

Evaluating Mathematical Reasoning Chains

Tanay Dixit (tanayd2)* Neeraja Kirtane (kirtane3) Rishabh Adiga (radiga2)
Aakriti (aa117) Akshat Sharma (akshat7) Omkar Gurjar (ogurjar2)

Abstract

There has been a growing interest in teaching large language models to perform Chain of thought (CoT) reasoning to solve complex tasks (Chung et al., 2022; Wei et al., 2022; Wang et al., 2022), yet assessing the quality of these step-by-step reasoning chains remains a challenge. Current approaches tend to focus on the correctness of the final answer, potentially overlooking the intricate facets of genuine reasoning. This also raises issues for evaluation when the final answer is not available. Existing methods for CoT evaluation either perform poorly on math-based tasks (Prasad et al., 2023) or fail to measure the logical correctness of steps effectively since they only check math calculations (Chern et al., 2023). To bridge this gap, we propose MARoC (Mathematical Reasoning Chain Evaluator), a set of metrics particularly trained to evaluate math-based reasoning chains. We make use of large-scale MathInstruct dataset (Yue et al., 2023a) and generate incorrect reasoning chains in order to train the metrics using Contrastive and Direct Preference Optimization techniques. By specifically training MARoC to differentiate between correct and incorrect solutions, MARoC learns the nuances of mathematical reasoning chains and learns to identify the errors better. Analysis on the ROSCOE benchmark shows that the majority of the MARoC variants show stronger correlations with human scores than prior metrics, with MARoC-SimCSE-SS performing the best with an 8% gain on average.

*

1 Introduction

The Chain of Thought (CoT) prompting (Wei et al., 2022) involves the systematic progression of ideas, establishing a logical sequence for problem-solving and concept exploration. CoT facilitates the structured breakdown of complex problems, revealing

insights into underlying sub-problems. It enhances the explainability of Language Models (LLMs), thereby improving their performance in various tasks by encouraging transparent reasoning pathways. However, evaluating the accuracy of these reasoning steps is challenging due to the lack of reliable automatic evaluation methods or gold reference chains (Prasad et al., 2023). Previous works have mainly concentrated on evaluating the CoT based on the model’s downstream performance for the given task. Focusing only on the final answer of the reasoning chain may mix up the quality of reasoning with unrelated shortcuts or spurious steps used to arrive at the answer. This also raises issues for evaluation when the final answer is not available.

In a mathematical setting, correct answers heavily rely on intermediate steps, distinguishing it from other natural language tasks (Zhang et al., 2023). Doing a step-wise assessment facilitates the identification and correction of model errors. However, few studies have addressed this critical aspect. Existing metrics that use generic entailment models tend to perform poorly on numeric-based inputs (Prasad et al., 2023). Assessing the correctness of only the arithmetic operations in each step (Chern et al., 2023) is not enough, as LLMs may make errors beyond numerical calculations (Golovneva et al., 2022a). Since math-based tasks contain several such nuances, they warrant evaluation metrics that have been solely optimized to evaluate mathematical reasoning chains.

To address this, we introduce MARoC, a series of metrics fine-tuned with different techniques with the aim of evaluating mathematical reasoning chains. In order to leverage contrastive and Direct Preference Optimization (DPO) fine-tuning techniques, we generate datasets consisting of both correct and incorrect reasoning chains. As a first step, we generate several incorrect reasoning chains

* Team Leader

*Code available at <https://github.com/neerajal504/Maroc>

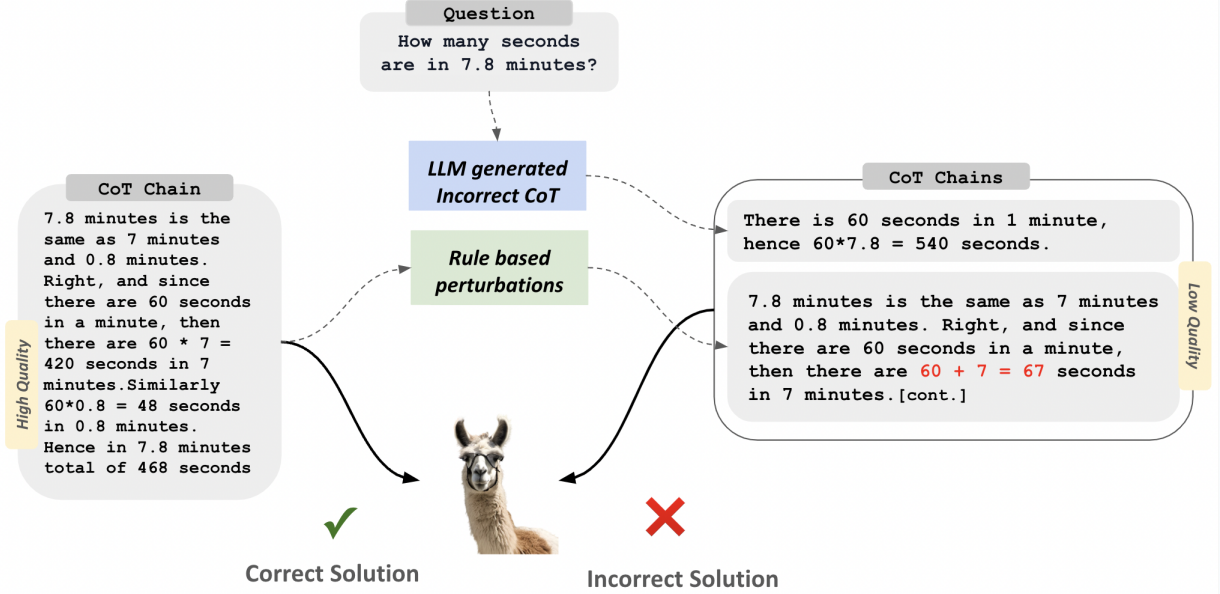


Figure 1: Overview of MARoC metric data generation and training process. Every given question in MathInstruct comes with its corresponding correct solution. In order to generate the incorrect solution, we adopt two techniques: (1) Incorrect LLM-generated chains and (2) Perturbed chains. Using these correct and incorrect solutions, we fine-tune MARoC with either DPO or contrastive learning in order to learn the nuances of mathematical reasoning.

for every question in the MathInstruct dataset (Yue et al., 2023b) using different techniques (§3.2). Next, we train multiple variants of MARoC, namely MARoC-DPO and MARoC-SimCSE using Direct Preference Optimization and contrastive learning techniques respectively. We evaluate these metrics on the ROSCOE benchmark (Golovneva et al., 2022a) and observe stronger correlations with human scores as compared to competitive baselines. In summary, we make the following contributions -

- We introduce MARoC, a novel set of metrics trained using Direct Preference Optimization and Contrastive Loss techniques.
- We generate a diverse set of incorrect reasoning chains for every question in the MathInstruct dataset.
- We demonstrate the superior performance of MARoC by showcasing stronger correlations with human scores on the ROSCOE benchmark

2 Related Work

2.1 Chain of Thought Prompting

Inspired by the human thought processing of breaking down complex reasoning problems into multiple intermediate steps, (Wei et al., 2022) shows

that prompting large language models with a few demonstrations can unlock the reasoning capabilities of LLMs. They show improvements in problem-solving performance on various math problem benchmarks (Cobbe et al., 2021; Hendrycks et al., 2021; Lightman et al., 2023; Yue et al., 2023b). Other works, such as Kojima et al. (2022) propose a zero-shot approach of prompting LLMs to solve complex reasoning tasks such as Arithmetic, Common Sense, and Symbolic reasoning. Combining the methodologies of the above two works (Zhang et al., 2022) eliminates the need for manually crafted exemplar CoT demonstrations by generating automatic demonstrations leveraging (Kojima et al., 2022)’s single prompting technique.

2.2 Evaluation of CoT

With more and more works exploring LLMs CoT capabilities, recent research has also started focusing on evaluating the quality of the generated CoT sequences. (Lightman et al., 2023) provide the first large-scale dataset for step-wise CoT evaluation (PRM800K), in which they rate GPT-4 generated CoT as positive/negative or neutral. (Golovneva et al., 2022a) proposes a suite of metrics (ROSCOE) for evaluating the reasoning steps on several dimensions, including semantic alignment (factuality, hallucination, etc.), semantic similarity, logical (self

and source) consistency, and language coherence. Additionally they lay the foundations of the developing evaluation metrics for CoT by contributing a benchmark dataset of human annotations. Moving towards reference-free metrics, (Prasad et al., 2023) propose an evaluation technique RECEVAL based on the correctness and informativeness of the generated CoT. Our work is along these lines but particularly focuses on math-based tasks.

2.3 Meta Evaluation

Evaluating the effectiveness of evaluation metrics is extremely challenging, with several possible techniques being proposed (Sai et al., 2021; Fabbri et al., 2021). The widely adopted approach is measuring the correlation of metric scores with human scores. Most of the existing NLG metrics show strong correlations on evaluating standard NLP tasks but extremely poor correlation on evaluating CoT (Golovneva et al., 2022b). The only existing benchmark for evaluating evaluation metrics is the one provided by Golovneva et al. (2022b). We report our results on the same.

3 MARoC: Mathematical Reasoning Chain Evaluator

Problem Formulation Our goal is to score a step-by-step reasoning chain generated by a language model. We assume that the model is given a *question* q and is prompted to generate step-by-step reasoning. We refer to this as a *hypothesis* $h = \{h_1, \dots, h_N\}$ of N -steps, including a final answer as the last step. We do not assume the availability of gold step-by-step reasoning *references* $r = \{r_1, \dots, r_K\}$ of K -steps as these are not available for majority datasets (Prasad et al., 2023). We train a metric M that learns to give a score s_i to h_i given q and $h_{<i}$. Formally

$$s_i = M(h_i|q, h_{<i}) \quad (1)$$

s_i corresponds to the step-level score, but we can combine the step-level scores in order to determine the overall quality of a reasoning chain. Following the scoring setup in (Golovneva et al., 2022a), we consider a reasoning chain to be only as good as its least correct step. Therefore, given a reasoning chain and an evaluation metric, we aggregate step-level scores using a \min operation, $s = \min(s_i)$

3.1 Training

Given a dataset $D : \{X, Y_+\}$, where X is a mathematical question, and Y_+ 's are the corresponding correct solutions. In order to fine-tune a metric to reward correct solutions and penalize incorrect ones, we need access to Y_- , which are the dispreferred or incorrect solutions. Since Y_- is not available as part of any dataset, it needs to be generated and be of high quality (hard negatives) (Karpukhin et al., 2020). As a results our training dataset corresponds to $D : \{X, Y_+, Y_-^1, Y_-^2, \dots\}$. The rest of this section elaborates on the training processes while §3.2 talks about the Y_- generation process.

Direct Preference Optimization Reinforcement learning (RL) has proven to be an effective approach to fine-tuning language models to extract complex, useful behaviors from their pre-trained knowledge (Ouyang et al., 2022). In the context of RL, a language model policy π_θ produces a conditional distribution $\pi_\theta(y | x)$ over responses y given an input query x . The goal of reinforcement learning is to maximize the average reward of outputs generated by the policy, where a reward function $r(x, y)$ assigns a scalar score to an input-output pair. However, past works have observed that fine-tuning language models with an objective of unconstrained reward maximization can lead to *overoptimization* (Gao et al., 2022). The most commonly used objective in practice, therefore, combines reward maximization with a KL-divergence penalty between the language model and its initialization:

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}_P, y \sim \pi_\theta(y|x)} [r(x, y) - \beta \log \frac{\pi_\theta(y | x)}{\pi_{\text{ref}}(y | x)}] \quad (2)$$

Most large language models fine-tuned with Equation 2 optimize a reward function that is *learned* from a dataset of preference rankings over possible model outputs. DPO enables straightforward fine-tuning of language models on objectives other than supervised imitation, using a preference ranking over possible model responses. The DPO algorithm simplifies RL on language models for this special case (Rafailov et al., 2023), using a dataset of preference pairs $\mathcal{D} = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$ of prompts x and candidate responses y_w and y_l , where y_w is preferred over y_l (denoted $y_w > y_l$).

The probability of observing a particular preference pair is assumed to follow a Bradley-Terry model:

$$p(y_w > y_l) = f(r(x, y_w) - r(x, y_l)) \quad (3)$$

where $f()$ is either a sigmoid or hinge loss function and $r(x, y)$ is an unobserved reward or scoring function. Rafailov et al. (2023) show that the optimal policy π^* for the problem in Eq. 2 can be found by optimizing a simple classification loss computed directly on the preference data:

$$\begin{aligned} L_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) \\ = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w | x)}{\pi_{\text{ref}}(y_w | x)} \right. \right. \\ \left. \left. - \beta \log \frac{\pi_\theta(y_l | x)}{\pi_{\text{ref}}(y_l | x)} \right) \right] \end{aligned} \quad (4)$$

DPO enables learning π_θ from a fixed dataset of preferences without fitting an explicit reward function. These advantages make DPO an attractive choice for fine-tuning language models. However, a challenge remains in constructing preference pairs. Recent works Tian et al. (2023); Mishra et al. (2023) show the effectiveness of using Direct Preference Optimization (DPO) with synthetically constructed preference datasets. Following these lines of work, we use generated data as a proxy for human preferences.

Inference: In order to evaluate a reasoning chain h for a question q , we use the conditional log probability to obtain a score (§5). We chose this particular scoring mechanism because it aligns with the reward function while training. Note this scoring technique allows for both step-level scores and chain-level scores, depending on the value of N .

$$\text{MARoC-DPO} = \sum_{t=1}^N \log p(h_t | h_{<t}, q, \theta) \quad (5)$$

Contrastive Learning aims to learn good representation by pulling semantically close neighbors together (question and correct reasoning chain) and pushing apart non-neighbors (question and incorrect reasoning chain). In our setting, we minimize the following objective

$$-\log \frac{e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^N \left(e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j^+)/\tau} + e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j^-)/\tau} \right)}. \quad (6)$$

where $\mathbf{h}_i, \mathbf{h}_i^+, \mathbf{h}_{i,j}^-$ are encoded reasoning chain embeddings corresponding to X, Y_+, Y_- , τ is a temperature hyperparameter and $\text{sim}(\cdot, \cdot)$ is the similarity metric between two embeddings.

Having generated the reasoning chain embeddings, following Golovneva et al. (2022a) we present three different variants - *semantic alignment (SA)*, *semantic similarity (SS)*, *logical inference (LI)*.

Semantic Alignment (SA): Semantic alignment metrics involve the derivation of the reasoning alignment vector from the hypothesis (h) to the source (s). We determine the alignment vector, denoted as $\text{r-align}(h \Rightarrow s)$, through the comparison of the source text and reasoning chains based on the embeddings of tokens and individual steps in the reasoning process.

Semantic Similarity (SS): Semantic similarity metrics measure the degree of semantic equivalence between the hypothesis chain and the question.

Logical Inference (LI): Logical inference metrics measure logical errors between pieces of text. We use an NLI model that was trained to classify hypothesis-context pairs into entailment, neutral, and contradiction classes to infer the contradiction probability.

3.2 Constructing the dataset

Both the training approaches mentioned required a correct solution and a wrong solution for every given question. The majority of the datasets provide correct solutions; hence the task remains the generate the negative data (wrong solutions), which is the most critical for effective learning (Karpukhin et al., 2020). We explore two techniques in order to get the best coverage of the possible and observed error types.

Perturbation based negatives Designing perturbations to generate incorrect instances has been a long-standing practice (Sai et al., 2021; Golovneva et al., 2022a). Similar to Golovneva et al. (2022a), we design 8 different perturbations that capture the possible errors models can depict while solving math-based problems. These are listed in Table 1. Each perturbation is based on a rule-based approach and guarantees that the perturbed solution would be incorrect. These perturbations are applied to the correct solution in order to generate the wrong solution. We ensure that each perturbation is equally represented in the final training dataset.

Perturbation Type	Error Type	Gold Reasoning Steps	Perturbed Reasoning Steps
Repeat a step	Repetition	$[r_1, r_2, r_3]$	$[r_1, r_2, r_2, r_3]$
Remove a step	Missing step	$[r_1, r_2, r_3]$	$[r_2, r_3]$
Shuffle steps	Self-coherency	$[r_1, r_2, r_3]$	$[r_3, r_1, r_2]$
Swap a step	Self-coherency	$[r_1, r_2, r_3]$	$[r_2, r_1, r_3]$
Negate a step	Factuality	$[r_1, r_2, r_3]$	$[r_1, \neg r_2, r_3]$
Hallucination	Hallucination	$[r_1, r_2, r_3]$	$[r_1, r_2, r_3, r_4]$
Grammar error	Grammatical	$[r_1, r_2, r_3]$	$[r_1, r_2, g^*(r_3)]$
Semantic change	Factuality	$[r_1, r_2, r_3]$	$[r_1, s^*(r_2), r_3]$

Table 1: Synthetic perturbations and corresponding error types of steps $\{r_i\}$ in reference chains used when constructing diagnostics datasets. $g^*(\cdot)$ represents grammar error, $s^*(\cdot)$ represents semantic change.

Error Type	Acronym	Definition
Grammar	QUAL	Faulty, unconventional, or controversial grammar usage
Factuality	FACT	Information about an object (i.e. quantity, characteristics) or a named entity doesn't match with the input.
Hallucination	HALL	Information is not provided in the problem statement and is irrelevant or wrong
Redundancy	RED	Explanation contains redundant information, even though might be factual, is not required to answer the question
Repetition	REP	Step paraphrases information already mentioned in previous reasoning steps
Missing step	MISS	The content of the reasoning is incomplete and lacks the required information to produce the correct answer.
Coherency	COH	Steps contradict each other or do not follow a cohesive story
Commonsense	COM	Model lacks relations that should be known from general world (e.g., "all ducks are birds")
Arithmetic	MATH	Error in math calculations

Table 2: Taxonomy of Step-by-Step Reasoning Errors (Golovneva et al., 2022a).

Sampling Incorrect solutions Generating perturbed solutions helps ensure coverage of diverse error types but does not guarantee coverage of *observed* error types. By only training using perturbed solutions, the model can face a distribution shift with the observed errors during inference. In order to avoid these issues, we also include incorrect model-generated solutions. We prompt LLMs like Llama-2² and Mistral-7B³ to generate solutions for every question in the dataset. We increase the temperature parameter while sampling in order to increase the possibility of generating wrong solutions. We use the final answer as a proxy for the correctness of the reasoning chain (this can lead to some false positives since a model can arrive at the correct final answer by performing incorrect reasoning, but we believe this percentage is small); this aligns with the process adopted by Lightman et al. (2023) for generating wrong solutions.

4 Experimental Setup

Meta Evaluation dataset The sole dataset available for meta-evaluating evaluation metrics in the context of reasoning chain assessment is ROSCOE

(Golovneva et al., 2022a). Comprising 200 annotated reasoning chains, the ROSCOE benchmark was created by prompting GPT-3.5 with questions from the GSM-8k test dataset. These annotations include evaluations for factuality issues (FACT), missing steps (MISS), hallucinations (HALL), redundant or irrelevant information (RED), unnecessary paraphrasing (REP), commonsense errors (COM), arithmetic errors (MATH), coherence (COH), and overall quality (QUAL). The definitions of each error type are outlined in Table 2. Each reasoning chain is assigned a binary label (correct or incorrect) across the nine specified dimensions. We use this dataset to evaluate our trained metrics.

Baselines We compare our approach against the metrics proposed by ROSCOE. For ROSCOE, we compare against semantic similarity (SS), semantic alignment (SA), and logical inference (LI) metrics. For ROSCOE-SA and -SS, we use the text-similarity models finetuned on reasoning chains⁴. In addition to this, we add competitive baselines that use Mistral, ChatGPT, and GPT-4 to evaluate the correctness of reasoning steps. We prompt these models to evaluate the correctness of every step given the question and previous steps. Prompt

²<https://huggingface.co/meta-llama/Llama-2-7b-hf>

³<https://huggingface.co/mistralai/Mistral-7B-v0.1>

⁴We use facebook/roscoe-512-roberta-base models for computing chain embeddings

Metric	QUAL	COH	COM	FACT	HALL	RED	REP	MISS	MATH
ROSCOE-SA	0.20	0.19	0.19	0.08	0.22	0.39	0.79	0.27	0.44
-SS	0.20	0.17	0.17	0.14	0.25	0.51	0.87	0.22	0.23
-LI	0.28	0.26	0.18	0.34	0.22	0.35	0.98	0.22	0.09
Mistral	0.12	0.13	0.15	0.20	-0.06	-0.11	-0.19	0.09	-0.04
MaRoC - DPO	<u>0.19</u>	<u>0.14</u>	<u>0.21</u>	<u>0.23</u>	<u>0.13</u>	<u>0.05</u>	<u>-0.16</u>	0.09	<u>0.12</u>
MaRoC-SimCSE-SA	<u>0.21</u>	<u>0.21</u>	<u>0.21</u>	<u>0.10</u>	0.22	<u>0.44</u>	0.79	<u>0.29</u>	0.44
-SS	<u>0.29</u>	<u>0.25</u>	<u>0.20</u>	<u>0.3</u>	<u>0.26</u>	0.49	0.26	<u>0.30</u>	0.63
-LI	0.28	<u>0.27</u>	0.18	<u>0.35</u>	0.21	0.34	0.98	0.22	0.09
ChatGPT	-	0.21	0.17	0.32	0.44	0.41	0.77	-	0.27
GPT-4	0.57	0.47	0.51	0.51	0.61	0.33	0.82	0.66	0.27

Table 3: We report the Somer’s-D correlations for the various metrics. ROSCOE represents the results for the baseline ROSCOE model. MaRoC - DPO and MaRoC-SimCSE are the results for the contrastive and DPO versions of our model. The shaded rows represent upper bound achieved using API based GPT models. SOTA values are bolded and values that outperform the baseline are underlined. ($p < 0.05$)

provided in Tab. 4.

Correlation Measure After scoring each reasoning chain with either metric, we assess whether the score can indicate the presence or absence of each type of error. Following Golovneva et al. (2022a), we use the Somer’s-D correlation (Somers, 1962), which measures the ordinal association between two dependent quantities. In our case, we evaluate a metric M against the random variable indicating whether the chain is erroneous ($E \in \{0, 1\}$). Using Kendall’s τ coefficient, Somer’s-D correlation is defined as:

$$D_{SE} = \tau(E, S) / \tau(E, E).$$

MaRoC Setup For our training data, we use the MathInstruct dataset (Yue et al., 2023a). The MathInstruct dataset is a combination of several datasets, namely GSM8k (Cobbe et al., 2021), AQuA (Ling et al., 2017), MATH (Hendrycks et al., 2021), and MathQA (Amini et al., 2019). Every question in this dataset consists of a gold reference solution. We make use of the negative data generation process mentioned in §3.2. In all our training dataset consists of 79k tuples of question, correct, wrong solution.

For the **MaRoC-DPO** metric, we fine-tune the Mistral-7B model on our training dataset. We use a hinge loss instead of a sigmoid loss in the DPO loss equation, as we observed higher correlations with it. We fine-tuned the model for 1 epoch which was the equivalent to 3.5k steps. For scoring the reasoning chains using a fine-tuned Mistral-7B

model, we use Equation 5. For **MaRoC-SimCSE** metric, we fine-tune the RoBERTa-base model on our training dataset. We reuse the training implementation of the authors⁵. We train the model using the supervised-SimCSE method using the loss described in Equation 6 for 5 epochs with a learning rate of 10^{-5} . Next, we use the fine-tuned SimCSE model to generate embeddings for our reasoning steps, which are then used to calculate the SA(semantic alignment), SS(semantic similarity), and LI(logical inference) scores defined in §3.1. We follow the same implementation procedure as adopted by Golovneva et al. (2022a).

5 Results

Table 3 contains the results of various approaches and baselines. The first three rows are the ROSCOE baseline, while the next five are the various MaRoC versions. The last two rows in grey are the closed-sourced models. We can observe that MaRoC outperforms the Mistral baseline across all except the MISS criteria. Maximum gains observed along HALL and MATH criteria. This implies that models fine-tuned using DPO help make the models better at evaluating reasoning chains. Also, a similar story holds for MaRoC-SimCSE and its variants. We see consistent gains across all evaluation dimensions except RED and REP. Additionally, MaRoC-SimCSE-SA achieves the highest correlation on MATH based errors, outperforming ChatGPT and GPT-4. We can observe that the MaRoC-SimCSE-SA performs the best, with

⁵<https://github.com/princeton-nlp/SimCSE>

an 8% gain on average across all dimensions as compared to ROSCOE-SA.

6 Conclusion

We present MaRoC, a novel metric for evaluating math-based reasoning chains. By utilizing Direct Preference Optimization and Contrastive Loss techniques for fine-tuning, MaRoC outperforms existing metrics on the ROSCOE benchmark. Additionally, we contribute a new dataset consisting of diverse error types, which future works could use to develop robust evaluation metrics for math-based tasks. Overall, our work addresses the critical need for robust evaluation in the realm of step-by-step reasoning, providing a valuable tool for assessing the correctness of mathematical reasoning chains generated by large language models.

7 Limitations

We highlight a few limitations of our work and suggest possible approaches to fix them.

Unable to capture numeric based errors: On analyzing the failure cases of MaRoC we observe that it is unable to accurately capture numeric calculations-based errors. A simple way to overcome this issue is to couple the MaRoC score with FactTool (Chern et al., 2023). FactTool only evaluates numeric consistency, and the two metrics would perfectly complement each other.

ROSCOE Benchmark: ROSCOE benchmark is generated by sampling reasoning chains from GPT-3.5 and annotating them. The absence of a diverse benchmark dataset for Chain of Thought (CoT) chains generated by open-source Language Models (LLMs) is a notable limitation, as the varied distributions and error patterns among different LLMs may lead to biased conclusions when relying solely on chains from a specific model (Peyrard, 2019).

8 Acknowledgments

We would like to thank Yi Fung, Prof. Hao Peng, and Prof. Heng Ji for the insightful discussions and feedback on our work.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. [MathQA: Towards interpretable math word](#)

[problem solving with operation-based formalisms](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. 2023. Factool: Factuality detection in generative ai—a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

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

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Leo Gao, John Schulman, and Jacob Hilton. 2022. [Scaling laws for reward model overoptimization](#).

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2022a. Roscoe: A suite of metrics for scoring step-by-step reasoning. *arXiv preprint arXiv:2212.07919*.

Olga Golovneva, Pan Wei, Khadige Abboud, Charith Peris, Lizhen Tan, and Haiyang Yu. 2022b. [Task-driven augmented data evaluation](#). In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 18–25, Abu Dhabi, United Arab Emirates (Hybrid). 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. *arXiv preprint arXiv:2103.03874*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

- 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. *arXiv preprint arXiv:2305.20050*.
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.
- Prakamya Mishra, Zonghai Yao, Shuwei Chen, Beining Wang, Rohan Mittal, and Hong Yu. 2023. Synthetic imitation edit feedback for factual alignment in clinical summarization. *arXiv preprint arXiv:2310.20033*.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.
- Maxime Peyrard. 2019. Studying summarization evaluation metrics in the appropriate scoring range. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5093–5100.
- Archiki Prasad, Swarnadeep Saha, Xiang Zhou, and Mohit Bansal. 2023. [Receval: Evaluating reasoning chains via correctness and informativeness](#).
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.
- Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. [Perturbation CheckLists for evaluating NLG evaluation metrics](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7219–7234, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Robert H Somers. 1962. [A new asymmetric measure of association for ordinal variables](#). *American sociological review*, pages 799–811.
- Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. 2023. Fine-tuning language models for factuality. *arXiv preprint arXiv:2311.08401*.
- Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krithika Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. [Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.
- Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhui Chen. 2023a. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.
- Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhui Chen. 2023b. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.
- Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A Smith. 2023. How language model hallucinations can snowball. *arXiv preprint arXiv:2305.13534*.
- Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

A Appendix

Table 5 is the prompt used for generating the incorrect step-by-step chains, and Table 4 shows the prompts given to ChatGPT and GPT-4 for evaluation of the steps in a CoT.

You are given a math problem and a step-by-step partial solution to the problem. The given solution could be correct or incorrect. Identify whether the next step to the given solution will lead to the correct final answer or not. If the next step is correct, you should generate "correct". If it is incorrect, you should generate "incorrect". I will give you a few examples to get you started.

Question: Siobhan has 2 fewer jewels than Aaron. Aaron has 5 more jewels than half of Raymond's jewels. If Raymond has 40 jewels, how many jewels does Siobhan have?

Solution: Aaron has 5 more jewels than half of Raymond's jewels, meaning he has $40 + 5 = 45$ jewels. Next Step: Siobhan has 2 fewer jewels than Aaron, meaning she has $45 - 2 = 43$ jewels.

Output: correct

Question: A teacher teaches 5 periods a day and works 24 days a month. He is paid \$5 per period. If he has been working for 6 months now, how much has he earned in total?

Solution: The amount paid to the teacher per day is 5 periods * \$5/period = \$25 per day. The amount paid for 24 days is \$25/day * 24 days = \$600.

Next Step: The total amount for 6 months is $\$600 * 6 = \1800 .

Output: incorrect

Question: Brandon's iPhone is four times as old as Ben's iPhone. Ben's iPhone is two times older than Suzy's iPhone. If Suzy's iPhone is 1 year old, how old is Brandon's iPhone?

Solution:

Next Step: Ben's iPhone is $2 * 1$ year = 2 years older than Suzy's iPhone. Output: correct

Question: Brandon's iPhone is four times as old as Ben's iPhone. Ben's iPhone is two times older than Suzy's iPhone. If Suzy's iPhone is 1 year old, how old is Brandon's iPhone?

Solution: Ben's iPhone is $2 * 1$ year = 2 years older than Suzy's iPhone.

Next Step: Thus, Brandon's iPhone is $2 + 4$ years = 6 years old.

Output: incorrect

Question: Wynter went to her local town bike shop to buy her sister a bicycle as her birthday gift. While at the shop, Wynter counted 50 bicycles and 20 tricycles. How many wheels in total did the vehicles she saw have?

Solution: There are 50 bicycles at the shop. Each bicycle has 2 wheels. So, there are $50 * 2 = 100$ wheels. There are 20 tricycles at the shop. Each tricycle has 3 wheels. So, there are $20 * 3 = 60$ wheels.

Next Step: The total number of wheels is $100 + 60 = 160$.

Output: correct

Question: Elsa and her sister watch an opera show every year at Central City Opera. Last year, opera seating cost \$85 per ticket. This year, it costs \$102. What is the percent increase in the cost of each ticket? Solution: Last year, the opera seating cost \$85 per ticket (initial cost) and this year, it costs \$102 per ticket (new cost). Hence the increase in cost, is nothing but new cost - initial cost. Increase in cost = $\$102 - \$85 = 17$. Next Step: Percent Increase = $(\$17 / \$85) * 100 = 20\%$

Output: correct

Question: Two runners are competing in a 10-mile race. The first runs at an average pace of 8 minutes per mile, while the second runs at an average pace of 7 minutes per mile. After 56 minutes, the second runner stops for a drink of water. For how many minutes could the second runner remain stopped before the first runner catches up with him?

Solution: The first runner runs at an average pace of 8 minutes per mile. So, after 56 minutes, the first runner has covered. Distance of the first runner = $(56 \text{ minutes}) / (8 \text{ minutes/mile}) = 7$ miles.

Next Step: The second runner runs at an average pace of 7 minutes per mile. After 56 minutes, the second runner has covered: Distance of the second runner = $(56 * 7) = 392$ miles.

Output: incorrect

Table 4: Prompt used for using ChatGPT and GPT-4 as evaluation metrics.

Given a math word problem, solve it while showing your work.

Question: The Dark Lord needs to transport 1200 pounds of swords to the Enchanted Forest for a battle with the elves. He has 10 squads of 8 orcs each. How many pounds of swords does each orc have to carry?

Lets think step by step

Solution: The Dark Lord has 10 squads of 8 orcs each. To find the total number of orcs, we'll multiply the number of squads by the number of orcs in each squad. Hence total orcs = 10 squads * 8 orcs/squad = 80 orcs. To find out how many pounds of swords each orc has to carry, we'll divide the total pounds of swords by the total number of orcs: Pounds per orc = Total pounds of swords / Total orcs = 1200 / 80 = 15 pounds

Answer: 15

Question: Elsa and her sister watch an opera show every year at Central City Opera. Last year, opera seating cost \$85 per ticket. This year, it costs \$102. What is the percent increase in the cost of each ticket?

Lets think step by step

Solution: Last year, the opera seating cost \$85 per ticket (initial cost) and this year, it costs \$102 per ticket (new cost). Hence the increase in cost, is nothing but new cost - initial cost. Increase in cost = \$102 - \$85 = 17. To find the percent increase, divide the increase in cost by the initial cost and then multiply by 100 to express it as a percentage. Percent Increase = $(17/85) * 100 = 20\%$

Answer: 20

Question: John and his best friend Steve bought 12 cupcakes together. Each cupcake cost 1.50. If they split the costs evenly, how much did each person pay?

Lets think step by step

Solution: Each cupcake costs \$1.50. To find the total cost, we'll multiply the cost per cupcake by the number of cupcakes. Hence $\$1.50/\text{cupcake} \times 12 \text{ cupcakes} = \18 . Since John and Steve bought the cupcakes together, they need to split the cost evenly. To do this, we'll divide the total cost by 2 (since there are two people). Each person's share = $\$18 / 2 = \9

Answer: 9

Question: Two runners are competing in a 10-mile race. The first runs at an average pace of 8 minutes per mile, while the second runs at an average pace of 7 minutes per mile. After 56 minutes, the second runner stops for a drink of water. For how many minutes could the second runner remain stopped before the first runner catches up with him?

Lets think step by step

Solution: The first runner runs at an average pace of 8 minutes per mile. So, after 56 minutes, the first runner has covered: Distance of the first runner = $(56 \text{ minutes}) / (8 \text{ minutes/mile}) = 7 \text{ miles}$. The second runner runs at an average pace of 7 minutes per mile. After 56 minutes, the second runner has covered: Distance of the second runner = $(56 \text{ minutes}) / (7 \text{ minutes/mile}) = 8 \text{ miles}$. Hence at the 56 minute, the gap between the two runners is $8 - 7 = 1 \text{ mile}$. The first runner is running at a pace of 8 minutes per mile, and he needs to cover the 1-mile gap to catch up with the second runner. Time to close the gap = $(1 \text{ mile}) * (8 \text{ minutes/mile}) = 8 \text{ minutes}$. Therefore, the second runner can remain stopped for a maximum of 8 minutes before the first runner catches up with him.

Answer: 8

Table 5: Prompt given to each LLM for generating the step-by-step chains that are used in §3.2.