RATIONALYST: Pre-training Process-Supervision for Improving Reasoning

Dongwei Jiang[♠] **Guoxuan Wang**[♠] **Yining Lu**[♢] **Andrew Wang**[♠]
Jingyu Zhang[♠] **Chuyu Liu**[♠] **Benjamin Van Durme**[♠] **Daniel Khashabi**[♠]
[♠]Johns Hopkins University, [♢]University of Notre Dame
djiang21@jhu.edu

Abstract

The reasoning steps generated by LLMs might be incomplete, as they mimic logical leaps common in everyday communication found in their pre-training data: underlying rationales are frequently left *implicit* (unstated). To address this challenge, we introduce RATIONALYST, a model for process-supervision of reasoning based on pre-training on a vast collection of rationale annotations extracted from unlabeled data. We extract 79k rationales from web-scale unlabelled dataset (the Pile) and a combination of reasoning datasets with minimal human intervention. This web-scale pre-training for reasoning allows RATIONALYST to consistently generalize across diverse reasoning tasks, including mathematical, commonsense, scientific, and logical reasoning. Fine-tuned from LLaMa-3-8B-Instruct, RATIONALYST improves the accuracy of reasoning by an average of 3.9% on 7 representative reasoning benchmarks. It also demonstrates superior performance compared to significantly larger verifiers like GPT-4 and similarly sized models fine-tuned on matching training sets. [1]

1 Introduction

Rationales play a crucial role in human reasoning and its accuracy (Rips, 1994; Mercier and Sperber, 2011). In reasoning problems, having accurate rationales often correlates with accurate outcomes (Tversky et al., 1982; Davis, 1984). This importance of rationales extends to Large Language Models (LLMs) as well. Wei et al. (2022) were among the first to show that generating chain-of-thought rationales significantly improves LLMs' reasoning performance. Subsequent research has further refined the methods for eliciting rationales, leading to improved performance (Fu et al., 2023; Zhou et al., 2022).

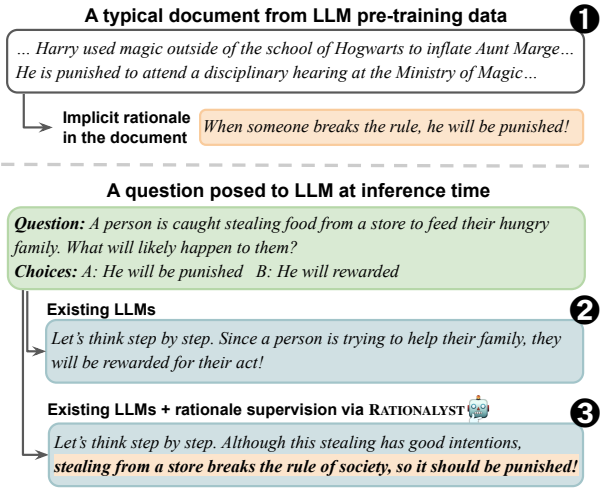

Figure 1: **A simplified example showing how implicit rationales in pre-training data can be leveraged to improve reasoning.** ①: Implicit rationales (unstated logical connections) occur frequently in LLM pre-training data. ②: Existing LLMs pre-trained to replicate their pretraining data tend to omit these logical steps as well. ③: However, RATIONALYST learns to generate these rationales at inference time to supervise the chain-of-thought process for better reasoning.

In the context of LLM reasoning, these rationales are typically employed through a chain-of-thought process that makes reasoning *explicit* by articulating them as plain-text rationales. In this approach, each subsequent rationale is generated conditioned on rationales produced in preceding steps, effectively using them as a form of supervision. However, the generated reasoning chains might be incomplete, containing potential logical leaps while leaving some rationales *implicit* (or hidden) during the generation process. These gaps in the reasoning chain can weaken the LLM's reasoning ability throughout the problem-solving process.

One reason why chain-of-thought methods might miss implicit steps is that models trained with "next-token prediction" often replicate the omissions present in their training data. Implicit rationales–underlying logical connections that are often not explicitly stated–are frequently missing

[1]Our code, data, and model can be found at this repository: https://anonymous.4open.science/r/RATIONALYST-B5CD

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26547–26566

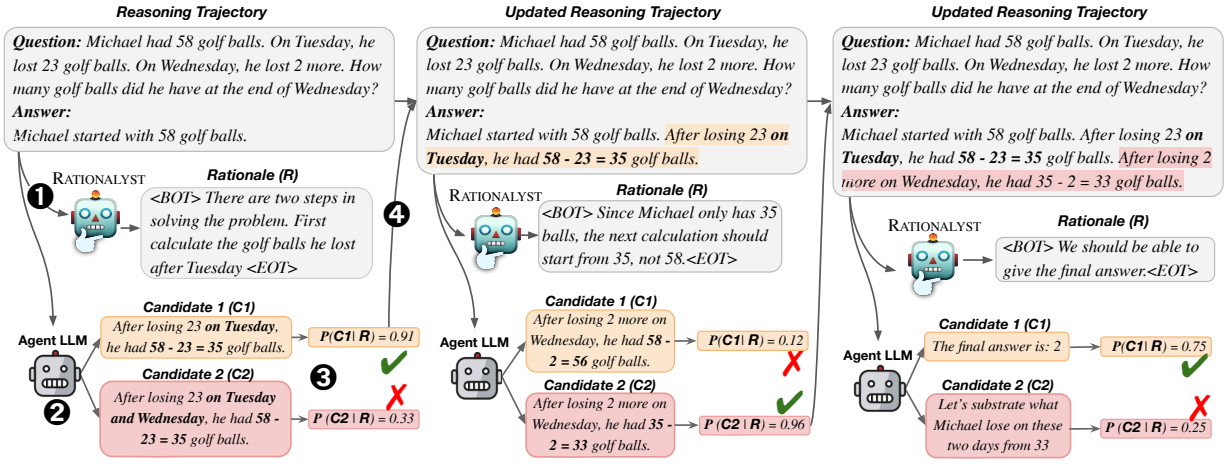

Figure 2: **An example showing how RATIONALYST works at inference time.** RATIONALYST generates implicit rationales given the current reasoning trajectory, which includes both the question and the reasoning steps generated so far ①. Agent LLM generates multiple next-step candidates for reasoning, also based on the current reasoning trajectory ②. Implicit rationale generated by RATIONALYST is used to provide heuristics for choosing the next step candidates proposed by the agent LLM by estimating the probability of the next step candidate given the rationale ③. The reasoning trajectory is updated iteratively with the highest scoring next step candidate ④.

in daily communication and web text. Figure 1 ① illustrates this concept using a typical document from LLM pre-training data. In this example, we see a passage from Harry Potter: "Harry used magic outside... He is punished to attend..." The text contains the implicit (unstated) rationale: *"When someone breaks the rule, he will be punished!"* This implicit rationale is crucial in inferring the causal reasoning that connects the cause (*Harry breaking rules*) to its effect (*punishment*), but is also left unstated in the context. As a result, existing LLMs trained to mimic web text will have difficulty surfacing these implicit statements during the reasoning process, which can lead to flawed conclusions, such as erroneously justifying theft as a praiseworthy act when done to support one's family (② in Figure 1).

This paper presents RATIONALYST, a model tailored for process-supervision of reasoning. RATIONALYST is trained on a vast collection of implicit rationales extracted from a mixture of web-scale unlabeled datasets and existing reasoning datasets. Although existing LLMs may miss crucial details in their reasoning, leading to flawed conclusions (② in Figure 1), using RATIONALYST provides additional supervision mechanism to guide their reasoning processes, resulting in more robust conclusions (③ in Figure 1).

RATIONALYST is developed and used in three stages: (1) we employ LLMs to extract implicit rationales from unlabeled text corpora without human annotation. These rationales are subsequently filtered based on their helpfulness in predicting subsequent text (§3.1); (2) we train RATIONALYST to predict those rationales given the preceding context (§3.2); and then (3) as depicted in Figure 2, during inference, we assume reasoning is done incrementally in a chain-of-thought fashion (Wei et al., 2022) by another agent model, and we use RATIONALYST to provide supervision for the agent model at each reasoning step throughout the reasoning process §3.3. By adopting a data-centric approach, RATIONALYST utilizes abundant unlabelled data to provide process supervision (Lightman et al., 2023) across various reasoning tasks without the need for human annotation.

Our method extracts 65k implicit rationales from the web-scale unlabelled dataset The Pile (Gao et al., 2020). To adapt the extracted rationales to our tested domain and stabilize training, we additionally extract a much smaller set of 14k implicit rationales from the question-answer pairs in the training sets of two reasoning datasets: GSM8K (Cobbe et al., 2021a) and ECQA (Aggarwal et al., 2021). Our extraction process controls for answer leakage to prevent artificial amplification of performance. Using this curated set of rationales, RATIONALYST is then fine-tuned from LLaMa-3-8B. To assess the effectiveness of our approach, we evaluate RATIONALYST on a diverse set of reasoning tasks, including mathematical, commonsense, scientific, and logical reasoning. Our results show that RATIONALYST improves the accuracy of reasoning by an average of 3.9% (§5.1). To understand the

contribution of different data sources, we conduct an ablation study that demonstrates the utility of rationales from both the large-scale Pile dataset and the smaller, specialized reasoning datasets (§5.2). Notably, RATIONALYST exhibits superior performance when compared to strong general-purpose verifiers like GPT-4 and similar capacity models specifically fine-tuned on matching training sets (§5.4).

Implicit rationales generated by RATIONALYST are also designed to provide supervision in a human-readable form, offering improved interpretability for LLM generation. This added interpretability is particularly beneficial when reasoning over complex domains such as mathematics or coding, where the step-by-step logic can be difficult for humans to follow without explicit explanations. As shown in §5.5, our model is capable of generating human-understandable rationales for unseen data from complex math reasoning.

2 Related Work

Supervising reasoning. Supervision-based approaches have been shown to enhance the reasoning abilities of LLMs. Cobbe et al. (2021b) and Snell et al. (2024) demonstrate that training a "verifier" to supervise reasoning can be more parameter-efficient than simply expanding the parameters of the "reasoner" responsible for solving the reasoning task. Ground-truth feedback from interaction with the environment is an effective form of supervision (Wang et al., 2023), but it works only in controlled environments like simulated world. General-purpose verifiers (Dhuliawala et al., 2023; Weir et al., 2024, 2023; Vacareanu et al., 2024) offer broader applicability utilizing principles like compositional reasoning. However, they don't fully capitalize on the vast amount of unlabelled data in the way a data-driven approach might. Process-based supervision (Lightman et al., 2023) offers supervision at each reasoning step rather than just at the final result. While promising, it requires substantial human annotation for the correctness of intermediate steps, making it resource-intensive. Our work aims to address these challenges by proposing a data-centric process-supervision method without the need for human annotation.

Knowledge extraction from unlabelled data. LLMs are conventionally trained on extensive web data using autoregressive next-token prediction. While effective, this approach may not fully harness the potential of the pre-training data, as latent information within this data could be better accessed using techniques beyond simple next-token prediction. Recent research has demonstrated several approaches to utilize this latent information to develop more sophisticated language model capabilities (Jiang et al., 2024c). Schick et al. (2023) introduced Toolformer, which autonomously annotates and extracts appropriate positions, names, and inputs for tool use by leveraging supervision from future tokens. Similarly, Cornille et al. (2024) developed a method for learning to plan coherent article writing through self-supervised learning in text. More closely related to our work, Zelikman et al. (2024) proposed Quiet-Star, which applied a comparable technique to uncover underlying rationales in daily communication to enhance reasoning capabilities. Our work adopts a strategy similar to Quiet-Star for extracting rationales in an unsupervised manner. However, our approach diverges in its primary objective: we aim to train a "supervisor" that can utilize these rationales to provide process supervision for any "reasoner." This focus enables us to implement a simpler and more reliable method, as we don't need to directly integrate rationale extraction with "reasoner" training. Our approach thus offers a novel perspective on leveraging latent information in language models to enhance their capabilities.

Rationales as the basis for reasoning. Various studies have focused on improving the use of rationales to elicit reasoning. Fu et al. (2023) refine rationales for more effective reasoning elicitation, while Li et al. (2023) explore different approaches to leveraging rationales to enhance reasoning. Other works, such as Hwang et al. (2024), examine the verification of rationales produced by LLMs during reasoning to improve performance. Additionally, training LLMs on rationale-rich data is a common strategy for enhancing reasoning skills. As highlighted by Lewkowycz et al. (2022) and Jiang et al. (2024a), LLMs trained on science and math data tend to perform better on reasoning tasks, particularly when CoT prompting is used. In this work, we build on this foundation by using rationales as the core of our method to supervise reasoning.

3 Building RATIONALYST

We discuss the construction of RATIONALYST and its usage at inference time. First, we describe ex-

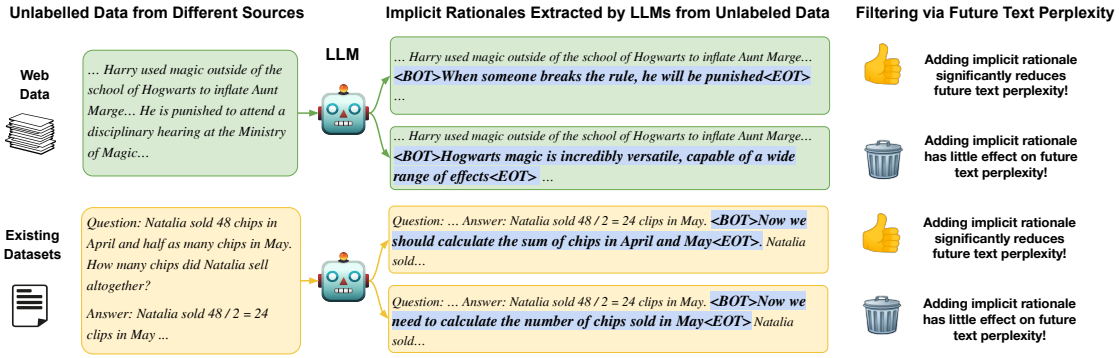

Figure 3: We use LLMs to extract implicit rationales (enclosed by <BOT> and <EOT> in bold) that capture reasoning in unlabelled text (§3.1 Ⓐ and Ⓑ). The sample at the top is taken from unlabelled web-scale pre-training datasets The Pile and the sample at the bottom is taken from existing datasets (GSM8K). These rationales are subsequently filtered based on whether they are useful for predicting future text (§3.1 Ⓒ).

tracting rationales from unlabeled text (§3.1), then use them to train RATIONALYST (§3.2), and finally, employ RATIONALYST to supervise reasoning during inference (§3.3).

Setup. As we will be using multiple LLMs throughout the process, we define them here: M_{Ra} is the trained rationale generation model (RATIONALYST) that generates rationales and heuristics during inference. M_{Agent} is a general-purpose reasoning agent that produces candidate reasoning steps and incorporates rationales during inference. We use one additional model M for initial rationale extraction, rationale filtration, and probability estimation of potential next reasoning steps during inference. These LLMs can be implemented using various state-of-the-art models, allowing for adaptability to specific research needs and computational resources.

3.1 Large-scale Rationale Extraction

Implicit rationales are often embedded in unlabelled text, reflecting natural thought processes in daily communication. Our extraction process, illustrated in Figure 3, aims to make these rationales explicit. Using an aligned language model M, we generate rationales from text and then use M to filter these rationales to retain only those that are useful, akin to the self-supervised "tool" learning approach described by Schick et al. (2023). The same M is subsequently used to train M_{Ra}.

Ⓐ Extracting rationales from pre-training data. We employ M to generate rationales from the Pile. Due to the size of this dataset, we implement a pre-filtering process to identify reasoning-rich documents by (1) computing the average semantic embedding of representative reasoning training sets using a paragraph embedding model, and (2) selecting documents from unlabelled datasets that exceed a cosine similarity threshold α when compared to this average embedding. After pre-filtering, we segment the selected paragraphs into 2000-word segments and instruct M to generate rationales at the end of each sentence, using prompts with demonstrations. Detailed information on the prompts and in-context learning demonstrations used for rationale extraction can be found in Appendix B.

Ⓑ Extracting rationales from reasoning datasets. In parallel to Ⓐ, we also extract rationales from existing reasoning datasets to adapt the extracted rationales to our tested domain and stabilize training. For a given reasoning dataset with pairs of questions and final answers $D = \{(q_i, a_i)\}_{i=1}^{m}$, we create a prompt P that instructs M to generate rationales for each reasoning step in the final answer a_i. The input of the prompt consists of the entire question and answer, and the output includes implicit rationales that can be inferred from the reasoning process in the answer. Consider the concrete example from existing datasets (bottom) in Figure 3. The solution involves two reasoning steps: "Natalia sold 48 / 2 = 24 clips in May" and "Natalia sold 48 + 24 = 72 chips altogether." Here, the implicit rationale that connects the first and second steps, *"Now we should calculate the sum of chips in April and May,"* is implicit yet helpful for the prediction of the second step. These rationales are subsequently filtered and used to train RATIONALYST.

Ⓒ Filtering extracted rationales. Generated rationales in Ⓐ and Ⓑ may not always be accurate or helpful. In reasoning tasks, our objective is for the extracted rationales to effectively aid in future

reasoning, which means a good rationale should enhance the likelihood of accurately predicting the following text. Let i be the position of the rationale r in the sequence $\mathbf{x} = x_1, \ldots, x_n$. Given a sequence of weights $(w_k)_{k \in \mathbb{N}}$, the cross-entropy loss for future token prediction is defined as:

$$L_i(\mathbf{r}) = -\sum_{j=i}^{n} w_{j-i} \cdot \log p_M(x_j \mid \mathbf{r}, x_{1:j-1}),$$

where M, in a different role from its previous use, is employed to estimate the probability over tokens $x_i, \ldots, x_n$ prefixed by preceding tokens $x_{1:i-1}$ and rationale $\mathbf{r}$. The weight assigned to each future token decreases exponentially by a factor of 0.9 for each step further away it is from the rationale. We compute $L_i = L_i(r_i) - L_i(\varepsilon)$, where ε represents an empty rationale (i.e. predicting following tokens based only on preceding tokens). A rationale is considered helpful if it makes the prediction of future tokens easier, indicated by $L_i \geq \tau_f$, where τ_f is a filtering threshold. We retain rationales for which adding the rationale *reduces* the loss by at least τ_f compared to having no rationale.

It's crucial to clarify two key aspects of our rationale extraction process. First, while M extracts rationales from the training sets of reasoning datasets, these training sets are not directly used as targets when training M_{Ra}. Instead, the target of training is the rationale generated by M. Second, we explicitly instruct M to exclude answers from the extracted rationales. This precaution prevents answer leakage in our prompts.

3.2 RATIONALYST Training

The goal of RATIONALYST (denoted by M_{Ra}) training is to develop a model that can generate implicit rationales to guide stepwise problem-solving during inference time. For web-scale datasets like The Pile, the input context consists of a segment of text from a document. M_{Ra} learns to generate an implicit rationale that can guide the prediction of the next segment of text in the document's flow. In the case of structured reasoning datasets such as GSM8K or ECQA, the input context includes the question and any preceding reasoning steps toward the answer. Here, M_{Ra} learns to generate rationales that could guide the next step in the problem-solving sequence.

Given the appropriate context from either source, the implicit rationales, extracted and filtered as described in §3.1, serve as the target outputs during training. The overall training objective is to minimize the per-token cross-entropy loss between the generated rationales and their ground truth values from the extracted and filtered rationales.

3.3 Inference with the Help of RATIONALYST

During inference, any general-purpose LLM (the "agent model" or M_{Agent}) can be employed for reasoning across various problems. Algorithm 1 outlines the procedure.

M_{Agent} generates reasoning incrementally in a chain-of-thought fashion, producing multiple candidates for the next reasoning step. These steps and the question form a "reasoning trajectory" T that aims to solve the problem, which also serves as input to M_{Ra}. M_{Ra} then generates r, the implicit rationale (line 3) With the help of implicit rationale, we provide supervision for the next reasoning step. Two supervision methods we considered are:

Implicit supervision. For this supervision, M_{Agent} generates the next reasoning steps conditioned on the trajectory T (line 6). We then use M to estimate the probability of potential next reasoning steps given rationale r and reasoning trajectory T (line 13). This probability-based heuristic aligns with our rationale filtration process used during M_{Ra} training: just as we identified rationales that improved the prediction of future text during filtration, here we use rationales to improve the selection of future reasoning steps. By leveraging the probability estimates as a heuristic, we can discriminate between more and less likely next steps in the reasoning process, guiding the overall trajectory towards more accurate conclusions.

Explicit supervision. Another approach is to directly incorporate the implicit rationale into the generation of the next reasoning steps. This method makes the previously implicit rationale an explicit part of the reasoning process. To do that, we ask M_{Agent} to generate multiple candidate next steps by temporarily appending r to the trajectory T, and then producing potential continuations based on this augmented context (line 8). Then, we estimate the probability of candidate generations according to M_{Agent} (line 15). This approach allows M_{Agent} to make the final decision on the next reasoning step, as in normal beam search (Snell et al., 2024; Yao et al., 2023), while benefiting from the additional context provided by M_{Ra}'s rationales.

Algorithm 1 Inference with RATIONALYST

Input: Question q, RATIONALYST M_{Ra}, Agent model M_{Agent}, Probability estimation model M;
Functions: Heuristic function $H(M_{\text{Ra}}, q, T)$, stopping condition $stop\_condition()$
Hyperparameters: Sampling temperature t and number of sampled rationales N

1: $T \leftarrow q$ ▷*Initialize reasoning trajectory as the question.*
2: **repeat**
3: $r \leftarrow M_{\text{Ra}}(T)$ ▷*Generate rationale given trajectory.*
4: $heuristic\_list = \emptyset$
5: **if** $supervision == implicit$ **then**
6: $next\_steps \leftarrow M_{\text{Agent}}(T)$
7: **else if** $supervision == explicit$ **then**
8: $next\_steps \leftarrow M_{\text{Agent}}(T, r)$
9: **end if**
10: **for** $n = 1 \ldots N$ **do**
11: $x \leftarrow next\_steps[n]$ ▷*Take next step generation.*
12: **if** $supervision == implicit$ **then**
13: $h \leftarrow M(x|T, r)$ ▷*Estimate probability of next reasoning step.*
14: **else if** $supervision == explicit$ **then**
15: $h \leftarrow M_{\text{Agent}}(x|T, r)$
16: **end if**
17: $heuristic\_list.append(h)$
18: **end for**
19: $m\_idx \leftarrow argmax(heuristic\_list)$
20: $T \leftarrow T + next\_steps[m\_idx]$ ▷*Extend trajectory with the highest scoring step.*
21: **until** $stop\_condition(T)$ ▷*E.g., the trajectory contains strings like "The final answer is:"*
22: **return** T

After providing heuristics for the next reasoning steps, the step with the highest heuristic (line 19) is selected. The reasoning trajectory is then extended with this highest-scoring step (line 20). The reasoning process concludes when the stop condition is satisfied (line 21), which varies by dataset and often includes cues like "The final answer is:" that can be specified in system prompts for M_{Agent} for different tasks.

The computational cost of M_{Ra} is comparable to a normal beam search, with the only additional cost being the generation of rationales, which are typically quite short.

4 Experimental Setup

4.1 Setup for Training RATIONALYST

Rationale extraction. As discussed in §3.1 Ⓐ, we perform pre-filtering on The Pile, an unlabelled web-scale dataset, to identify documents with extensive reasoning content before rationale extraction. This is achieved by computing the average semantic embedding from the training sets of the reasoning datasets we test, filtering documents that exceed the cosine similarity threshold α of 0.3, and keeping only the documents with length under 2000 tokens to fit within LLaMa-3 models' context length. The model we used to calculate these embeddings is `MPNet-base` (Song et al., 2020).

Following the recipe in §3.1 Ⓑ, we also extract rationales from existing reasoning datasets. GSM8K (Cobbe et al., 2021b) and ECQA (Aggarwal et al., 2021) were selected for their complementary coverage of mathematical and commonsense reasoning, respectively. This combination ensures RATIONALYST is trained on diverse reasoning patterns, enhancing its versatility across various tasks.

Rationale annotation and filtration. The model M used for rationale extraction and rationale filtering are both LLama-3-8B-Instruct (MetaAI, 2024). On GSM8K and ECQA, we manually annotated 100 pairs of {preceding_context, rationale, following_context} to determine an appropriate filtration threshold. The annotations include 50 positive and 50 negative rationale examples. Since it's straightforward to scale up the extraction of rationales from unlabelled data for filtration, we prioritize maximizing the precision of our filtered rationales, even if it means extracting fewer of them. We set the threshold τ_f to ensure that 95% of the filtered rationales are accurate. On The Pile, we do not perform rationale annotation due to its diverse composition of corpora with varying characteristics. So the filter threshold τ_f for the Pile is set to 0 for all of its subdomains. The results of rationale extraction and filtration on GSM8K, ECQA, and The Pile are presented in Table 1. From our extraction and filtration process, we obtained approximately 14k rationales from GSM8K and ECQA combined, and about 65k from The Pile after filtration. Detailed statistics of rationale extraction and filtration results for each dataset, including the number of rationales per document and filtration rates, can be found in Appendix H.

Dataset	Subdomains	# Docs.	# Rationales	Rationales Left (%)	τ_f
GSM8K	N/A	7473	17566	19.5	1.2
ECQA	N/A	7600	19669	57.6	0.5
The Pile	Pile-CC	266.6K	853.2K	2.9	0
	StackExchange	21.8K	113.6K	29.8	0
	Github	19.9K	45.8K	2.6	0
	HackerNews	5.8K	24.4K	9.4	0
	PubMed Central	4.9K	18.6K	3.2	0
	Wikipedia (en)	4.2K	23.0K	7.8	0

Table 1: **The statistics on rationale sampling and filtration.** We provide the total number of documents and rationales before filtering, and the percentage of leftover rationales after filtering.

RATIONALYST training. RATIONALYST is fine-tuned with LLaMa-3-8B-Instruct as the base model. We use the default hyperparameters as specified in the LLaMa-3 technical report (MetaAI, 2024) for

fine-tuning. After training, we conducted manual annotation of the model's output and found that the accuracy of rationales generated on unseen test data closely matches the filtration accuracy we specified for training data through our filtration parameters.

4.2 Setup for Evaluating RATIONALYST

We evaluate RATIONALYST across diverse reasoning tasks including mathematical (GSM8K (Cobbe et al., 2021b), MATH (Hendrycks et al., 2021), commonsense (ECQA(Aggarwal et al., 2021) and HellaSwag (Zellers et al., 2019)), logical (ProofWriter (Tafjord et al., 2021)), scientific (ARC (Clark et al., 2018)), and multi-task reasoning (MMLU-Pro (Wang et al., 2024b)). For inference, we use LLaMa-3-8B-Instruct as our base agent model with temperature 0.7 and top-k sampling of 3. We compare RATIONALYST against both process supervision approaches (using LLaMa-3-8B-Instruct and GPT-4) and outcome-based verifiers. Complete experimental details are provided in Appendix A.

4.3 Setup for other verifiers.

To evaluate the effectiveness of RATIONALYST's process supervision, we compare it with other approaches. For *process supervision* with other models, we include LLaMa-3-8B-Instruct and GPT-4 in our comparison. These models are prompted to rerank partial reasoning trajectories as reasoning steps are generated. The prompts and in-context learning demonstrations used for these models on representative datasets are provided in Appendix E. For *outcome supervision*, we also compare with outcome-based verifiers derived from LLaMa-3-8B-Instruct. These verifiers are fine-tuned on the training sets of each reasoning dataset. Following the approach outlined by Cobbe et al. (2021b), they assess the correctness of the final prediction by directly evaluating the question and final solution. This comparison allows us to assess the performance of RATIONALYST against both process-based and outcome-based supervision methods.

5 Empirical Results

5.1 Main result: RATIONALYST Improves Performance on Various Tasks

In this section, we train RATIONALYST using a combination of rationales extracted from GSM8K and ECQA, as well as from The Pile, as outlined in Table 1. The baseline does not use any verifier. We use implicit supervision for this experiment. The main result is shown in Table 2.

Reasoning Type	Dataset	Baseline Acc.	RATIONALYST Acc.	RATIONALYST Δ Acc.
Mathematical	GSM8K	77.6	**81.6**	**4.0**
	Math	28.0	**32.5**	**4.5**
CommonSense	ECQA	72.6	**75.2**	**2.6**
	HellaSwag	58.2	**60.3**	**2.1**
Logical	ProofWriter	86.4	**90.7**	**4.3**
Scientific	ARC	77.6	**80.7**	**3.1**
Combined	MMLU-Pro	39.6	**45.3**	**5.7**

Table 2: Accuracy and absolute improvement over baseline using RATIONALYST. **RATIONALYST generalizes improved performance across different reasoning tasks.**

Evaluation of RATIONALYST shows that training with rationales from GSM8K, ECQA, and The Pile improves performance not only on GSM8K and ECQA, but also on other reasoning tasks (e.g. scientific reasoning, logical reasoning, etc) not directly used in rationale extraction. This supports the idea that rationales can be broadly applicable across different reasoning tasks. In addition, since we use the same model (LLaMa-3-8B-Instruct) for rationale extraction, filtering, RATIONALYST training, and inference, our results do not leverage external knowledge from stronger models like LLaMa-3-70B-Instruct or GPT-4. Future work might change M to stronger models, with the expectation that higher-quality rationales will lead to better performance.

5.2 Ablation: Web-scale Rationales Enhance Performance Across Tasks

To assess the benefit of web-scale rationales, we train another model: RATIONALYST w/o Pile solely on rationales extracted from the training sets of GSM8K and ECQA. We re-ran the experiments on the same reasoning datasets using implicit supervision. The results are detailed in Table 3.

We find that training the model on web-scale data results in better performance compared to training only on the rationales extracted from GSM8K and ECQA. This improvement is consistent and particularly significant on MMLU-Pro. Web-scale data likely provides exposure to more diverse reasoning types and content, including specialized knowledge, complex real-world scenarios, and interdisciplinary connections not present in the more focused datasets.

Dataset	RATIONALYST	RATIONALYST (w/o Pile)	ΔAcc.
GSM8K	**81.6**	80.3	**-1.3**
Math	**32.5**	31.4	**-1.1**
ECQA	**75.2**	74.5	**-0.7**
HellaSwag	**60.3**	59.1	**-1.2**
ProofWriter	**90.7**	88.2	**-2.5**
ARC	**80.7**	78.8	**-1.9**
MMLU-Pro	**45.3**	41.2	**-4.1**

Table 3: An ablation study on the benefit of rationales extracted from pre-training data (The Pile). The consistent accuracy drop shows that, **utilizing web-scale rationales improves performance on various reasoning datasets.**

5.3 Ablation: Implicit Supervision Works Better than Explicit Supervision

In this section, we conduct ablation studies to test the effectiveness of different supervision methods. To isolate the impact of supervision methods and minimize confounding variables, we focus on GSM8K and ECQA as representative benchmarks for mathematical and commonsense reasoning, respectively. We train two versions of RATIONALYST: one on rationales extracted from the GSM8K training set (RATIONALYST - GSM8K) and another on rationales from the ECQA training set (RATIONALYST - ECQA). These models are used to supervise M_{Agent} during inference on their respective tasks.

As shown in Table 4, implicit supervision outperforms explicit supervision. Our manual analysis revealed that implicit supervision's superior performance stems from its greater robustness to errors. When RATIONALYST generates an imperfect rationale, the probability-based heuristic used in implicit supervision can still provide useful guidance even if the rationale itself is not ideal. This approach is less likely to lead M_{Agent} to produce incorrect next steps. In contrast, explicit supervision directly incorporates potentially flawed rationales into the reasoning process, which can cause M_{Agent} to produce incorrect next steps. Essentially, implicit supervision acts as a softer guide, allowing for some imperfection in rationales, while explicit supervision more strictly adheres to potentially flawed rationales, making it more susceptible to errors.

5.4 RATIONALYST Outperforms Other Verifiers

Table 5 presents an analysis of RATIONALYST against various verifiers. Our findings reveal several insights:

Heuristic↓ - Evaluation task→	GSM8K	ECQA
Implicit Supervision	**80.3**	**74.5**
Explicit Supervision	77.5	72.2

Table 4: Comparison of implicit and explicit supervision methods on GSM8K and ECQA tasks. **Implicit supervision outperforms explicit supervision due to its robustness to errors.**

RATIONALYST outperforms vanilla LLaMa-3-8B-Instruct using process supervision: RATIONALYST, even without leveraging The Pile dataset, outperforms process-based verifiers using vanilla LLaMa-3-8B-Instruct. A manual examination of reasoning trajectories suggests that LLaMa-3-8B-Instruct faces difficulties in reranking partial reasoning steps. This challenge likely stems from the model's struggle to differentiate among its own generated outputs, a phenomenon observed in recent studies (Jiang et al., 2024b; Huang et al., 2023).

RATIONALYST shows superior process-supervision performance than much bigger models like GPT-4: We observe consistent superior performance of RATIONALYST compared to GPT-4's process supervision. We hypothesize that this advantage arises from RATIONALYST's specialized design for providing supervision, in contrast to GPT-4's general-purpose training.

RATIONALYST surpasses outcome-based verifiers trained using matching data: Notably, our method surpasses the performance of fine-tuned outcome-based verifiers on both GSM8K and ECQA datasets, despite these verifiers being trained on matching data. We attribute this success to the richer feedback provided by process-based supervision compared to outcome-based approaches.

We would also like to mention that methods like self-consistency and other prompting techniques (e.g. STEP-BACK Prompting (Zheng et al., 2024)) are complementary to our approach. RATIONALYST can be integrated with these methods to potentially achieve even better results. For more detail, please refer to Appendix I.

5.5 RATIONALYST Generates Accurate and Easy-to-understand Rationals

We annotate some samples from the test set of Math (Hendrycks et al., 2021) at inference time, which was not part of the rationale sampling datasets. Through manual observation, we find that our

Supervision	GSM8K	ECQA
N/A	77.6	72.6
Process Supervision w/ LLaMa-3	77.4	71.5
Process Supervision w/ GPT-4	80.0	74.7
Outcome Supervision w/ LLaMa-3 + FT	79.2	74.3
RATIONALYST w/o Pile	80.3	74.5
RATIONALYST	**81.6**	**76.2**

Table 5: Comparison of different supervision methods. **RATIONALYST outperforms both strong verifiers like GPT-4 and similarly-sized models fine-tuned on matching training data.**

model can generate useful rationales that is helpful for understanding LLM's reasoning process on Math (an example is provided in Appendix D). Comparing the rationales generated by RATIONALYST with those generated by Quiet-Star (Zelikman et al., 2024) on the same problems, we find that our method produces more human-understandable rationales. We believe this happens because Quiet-Star optimizes rationales during training using the accuracy of the final prediction as a reward. This approach, while effective for improving task performance, does not explicitly prioritize human interpretability. In addition, this appraoch might inadvertently develop shortcuts or non-intuitive patterns that optimize for accuracy but not necessarily for clarity or human understanding.

6 Conclusion and Future Work

In this paper, we introduced RATIONALYST, a novel self-supervised model designed to enhance the reasoning capabilities of LLMs by leveraging hidden rationales extracted from unlabeled text. While our results demonstrate the effectiveness of this approach, there are several promising directions for future work, including scaling to larger models and datasets, integrating with test-time compute optimization techniques, and exploring the role of implicit rationales in model training. We discuss these future directions in detail in Appendix F. Through this work, we have demonstrated that extracting and utilizing implicit rationales from unlabeled text can meaningfully improve LLMs' reasoning abilities. We hope these insights will inspire further research into leveraging latent knowledge in pre-training data to enhance model capabilities.

7 Limitations

One limitation of this work is the comprehensiveness of our experiments. In future research, we plan to extend our experiments to a broader range of reasoning tasks and compare RATIONALYST with other outcome-based and process-based verifiers. We also plan to adjust the combination of rationales used to train RATIONALYST by (1) sampling from different reasoning tasks and (2) altering the mix of rationales in unlabelled web-scale pre-training data to better understand its generalizability. An additional limitation is that we could potentially improve performance through preference tuning (e.g., DPO) where the model learns to distinguish between valid and invalid rationales, but we have not explored this direction.

References

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for CommonsenseQA: New Dataset and Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*.

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. 2021a. Training verifiers to solve math word problems. *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.

Nathan Cornille, Marie-Francine Moens, and Florian Mai. 2024. Learning to plan for language modeling from unlabeled data. *Preprint*, arXiv:2404.00614.

Randall Davis. 1984. Diagnostic reasoning based on structure and behavior. *Artificial intelligence*, 24(1-3):347–410.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *Preprint*, arXiv:2309.11495.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *ICLR*. OpenReview.net.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *Preprint*, arXiv:2306.11644.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *NeurIPS*, Menlo Park, Calif. AAAI Press.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet.

Hyeonbin Hwang, Doyoung Kim, Seungone Kim, Seonghyeon Ye, and Minjoon Seo. 2024. Self-explore to avoid the pit: Improving the reasoning capabilities of language models with fine-grained rewards. *Preprint*, arXiv:2404.10346.

Dongwei Jiang, Marcio Fonseca, and Shay B. Cohen. 2024a. Leanreasoner: Boosting complex logical reasoning with lean. *Preprint*, arXiv:2403.13312.

Dongwei Jiang, Jingyu Zhang, Orion Weller, Nathaniel Weir, Benjamin Van Durme, and Daniel Khashabi. 2024b. Self-[in]correct: Llms struggle with refining self-generated responses. *Preprint*, arXiv:2404.04298.

Zhengping Jiang, Yining Lu, Hanjie Chen, Daniel Khashabi, Benjamin Van Durme, and Anqi Liu. 2024c. RORA: Robust free-text rationale evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1070–1087, Bangkok, Thailand. Association for Computational Linguistics.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *NeurIPS*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. Making large language models better reasoners with step-aware verifier. *Preprint*, arXiv:2206.02336.

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

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

Hugo Mercier and Dan Sperber. 2011. Why do humans reason? arguments for an argumentative theory. *Behavioral and brain sciences*, 34(2):57–74.

MetaAI. 2024. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. 2023. Openwebmath: An open dataset of high-quality mathematical web text. *Preprint*, arXiv:2310.06786.

Lance J Rips. 1994. *The psychology of proof: Deductive reasoning in human thinking*. Mit Press.

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*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Preprint*, arXiv:2004.09297.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *ACL*.

Amos Tversky, Daniel Kahneman, and Paul Slovic. 1982. *Judgment under uncertainty: Heuristics and biases*. Cambridge.

Robert Vacareanu, Anurag Pratik, Evangelia Spiliopoulou, Zheng Qi, Giovanni Paolini, Neha Anna John, Jie Ma, Yassine Benajiba, and Miguel Ballesteros. 2024. General purpose verification for chain of thought prompting. *Preprint*, arXiv:2405.00204.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *Preprint*, arXiv:2305.16291.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. 2024a. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *Preprint*, arXiv:2312.08935.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

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* (NeurIPS), 35:24824–24837.

Nathaniel Weir, Peter Clark, and Benjamin Van Durme. 2023. Nellie: A neuro-symbolic inference engine for grounded, compositional, and explainable reasoning. *Preprint*, arXiv:2209.07662.

Nathaniel Weir, Kate Sanders, Orion Weller, Shreya Sharma, Dongwei Jiang, Zhengping Jiang, Bhavana Dalvi Mishra, Oyvind Tafjord, Peter Jansen, Peter Clark, and Benjamin Van Durme. 2024. Enhancing systematic decompositional natural language inference using informal logic. *Preprint*, arXiv:2402.14798.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. An empirical analysis of compute-optimal inference for problem-solving with language models. *Preprint*, arXiv:2408.00724.

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. In *Advances in Neural Information Processing Systems* (NeurIPS).

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D. Goodman. 2024. Quiet-star: Language models can teach themselves to think before speaking. *Preprint*, arXiv:2403.09629.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *ACL*.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V Le, and Denny Zhou. 2024. Take a step back: Evoking reasoning via abstraction in large language models. *Preprint*, arXiv:2310.06117.

Fan Zhou, Haoyu Dong, Qian Liu, Zhoujun Cheng, Shi Han, and Dongmei Zhang. 2022. Reflection of thought: Inversely eliciting numerical reasoning in language models via solving linear systems. *Preprint*, arXiv:2210.05075.

A Evaluation Setup Details

A.1 Task Configuration and Metrics

We evaluate RATIONALYST on a diverse set of reasoning tasks, carefully selected to assess performance across different reasoning domains. Table 6 provides a comprehensive overview of our evaluation configuration.

Task	#Eval	#Shots	Reasoning Type
GSM8K	1319	8	Math
Math	5000	5	Math
ECQA	17944	6	CommonSense
HellaSwag	10000	4	CommonSense
ProofWriter	600	2	Logical
ARC	1172	4	Scientific
MMLU-Pro	12000	5	Mixed

Table 6: **Detailed configuration of evaluation tasks.** For each dataset, we report the training set size (#Train), evaluation set size (#Eval), and number of few-shot demonstrations (#Shots) used during evaluation. We use exact match accuracy across all tasks, following standard evaluation protocols.

For each dataset, we implement specific evaluation protocols:

- **GSM8K**: We evaluate on the standard test split, using 8-shot demonstrations that showcase step-by-step mathematical reasoning.
- **MATH**: Our evaluation encompasses all difficulty levels and mathematical topics in the dataset, utilizing 5-shot demonstrations calibrated for complexity.
- **ECQA**: We use the validation split due to evaluation server constraints for the test set, maintaining consistency with prior work.
- **HellaSwag**: Evaluation follows the original setup with 4-shot demonstrations focusing on commonsense inference.
- **ProofWriter**: We specifically evaluate on proofs with depth more than 5 to assess complex logical reasoning capabilities.
- **ARC**: Our evaluation focuses on the more challenging ARC-Challenge subset, employing 4-shot demonstrations.
- **MMLU-Pro**: Following the original paper's protocol, we evaluate across all reasoning categories using 5-shot demonstrations.

A.2 Inference Configuration

Our inference setup utilizes LLaMa-3-8B-Instruct as the base agent model (M_{Agent}). Key configuration parameters include:

Sampling Parameters. We employ temperature = 0.7 and top-k = 3 during inference. This configuration allows for diverse reasoning path exploration while maintaining coherence. The temperature setting represents a calculated trade-off between deterministic output (temperature = 0) and exploration, showing particular effectiveness on logical reasoning tasks like ProofWriter.

Chain-of-Thought Implementation. For each task category, we implement specialized chain-of-thought prompting:

- **Mathematical and Procedural Tasks**: Demonstrations break down solutions into atomic reasoning steps, with explicit intermediate calculations.
- **Multiple-Choice Tasks**: Prompts guide systematic analysis of each answer choice, encouraging comparative reasoning.
- **Logical Reasoning**: Step-by-step deductive reasoning chains are demonstrated, emphasizing premise utilization.

B Prompts used for rationale sampling

In this section, we provide the prompts we used for rationale sampling on GSM8K (Figure 4), ECQA (Figure 5), and The Pile (Figure 6).

C Prompts used during inference

In this section, we provide the prompts used during inference time to encourage the agent model reason step by step for GSM8K (Figure 7) and ECQA (Figure 8). Note that the input to the agent model appends the last rationale generated by the agent model.

D Examples of rationales generated at inference time

In this section, we provide rationales generated by RATIONALYST from the test set of MATH during inference time Figure 9, which was not part of the rationale sampling datasets, and observe that our model can still generate useful rationales that help to understand LLM's reasoning process.

E Prompts used for LLaMa-3 reranking

In this section, we provide the prompts and in-context-learning demonstrations used to instruct

System Prompt:
Your task is to add rationals to a piece of text. The rationals should help you with predicting future text. You can add rationals by writing "<BOT>rational<EOT>". Here are some examples of rationale generation:

Example Input 1:
Question: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?
Answer: Michael started with 58 golf balls. After losing 23 on tuesday, he had 58 - 23 = 35. After losing 2 more, he had 35 - 2 = 33 golf balls. The answer is 33

Example Output 1:
Question: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?
Answer: **<BOT>First, we need to calculate how many golf balls Michael had after losing 23 on tuesday<EOT>** Michael started with 58 golf balls. After losing 23 on tuesday, he had 58 - 23 = 35. **<BOT>After losing 2 more on wednesday, we need to calculate how many golf balls he had left<EOT>** After losing 2 more, he had 35 - 2 = 33 golf balls. **<BOT>We are ready to output the final answer<EOT>** The answer is 33

Example Input 2:
Question: Brennan was researching his school project and had to download files from the internet to his computer to use for reference. After downloading 800 files, he deleted 70% of them because they were not helpful. He downloaded 400 more files but again realized that 3/5 of them were irrelevant. How many valuable files was he left with after deleting the unrelated files he downloaded in the second round?
Answer: The number of non-valuable files Brennan downloaded in the first round is 70/100*800 = <<70/100*800=560>>560 files. The number of valuable files Brennan downloaded in the first round is 800-560 = <<800-560=240>>240 When he downloaded 400 new files, there were 3/5*400= <<3/5*400=240>>240 non-useful files, which he deleted again. The total number of valuable files he downloaded in the second round is 400-240 = <<400-240=160>>160 To write his research, Brennan had 160+240 = <<160+240=400>>400 useful files to reference to write his research. The answer is 400

Example Output 2:
Question: Brennan was researching his school project and had to download files from the internet to his computer to use for reference. After downloading 800 files, he deleted 70% of them because they were not helpful. He downloaded 400 more files but again realized that 3/5 of them were irrelevant. How many valuable files was he left with after deleting the unrelated files he downloaded in the second round?
Answer: **<BOT>First, we need to calculate how many non-valuable files Brennan downloaded in the first round<EOT>** The number of non-valuable files Brennan downloaded in the first round is 70/100*800 = <<70/100*800=560>>560 files. **<BOT>Next, we need to calculate how many valuable files Brennan downloaded in the first round<EOT>** The number of valuable files Brennan downloaded in the first round is 800-560 = <<800-560=240>>240 **<BOT>After downloading 400 new files, we need to calculate how many non-valuable files he downloaded<EOT>** When he downloaded 400 new files, there were 3/5*400= <<3/5*400=240>>240 non-useful files, which he deleted again. **<BOT>Finally, we need to calculate how many valuable files he was left with<EOT>** The total number of valuable files he downloaded in the second round is 400-240 = <<400-240=160>>160 **<BOT>Now we need to calculate the total number of valuable files he has left to write his research<EOT>** To write his research, Brennan had 160+240 = <<160+240=400>>400 useful files to reference to write his research. **<BOT>We are ready to output the final answer<EOT>** The answer is 400

Figure 4: The prompt and in-context learning examples used for sampling rationales for GSM8K. The bolded rationales represent implicit rationales in the document.

LLaMa-3-8B-Instruct and GPT-4 to provide feedback by directly reranking partial reasoning traces given the question (Figure 10).

F Discussion

Scaling up RATIONALYST. Scaling RATIONALYST with stronger models and increased computational resources is a logical next step. Utilizing stronger models, such as LLaMa-3-70B or GPT-4, would enhance the quality of extracted rationales, improve filtration accuracy, and ultimately strengthen RATIONALYST. However, due to computational constraints, we have not pursued this, which remains a limitation of this paper. Additionally, using larger unlabelled datasets with more extensive reasoning content, such as OpenWebMath (Paster et al., 2023), is currently infeasible due to the significant computational and time requirements for pre-filtering and training. These enhancements are planned for future work.

Connection to research on scaling test-time compute. Recent research has focused on extending computational resources at test-time (Snell et al., 2024; Wu et al., 2024), particularly for complex reasoning tasks. In our experiments, we focus on developing heuristics and employ a straightforward approach of sampling multiple candidates and reranking them based on RATIONALYST's guidance. However, RATIONALYST's framework is compatible with more sophisticated test-time compute techniques. Its heuristics can be integrated into existing algorithms like beam-search or look-ahead search, potentially enhancing their performance without significantly increasing computational cost.

Is training on extracted rationales necessary? In our approach, we first select a subset of unlabelled data that contains strong reasoning signals, then extract implicit rationales from this data for model fine-tuning. While it has been demonstrated

System Prompt:
Your task is to add rationals to a piece of text. The rationals should help you with predicting future text. You can add rationals by writing "<BOT>rational<EOT>". Here are some examples of rationale generation:

Example Input 1:
Question: He came across a raw item but his pack was full, he had to abandon something if he was to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Acquiring an item requires one to have space to carry it. As he had box full of items, he had to abandon one of them in order to acquire new one. Join and Engage is not related to item. Maintain an item does not require one to abandon existing item. One cannot remit raw item. The answer is B - acquire.

Example Output 1:
Question: He came across a raw item but his pack was full, he had to abandon something if he was to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Acquiring an item requires one to have space to carry it. **<BOT>We also need to establish relationship between acquire and the question itself<EOT>** As he had box full of items, he had to abandon one of them in order to acquire new one. **<BOT>Let's check whether other answer choices are related to the word item in question<EOT>** Join and Engage is not related to item. **<BOT>Let's check whether other answer choices are contradictory to the word abandon in question<EOT>** Maintain an item does not require one to abandon existing item. **<BOT>Let's check whether other answer choices are related to the word raw item in question<EOT>** One cannot remit raw item. **<BOT>We are ready to make final prediction<EOT>** The answer is B - acquire.

Example Input 2:
Question: Where is aberdeen in the US located?
Choices: A - washington, B - europe, C - scotland, D - maryland, E - south dakota
Answer: Aberdeen is located in Washington state which is in US. Aberdeen is also located in Scotland which is part of Europe. However, Scotland or Europe are not inside US. Aberdeen is not located in Maryland and South Dakota states. The answer is A - washington.

Example Output 2:
Question: Where is aberdeen in the US located?
Choices: A - washington, B - europe, C - scotland, D - maryland, E - south dakota
Answer: Aberdeen is located in Washington state which is in US. **<BOT>We need to think whether there is another aberdeen with the same name<EOT>** Aberdeen is also located in Scotland which is part of Europe. **<BOT>The answer choice need to be in the U.S. as mentioned in the question<EOT>** However, Scotland or Europe are not inside US. **<BOT>Let's check whether aberdeen is located in other answer choices<EOT>** Aberdeen is not located in Maryland and South Dakota states. **<BOT>We are ready to make final prediction<EOT>** The answer is A - washington.

Figure 5: The prompt and in-context learning examples used for sampling rationales for ECQA. The bolded rationales represent implicit rationales in the document.

that training on data with robust reasoning signals can enhance reasoning capabilities on its own (Gunasekar et al., 2023; Jiang et al., 2024a), we believe our method offers additional performance benefits for two reasons. First, many language models have already been trained on datasets like The Pile. The value of fine-tuning on previously encountered text is likely lower than the value of fine-tuning on newly incorporated rationales. Second, implicit rationales encapsulate the reasoning process. Pre-training on these rationales enhances reasoning more effectively than focusing on the whole document.

G Extended Related Work

Supervising reasoning. Supervision-based approaches have been shown to enhance the reasoning abilities of LLMs. Cobbe et al. (2021b) and Snell et al. (2024) demonstrate that training a "verifier" to supervise reasoning can be more parameter-efficient than simply expanding the parameters of the "reasoner" responsible for solving the reasoning task. Ground-truth feedback from interaction with the environment is an effective form of supervision (Wang et al., 2023), but it works only in controlled environments like simulated world. Some verifiers employ principles like compositional reasoning to validate the reasoning process (Dhuliawala et al., 2023; Weir et al., 2024, 2023; Vacareanu et al., 2024), these general-purpose approaches do not fully leverage the vast amount of available unlabeled data that a data-driven approach could utilize. Process-based supervision (Lightman et al., 2023; Luo et al., 2024; Wang et al., 2024a) offers supervision at each reasoning step rather than just at the final result. While promising, it requires substantial human annotation for the correctness of intermediate steps or the existence of ground-truth answers for data collection. Our work aims to address these challenges by proposing a data-centric process-supervision method without the need for human annotation.

Knowledge extraction from unlabelled data. LLMs are conventionally trained on extensive web data using autoregressive next-token prediction. While effective, this approach may not fully harness the potential of the pre-training data, as latent information within this data could be better ac-

cessed using techniques beyond simple next-token prediction. Recent research has demonstrated several approaches to utilize this latent information to develop more sophisticated language model capabilities. Schick et al. (2023) introduced Toolformer, which autonomously annotates and extracts appropriate positions, names, and inputs for tool use by leveraging supervision from future tokens. Similarly, Cornille et al. (2024) developed a method for learning to plan coherent article writing through self-supervised learning in text. More closely related to our work, Zelikman et al. (2024) proposed Quiet-Star, which applied a comparable technique to uncover underlying rationales in daily communication to enhance reasoning capabilities. Our work adopts a strategy similar to Quiet-Star for extracting rationales in an unsupervised manner. However, our approach diverges in its primary objective: we aim to train a "supervisor" that can utilize these rationales to provide process supervision for any "reasoner." This focus enables us to implement a simpler and more reliable method, as we don't need to directly integrate rationale extraction with "reasoner" training. Our approach thus offers a novel perspective on leveraging latent information in language models to enhance their capabilities.

Rationales as the basis for reasoning. Various studies have focused on improving the use of rationales to elicit reasoning. Fu et al. (2023) refine rationales for more effective reasoning elicitation, while Li et al. (2023) explore different approaches to leveraging rationales to enhance reasoning. Other works, such as Hwang et al. (2024), examine the verification of rationales produced by LLMs during reasoning to improve performance. Additionally, training LLMs on rationale-rich data is a common strategy for enhancing reasoning skills. As highlighted by Lewkowycz et al. (2022) and Jiang et al. (2024a), LLMs trained on science and math data tend to perform better on reasoning tasks, particularly when CoT prompting is used. In this work, we build on this foundation by using rationales as the core of our method to supervise reasoning.

H The Resulting Data from Extraction/Filteration.

On GSM8K, our method generates an average of 2.34 rationales per document, while on ECQA, it generates 2.58 rationales per document. The filtration process removes 80.5% of the generated rationales on GSM8K and 42.4% on ECQA.

For The Pile, we report the number of rationales per document and the number after filtration for each subdomain. The Pile's documents, being longer than those in GSM8K and ECQA, yield a higher average number of rationales per document. Among the subdomains, StackExchange retains the highest percentage of rationales, likely due to its question-answering format aligning well with our reasoning tasks and containing more inherent reasoning. However, The Pile as a whole contains less reasoning content, making rationale extraction challenging. Setting the threshold to 0 accepts all rationales more helpful than not having them, but the yield remains low. A manual review shows that most filtered rationales describe the preceding context rather than guiding future reasoning.

In total, we extracted approximately 14k rationales from GSM8K and ECQA combined, and about 65k from The Pile after filtration.

I RATIONALYST in Combination with General Methods for Reasoning Enhancement

We have conducted additional experiments focusing on GSM8K and Math datasets to verify our claim. We specifically chose these arithmetic reasoning benchmarks over multiple-choice questions, as using self-consistency on multiple-choice tasks could artificially inflate performance. Using 64 generations for self-consistency, we observed improved accuracy on both benchmarks with the help of RATIONALYST.

System Prompt:
Your task is to add rationals to a piece of text. The rationals should help you with predicting future text. You can add rationals by writing "<BOT>rational<EOT>". Here are one example of rationale generation:

Example Input:
\n\nNot applicable.\n\nJZ takes responsibility for drafting the manuscript. JZ, DY and XS are the attending doctors of this patient. QS is responsible for Pathological results and WW is for Microbiological results. YS and XS is responsible for revision of the manuscript. All authors read and approved the final manuscript.\n\nAuthors' information
{#FPar1}\n====================\n\nJiangnan Zhao and Dongmei Yuan are resident physicians-in-training, and Yi Shi, Qunli Shi, Weiping Wang and Xin Su are attending specialist physicians who dedicate their time to mentoring trainees

Example Output:
\n\nNot applicable.\n\nJZ takes responsibility for drafting the manuscript. **<BOT>Indicates JZ's role, suggesting the next roles to be described will involve other tasks like attending doctors and result analysis<EOT>** JZ, DY and XS are the attending doctors of this patient. QS is responsible for Pathological results and WW is for Microbiological results. **<BOT>Following the pattern, specific tasks related to results analysis are now described, predicting further details on manuscript revision<EOT>** YS and XS is responsible for revision of the manuscript. **<BOT>The revision responsibility follows naturally after result analysis, likely leading to a final approval statement<EOT>** All authors read and approved the final manuscript. **<BOT>Final approval statement indicates closure of the roles and tasks, suggesting a summary or author information follows<EOT>** \n\nAuthors\' information \{#FPar1\} **<BOT>Transition to detailed author information section, predicting descriptions of each author\'s background<EOT>**
\n====================\n\nJiangnan Zhao and Dongmei Yuan are resident physicians-in-training, **<BOT>Starting with the roles of resident physicians-in-training, predicting subsequent details about attending specialist physicians<EOT>** and Yi Shi, Qunli Shi, Weiping Wang and Xin Su are attending specialist physicians who dedicate their time to mentoring trainees

Example Input:
Suppose -s*w = -26*w. Suppose -15 = -5*j - w*j. Let g(b) = b**3 - 3*b**2 - 2*b + 3. What is g(j)?\n-3\nLet o(w) = -9 - 31214*w + 31220*w - 3. Determine o(1).\n-6\nLet y = -11 + 15. Suppose -4*c + 30 = -y*r + 50, 0 = -5*c - r - 25. Let q(d) = d + 5. Give q(c).\n0\nLet r = 2899 - 2906. Let i(s) = 2*s**2 + 12*s - 3. Calculate i(r).\n11\nLet q be 6/(-105) + 4199/595. Let m(t) = t**3 - 9*t**2 + 5*t + 5. Give m(q).\n-58\nLet d = 440 - 92

Example Output:
Suppose -s*w = -26*w. Suppose -15 = -5*j - w*j. Let g(b) = b**3 - 3*b**2 - 2*b + 3. What is g(j)?\n**<BOT>First solve the equations to find j, then substitute j into g(b) to calculate g(j)<EOT>**\n-3\nLet o(w) = -9 - 31214*w + 31220*w - 3. Determine o(1).\n**<BOT>Substitute w = 1 into the function o(w) and simplify to determine its value<EOT>**\n-6\nLet y = -11 + 15. Suppose -4*c + 30 = -y*r + 50, 0 = -5*c - r - 25. Let q(d) = d + 5. Give q(c).\n**<BOT>First calculate y, then solve the equations to find c, and finally substitute c into q(d) to calculate q(c)<EOT>**\n0\nLet r = 2899 - 2906. Let i(s) = 2*s**2 + 12*s - 3. Calculate i(r).\n**<BOT>First calculate r, then substitute r into the function i(s) to calculate i(r)<EOT>**\n11\nLet q be 6/(-105) + 4199/595. Let m(t) = t**3 - 9*t**2 + 5*t + 5. Give m(q).\n**<BOT>First calculate q, then substitute q into the function m(t) to determine m(q)<EOT>**\n-58\nLet d = 440 - 92\n**<BOT>Calculate the value of d as 440 - 92<EOT>**

Figure 6: The prompt and in-context learning examples used for sampling rationales for The Pile. The bolded rationales represent implicit rationales in the document.

```
System Prompt:
You are a smart assistant that solves math word problems. You will only generate one sentence that
extends the reasoning trajectory that solves the question given the question and partial answer
reasoning trajectory. Please don't repeat your previous generation while you're generating the
sentence. If you think you're ready to output the answer, you can finish the response with The
answer is:

Example Input:
Question: Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of
babysitting. How much did she earn?
Answer:

Example Output:
Weng earns 12/60 = $<<12/60=0.2>>0.2 per minute.

Example Input:
Question: Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of
babysitting. How much did she earn?
Answer: Weng earns 12/60 = $<<12/60=0.2>>0.2 per minute.

Example Output:
Working 50 minutes, she earned 0.2 x 50 = $<<0.2*50=10>>10. The answer is: 10
```

Figure 7: The prompt and in-context-learning demonstrations used during inference time to encourage the agent model reason step by step on GSM8K.

```
System Prompt:
You are a smart assistant that solves commonsense reasoning problems. You will only generate one
sentence that extends the reasoning trajectory that solves the question given the question and
partial answer reasoning trajectory. Please don't repeat your previous generation while you're
generating the sentence. Please analyze all answer choices before finishing reasoning. If you
think you're ready to output the answer, you can finish the response with The answer is:

Example Input:
Question: He came across a raw item but his pack was full, he had to abandon something if he was
to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer:

Example Output:
Join is not the type of the activity associated with item.

Example Input:
Question: He came across a raw item but his pack was full, he had to abandon something if he was
to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Join is not the type of the activity associated with item.

Example Output:
Acquiring requires space in a pack. To create space in a full pack, one has to abandon an item.

Example Input:
Question: He came across a raw item but his pack was full, he had to abandon something if he was
to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Join is not the type of the activity associated with item. Acquiring requires space in a
pack. To create space in a full pack, one has to abandon an item.

Example Output:
Engage is not related to item.

Example Input:
Question: He came across a raw item but his pack was full, he had to abandon something if he was
to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Join is not the type of the activity associated with item. Acquiring requires space in a
pack. To create space in a full pack, one has to abandon an item. Engage is not related to item.

Example Output:
Maintain an item does not require one to make space for it.

Example Input:
Question: He came across a raw item but his pack was full, he had to abandon something if he was
to what the item?
Choices: A - join, B - acquire, C - engage, D - maintain, E - remit
Answer: Join is not the type of the activity associated with item. Acquiring requires space in a
pack. To create space in a full pack, one has to abandon an item. Engage is not related to item.
Maintain an item does not require one to make space for it.

Example Output:
One cannot remit raw item. The answer is B - acquire.
```

Figure 8: The prompt and in-context-learning demonstrations used during inference time to encourage the agent model reason step by step on ECQA.

```
System Prompt:
You are a smart assistant that solves math problems. You will only generate one sentence that
extends the reasoning trajectory that solves the question given the question and partial answer
reasoning trajectory. Please don't repeat your previous generation while you're generating the
sentence.

Example Input:
Question: If $2x - 9y = 14$ and $6x = 42 +y$, what is the value of the product $xy$?
Answer:

Generated Rationale:
Let's first rewrite the second equation to isolate $y$ in terms of $x$

Example Output:
Rewrite the second equation to express $y$ in terms of $x$: $6x = 42 + y \implies y = 6x - 42$

Example Input:
Question: If $2x - 9y = 14$ and $6x = 42 +y$, what is the value of the product $xy$?
Answer: Rewrite the second equation to express $y$ in terms of $x$: $6x = 42 +y$ \implies y = 6x -
42 $

Generated Rationale:
Substitute the expression for $y$ into the first equation

Example Output:
Substitute $y = 6x - 42$ into the first equation: $2x - 9(6x - 42) = 14$

Example Input:
Question: If $2x - 9y = 14$ and $6x = 42 +y$, what is the value of the product $xy$?
Answer: Rewrite the second equation to express $y$ in terms of $x$: $ 6x = 42 + y \implies y = 6x
- 42 $. Substitute $y = 6x - 42$ into the first equation: $2x - 9(6x - 42) = 14$

Generated Rationale:
We should simplify the equation and solve for $x$

Example Output:
Simplify and solve for $x$: $2x - 54x + 378 = 14 \implies -52x + 378 = 14 \implies -52x = -364
\implies x = 7$

Example Input:
Question: If $2x - 9y = 14$ and $6x = 42 +y$, what is the value of the product $xy$?
Answer: Rewrite the second equation to express $y$ in terms of $x$: $6x = 42 + y \implies y = 6x -
42$. Substitute $y = 6x - 42$ into the first equation: $ 2x - 9(6x - 42) = 14 $. Simplify and
solve for $x$: $2x - 54x + 378 = 14 \implies -52x + 378 = 14 \implies -52x = -364 \implies x = 7$

Generated Rationale:
Use the value of $x$ to calculate $y$

Example Output:
Substitute $x = 7$ back into $y = 6x - 42$ to find $y$: $y = 6(7) - 42 = 42 - 42 = 0$

Example Input:
Question: If $2x - 9y = 14$ and $6x = 42 +y$, what is the value of the product $xy$?
Answer: Rewrite the second equation to express $y$ in terms of $x$: $6x = 42 + y \implies y = 6x -
42$. Substitute $y = 6x - 42$ into the first equation: $2x - 9(6x - 42) = 14$. Simplify and solve
for $x$: $2x - 54x + 378 = 14 \implies -52x + 378 = 14 \implies -52x = -364 \implies x = 7$.
Substitute $x = 7$ back into $y = 6x - 42$ to find $y$: $y = 6(7) - 42 = 42 - 42 = 0$

Generated Rationale:
Now we can just multiply the values of $x$ and $y$ to find the product

Example Output:
Finally, calculate the product $xy$: $xy = 7 \cdot 0 = 0$. The value of the product $xy$ is
$\boxed{0}$.
```

Figure 9: Rationales generated by RATIONALYST for the test set of MATH.

System Prompt:
Your task is to assign rewards to the reasoning trajectory that solves the question given the question and partial answer reasoning trajectory. The more probable the reasoning trajectory is correct, the higher the reward should be. The reward should be an integer in the range of 0 to 3. End your generation with So the reward is:

Example Input:
Question: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?
Answer: Michael started with 58 golf balls. After losing 23 on tuesday, he had 58 - 23 = 35.

Example Output:
This reasoning trajectory is correct and reasonable. So the reward is: 3

Example Input:
Question: Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny?

Answer: Jason started with 20 lollipops. After giving some to Denny, he had 12 left. To find out how many he gave away, we need to add 12 + 20, which gives us 32. So, Jason gave Denny 32 lollipops.

Example Output:
This reasoning trajectory incorrectly adds instead of subtracting the lollipops given to Denny, leading to an illogical result. So the reward is: 1.

Figure 10: The prompt and in-context-learning demonstrations used during process supervision to elicit the feedback by directly reranking partial reasoning trajectory.