Learning to Rank Chain-of-Thought: An Energy-Based Outcome Reward Model

Anonymous ACL submission

Abstract

Large Language Models (LLMs) struggle with reliable mathematical reasoning, and current verification methods are often computationally expensive. This paper introduces the Energy Outcome Reward Model (EORM), a highly efficient, lightweight post-hoc verifier designed to address this challenge. EORM uses an energy-based framework to rank Chain-of-Thought (CoT) solutions, learning to distinguish correct from incorrect reasoning using only simple outcome labels, thus eliminating the need for expensive annotations. With only 55M parameters, over 127 times smaller than typical reward models, EORM boosts the accuracy of Llama 3 8B to 90.7% on GSM8k and 63.7% on MATH. This performance is achieved by efficiently selecting the optimal reasoning path from a pool of candidates, allowing it to match or exceed the accuracy of far more resource-intensive Best-of-N sampling techniques. Crucially, our experiments show that EORM generalizes effectively to out-of-distribution problems and unseen models, indicating it learns fundamental principles of valid reasoning. This robustness, combined with its efficiency, establishes EORM as a practical tool for deploying more dependable LLMs in complex, real-world applications. Our code is available at `https://anonymous.4open.science/r/Learning-to-rank-COT-anynomized-7718/`.

1 Introduction

Mathematical reasoning remains a critical and challenging domain for Large Language Models (LLMs), demanding a high degree of logical consistency and multi-step inferential accuracy that often eludes current architectures (Guo et al., 2025; Tong et al., 2024). While Chain-of-Thought (CoT) (Wei et al., 2022; Kojima et al., 2022) has significantly advanced the ability of LLMs to articulate intermediate reasoning steps, thereby improving performance on complex tasks (Wang et al., 2023; Huang et al., 2022; Yao et al., 2022), it offers no intrinsic guarantee of correctness. This leads to a fundamental challenge: among the multiple Chain-of-Thought produced by the LLMs, how can we reliably identify the single most accurate one? This selection problem, widely referred to as the *Best-of-N challenge*, has become a key bottleneck in advancing the reliability of mathematical reasoning with LLMs (Tong et al., 2024; Xiong et al., 2024; Press et al., 2022).

To solve the Best-of-N problem, researchers have tried many approaches, and currently, existing approaches fall into two categories: (1) majority voting and (2) post-training with reinforcement learning (RL) to train a reward model for answer selection. Majority voting (Toh et al., 2025; Wang et al., 2023) is the simplest method: it chooses the answer most frequently generated by the model. While this method is easy to implement, it tends to reflect the model's biases rather than identifying the truly correct answer. The RL-based approaches are more sophisticated, with three prominent variants: **(1) Preference Optimization (PO)**, **(2) Outcome Reward Models (ORM)**, and **(3) Process Reward Models (PRM)**. Preference Optimization (Pang et al., 2024; Zhang et al., 2024) uses pairwise comparisons between correct and incorrect answers to train a reward model. Although it can improve answer quality, it requires training a new model for each task and struggles with scenarios involving multiple complex outputs. Outcome Reward Models (Lyu et al., 2025; Hosseini et al., 2024; Cobbe et al., 2021b) are similar but allow supervision from multiple positive and negative examples, making them more flexible than pairwise-only methods. Process Reward Models (She et al., 2025; Wang et al., 2025) go one step further by assigning rewards to each step in the reasoning process. However, this requires extensive annotation of every Chain-of-Thought (CoT) trace, making data preparation labor-intensive. We provide a more in-

Feature	ORM	PRM	PO	EORM (ours)
Does not require Step-by-Step Labels?	✓	✗	✓	✓
Does not require Preference Pair Labels?	✓	✓	✗	✓
Uses Binary Outcome Labels as Reward?	✗	✗	✗	✓
Low Annotation Cost?	✓	✗	✓	✓
Lightweight Model	✗	✗	✗	✓

Table 1: **EORM's feature benefit compared to other reasoning baselines.** We compare EORM with ORM (Outcome Reward Model), PRM (Process Reward Model), and PO (Preference Optimization) to demonstrate the feature benefit of our method over these baselines.

depth comparison of the limitations of the different types of reward model in Table 1.

This motivates the need for a more efficient and generalizable alternative. Our approach begins with the observation that model outputs are sequences of tokens, which allows us to tokenize these responses and analyze their statistical patterns. Each output can be interpreted as a sample from an underlying probability distribution. Leveraging the Universal Approximation Theorem (Liu et al., 2025b; Kratsios, 2021), we recognize that a sufficiently expressive neural network can, in principle, learn to distinguish between these distributions. Building on this insight, we propose training a lightweight neural network model to classify responses as good or bad, effectively select the best answer using Best-of-N, acting like a "LLM-as-a-Judge". Crucially, because this model learns from token-level patterns rather than task-specific features, it is capable of generalizing across outputs from different base LLMs. To further enhance our neural network's ability to capture complex distributions and support robust generalization, we adopt an Energy-Based Model (EBM) architecture (Deng et al., 2020a; Du and Mordatch, 2020). EBMs have demonstrated strong performance in related tasks, utilizing their uniqueness of using Energy Probabilistic Landscape, and in our experiments, our EBM-based judge consistently outperforms both majority voting and traditional Outcome Reward Models in identifying the best answer.

In this paper, we introduce the Energy Outcome Reward Model (EORM), an efficient post-hoc verifier for CoT outputs. EORM leverages principles from Energy-Based Models (EBMs) (Du and Mordatch, 2019; Liu et al., 2020) to effectively rerank and select the best candidate solutions. Our methodology trains a lightweight model to assign a scalar energy score to each candidate, learning to distinguish correct from incorrect reasoning using only simple binary outcome labels. Critically, this approach allows us to drastically reduce the model size from the typical 7B to 8B parameters of reward models to just 55M, achieving a greater than 127x reduction in parameter count while maintaining high performance.

Our main contributions are threefold:

① We propose EORM, a lightweight and efficient verifier that assigns a scalar energy score to each CoT solution. By training on simple binary outcome labels (correct/incorrect), EORM eliminates the need for costly step-by-step annotations or preference pairs required by Process and Preference-based Reward Models.

② We demonstrate that EORM significantly boosts the performance of various open-source LLMs on challenging mathematical reasoning benchmarks, including GSM8k and MATH. Our approach consistently achieves superior accuracy by effectively reranking candidate solutions, often matching or exceeding the performance of more computationally intensive methods.

③ We empirically validate that EORM generalizes effectively to both out-of-distribution datasets and unseen LLM architectures. This highlights its ability to learn underlying principles of logical reasoning rather than overfitting to the stylistic patterns of the models in its training set.

2 Related Work

Our work intersects with advancements in Chain-of-Thought (CoT) reasoning, the verification and reranking of Large Language Model (LLM) outputs, and Energy-Based Models (EBMs). CoT reasoning (Wei et al., 2022; Kojima et al., 2022) has significantly improved multi-step reasoning in LLMs (Bai et al., 2023; Touvron et al., 2023; OpenAI, 2023; Guo et al., 2025), with further refinements like Least-to-Most prompting (Zhou et al., 2022) and Tree-of-Thoughts (Yao et al., 2023; Zhang et al., 2023). However, the fallibility of CoT

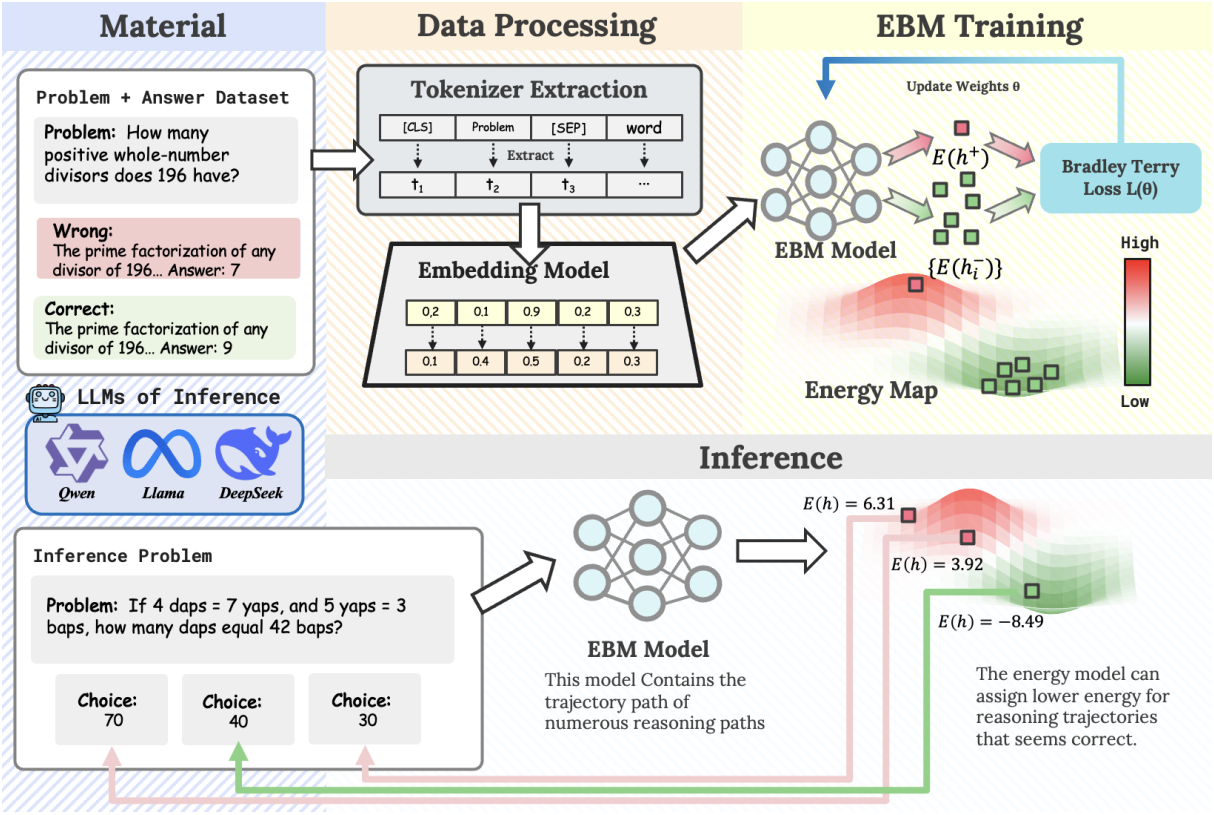

Figure 1: **An overview of the flow chart of EORM.** In the EORM process, the model tokenizes the question–answer pair, then computes an energy score using an Energy-Based Model (EBM). The Bradley-Terry loss serves as the objective for reward-based fine-tuning. During deployment, the trained energy reward model computes energy scores for classification tasks.

outputs (Tong et al., 2024), where a single error can invalidate a solution, necessitates robust verification. Common approaches like self-consistency (Wang et al., 2023) improve reliability but incur substantial computational costs by sampling numerous solutions (Wu et al., 2024). To address this, various reranking and verification techniques have emerged, including training separate verifier models (Luo et al., 2024; Khalifa et al., 2023; Liu et al., 2024) or adopting learning-to-rank perspectives (Liu, 2009; Deng et al., 2020b). These methods, while effective, often introduce their own complexities or computational demands. Our approach, EORM, draws inspiration from EBMs (Du and Mordatch, 2019), which assign scalar energy scores to configurations and are well-suited for ranking (Grathwohl et al., 2019; Liu, 2009). Specifically, we leverage the insight that classifier logits can be interpreted as negative energies (Grathwohl et al., 2019; Liu et al., 2020). By training a lightweight EBM with a pairwise Bradley-Terry loss (Liu, 2009) on outcome labels, EORM efficiently reranks CoT candidates, offering a distinct, streamlined post-hoc verification mechanism that complements retrieval-augmented methods (Rakin et al., 2024; Schick et al., 2023; Yao et al., 2022; Press et al., 2022; Khattab et al., 2022) and other specialized verifiers like Chain-of-Actions (Pan et al., 2025) or Search-in-the-Chain (Xu et al., 2023). For a more detailed discussion of related literature, please refer to Appendix E.

3 Methodology

In this section, we introduce the architectural design and technical foundations of our proposed **Energy Outcome Reward Model (EORM)**, a lightweight verifier for Chain-of-Thought (CoT) reasoning. We begin by outlining the principles of Energy-Based Models (EBMs), which form the foundation of our approach. We then detail the specific architecture and training objective of EORM, showing how it learns to rank CoT candidates effectively using only simple outcome-based supervision.

3.1 Preliminaries: Energy-Based Models for Ranking

Evaluating CoT reasoning requires a flexible mechanism that can distinguish high-quality reasoning from incorrect or incoherent paths. Instead of treating this as a classification task, we model it as a preference-ranking problem using Energy-Based

Models (EBMs).

Energy-Based Models (EBMs) provide a flexible approach to modeling distributions by assigning a scalar energy $E_\theta(y)$ to each configuration y from a space $\mathcal{Y}$. This energy function is parameterized by θ. Lower energy typically corresponds to more desirable or probable configurations. In this work, $\mathcal{Y}$ is the space of possible Chain-of-Thought text sequences y, and θ represents the learnable parameters of our EORM model.

Given an energy function $E_\theta(y)$, the corresponding Boltzmann (or Gibbs) distribution $p_\theta(y)$ assigns a probability to each configuration $y \in \mathcal{Y}$ as:

$$p_\theta(y) \;=\; \frac{\exp\big(-E_\theta(y)\big)}{Z_\theta}, \tag{1}$$

where Z_θ is the partition function, a normalization constant that ensures the probability distribution sums to unity:

$$Z_\theta \;=\; \sum_{y' \in \mathcal{Y}} \exp\big(-E_\theta(y')\big) \quad \text{(for discrete } \mathcal{Y}\text{)}. \tag{2}$$

A key challenge in working with EBMs is that computing the partition function Z_θ is often computationally intractable. However, for tasks involving ranking or selecting the best candidate from a finite set $\mathcal{Y}_{\text{cand}} \subset \mathcal{Y}$, the explicit computation of Z_θ is unnecessary. Since Z_θ is constant for all $y \in \mathcal{Y}_{\text{cand}}$, comparing probabilities $p_\theta(y)$ is equivalent to comparing the unnormalized scores $\exp(-E_\theta(y))$, which in turn relates directly to comparing the energies $E_\theta(y)$. This equivalence implies that energy minimization provides a direct mechanism for identifying the most probable (preferred) solution within a candidate set, without needing to compute Z_θ:

$$y^* \;=\; \arg\min_{y \in \mathcal{Y}_{\text{cand}}} E_\theta(y) \;=\; \arg\max_{y \in \mathcal{Y}_{\text{cand}}} p_\theta(y). \tag{3}$$

This makes EBMs well-suited for our task of reranking CoT solutions based on learned preferences encoded in the energy function $E_\theta(y)$.

3.2 EORM: Architecture and Training Objective

Building on the EBM framework, EORM is designed as a lightweight yet powerful verifier for CoT reasoning. The core idea is to train a small neural network to learn an energy function $E_\theta(y)$ that maps any given CoT solution y to a scalar energy score. By design, this function assigns low energy to correct reasoning paths and high energy to incorrect ones, enabling effective reranking of candidate solutions.

Architecture. To implement the energy function $E_\theta(y)$, EORM uses a lightweight Transformer encoder. A given CoT solution y is first tokenized, and a special classification token (e.g., `[CLS]`) is prepended. The sequence is then passed through the Transformer encoder. The final hidden state corresponding to the `[CLS]` token, denoted $\mathbf{h}_{\text{CLS}}$, serves as a holistic representation of the entire reasoning path. This representation is fed into a simple energy head—a Multi-Layer Perceptron (MLP) with Layer Normalization—which projects it to the final scalar energy value:

$$E_\theta(y) \;=\; \text{MLP}\Big(\text{LayerNorm}\big(\mathbf{h}_{\text{CLS}}\big)\Big) \;\in\; \mathbb{R}. \tag{4}$$

This scalar output $E_\theta(y)$ represents the energy assigned to the sequence y. Lower values indicate higher assessed quality or correctness.

Training Objective. EORM is trained to distinguish between correct and incorrect solutions using a simple yet effective pairwise ranking loss. This approach only requires binary outcome labels for each CoT, avoiding the need for expensive, fine-grained annotations required by process-based reward models. For a given problem, we collect a set of generated CoT solutions $\mathcal{Y}_n$ and partition it into a subset of correct solutions, $\mathcal{Y}_+$, and a subset of incorrect solutions, $\mathcal{Y}_-$, where:

$$\mathcal{Y}_+ = \big\{y \in \mathcal{Y}_n \mid l(y) = 1\big\}, \quad \mathcal{Y}_- = \big\{y \in \mathcal{Y}_n \mid l(y) = 0\big\}. \tag{5}$$

The model's parameters θ are then optimized to ensure that for any pair of solutions (y_+, y_-) where $y_+ \in \mathcal{Y}_+$ and $y_- \in \mathcal{Y}_-$, the energy of the correct solution is lower than that of the incorrect one. We formalize this using the Bradley-Terry loss, which is summed over all possible pairs within the group:

$$\mathcal{L}(\theta; \mathcal{Y}_n) = \frac{1}{|\mathcal{Y}_+||\mathcal{Y}_-|} \sum_{y_+ \in \mathcal{Y}_+} \sum_{y_- \in \mathcal{Y}_-} \log\Big(1 + \exp\big(E_\theta(y_+) - E_\theta(y_-)\big)\Big). \tag{6}$$

Minimizing this loss encourages a clear separation in the energy landscape, pushing the model to assign consistently lower energy scores to correct solutions. The complete training procedure is summarized in Algorithm 1. For a more detailed theoretical analysis, please refer to Appendix F.

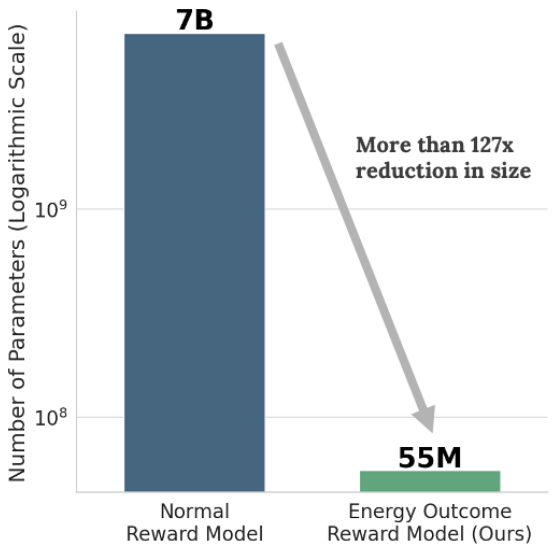

Figure 2: **A comparison of the parameter sizes between a standard reward model and our EORM Model.** A typical reward model has approximately 7 billion parameters, while EORM has only 55 million, demonstrating a size reduction of over 127 times and highlighting EORM's efficiency.

Implementation. It is crucial to distinguish between the training and deployment phases of EORM. During the **offline training phase**, the model requires ground-truth labels to learn the energy landscape that distinguishes valid reasoning. However, during the **online deployment phase**, EORM functions as a standalone verifier for open-domain problems where ground-truth answers are unavailable. By internalizing the fundamental principles of validity during training, EORM can rank and select the most plausible solution from unlabeled candidates at inference time.

4 Experiments

In this section, we empirically evaluate our proposed Energy Outcome Reward Model (EORM). A key advantage of EORM, as highlighted in our introduction, is its exceptional efficiency; with only 55M parameters, it is over 127 times smaller than typical 7B reward models, as shown in Figure 2. Our evaluation is structured into three main parts. **(1)** First, in Section 4.2, we assess its performance on in-distribution mathematical reasoning tasks. **(2)** Second, in Section 4.3, we test its robustness on out-of-distribution benchmarks. **(3)** Finally, in Section 4.4, we analyze its ability to generalize to outputs from unseen LLM architectures.

4.1 Experimental Setup

For our experiments, we use the GSM8k (Cobbe et al., 2021a) and MATH (Hendrycks et al., 2021) datasets for in-distribution tasks, and AIME 2024, AMC, and AGIEval's SAT Math and Gaokao Math subsets (Zhong et al., 2023) for out-of-distribution evaluation. We applied EORM to several open-source Large Language Models (LLMs): Mistral-7B-v0.1 (Jiang et al., 2023), DeepSeekMath-7B (Shao et al., 2024), Llama 3 8B (Grattafiori et al., 2024), Qwen 2.5 7B (Yang et al., 2024), and Llama 2 7B (Touvron et al., 2023).

EORM was trained using a pairwise Bradley-Terry loss function (Liu, 2009). The training dataset was constructed from Chain-of-Thought (CoT) solutions for problems in the in-domain GSM8k and MATH training splits. These solutions were generated using all five aforementioned LLMs. For each training problem, we generated $n = 256$ CoT candidates with a temperature of 0.7 and a top-p (nucleus) sampling of 0.9, ensuring all attempts were included, regardless of their correctness. Each training instance comprised the original question, a generated solution, and a label indicating whether the solution was correct (y^+) or incorrect (y^-). EORM learned from these pairs to assign lower energy scores to preferred solutions by interpreting classifier logits as negative energies (Grathwohl et al., 2019; Liu et al., 2020). The GPT-2 tokenizer (Radford et al., 2019) was employed for the energy-based tokenizer.

To provide a rigorous comparison, our "ORM" baseline represents a standard, large-scale reward model setup. It utilizes the Llama 3 8B base model equipped with a reward head and is trained using the standard pairwise ranking loss (identical to the ranking objective used in WizardMath). This allows us to compare our 55M parameter EORM directly against a standard reward model that is over 145x larger.

For evaluation, we generated Chain-of-Thought (CoT) candidates across all models using a temperature of 0.7 and a top-p sampling of 0.9. Specifically, $n = 256$ candidates were generated for each in-distribution problem, while $n = 64$ candidates were generated for each out-of-distribution problem. Subsequently, EORM was used post-hoc to select the candidate with the lowest energy. The final answer accuracy is reported in the two subsections that follow.

4.2 In-Distribution Learning

Our primary evaluation focused on EORM's performance within the domains it was trained on, namely the GSM8k and MATH test sets. The quan-

Model	Base Model	Params	GSM8k	MATH	Avg.
Mistral-v0.1 (Jiang et al., 2023)	Mistral-v0.1	7B	42.9$_{+0.0}$	12.9$_{+0.0}$	27.9
MathScale (Tang et al., 2024)	Mistral-v0.1	7B	74.8$_{+31.9}$	35.2$_{+22.3}$	55.0
MMIQC (Liu et al., 2025a)	Mistral-v0.1	7B	74.8$_{+31.9}$	36.0$_{+23.1}$	55.4
MetaMath (Yu et al., 2024)	Mistral-v0.1	7B	77.9$_{+35.0}$	28.6$_{+15.7}$	53.3
KPMath-Plus (Huang et al., 2024)	Mistral-v0.1	7B	82.1$_{+39.2}$	46.8$_{+33.9}$	64.5
DART-Math (Tong et al., 2024)	Mistral-v0.1	7B	82.6$_{+39.7}$	43.5$_{+30.6}$	63.1
Math-Shepherd (Wang et al., 2024)	Mistral-v0.1	7B	84.1$_{+41.2}$	33.0$_{+20.1}$	58.6
MAmmoTH2-Plus (Yue et al., 2024)	Mistral-v0.1	7B	84.7$_{+41.8}$	45.0$_{+32.1}$	64.9
Xwin-Math (Li et al., 2024a)	Mistral-v0.1	7B	89.2$_{+46.3}$	43.7$_{+30.8}$	66.5
WizardMath-Mistral (Luo et al., 2025)	Mistral-v0.1	7B	90.7$_{+47.8}$	55.4$_{+42.5}$	73.1
EORM (Ours)	**Mistral-v0.1**	7B	**91.0**$_{+48.1}$	**48.8**$_{+35.9}$	**69.9**
Llama2 (Touvron et al., 2023)	Llama 2	7B	14.6$_{+0.0}$	2.5$_{+0.0}$	8.6
MAmmoTH-CoT (Yue et al., 2024)	Llama 2	7B	50.5$_{+35.9}$	10.4$_{+7.9}$	30.5
MetaMath (Yu et al., 2024)	Llama 2	7B	66.5$_{+51.9}$	19.8$_{+17.3}$	43.2
Math-Shepherd (Wang et al., 2024)	Llama 2	7B	73.2$_{+58.6}$	21.6$_{+19.1}$	47.4
EORM (Ours)	**Llama 2**	7B	**75.6**$_{+61.0}$	**21.8**$_{+19.3}$	**48.7**
DeepSeekMath-Base (Shao et al., 2024)	DeepSeekMath	7B	64.2$_{+0.0}$	36.2$_{+0.0}$	50.2
NuminaMath-CoT (Li et al., 2024b)	DeepseekMath	7B	75.4$_{+11.2}$	55.2$_{+19.0}$	65.3
MMIQC (Liu et al., 2025a)	DeepSeekMath	7B	79.0$_{+14.8}$	45.3$_{+9.1}$	62.2
KPMath-Plus (Huang et al., 2024)	DeepSeekMath	7B	83.9$_{+19.7}$	48.8$_{+12.6}$	66.4
DeepSeekMath-RL (Shao et al., 2024)	DeepSeekMath	7B	88.2$_{+24.0}$	51.7$_{+15.5}$	70.0
EORM (Ours)	**DeepSeekMath**	7B	**84.2**$_{+20.0}$	**58.7**$_{+22.5}$	**71.5**
Llama 3 (Team, 2024)	Llama 3	8B	76.6$_{+0.0}$	28.9$_{+0.0}$	52.8
MetaMath (Yu et al., 2024)	Llama 3	8B	77.3$_{+0.7}$	30.8$_{+1.9}$	54.1
MMIQC (Liu et al., 2025a)	Llama 3	8B	77.6$_{+1.0}$	29.5$_{+0.6}$	53.6
DART-Math (Tong et al., 2024)	Llama 3	8B	82.5$_{+5.9}$	45.3$_{+16.4}$	63.9
MAmmoTH2-Plus (Yue et al., 2024)	Llama 3	8B	84.1$_{+7.5}$	42.8$_{+13.9}$	63.5
Llama 3.1-Instruct (Team, 2024)	Llama 3	8B	84.5$_{+7.9}$	51.9$_{+23.0}$	68.2
JiuZhang3.0 (Zhou et al., 2024)	Llama 3	8B	88.6$_{+12.0}$	51.0$_{+22.1}$	69.8
WizardMath-Llama (Luo et al., 2025)	Llama 3	8B	90.3$_{+13.7}$	58.8$_{+29.9}$	74.6
EORM (Ours)	**Llama 3**	8B	**90.7**$_{+14.1}$	**63.7**$_{+34.8}$	**77.2**
Qwen2.5 7B (Yang et al., 2024)	Qwen 2.5	7B	89.5$_{+0.0}$	63.4$_{+0.0}$	76.5
EORM (Ours)	**Qwen 2.5**	7B	**92.8**$_{+3.3}$	**65.8**$_{+2.4}$	**79.3**

Table 2: **Performance comparison on math reasoning benchmarks.** We evaluate EORM on GSM8K and MATH using five LLM structures. Note that baselines such as WizardMath and MetaMath involve extensive specialist fine-tuning. This table is intended to provide a contextual comparison, demonstrating that EORM applied post-hoc to a base model can achieve performance competitive with these computationally intensive methods.

titative outcomes, detailed in Table 2, consistently show that employing EORM as a post-hoc reranker leads to substantial improvements in final answer accuracy compared to the baseline performance of the underlying LLMs generating the candidate solutions. By effectively identifying and selecting the most plausible reasoning path from the generated set, EORM significantly enhances problem-solving capabilities. For instance, when integrated with Llama 3 8B and reranking $n = 256$ candidate solutions, EORM achieved high accuracy levels of 90.7% on GSM8k and 63.7% on MATH, demonstrating its ability to leverage multiple reasoning attempts effectively. The relationship between the number of candidate samples (n) and the resulting accuracy improvement is visualized in Figure 3, illustrating how performance generally scales with more samples but strong gains are often achievable with moderate sampling budgets. In addition, to assess the upper bound of reranking, we analyzed how often a correct answer exists in the candidate pool. At $n = 256$, the oracle accuracy for Llama 3 8B is 94.6% on GSM8k and 78.8% on MATH. EORM achieves 90.7% and 63.7% respectively, effectively recovering 95.9% (GSM8k) and 80.8% (MATH) of the theoretically possible performance.

4.3 Out-of-Distribution Learning

A critical aspect of evaluating any reasoning model is its ability to generalize to new, unseen problems

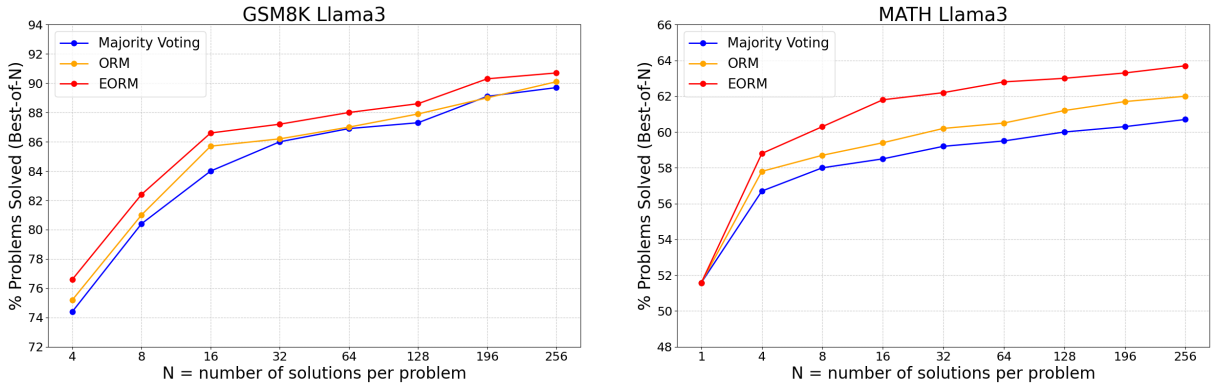

Figure 3: **Scaling behavior with respect to the number of sampled solutions.** Best-of-N accuracy on GSM8K (left) and MATH (right) under LLaMA-3 8B. While all methods improve as N increases, EORM achieves consistently higher accuracy than Majority Voting and ORM, particularly when the number of samples is limited.

and potentially different reasoning styles. To rigorously test this, we assessed the performance of EORM (trained exclusively on GSM8k and MATH data) on a suite of OOD benchmarks known for their difficulty: AIME 2024, AMC, and AGIEval's SAT Math and Gaokao Math subsets (Zhong et al., 2023). These tasks often require more complex mathematical insights or different problem-solving strategies than those predominant in the training datasets. For this phase, evaluations primarily utilized Llama-3 8B and Qwen-2.5 7B as the base models, generating $n = 64$ candidate solutions for each OOD problem instance. The results, documented in Table 3, demonstrate EORM's strong generalization capabilities. When applied to the Llama-3 base model, EORM generally yielded superior performance compared to alternative baseline reranking methods like TTRL (Zuo et al., 2025) and MathWizard (Luo et al., 2025) across the majority of the evaluated OOD datasets under the same experimental setup. Impressively, EORM achieved significant scores on highly challenging benchmarks such as AIME 2024 (reaching 10.0%) and the AGIE Gaokao Math problems (achieving 70.3%). Additionally, the Qwen-2.5 base model shows similar findings, providing further evidence that EORM is not merely overfitting to patterns specific to GSM8k and MATH. Instead, it appears to learn more fundamental, transferable principles related to the structure and validity of mathematical reasoning, allowing it to effectively discern higher-quality solutions even when faced with novel problem types and domains.

4.4 Generalization Capabilities

To further assess EORM's robustness, we analyze its generalization capabilities when encountering outputs from LLMs not seen during training. This is crucial for real-world applicability where new models are constantly emerging. We test two scenarios: generalization to unseen models on in-distribution tasks, and a more challenging test of generalization to unseen models on out-of-distribution tasks.

Generalization to Unseen Models on In-Distribution Tasks. We employ a leave-one-out training strategy where a variant of EORM, termed "EORM Generalize," is trained on CoT data from four of the five LLMs and tested on the held-out model. The results, detailed in Table 4, show that this variant consistently improves accuracy over the base performance. For instance, when Mistral-v0.1 is the held-out model, its base accuracy on GSM8k is 42.9%. "EORM Generalize" elevates this to 76.0%, while the standard EORM (trained on all models) reaches 91.0%. The significant uplift from "EORM Generalize" demonstrates that our model learns generalizable features of correct reasoning, rather than merely overfitting to the stylistic patterns of the models in its training set. The performance gap between the standard EORM and "EORM Generalize" is expected, as the latter lacks exposure to the target LLM's specific error patterns and stylistic nuances. However, the crucial finding is that "EORM Generalize" still provides a substantial boost, indicating that it captures fundamental principles of logical consistency and procedural correctness. This ability to transcend the idiosyncrasies of individual LLMs confirms that EORM is

Model	AIME 2024	AMC	SAT Math	Gaokao Math	Avg
Llama-3 8B	3.3 ▲+0.0	19.3 ▲+0.0	77.3▲+0.0	48.7▲+0.0	37.2
+ TTRL (Zuo et al., 2025)	3.3▲+0.0	32.5▲+13.2	89.1▲+11.8	61.0▲+12.3	46.5
+ DART-MATH (Prop2Diff) (Tong et al., 2024)	6.7▲+3.4	26.5▲+7.2	**90.5**▲+13.2	67.6▲+18.9	47.8
+ **EORM Ours**	**10.0**▲+6.7	**28.9**▲+9.6	**90.5**▲+13.2	**70.3**▲+21.6	**49.9**
Qwen-2.5 7B	16.7▲+0.0	53.0▲+0.0	91.4▲+0.0	83.3▲+0.0	61.1
+ TTRL (Zuo et al., 2025)	**43.3**▲+26.6	67.5▲+14.5	95.0▲+3.6	88.2▲+4.9	73.5
+ **EORM Ours**	**43.3**▲+26.6	**68.7**▲+15.7	**96.4**▲+5.0	**88.9**▲+5.6	**74.3**

Table 3: **Comparison of EORM with other reasoning methods.** We evaluate EORM against two reasoning baselines, TTRL and DART-MATH, using accuracy as the metric across four mathematical datasets. Each method uses 64 samples per question for a fair comparison. EORM consistently achieves the highest performance, with the best accuracy values highlighted in bold.

learning the underlying structure of valid mathematical arguments.

Model	GSM8k	MATH
Mistral-v0.1 7B (Jiang et al., 2023)	42.9▲+0.0	12.9▲+0.0
+ EORM Generalize	76.0▲+33.1	23.0▲+10.1
+ EORM	91.0▲+48.1	48.8▲+35.9
Llama2 7B (Touvron et al., 2023)	14.6▲+0.0	2.5▲+0.0
+ EORM Generalize	38.5▲+23.9	7.8▲+5.3
+ EORM	75.6▲+61.0	21.8▲+19.3
DeepSeekMath-Base 7B (Shao et al., 2024)	64.2▲+0.0	36.2▲+0.0
+ EORM Generalize	81.6▲+17.4	41.7▲+5.5
+ EORM	84.2▲+20.0	58.7▲+22.5
Llama3 8B (Team, 2024)	76.6▲+0.0	28.9▲+0.0
+ EORM Generalize	76.6▲+0.0	51.6▲+31.5
+ EORM	90.7▲+47.8	63.7▲+43.6
Qwen2.5 7B (Yang et al., 2024)	89.5▲+0.0	63.4▲+0.0
+ EORM Generalize	90.2▲+0.7	63.8▲+0.4
+ EORM	92.8▲+3.3	65.8▲+2.4

Table 4: **Performance on in-distribution benchmarks (GSM8k, MATH) when EORM is trained without exposure to the target model's outputs.** EORM Generalize is trained on CoT data from four LLMs and tested on the fifth (target) LLM. EORM (from Table 2) is trained on data from all five LLMs. Accuracy is reported.

Generalization to Unseen Models and Tasks (Out-of-Distribution). We test a more demanding scenario where the "EORM Generalize" variant is evaluated on challenging OOD benchmarks using CoT solutions from the held-out LLM. This setup assesses EORM's ability to transfer learned principles to both novel tasks and unfamiliar model outputs simultaneously. The results in Table 5 show that even under this compounded challenge, our model provides tangible improvements. For example, with Llama 3 8B as the held-out model, "EORM Generalize" raises the average OOD accuracy from a base of 37.2% to 40.5%. This suggests EORM internalizes fundamental and transferable heuristics about the structure and coherence of valid reasoning that apply broadly. As anticipated, the standard EORM—which benefited from seeing the target model's in-distribution outputs during training—achieves superior OOD performance. Nevertheless, the fact that "EORM Generalize" still delivers tangible improvements in this dual-unseen scenario is a strong testament to its robustness.

Model	AIME	AMC	SAT	Gaokao	Avg
Llama-3 8B	3.3	19.3	77.3	48.7	37.2
+ EORM Gen.	6.7	20.5	82.3	52.6	40.5
+ EORM	10.0	28.9	90.5	70.3	49.9
Qwen-2.5 7B	16.7	53.0	91.4	83.3	61.1
+ EORM Gen.	26.7	54.2	92.3	84.5	64.4
+ EORM	43.3	68.7	96.4	88.9	74.3

Table 5: **Out-of-distribution (OOD) generalization on unseen mathematics benchmarks.** Performance on AIME, AMC, SAT, and Gaokao when EORM is trained without access to the target model's outputs. *EORM Gen.* is trained on reasoning data from other base models only, while *EORM* additionally includes target-model data. Results show consistent gains over the base model, with further improvements when target-model supervision is available.

5 Conclusion

We introduced EORM, a lightweight energy-based model for post-hoc verification of Chain-of-Thought reasoning. Trained solely on outcome labels, EORM efficiently reranks candidate solutions to significantly boost mathematical reasoning performance across various benchmarks. Its effectiveness, combined with a parameter count over 127x smaller than typical reward models, presents a practical path toward more dependable and accessible LLMs for complex reasoning. In future work, we plan to extend this framework to broader domains and explore hybrid approaches that combine the efficiency of outcome supervision with the granularity of process-level feedback.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Karl Cobbe, Jacob Hilton, and John Schulman. 2021a. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

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. 2021b. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020a. Residual energy-based models for text generation. *Preprint*, arXiv:2004.11714.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020b. Residual energy-based models for text generation. In *International Conference on Learning Representations (ICLR)*.

Yilun Du and Igor Mordatch. 2019. Implicit generation and modeling with energy-based models. *Advances in Neural Information Processing Systems*, 32.

Yilun Du and Igor Mordatch. 2020. Implicit generation and generalization in energy-based models. *Preprint*, arXiv:1903.08689.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. 2019. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Colin Burns, Spencer Chen, , and 1 others. 2021. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 3512–3524.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *Preprint*, arXiv:2402.06457.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR.

Yiming Huang, Xiao Liu, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, and Weizhu Chen. 2024. Key-point-driven data synthesis with its enhancement on mathematical reasoning. *Preprint*, arXiv:2403.02333.

Alon "Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor" Geva. 2024. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Eric Hanchen Jiang, Zhi Zhang, Dinghuai Zhang, Andrew Lizarraga, Chenheng Xu, Yasi Zhang, Siyan Zhao, Zhengjie Xu, Peiyu Yu, Yuer Tang, Deqian Kong, and Ying Nian Wu. 2024. Dodt: Enhanced online decision transformer learning through dreamer's actor-critic trajectory forecasting. *Preprint*, arXiv:2410.11359.

Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, , and 1 others. 2023. GRACE: Discriminator-guided chain-of-thought reasoning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 11710–11725, Singapore.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*.

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.

Anastasis Kratsios. 2021. The universal approximation property: Characterization, construction, representation, and existence. *Annals of Mathematics and Artificial Intelligence*.

Yann LeCun, Sumit Chopra, and Raia Hadsell. 2006. A tutorial on energy-based learning. In *Predicting Structured Data*, pages 1–59. MIT Press.

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. 2024a. Common 7b language models already possess strong math capabilities. *Preprint*, arXiv:2403.04706.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024b. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.

Zongyu Lin, Yao Tang, Xingcheng Yao, Da Yin, Ziniu Hu, Yizhou Sun, and Kai-Wei Chang. 2025. Qlass: Boosting language agent inference via q-guided stepwise search. *arXiv preprint arXiv:2502.02584*.

Haoxiong Liu, Yifan Zhang, Yifan Luo, and Andrew C. Yao. 2025a. Augmenting math word problems via iterative question composing.

Hude Liu, Jerry Yao-Chieh Hu, Zhao Song, and Han Liu. 2025b. Attention mechanism, max-affine partition, and universal approximation. *Preprint*, arXiv:2504.19901.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. In *Foundations and Trends® in Information Retrieval*, volume 3, pages 225–331. Now Publishers Inc.

Weitang Liu, Xiaoyun Wang, John D Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475.

Xiangyang Liu, Junliang He, and Xipeng Qiu. 2024. Making large language models better reasoners with orchestrated streaming experiences.

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. In *The Thirteenth International Conference on Learning Representations*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, and 1 others. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, and 1 others. 2025. Exploring the limit of outcome reward for learning mathematical reasoning. *arXiv preprint arXiv:2502.06781*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Zhenyu Pan, Haozheng Luo, Manling Li, and Han Liu. 2025. Chain-of-action: Faithful and multimodal question answering through large language models. In *The Thirteenth International Conference on Learning Representations*.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. In *Advances in Neural Information Processing Systems*, volume 37.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

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

Salman Rakin, Md AR Shibly, Zahin M Hossain, Zeeshan Khan, and Md Mostofa Akbar. 2024. Leveraging the domain adaptation of retrieval augmented generation models for question answering and reducing hallucination. *arXiv preprint arXiv:2410.17783*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shuaijie She, Junxiao Liu, Yifeng Liu, Jiajun Chen, Xin Huang, and Shujian Huang. 2025. R-prm: Reasoning-driven process reward modeling. *Preprint*, arXiv:2503.21295.

Yang Song and Diederik P. Kingma. 2021. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *Preprint*, arXiv:2403.02884.

Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Vernon Toh, Deepanway Ghosal, and Soujanya Poria. 2025. Not all votes count! translated program for verification improves self-consistency of language models for math reasoning. In *2nd AI for Math Workshop @ ICML 2025*.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. DART-math: Difficulty-aware rejection tuning for mathematical problem-solving. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. 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)*.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, and 1 others. 2025. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*.

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*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Haoran Wu, Avishkar Balika Singh, McKay Andrus, Bohyung Yang, Abdelrahman Mourad, , and 1 others. 2024. Mitigating misleading chain-of-thought reasoning with selective filtering. *arXiv preprint arXiv:2403.19167*.

Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. 2024. Large language models can learn temporal reasoning. *arXiv preprint arXiv:2401.06853*.

Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-seng Chua. 2023. Search-in-the-chain: Towards the accurate, credible and traceable content generation for complex knowledge-intensive tasks. *arXiv preprint arXiv:2304.14732*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Xiang Yue, Tianyu Zheng, Ge Zhang, and Wenhu Chen. 2024. MAmmoTH2: Scaling instructions from the web. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024. Chain of preference optimization: Improving chain-of-thought reasoning in llms. *Advances in Neural Information Processing Systems*, 37:333–356.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Kun Zhou, Beichen Zhang, jiapeng wang, Zhipeng Chen, Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, and Ji-Rong Wen. 2024. Jiuzhang3.0: Efficiently improving mathematical reasoning by training small data synthesis models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yuxin Zuo, Kaiyan Zhang, Shang Qu, Li Sheng, Xuekai Zhu, Biqing Qi, Youbang Sun, Ganqu Cui, Ning Ding, and Bowen Zhou. 2025. Ttrl: Test-time reinforcement learning. *Preprint*, arXiv:2504.16084.

A Limitations

EORM is a post-hoc verifier and therefore inherits the candidate diversity of the underlying base LLM: when the candidate pool contains no correct solution, EORM cannot recover one. Our experiments focus on mathematical reasoning (GSM8k, MATH, AIME, AMC, SAT/Gaokao Math); generalization to other reasoning domains (e.g., code, commonsense, multilingual reasoning) is plausible but not empirically verified here. Training relies on automatic correctness labels derived from final-answer matching, which may be noisy on benchmarks with open-form answers. Finally, EORM is a scalar scoring model and does not explain *why* a solution is preferred; producing such interpretable rationales is left to future work.

B Ethics Statement

This research aims to improve the correctness and efficiency of mathematical reasoning in LLMs, which can yield positive societal impacts such as accelerating scientific discovery, enhancing educational tools, and reducing the energy consumption of large-scale AI computations by minimizing flawed outputs. However, a powerful and efficient reasoning verifier is a dual-use technology: in malicious hands it could be used to automate the selection of the most effective or deceptive generated content. The performance of EORM is intrinsically tied to the datasets used for training; any biases present in problem sets could lead to inequitable behavior across domains. We advocate for strong ethical guidelines for the development and use of verifier and reward models in advanced LLM systems.

C Algorithm

Algorithm 1 Training EORM using Pairwise Bradley-Terry Loss

Require: Labeled groups $\{\mathcal{Y}_n\}_{n=1}^{N}$; learning rate η; number of epochs E

Initialize EBM parameters θ
for epoch = 1 to E **do**
  for each group $\mathcal{Y}_n$ in the training set **do**
    $\mathcal{Y}_+ \leftarrow \{\text{positives in } \mathcal{Y}_n\}$, $\mathcal{Y}_- \leftarrow \{\text{negatives in } \mathcal{Y}_n\}$
    if $\mathcal{Y}_+$ and $\mathcal{Y}_-$ are both non-empty **then**
      Compute energies $E_\theta(y)$ for all $y \in \mathcal{Y}_+ \cup \mathcal{Y}_-$
      $\mathcal{L}_n \leftarrow \frac{1}{|\mathcal{Y}_+||\mathcal{Y}_-|} \sum_{y_+ \in \mathcal{Y}_+} \sum_{y_- \in \mathcal{Y}_-} \log\big(1 + \exp(E_\theta(y_+) - E_\theta(y_-))\big)$
      $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_n$
    end if
  end for
end for

D Ablation Studies

We conducted comprehensive ablation studies to validate the design choices of EORM (detailed in **??**). Our analysis reveals several key findings: (1) **Architecture**: The Transformer backbone significantly outperforms MLP and RNN alternatives (90.7% vs. 82.6% on GSM8k), confirming the importance of modeling long-range dependencies in mathematical reasoning chains. (2) **Tokenizer robustness**: EORM achieves nearly identical performance with both GPT-2 and Llama-native tokenizers, enabling deployment across heterogeneous model ecosystems without retraining. (3) **Training objective**: The pairwise Bradley-Terry ranking loss consistently outperforms BCE and InfoNCE losses, validating that relative comparison better captures subtle reasoning quality differences. (4) **Data efficiency**: While EORM achieves 99% of peak GSM8k performance with only 1.5% of training data (224k samples), the full 15M dataset provides a substantial 12% boost on the harder MATH benchmark, indicating that large-scale data remains essential for complex reasoning tasks.

Architecture	GSM8k	MATH
RNN Verifier	78.3 ▲+0.0	46.2 ▲+0.0
MLP Verifier	82.6 ▲+4.3	52.1 ▲+5.9
Transformer (EORM)	**90.7** ▲+12.4	**63.7** ▲+17.5

Table 6: **Impact of Architecture.** The Transformer backbone significantly outperforms MLP and RNN baselines due to better modeling of long-range dependencies.

Architecture and Pooling. We assessed the importance of the Transformer architecture by comparing our standard EORM against a simpler baseline where the Transformer encoder was replaced with a standard Multi-Layer Perceptron (MLP) verifier. To handle variable-length CoT sequences for the MLP baseline, we applied average (mean) pooling across the token embeddings to produce

a fixed-size input vector. As shown in Table 6, the Transformer-based EORM significantly outperforms the MLP variant, achieving accuracies of 90.7% on GSM8k and 63.7% on MATH, compared to the MLP's 82.6% and 52.1%, respectively. To further validate the architecture, we expanded our comparison to include Recurrent Neural Networks (RNNs). The RNN verifier significantly underperforms (78.3% on GSM8k), confirming that the Transformer's ability to model long-range dependencies is superior to recurrent approaches for this task.

Tokenizer	GSM8k	MATH
Llama 3 (Native)	86.8$^{\blacktriangle +0.0}$	61.8$^{\blacktriangle +0.0}$
GPT-2 (Universal)	**86.6**$^{\blacktriangledown -0.2}$	**61.8**$^{\blacktriangle +0.0}$

Table 7: **Impact of Tokenizer.** EORM is robust to tokenization choice, performing equally well with a universal tokenizer versus a model-native one.

Tokenizer Sensitivity. Next, we examined EORM's sensitivity to the tokenizer by training it with two different tokenizers: a universal GPT-2 tokenizer and the native tokenizer from the Llama 3 model family. As presented in Table 7, the choice of tokenizer has a negligible impact on performance. The model achieves nearly identical scores on both GSM8k (86.6% vs. 86.8%) and MATH (61.8% for both). This robustness suggests that EORM learns fundamental, transferable features of valid reasoning that are not dependent on specific tokenization schemes, underscoring its generalizability.

Loss Function	GSM8k	MATH
BCE Loss	89.8$^{\blacktriangle +0.0}$	59.1$^{\blacktriangle +0.0}$
InfoNCE Loss	90.3$^{\blacktriangle +0.5}$	62.5$^{\blacktriangle +3.4}$
Bradley-Terry (EORM)	**90.7**$^{\blacktriangle +0.9}$	**63.7**$^{\blacktriangle +4.6}$

Table 8: **Impact of Training Objective.** The pairwise Bradley-Terry ranking loss consistently outperforms standard classification (BCE) and contrastive (InfoNCE) losses.

Loss Function Analysis. We investigated the impact of the training objective by comparing EORM against alternative loss functions. As shown in Table 8, the Bradley-Terry ranking loss outperforms both Binary Cross-Entropy (BCE) and InfoNCE losses, particularly on the more complex MATH dataset. This confirms that the pairwise ranking objective is better suited for discriminating between subtle reasoning differences than standard classification or contrastive losses.

Dataset Size	GSM8k	MATH
Subset (224k)	89.9$^{\blacktriangle +0.0}$	51.8$^{\blacktriangle +0.0}$
Subset (938k)	90.3$^{\blacktriangle +0.4}$	58.3$^{\blacktriangle +6.5}$
Full Dataset (15M)	**90.7**$^{\blacktriangle +0.8}$	**63.7**$^{\blacktriangle +11.9}$

Table 9: **Data Efficiency.** EORM learns simple tasks (GSM8k) with very little data, but large-scale data is essential for complex tasks (MATH).

Data Efficiency. Finally, we evaluated EORM's performance when trained on a subset (224k samples, $\approx 1.5\%$ of full data). As shown in Table 9, on the easier GSM8k benchmark, the model recovers 99% of its peak performance (89.9% accuracy), indicating high data efficiency for standard tasks. However, the full 15M dataset provides a substantial 12% boost on the harder MATH benchmark (51.8% vs. 63.7%), suggesting large-scale data is essential for learning complex reasoning patterns.

E Extended Related Work

This section provides a more detailed discussion of research areas related to our work on the Energy Outcome Reward Model (EORM), covering Chain-of-Thought reasoning, techniques for verifying and reranking LLM outputs, and Energy-Based Models.

E.1 Chain-of-Thought and Multi-Step Reasoning

Large Language Models (LLMs) (Bai et al., 2023; Touvron et al., 2023; OpenAI, 2023; Guo et al., 2025) have demonstrated remarkable capabilities, particularly when prompted to generate step-by-step solutions. Chain-of-Thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) has become a cornerstone technique for eliciting complex multi-step reasoning from LLMs. This approach encourages models to "think aloud," breaking down problems into manageable intermediate steps. Various refinements to CoT have been proposed, such as Least-to-Most prompting (Zhou et al., 2022), which guides the model through progressively more complex steps, and Tree-of-Thoughts (ToT) (Yao et al., 2023; Zhang et al.,

2023), which explores multiple reasoning paths in a structured manner. Retrieval-augmented CoT methods (Pan et al., 2025; Rakin et al., 2024; Schick et al., 2023; Yao et al., 2022; Press et al., 2022; Khattab et al., 2022) further enhance reasoning by allowing models to consult external knowledge sources or tools. Despite these advancements, CoT outputs are not infallible; a single incorrect step can derail the entire reasoning process (Tong et al., 2024). This has led to the development of post-hoc processing techniques. Among the most prominent is self-consistency (Wang et al., 2023), which samples multiple CoT outputs and selects the final answer via majority vote. While effective, self-consistency and similar ensemble methods often require generating a large number of candidate solutions, leading to substantial computational overhead (Wu et al., 2024). Our work seeks to mitigate this cost by providing a more efficient mechanism for identifying high-quality CoT solutions.

E.2 Reranking and Verification of LLM Outputs

Given the variability in the quality of CoT paths, a subsequent verification or reranking stage is often crucial for improving the final answer accuracy (Wu et al., 2024; "Jacovi et al., 2024; Liu et al., 2024; Lin et al., 2025). Researchers have explored various strategies for this purpose. Some approaches focus on training separate verifier models to score the correctness of generated solutions or even individual reasoning steps (Luo et al., 2024; Khalifa et al., 2023; Jiang et al., 2024). For example, DI-VeRSe (Liu et al., 2024) trains a model to perform step-aware verification. Other methods adopt a learning-to-rank perspective (Liu, 2009; Deng et al., 2020b), training models to compare and order candidate solutions based on their perceived quality using pairwise or listwise objectives. While these verification and reranking techniques can be effective, many existing methods either rely on large, specialized verifier models that add significant computational load or employ complex, computationally expensive decoding strategies. The goal of EORM is to provide a lightweight yet powerful verifier that can efficiently identify correct CoT solutions without imposing such heavy overhead.

E.3 Energy-Based Models (EBMs)

Energy-Based Models (EBMs) provide a flexible and powerful framework for modeling complex data distributions by assigning a scalar "energy" value to each input configuration (LeCun et al., 2006; Du and Mordatch, 2019). Lower energy values typically correspond to more plausible or desirable configurations. EBMs have found successful applications in diverse domains, including computer vision (Grathwohl et al., 2019), generative modeling (Song and Kingma, 2021; Du and Mordatch, 2019), out-of-distribution detection (Liu et al., 2020), and ranking tasks (Grathwohl et al., 2019; Liu, 2009). A pivotal insight, leveraged by our work, is that the logits produced by standard discriminative classifiers can be interpreted as negative unnormalized log-probabilities, or effectively, as negative energies (Grathwohl et al., 2019). EORM builds on this by using a pairwise Bradley-Terry loss (Liu, 2009), a common technique in learning-to-rank, to train the energy function to distinguish between correct and incorrect CoT solutions.

F Theoretical Details

F.1 Preliminaries

Definition F.1 (Energy Function)**.** *An* energy function $E_\theta : \mathcal{Y} \to \mathbb{R}$ *is a function, parameterized by* θ*, that assigns a scalar value* $E_\theta(y)$ *to each possible configuration* y *from a space* $\mathcal{Y}$*. By convention, lower energy values typically correspond to more desirable or probable configurations.*

Definition F.2 (Boltzmann Distribution)**.** *Given an energy function* $E_\theta(y)$*, the corresponding* Boltzmann (or Gibbs) distribution $p_\theta(y)$ *assigns a probability to each configuration* $y \in \mathcal{Y}$ *as:*

$$p_\theta(y) \;=\; \frac{\exp\big(-E_\theta(y)\big)}{Z_\theta}, \tag{7}$$

where Z_θ *is the partition function.*

Theorem F.1 (Energy–Probability Equivalence for Ranking)**.** *For any finite candidate set* $\mathcal{Y}_{\text{cand}} \subset \mathcal{Y}$*,*

$$y^* \;=\; \arg\min_{y \in \mathcal{Y}_{\text{cand}}} E_\theta(y) \;=\; \arg\max_{y \in \mathcal{Y}_{\text{cand}}} p_\theta(y). \tag{8}$$

Proof. Since Z_θ is a positive constant w.r.t. the choice of $y \in \mathcal{Y}_{\text{cand}}$,

$$\arg\max_y p_\theta(y) = \arg\max_y \exp(-E_\theta(y)) = \arg\min_y E_\theta(y),$$

using strict monotonicity of $\exp(-\cdot)$. $\square$

F.2 Gradient of the Pairwise Bradley-Terry Loss

Theorem F.2 (Gradient of Pairwise Bradley-Terry Loss)**.** *Let σ denote the sigmoid function. Then*

$$\nabla_\theta \mathcal{L}(\theta; \mathcal{Y}_n) = \frac{1}{|\mathcal{Y}_+||\mathcal{Y}_-|} \sum_{y_+ \in \mathcal{Y}_+} \sum_{y_- \in \mathcal{Y}_-} \sigma\big(E_\theta(y_+) - E_\theta(y_-)\big) \times \big(\nabla_\theta E_\theta(y_+) - \nabla_\theta E_\theta(y_-)\big). \tag{9}$$

Proof. The single-pair loss is $L_{pair} = \text{softplus}(E_\theta(y_+) - E_\theta(y_-))$. Since $\frac{d}{dz}\text{softplus}(z) = \sigma(z)$, the chain rule yields the stated expression. □

Remark F.1 (Pairwise Bradley-Terry Loss as Negative Log-Likelihood)**.** *Setting $\pi_y = \exp(-E_\theta(y))$, the Bradley-Terry preference $P(y_+ \succ y_-) = \frac{\pi_{y_+}}{\pi_{y_+} + \pi_{y_-}} = \sigma(E_\theta(y_-) - E_\theta(y_+))$. The negative log-likelihood of this preference equals the per-pair loss in Eq. (6). Thus, minimizing the pairwise loss is equivalent to maximum likelihood estimation under the Bradley-Terry preference model with energy-defined scores.*

G Implementation Details

G.1 EORM Model Architecture (`TransEBM`)

Listing 1: PyTorch implementation of the TransEBM model.

```
import torch
import torch.nn as nn

class TransEBM(nn.Module):
    """Lightweight Transformer-based EBM
        ."""
    def __init__(self, vocab_size,
        dim_model,
                 n_heads, n_layers,
                     dropout):
        super().__init__()
        self.dim_model = dim_model
        self.emb = nn.Embedding(
            vocab_size, dim_model)
        encoder_layer = nn.
            TransformerEncoderLayer(
            d_model=dim_model,
            nhead=n_heads,
            dim_feedforward=4 *
                dim_model,
            activation="gelu",
            dropout=dropout,
            batch_first=True,
            norm_first=True,
        )
        self.enc = nn.TransformerEncoder
            (
            encoder_layer, num_layers=
                n_layers)
        self.head = nn.Sequential(
            nn.LayerNorm(dim_model),
            nn.Linear(dim_model,
                dim_model),
            nn.GELU(),
            nn.Linear(dim_model, 1),
        )

    def forward(self, ids, mask):
        x = self.emb(ids) * (self.
            dim_model ** 0.5)
        padding_mask = (mask == 0)
        x = self.enc(x,
            src_key_padding_mask=
            padding_mask)
        cls = x[:, 0]
        return self.head(cls).squeeze
            (-1)
```

G.2 Pairwise Bradley-Terry Loss

Listing 2: PyTorch implementation of the Pairwise Bradley-Terry loss.

```
import torch
import torch.nn.functional as F

def bradley_terry_loss(energies, labels)
    :
    pos = torch.where(labels == 1.0)[0]
    neg = torch.where(labels == 0.0)[0]
    if len(pos) == 0 or len(neg) == 0:
        return None
    diffs = energies[pos].unsqueeze(1) \
            - energies[neg].unsqueeze(0)
    return F.softplus(diffs).mean()
```

G.3 Hyperparameter Settings

H Dataset Details

This section outlines the dataset utilized for training and evaluating our EORM model. The core of our dataset comprises (question, Chain-of-Thought solution, correctness label) triplets, generated by employing a suite of LLMs to solve problems sourced from established mathematical reasoning benchmarks, principally the training splits of GSM8k (Cobbe et al., 2021a) and MATH (Hendrycks et al., 2021).

H.1 Data Format

Each instance in our dataset is represented as a JSON object, with multiple instances typically stored in a JSON Lines (`.jsonl`) file. An illustrative example of a single data instance is presented below.

Table 10: **Key hyperparameters for EORM model architecture and training.**

Category	Hyperparameter	Value
Model	Embedding Dim.	4096
	Encoder Layers	2
	Attention Heads	4
	Dropout	0.2
	FFN Dim.	$4 \times$ d_model
	Activation	GELU
	Normalization	Pre-LN
Tokenizer	Base Tokenizer	GPT-2
	Max Seq. Length	4096
	CLS ID	BOS token
	PAD ID	EOS token
Training	Optimizer	AdamW
	Learning Rate	1×10^{-4}
	Weight Decay	0.01
	LR Scheduler	Cosine + Warmup
	Warmup Ratio	0.2
	Epochs	50
	Grad Clip	1.0
	Mixed Precision	FP16

Training Dataset Format Example

label: 1

question: How many vertical asymptotes does the graph of $y = \frac{2}{x^2+x-6}$ have?

gen_text: To determine the number of vertical asymptotes for the function $y = \frac{2}{x^2+x-6}$, we need to identify the values of x that make the denominator zero.

1. *Factor the denominator*: $x^2 + x - 6 = (x + 3)(x - 2)$.
2. *Solve for zeros*: $(x + 3)(x - 2) = 0 \Rightarrow x = -3$ or $x = 2$.
3. *Check for holes*: The numerator is the constant 2, which is non-zero at $x = -3$ and $x = 2$, so no factors cancel.

Therefore, the graph has $\boxed{2}$ vertical asymptotes.

H.2 Comparison Between Other Datasets

Many existing datasets leverage powerful LLMs like GPT-4 or GPT-3.5 to synthesize novel mathematical problems. In contrast, our approach does not involve the synthesis of new problems: we generate a large corpus of CoT solutions by prompting existing LLMs with problems from open benchmarks (GSM8k and MATH), and each solution is labeled for correctness based on its final answer. The resulting corpus of ~ 14.96M triplets provides a diverse basis for training a robust verifier.

Table 11: **Comparison of the EORM training data scale and problem synthesis approach with other mathematical reasoning datasets.** Dataset sizes are approximate, in thousands (k) of samples.

Dataset	Synthesis Agent	Size (k)
WizardMath	GPT-4	96
MetaMathQA	GPT-3.5	395
MMIQC	GPT-4 + Human	2294
Orca-Math	GPT-4	200
Xwin-Math-V1.1	GPT-4	1440
KPMath-Plus	GPT-4	1576
MathScaleQA	GPT-3.5 + Human	2021
DART-Math	DeepSeekMath-7B-RL	591
EORM (Ours)	**None**	**14958**

I Hardware Resources

For CoT candidate generation we used NVIDIA A800 (40GB), A100 (40GB/80GB), RTX A6000 Ada (48GB), and H100 (80GB) GPUs. The training of EORM itself was conducted on 2–4 NVIDIA H100 (80GB) GPUs; with 4 H100s, end-to-end training takes approximately 36 hours.

J Use of Large Language Models

LLMs play two roles in this work. First, they are the object of study: our framework evaluates and improves the reasoning outputs of base models such as Llama 3, Mistral, and DeepSeekMath. Second, they served as a tool to generate the large-scale dataset of CoT solutions used for training EORM. For manuscript preparation, LLM use was strictly limited to language polishing and figure generation. All research design, theoretical analysis, and experimental work were carried out by the authors.