accuracy and F1 metric for the ♫ MuSiQue-Ans dataset, and the macro-F1 for the SpaRP dataset.

In the finetuning scenario, we evaluate our REPA-ORPO iteration trained on preference pairs formed using both outcome supervision and the mean reasoning scores of the step-by-step annotations. we benchmark REPA-ORPO iteration against Outcome-ORPO (preference pairs formed only by outcome supervision) and Supervised Fine-Tuning (SFT) iterations with an equivalent number of ground-truth reasoning traces.

In the verification scenario, we train and evaluate – (a) *pairwise* REPA-ORMs, equivalent to the implicit reward model of the REPA-ORPO finetuning, with preference pairs formed using both outcome supervision and mean reasoning scores of the step-by-step annotations, and (b) REPA-PRMs trained under a *pointwise* classification setting at each $\mathbb{EOS}$ token. We benchmark these against the majority-voted self-consistency (Wang et al., 2023), pairwise and pointwise ORMs. While pairwise RMs are balanced with both positive and negative examples, we randomly sample equal number of positive and negative examples to train a balanced pointwise RMs. The metrics for ORMs and PRMs are reported under both the settings – (a) with weighted-aggregation, i.e., RM-weighted self-consistency, and (b) without aggregation, i.e., Best-of-N (BoN) sampling with *only* the best scored solution considered for evaluation.

We report these RM-based evaluations on $N = 20$ solutions generated using a 1^{st} iteration SFT model. While prior work (Lightman et al., 2024; Wang et al., 2024a; Luo et al., 2024) have shown the min or prod aggregation to be the better performing aggregation strategies, other work (Wang et al., 2024c) have reported these to underperform ORM when the annotation process differs. For their annotation process, they reported last aggregation strategy, among others, to outperform ORM. We also found the min and prod aggregation strategies underperforming the ORMs, while last aggregation strategy performing the best. Hence, all the metrics are reported using the last aggregation strategy for PRMs.

4.2 Results and Discussion

REPA Helps Base Instruct Finetuning. We report the performances of finetuning LLM followed by greedy decoding in Table 1. REPA-ORPO is shown to perform the best across three of the four datasets, with an improvement of at least 0.26 points for MATH dataset and at most 1.21 points for the SpaRP-S dataset, compared to the next best Outcome-ORPO models. This underscores the effectiveness of REPA in reasoning step annotation and identifying *superior* preference pairs than outcome-only preference pairs. Both these ORPO models significantly outperform the SFT models trained on ground-truth reasoning traces, except for the GSM8K dataset. In the case of GSM8K, we observe a marginal performance decrease for both the ORPO finetuned models and a performance stagnation for the 2^{nd} SFT iteration compared to the 1^{st} SFT iteration. This may indicate the base-instruct Llama-3 8B model is saturated with the GSM8K data and may require working with larger, extended and augmented versions such as MetaMATH (Yu et al., 2024b).

REPA Improves Reward Model Training. Table 2 shows that REPA-PRM performs the best

[3]Exact Match as accuracy for GSM8K. *competition_math* metric as implemented in the 🤗 evaluate library for the MATH dataset. Implementation from the source github repository for the ♫ MuSiQue-Ans dataset. The F1 metric implementation in the scikit-learn library for the SpaRP dataset.

Training Method	Mathematical Reasoning GSM8K	Mathematical Reasoning MATH	Question Answering ♫ MuSiQue-Ans	Spatial Reasoning SpaRP-S
	Acc. (↑)	Acc. (↑)	Acc. (↑) / F1 (↑)	F1 (↑)
SFT 1^{st} Iteration	**70.43**	21.22	23.58 / 32.53	35.00
SFT 2^{nd} Iteration	**70.43**	22.08	26.31 / 35.12	47.13
SFT + Outcome-ORPO	69.07	<u>23.16</u>	<u>38.15</u> / <u>49.85</u>	<u>49.75</u>
SFT + REPA-ORPO	<u>69.75</u>	**23.42**	**38.89 / 50.53**	**50.96**

Table 1: Performance evaluations of Llama-3 8B Instruct model with *greedy decoding* under different training methods. Best values in **bold**, second best in <u>underline</u>.

Aggregation / Ranking Method	Mathematical Reasoning GSM8K	Mathematical Reasoning MATH-500	Question Answering ♫ MuSiQue-Ans	Spatial Reasoning SpaRP-S
	Acc. (↑)	Acc. (↑)	Acc. (↑) / F1 (↑)	F1 (↑)
Self-Consistency (SC)	74.91	23.40	19.74 / 25.18	34.37
pairwise-ORM	78.54	16.30	30.45 / 42.87	40.78
pairwise-ORM + SC	79.45	21.00	34.13 / 43.82	40.77
pointwise-ORM	79.76	20.20	33.43 / <u>45.42</u>	<u>49.79</u>
pointwise-ORM + SC	79.83	<u>23.80</u>	34.80 / 44.45	49.78
REPA-ORM	79.15	19.00	34.67 / 45.11	41.95
REPA-ORM + SC	79.22	20.60	**35.29** / 45.25	41.90
REPA-PRM	<u>79.98</u>	20.90	<u>34.84</u> / **45.52**	**50.08**
REPA-PRM + SC	**80.29**	**24.10**	32.11 / 40.43	46.92

Table 2: Performance evaluations of aggregators and RM verifiers on $N = 20$ sample output generations from Llama-3 8B SFT 1^{st} iteration. RM only entries indicate Best-of-N (BoN) sampling based results. Best values in **bold**, second best in <u>underline</u>. Mean of metrics reported on 3 groups of sampling results.

across all four datasets, outperforming multiple variants of ORMs as well as majority-voted Self-Consistency (SC). More specifically, REPA-PRM with Best-of-N (BoN) sampling leads the performance on the ♫ MuSiQue-Ans and SpaRP-S datasets, while REPA-PRM with SC leads the performance on the GSM8K and MATH datasets. In general, RMs with SC (i.e. aggregation) often leads to improved performance compared to RM with BoN sampling (i.e. ranking), except for a few cases (e.g. REPA-PRM) on the ♫ MuSiQue-Ans and SpaRP-S dataset where the aggregation harms the sufficiently powerful reward model. MATH-SHEPHARD (Wang et al., 2024a) observed similar trend with their RMs evaluated on GSM8K dataset. In the pairwise preference optimization setting, REPA-ORM with pairs identified and prioritized based on both final answers and aggregated reasoning scores, majorly outperforms its counterpart *pairwise*-ORM trained on pairs identified using only final answers. Both these showcase the effectiveness of our REPA framework for better reward modelling.

REPA Can Adapt Between Surface Form Precision and Flexibility. Since REPA evaluates multiple model outputs against a single ground-truth reasoning trace, there is a concern that it may overfit to specific surface-form patterns rather than capturing broader reasoning diversity. This concern is further reinforced by empirical observation of REPA-PRM with Best-of-N (BoN) sampling achieving the highest performance on datasets with limited surface form diversity, such as ♫ MuSiQue-Ans and SpaRP. Self-consistency (SC) underperforms when used with REPA-PRM on these datasets, whereas other reward models show comparable or improved performance. This suggests that REPA-PRM may struggle when evaluating outputs with varied reasoning expressions.

However, REPA's design mitigates this limitation through multi-step alignment and explicit explanations, allowing it to generalize beyond exact surface-form matches. In challenging datasets with more diverse reasoning structures such as MATH-500, REPA-PRM with aggregation-based methods (e.g., self-consistency) outperform both Best-of-N (BoN) sampling as well as other reward models.

Performance Scaling with the Number of Candidate Solutions. Table 3 shows consistent performance improvements across three strategies—self-

Agg./Rank.	N=4	N=16	N=64	N=128	N=256
SC	11.36	17.08	17.83	18.20	18.16
point.-ORM	**17.62**	**23.14**	24.00	25.64	25.22
REPA-PRM	16.88	22.34	**24.14**	**25.98**	**25.36**

Table 3: Accuracy of Llama-3-8B verifiers using BoN sampling across different no. of candidate solutions (N) on MATH-500 test-set. Mean of accuracy reported on 10 groups of sampling results.

consistency, *pointwise* ORM, and REPA-PRM—as the number of candidates increases from 4 to 256 on the MATH-500 test set. Notably, REPA-PRM emerges as the best performing strategy as the number of generations increases. However, all strategies peak at N=128 generations, with a slight performance decline afterwards.

Pointwise vs Pairwise RMs. While *pairwise*-loss RM training is generally considered more effective than *pointwise*-loss RMs (Liu et al., 2025), empirical evidence remains divided. For instance, even accounting for differences in annotation guidelines and human expectations, Liu et al.(2025) found pairwise RM training superior, whereas Wang et al.(2024b) reported better results with pointwise RM training. Our study adds to this debate with empirical evidence showing *pointwise*-ORM outperforming *pairwise*-ORM, significantly so on the MATH and SpaRP datasets. Both models are trained on a balanced set of positive and negative instances based on final answer outcomes, with *pairwise*-ORM forming pairs in reference to given contexts. Furthermore, REPA-ORM also underperforms *pointwise*-ORM, despite incorporating *superior* pairs selected based on both outcome and mean aggregated reasoning scores.

REPA-PRM Is Consistent Within Model Family. Table 4 reports performance on the Math-500 test set using two model sizes (3B and 32B) from the Qwen-2.5 family as candidate solution generators. We also evaluate verifiers from two different families, Llama-3 8B and Qwen-2.5 3B. REPA-PRM consistently outperforms self-consistency and ORM when applied to the same base model. Verifier effectiveness is model-family dependent—a verifier trained within a model family tends to perform better on generations from that family. For example, Llama-3 8B PRM underperforms compared to Qwen-2.5 3B ORM when evaluating generations from Qwen-2.5 32B. This discrepancy may stem from differences in model output distribution across families, suggesting that reward models generalize best within their own model lineage. These findings highlight the robustness of REPA-PRM within the same model family while also emphasizing the challenges of applying verifiers across different architectures.

Generator	Verifier	Agg./Rank.	Math–500
Qwen-2.5-3B	–	SC	31.4
	Llama-3-8B	REPA-PRM	32.2
		REPA-PRM + SC	34.2
	Qwen-2.5-3B	point.-ORM	33.4
		point.-ORM + SC	33.8
		REPA-PRM	32.2
		REPA-PRM + SC	**34.6**
Qwen-2.5-32B	–	SC	64.6
	Llama-3-8B	REPA-PRM	55.0
		REPA-PRM + SC	65.4
	Qwen-2.5-3B	point.-ORM	57.6
		point.-ORM + SC	65.6
		REPA-PRM	58.6
		REPA-PRM + SC	**66.0**

Table 4: Accuracy of REPA-PRMs across model families and sizes on MATH-500 test-set on 20 generations. Mean accuracy reported on 3 groups of sampling results.

5 Conclusions

Achieving high-quality and efficient automatic process supervision is crucial for enhancing the complex multi-step reasoning abilities of Large Language Models (LLMs). To this end, we propose **R**eference-**E**valuated **P**rocess **A**nnotation (**REPA**), a structured framework that enables per-step annotation in a single stage by evaluating each solution step against one or multiple ground-truth reference steps with explicit reasoning. Our experimental results demonstrate that fine-tuning a base-instruct model and training a reward model with REPA lead to improved reasoning performance under both single greedy decoding and ranking/aggregation of multiple solutions. Furthermore, we observe consistent improvements across four datasets spanning mathematical reasoning, multi-hop compositional question answering, and spatial reasoning. We also show that *pointwise*-ORMs still outperform REPA-ORMs. These findings highlight the potential of reference-guided automatic process supervision as a promising approach for enhancing LLM reasoning capabilities.

Limitations

REPA and its associated models depend on the availability of complete ground-truth reasoning chains to perform reference-guided step evaluations. While this reliance is a key limitation, we note that the ground-truth reasoning traces required by REPA are the same as those commonly used in Supervised Finetuning (SFT)—a foundational step in most finetuning methodologies. Thus, our approach does not introduce an additional annotation burden beyond what is typically required for training strong instruction-following models.

We also note that REPA, like other LLM-based automatic processes, is susceptible to some degree of noise. Nevertheless, we find that the structured, reference-evaluated step annotations it provides are effective for training both Process Reward Models (PRMs) and base-instruct models, leading to improved reasoning performance. Despite its limitations, REPA remains a practical and impactful approach for process supervision and model enhancement.

References

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

Zhen Bi, Ningyu Zhang, Yinuo Jiang, Shumin Deng, Guozhou Zheng, and Huajun Chen. 2024. When do program-of-thought works for reasoning? In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, and Akhil Mathur et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. ORPO: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189, Miami, Florida, USA. Association for Computational Linguistics.

Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. 2023. GRACE: Discriminator-guided chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15299–15328, Singapore. Association for Computational Linguistics.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353, Miami, Florida, USA. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333, Toronto, Canada. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Wenyu Qin, Menghan Xia, Xintao Wang, Xiaohong Liu, Fei Yang, Pengfei Wan, Di Zhang, Kun Gai, Yujiu Yang, and Wanli Ouyang. 2025. Improving video generation with human feedback. *Preprint*, arXiv:2501.13918.

Jianqiao Lu, Zhiyang Dou, Hongru WANG, Zeyu Cao, Jianbo Dai, Yunlong Feng, and Zhijiang Guo. 2024. AutoPSV: Automated process-supervised verifier. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. 2025. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *Preprint*, arXiv:2308.09583.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. 2024. Improve mathematical reasoning in language models by automated process supervision. *Preprint*, arXiv:2406.06592.

Roshanak Mirzaee and Parisa Kordjamshidi. 2022. Transfer learning with synthetic corpora for spatial role labeling and reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6148–6165, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sarah Pan, Vladislav Lialin, Sherin Muckatira, and Anna Rumshisky. 2023. Let's reinforce step by step. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Md Imbesat Rizvi, Xiaodan Zhu, and Iryna Gurevych. 2024. SpaRC and SpaRP: Spatial reasoning characterization and path generation for understanding spatial reasoning capability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4750–4767, Bangkok, Thailand. Association for Computational Linguistics.

Zhengxiang Shi, Qiang Zhang, and Aldo Lipani. 2022. Stepgame: A new benchmark for robust multi-hop spatial reasoning in texts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11321–11329.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. ♫ MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *Preprint*, arXiv:2211.14275.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024a. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Zhilin Wang, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. 2024b. Helpsteer2-preference: Complementing ratings with preferences. *Preprint*, arXiv:2410.01257.

Zihan Wang, Yunxuan Li, Yuexin Wu, Liangchen Luo, Le Hou, Hongkun Yu, and Jingbo Shang. 2024c. Multi-step problem solving through a verifier: An empirical analysis on model-induced process supervision. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7309–7319, Miami, Florida, USA. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. In *Advances in Neural Information Processing Systems*.

Fei Yu, Anningzhe Gao, and Benyou Wang. 2024a. OVM, outcome-supervised value models for planning in mathematical reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 858–875, Mexico City, Mexico. Association for Computational Linguistics.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024b. Metamath: Bootstrap your own mathematical questions for large

language models. In *The Twelfth International Conference on Learning Representations*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. ReST-MCTS*: LLM self-training via process reward guided tree search. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A Evaluation Prompt

An example system prompt with evaluation guidelines for MuSiQue dataset is provided in Table 5. Similarly, other datasets have their own domain and subject specific evaluation heuristics included in the System Prompt.

B Details of Finetuning

We used the Huggingface's TRL library and QLoRA for parameter efficient finetuning of various ORPO models with the values of parameters shown in Table 6.

C Details of Reward Model (RM) Training

We used $\sim 3\times \approx$ 20K of SFT training dataset size for mathematical datasets. This is still significantly smaller than other work (Lightman et al., 2024; Wang et al., 2024a) that are usually trained on $>$ 150K solutions. The number N of individual positive and negative samples (i.e. N pairs) for Reward Model training are presented in Table 7.

Role	Content
System	You are a teacher grading a student's assignment. You are given a ground-truth correct REFERENCE ANSWER and a STUDENT'S ANSWER. You are asked to match the steps of STUDENT'S ANSWER with that in the REFERENCE ANSWER. You are required to score the steps of STUDENT'S ANSWER as below: A step in the STUDENT'S ANSWER that matches or is factually consistent with one or more steps in the REFERENCE ANSWER is labelled as CORRECT. While matching steps, evaluate on the basis of: (a) whether the document name matches or not (b) whether the entities present in the steps match or not (c) whether the numbers mentioned match or not, and finally (d) whether the semantic relation between all these match or not. A step in the STUDENT'S ANSWER that doesn't match or is factually incorrect with respect to the provided REFERENCE ANSWER is labelled as INCORRECT. Thus a step that may be factually correct but is not matched to one or more steps in the REFERENCE ANSWER is also to be marked as INCORRECT. You need to evaluate ALL the steps of the STUDENT'S ANSWER. Provide your evaluation ONLY and ONLY in JSON format as a list of dictionaries whose keys and their intended purpose are: "student_step": The current step number of the STUDENT'S ANSWER. "reasoning": The reasoning expanding upon why or what part of the current `student_step` of the STUDENT'S ANSWER, either DIRECTLY and ENTIRELY in itself or probably in combination with other steps in the STUDENT'S ANSWER, is correct or incorrect in reference to one or more REFERENCE ANSWER steps. "student_combining_steps": A list of previous `student_step` that when combined with the current `student_step` will be part or whole of one or more steps in the REFERENCE ANSWER. Leave it as an empty list if the current `student_step` DIRECTLY and ENTIRELY matches with one or more steps in the REFERENCE ANSWER. If the number of steps in the STUDENT'S ANSWER is more than that in the REFERENCE ANSWER, then a single step in REFERENCE ANSWER can correspond to multiple steps in the STUDENT'S ANSWER and this list will be non-empty for some of the `student_step`. "matching_reference_steps": A list of steps in the REFERENCE ANSWER based on which the correctness or the incorrectness of the current `student_step` is reasoned and arrived at. If the number of steps in the STUDENT'S ANSWER is less than that in the REFERENCE ANSWER, then multiple steps in the REFERENCE ANSWER can correspond to a single step in the STUDENT'S ANSWER. "error_category": A list of type of errors from "DOCUMENT NAME", "ENTITY NAME", "NUMERIC", "INTENDED CATEGORY", "SEMANTIC RELATION" and "NO STEP MATCH" that caused the current `student_step` to be partially or fully incorrect. Leave it as an empty list if the current `student_step` is completely correct. "label": binary score of the current `student_step` as either CORRECT or INCORRECT.

Table 5: An example system prompt with evaluation guidelines for MuSiQue dataset. Similarly, other datasets have their own domain and subject specific evaluation heuristics included in the System Prompt.

Parameter Name	Value
QLoRA:	
α	16
Dropout	0.1
r	64
bias	None
task_type	CAUSAL_LM
Training Arguments:	
Effective Batch Size	32
lr	$1.0e-4$
weight decay	0.001
max_grad_norm	0.3
warm up ratio	0.03
lr_scheduler	cosine

Table 6: Values of the parameters and hyperparameters used while ORPO finetuning.

Dataset	N
GSM8K	20175
MATH	20,250
MuSiQue	5,000
SpaRP	8,000

Table 7: Training data sizes for Reward Models