

# UNIVERSAL REASONER: A SINGLE, COMPOSABLE PLUG-AND-PLAY REASONER FOR FROZEN LLMs

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

Large Language Models (LLMs) have demonstrated remarkable general capabilities, but enhancing skills such as reasoning often demands substantial computational resources and may compromise their generalization. While Parameter-Efficient Fine-Tuning (PEFT) methods offer a more resource-conscious alternative, they typically require retraining for each LLM backbone due to architectural dependencies. To address these challenges, here we propose Universal Reasoner (UniR) - a single, lightweight, composable, and plug-and-play reasoning module that can be used with larger frozen LLMs to endow it with specialized reasoning capabilities. Specifically, UniR decomposes the reward into a standalone reasoning module that is trained in a decoupled manner using predefined rewards, effectively translating trajectory-level signals into token-level guidance. Once trained, UniR can be combined with frozen LLMs at inference time by simply adding its output logits to those of the LLM backbones. This additive structure naturally enables modular composition: multiple UniR modules trained for different tasks can be jointly applied by summing their logits, enabling complex reasoning via composition. Experiments on mathematical reasoning and machine translation show that UniR surpasses existing baseline fine-tuning methods. Furthermore, UniR demonstrates weak-to-strong generalization: reasoning modules trained on smaller models effectively guide much larger LLMs. Beyond this, UniR generalizes across modalities such as in vision language models and domains such as medical reasoning. This makes UniR a cost-efficient, adaptable, and robust solution for enhancing reasoning in LLMs without compromising their core capabilities.

## 1 INTRODUCTION

Recent Large Language Models (LLMs), such as OpenAI-o1 (OpenAI, 2024), DeepSeek-R1 (DeepSeek-AI, 2025), Kimi-1.5 (Team et al., 2025), BOLT (Pang et al., 2025), and OpenReasoner-Zero (Hu et al., 2025), have demonstrated remarkable capabilities in complex reasoning tasks. These models leverage Chain-of-Thought (CoT) (Wei et al., 2022) to elicit step-by-step reasoning processes from LLMs, thereby significantly enhancing their ability to tackle intricate problems. A key driver of this progress has been Reinforcement Fine-Tuning (RFT), which aligns model outputs with desired criteria, often human preferences, through reinforcement learning algorithms such as PPO (Schulman et al., 2017), DPO (Rafailov et al., 2023), GRPO (Shao et al., 2024), etc.

Unfortunately, adapting these powerful LLMs to specialize in specific domains or further enhance their reasoning abilities through RFT presents significant hurdles. The process is often prohibitively expensive due to the substantial computational and memory resources required for fine-tuning large-scale models. While Parameter-Efficient Fine-Tuning (PEFT) techniques like LoRA (Hu et al., 2022) have been proposed to mitigate these costs, they introduce their own limitations: (1) Model Dependency: LoRA modules strongly depend on specific model architectures, limiting their transferability across models of different sizes (e.g., from 3B to 14B). (2) Poor Linearity Support: Combining multiple domain-specific LoRA adapters through linear operations lacks robust theoretical support and frequently yields unpredictable outcomes (Wang et al., 2023).

To address these challenges, we introduce **UniR (Universal Reasoner)**, a novel method that decouples the reasoning capability into a specialized reasoning module, independently trained and

054 designed to enhance the capabilities of a backbone LLM. This lightweight reasoning module can  
055 be seamlessly integrated into a backbone model with a compatible tokenizer, without requiring ex-  
056 plicit information about the model’s internal structure. This approach not only fosters modularity,  
057 preserving the core capabilities of the base model, but also enables effective weak-to-strong general-  
058 ization where a compact reasoning module can significantly enhance larger models. Moreover, this  
059 modularity opens avenues for combining distinct reasoning modules across multiple domains such  
060 as visual mathematical reasoning and medical reasoning.

061 The core innovation of UniR lies in its decoupling of reward model training from policy updates.  
062 **While many alignment methods rely on preference datasets, UniR is designed to leverage explicit,**  
063 predefined rewards that are often available for tasks with verifiable solutions, such as mathematical  
064 problem-solving or machine translation. We propose to model these typically trajectory-level re-  
065 wards, or rewards over a full response, as a sum of log-probabilities derived from our decomposed  
066 reasoning module. This formulation effectively translates a global reward signal into a sequence of  
067 implicit token-level guidance signals that steer the reasoning process. The reasoning module is then  
068 trained to maximize this reward using a policy gradient algorithm, ensuring that only the smaller  
069 reasoning module is updated while the backbone policy remains unchanged. During inference, rea-  
070 soning module provides guidance to a main backbone model by combining their outputs, typically  
071 by adding their respective logits before sampling. Furthermore, because the guidance occurs at the  
072 logit level, multiple reasoning modules trained for distinct tasks can be composed through simple  
073 logit addition—enabling composite probabilistic reasoning without additional training or architec-  
074 tural modification.

075 Our contributions are summarized as follows:

- 076 • **Computationally Efficient Training.** We introduce a method to train a decomposed rea-  
077 soning module by **optimizing verifiable rewards via token-level, policy-gradient guidance.**  
078 Furthermore, our algorithm backpropagates only through a smaller reasoning model while  
079 keeping the large backbone model frozen, reducing computational and memory overhead.
- 080 • **Transferability Across Model Sizes.** Although our approach trains only small models,  
081 the learned reasoning module can be effectively transferred to larger models at inference  
082 time. For example, a 1.5B reasoning module trained alongside a 3B backbone can be used  
083 in conjunction with a 14B model, providing token-level guidance that improves the larger  
084 model’s performance without requiring any fine-tuning of the 14B model itself.
- 085 • **Composability.** Our method allows seamless composition of multiple small reasoning  
086 modules specialized for specific domains or tasks. These specialized models can then be  
087 selectively deployed based on the domain context, enabling modular and efficient inference  
088 pipelines where domain expertise is encapsulated in lightweight, task-specific policies.
- 089 • **Superior Performance on Diverse Tasks.** Our experiments show that UniR substantially  
090 outperforms conventional fine-tuning methods, showcasing its efficacy in enhancing LLM  
091 capabilities on challenging mathematics, translation, and even medical benchmarks.

## 093 2 RELATED WORK

095 **Enhancing LLM Reasoning.** Improving LLM reasoning is key for complex reasoning and under-  
096 standing (Durante et al., 2024). One prominent strategy involves Supervised Fine-Tuning (SFT)  
097 on specialized, often small, datasets curated to foster reasoning skills; these datasets might be dis-  
098 tilled from larger models (Open Thoughts, 2025) or meticulously selected from high-quality ex-  
099 amples (Muennighoff et al., 2025; Ye et al., 2025). Recently, Reinforcement Learning (RL)-based  
100 Group Relative Policy Optimization (GRPO) and related methods (Shao et al., 2024; Liu et al.,  
101 2025b) have shown promise in enhancing LLM reasoning capabilities using rule-based rewards  
102 without requiring a separate value function. Furthermore, the concept of verifiable rewards is ex-  
103 tended by using generative models to provide soft reward signals for unstructured tasks (Su et al.,  
104 2025b). Our work leverages verifiable rewards to improve the reasoning ability of LLMs using the  
105 specialized reasoning module on tasks that require complex reasoning.

106 **Efficient Methods for Domain-Specific Tasks.** To mitigate the substantial computational costs  
107 of training LLMs, Parameter-Efficient Fine-Tuning (PEFT) methods like LoRA (Hu et al., 2022)  
are widely used, but their modules are tied to the model architecture, limiting transferability across

108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161

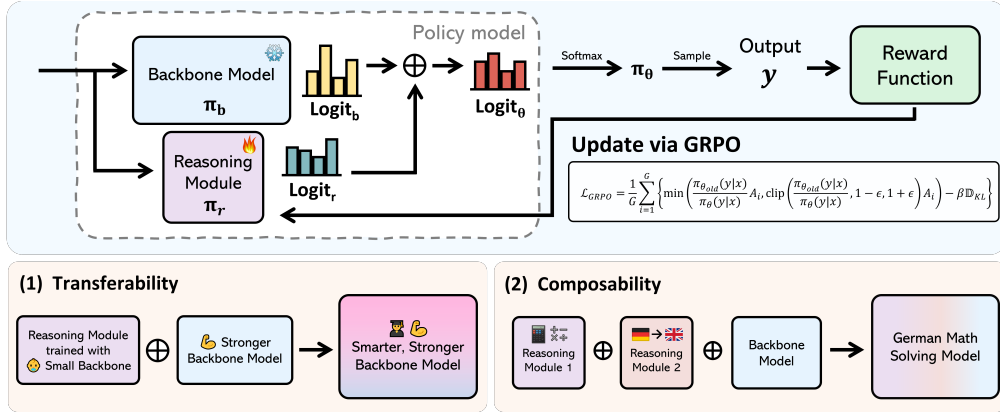


Figure 1: **UniR Framework Overview.** Our approach trains a lightweight, transferable reasoning module ( $\pi_r$ ) using predefined rewards to guide a frozen backbone model ( $\pi_b$ ), offering (1) transferability across different backbone models or tasks; and (2) composability by combining multiple specialized reasoning modules through reward optimization.

LLMs of different sizes. In contrast, UniR trains a compact reasoning module that interacts with a frozen backbone at the logit level, making training efficient and producing a flexible, architecture-agnostic component for transfer and guidance.

**Guided LLM generation.** Guiding the generation of frozen LLMs at inference offers a flexible way to improve outputs without full retraining (Li et al., 2023). Some plug-and-play approaches attach a separately trained module to a frozen backbone. TART (Bhatia et al., 2023) trains a separate module attached to a frozen base LLM to perform classification. While TART operates at the embedding level, UniR operates at the logit level to guide the base LLM. Recent techniques focus on finer-grained, token-level guidance (Xu et al., 2024; Yang et al., 2023). GenARM (Xu et al., 2024) proposes a token-level reward model, trained on human preference data with DPO (Rafailov et al., 2023), to guide a frozen LLM. While both GenARM and UniR employ a token-level guiding module, GenARM focuses on training the reward model separately using preference data. In contrast, UniR trains a reasoning module to maximize verifiable rewards, avoiding the need for costly preference data collection. In addition, several works (Mudgal et al., 2023; Han et al., 2024) train a value function using temporal difference learning. Instead, UniR directly trains the reasoning module using a policy gradient algorithm, collaborating with a backbone model.

### 3 PROBLEM DEFINITION

A growing trend in enhancing LLM reasoning is Reinforcement Learning with Verifiable Rewards (RLVR), where models are guided by predefined and automatically checkable signals—such as correctness or rule-based verification—rather than relying on human preference data. This approach has been adopted in models such as DeepSeek-R1 (DeepSeek-AI, 2025), and recent RLVR research has demonstrated its effectiveness across diverse domains (Su et al., 2025a; Zhao et al., 2025). Following this direction, we focus on tasks where performance can be assessed reliably via such verifiable rewards. We denote the predefined reward as  $r(x, y)$  for a given prompt  $x$  from a dataset  $\mathcal{D}$  and a generated output sequence  $y$ . Standard RL fine-tuning aims to find an optimal policy,  $\pi_{\theta}^*(y|x)$ , that maximizes the expected reward while remaining close to a backbone policy ( $\pi_b(y|x)$ ), typically a pre-trained base LLM. This objective is formulated as:

$$\max_{\pi_{\theta}} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_{\theta}(y|x)} [r(x, y)] - \beta \mathbb{D}_{KL}[\pi_{\theta}(y|x) || \pi_b(y|x)], \quad (1)$$

where  $\mathbb{D}_{KL}[\cdot || \cdot]$  represents the Kullback-Leibler divergence, and  $\beta > 0$  controls the strength of the regularization towards the backbone policy  $\pi_b$ . The optimal policy  $\pi_{\theta}^*(y|x)$  for this objective can be expressed in a closed form as (Ziebart et al., 2008; Rafailov et al., 2023):

$$\pi_{\theta}^*(y|x) = \frac{1}{Z(x)} \pi_b(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right), \quad (2)$$

where  $Z(x) = \sum_y \pi_b(y|x) \exp(\frac{1}{\beta} r(x, y))$  is a partition function. A key challenge in directly applying Eq. (2) for autoregressive generation is that the trajectory-level reward  $r(x, y)$  can be obtained

only after the full sequence  $y$  is complete. However, it does not provide token-specific feedback during the generation process. For fine-grained guidance of an LLM’s reasoning steps, per-token signals are more desirable. Theoretically, the optimal token-level decision rule for the objective in Eq. (1) can be expressed using the optimal soft Q-function,  $Q^*(a_t|s_t)$  where  $s_t = (x, y_{<t})$  is the current state and  $a_t = y_t$  is the next token. This  $Q^*(a_t|s_t)$  represents the maximum expected future return following the optimal policy  $\pi_{\theta}^*$ . The optimal token-level policy is then given by (Haarnoja et al., 2017; Ziebart et al., 2008):

$$\log \pi_{\theta}^*(y_t|x, y_{<t}) = \log \pi_b(y_t|x, y_{<t}) + \frac{1}{\beta} Q^*(y_t|x, y_{<t}) - \log Z'(x, y_{<t}), \quad (3)$$

where  $Z'(x, y_{<t})$  is the per-step partition function (Uehara et al., 2025; Haarnoja et al., 2017). However, guiding with  $Q^*$  is impractical since it requires knowledge of the optimal policy  $\pi_{\theta}^*$  itself. This challenge motivates the need to bridge observable trajectory-level rule-based rewards with a practical formulation that enables effective token-level guidance.

## 4 UNIR: UNIVERSAL REASONER

To circumvent the direct estimation of  $Q^*$ , our core idea is to learn a separate, typically smaller, reasoning module, denoted as  $\pi_r(y|x)$  with parameters  $\phi$ . This policy is specifically trained to capture the desired reasoning or task-specific behaviors that lead to high rewards. We then use this learned reasoning module to guide a larger LLM.

### 4.1 MODELING TRAJECTORY REWARDS VIA THE REASONING MODULE

To connect trajectory-level reward  $r(x, y)$  to our reasoning module  $\pi_r$  and enable its use in a token-by-token guidance mechanism, we posit that the reward can be represented by the sum of log-probabilities of the reasoning module  $\pi_r$ :

$$\frac{1}{\beta} r(x, y) = \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t}; \phi). \quad (4)$$

Here,  $\beta$  is the same temperature parameter from Eq. (1). This formulation implies that sequences  $y$  assigned a higher sum of token-level log-probabilities by  $\pi_r$  should correspond to sequences achieving higher rewards  $r(x, y)$ . The feasibility of representing a reward function via the log-likelihood of a policy is supported by theoretical work in (Xu et al., 2024), which claims that reward can be represented using the parameterization  $\log \pi_r(y|x)$  for some probability distribution  $\pi_r(y|x)$ . This assumption is further supported by recent works (Li et al., 2025; Wulfmeier et al., 2024) that language models trained via next-token prediction implicitly learn endogenous reward signals.

Since  $\log \pi_r(y|x) = \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t})$  by the chain rule, this suggests that a sufficiently expressive reasoning module  $\pi_r$  can indeed model any trajectory-level reward function  $r(x, y)$ . By substituting the definition from Eq. (4) into the optimal policy form of Eq. (2), we derive the structure of our guided policy for autoregressive generation of the next token  $y_t$ . This translates to combining their next-token predictions:

$$\log \pi_{\theta}(y_t|x, y_{<t}) = \log \pi_b(y_t|x, y_{<t}) + \log \pi_r(y_t|x, y_{<t}) - \log Z'(x, y_{<t}), \quad (5)$$

where  $Z'(x, y_{<t})$  is the per-step partition function. Accordingly, the logits from the frozen backbone  $\pi_b$  and the learned module  $\pi_r$  can be separable to produce the logits for the guided policy  $\pi_{\theta}$ .

This additive and separable guidance mechanism extends to scenarios involving multiple reward criteria. Instead of training a single monolithic model to jointly optimize for all objectives, UniR allows for the training of several distinct reasoning modules  $\{\pi_r^1, \dots, \pi_r^N\}$ , each specialized for a different reward function  $r_i(x, y)$ . At inference, these specialized modules can be combined by taking a weighted sum of their logits and adding them to the backbone model’s logits:

$$\log \pi_{\theta}(y_t|x, y_{<t}) \propto \log \pi_b(y_t|x, y_{<t}) + \sum_{i=1}^N \alpha_i \log \pi_r^i(y_t|x, y_{<t}), \quad (6)$$

where  $\alpha_i \geq 0$  controls the influence of each reasoning module  $\pi_r^i$ . This approach offers significant flexibility, as the weights  $\alpha_i$  can be adjusted at inference time to explore dynamic optimization across multiple objectives without any retraining. This additive combination of logits is not a heuristic but the principled solution to a standard KL-regularized optimization problem,  $\max_{\pi_\theta} \mathbb{E}_{y \sim \pi_\theta} [\sum_i \alpha_i r_i(x, y)] - \mathbb{D}_{KL}(\pi_\theta \| \pi_b)$ , a common framework for multi-objective alignment (Abdolmaleki et al., 2020; Dekoninck et al., 2023; Xu et al., 2024). The goal is to find the optimal policy  $\pi_\theta(y|x) \propto \pi_b(y|x) \prod_i \exp(\frac{1}{\beta} \alpha_i r_i(x, y))$ . By substituting our reward definition from Eq. (4), taking the logarithm of this solution directly yields the linear combination of logits in Eq. (6).

## 4.2 TRAINING $\pi_r$ WITH GRPO

The parameters  $\phi$  of the reasoning module  $\pi_r$  are trained to maximize the expected predefined reward  $r(x, y)$ , not the sum of its own log-probabilities defined in Eq. (4). We employ Group Relative Policy Optimization (GRPO) (Shao et al., 2024) for this purpose. For each input  $x \sim \mathcal{D}$ , we generate a group of  $G$  candidate responses  $\{y^1, \dots, y^G\}$  by sampling from  $\pi_\theta(y^i|x; \phi_{old})$ , where  $\phi_{old}$  denotes the parameters of the reasoning module from the previous iteration. For each generated sequence  $y^i$ , we compute its external reward  $r(x, y^i)$  (denoted  $r_i$  for brevity) and then calculate the normalized advantage  $A_i$ :

$$A_i = (r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})) / \text{std}(\{r_1, r_2, \dots, r_G\}). \quad (7)$$

In GRPO, the objective function for updating the parameters  $\phi$  of the reasoning module is then:

$$\begin{aligned} \mathcal{L}_{\text{GRPO}}(\phi) = & -\mathbb{E}_{x \sim \mathcal{D}, y^i \sim \pi_\theta(y^i|x; \phi_{old})} \left[ \frac{1}{G} \sum_{i=1}^G \left\{ \min \left( \frac{\pi_\theta(y^i|x)}{\pi_\theta(y^i|x; \phi_{old})} A_i, \right. \right. \\ & \left. \left. \text{clip} \left( \frac{\pi_\theta(y^i|x)}{\pi_\theta(y^i|x; \phi_{old})}, 1 - \epsilon, 1 + \epsilon \right) A_i \right) - \beta_{KL} \mathbb{D}_{KL}[\pi_\theta(y^i|x; \phi) \| \pi_\theta(y^i|x; \phi_0)] \right\} \right]. \end{aligned} \quad (8)$$

Here,  $\epsilon$  is the clipping hyperparameter from PPO-clip,  $\phi_0$  is initial value of  $\phi$ , and  $\beta_{KL}$  controls the KL-regularization term. The learning direction is determined by the ratio  $\frac{\pi_\theta(y^i|x, \phi)}{\pi_\theta(y^i|x; \phi_{old})}$ , which simplifies to  $\frac{\pi_r(y^i|x, \phi)}{\pi_r(y^i|x; \phi_{old})}$  as the terms from the  $\pi_b$  cancel out, thereby the gradient  $\nabla_\phi \mathcal{L}_{\text{GRPO}}$  only affects the parameters of  $\pi_r$  since  $\pi_b$  is frozen.

## 4.3 INTERPRETING $\pi_r$ WITH $Q^*$

We now provide theoretical grounding to understand what the learned reasoning module  $\pi_r$  captures at each token-level decision point. This provides a theoretical justification for interpreting  $\log \pi_r$  as a meaningful guidance signal that drives the policy towards optimality for the trajectory-level objective.

**Theorem 1.** (Informal) *If the reasoning module  $\pi_r$  satisfies Eq. (4), such that its summed per-token log-probabilities accurately approximate the scaled reward for trajectories  $y$  that are consistent with the optimal policy  $\pi_\theta^*$ , then*

$$\log \pi_r(y_t|x, y_{<t}) = \frac{1}{\beta} Q^*(y_t|x, y_{<t}). \quad (9)$$

*Proof.* We assume guided policy  $\pi_\theta$  has converged to the optimal policy  $\pi_\theta^*$  for all states  $s_t = (x, y_{<t})$  and actions  $a_t = y_t$  by optimizing the GRPO objective to maximize the reward  $r(x, y)$  since the KL-regularized objective typically admits a unique optimum. Given that the trajectory-level policies are identical, their per-token conditional distributions must also match for any  $s_t$ . Comparing Eq. (5) with the standard form of the optimal policy expressed in terms of the soft Q-function (Eq. (3)) shows that  $\log \pi_r(y_t | x, y_{<t})$  corresponds to the scaled optimal soft Q-value.  $\square$

# 5 EXPERIMENTS

## 5.1 EXPERIMENTAL SETTINGS

We evaluate UniR on two tasks: mathematical problem-solving and machine translation. Mathematical problem-solving requires explicit, step-by-step reasoning, while machine translation involves

Table 1: Zero-shot pass@1 performance with standard deviation on mathematic benchmarks. The out-of-distribution results are obtained using models trained on the MATH training dataset. Models marked with \* are taken from the report (Yang et al., 2024b).

Method	Trained Model	In-distribution		Out-of-distribution			Avg.	
		GSM8K	MATH-500	AIME24	Minerva	OlympiadBench		
Other 7B	Mathstral-7B-v0.1*	7B	84.9	56.6	-	16.2	21.5	-
	Internlm2-math-plus-7B*	7B	84.0	54.4	-	17.3	18.8	-
	NuminaMath-7B-CoT*	7B	75.4	55.2	-	19.1	19.9	-
	DeepSeekMath-7B-RL*	7B	88.2	52.4	-	20.6	19.0	-
Llama3.2-3B	Backbone only	-	65.6 ± 1.3	33.0 ± 2.3	3.7 ± 1.8	12.5 ± 2.3	8.5 ± 2.4	24.7
	Backbone + 1B	-	63.8 ± 0.7	35.7 ± 2.0	5.0 ± 2.7	13.1 ± 1.7	9.9 ± 0.9	25.5
	GRPO LoRA	3B	65.8 ± 0.9	32.1 ± 1.6	6.0 ± 2.5	12.0 ± 1.3	11.9 ± 0.7	25.6
	GRPO Full	3B	65.1 ± 1.1	34.6 ± 1.9	4.0 ± 3.6	12.6 ± 1.7	10.3 ± 0.6	25.3
	<b>Ours</b>	1B	<b>77.5 ± 1.2</b>	<b>48.8 ± 1.2</b>	<b>7.3 ± 2.9</b>	<b>16.6 ± 1.6</b>	<b>16.4 ± 0.5</b>	<b>33.3</b>
Qwen2.5-3B	Backbone only	-	74.4 ± 1.0	44.2 ± 1.3	6.3 ± 3.7	19.2 ± 1.7	24.6 ± 0.9	33.7
	Backbone + 1.5B	-	75.6 ± 0.9	50.6 ± 1.6	4.0 ± 2.1	20.7 ± 2.3	25.1 ± 0.9	35.2
	GRPO LoRA	3B	75.0 ± 0.9	60.0 ± 1.2	5.7 ± 3.2	24.7 ± 2.1	27.1 ± 0.6	38.5
	GRPO Full	3B	81.3 ± 0.5	62.9 ± 1.3	4.7 ± 2.3	25.0 ± 1.0	27.5 ± 1.3	40.3
	<b>Ours</b>	1.5B	<b>84.5 ± 0.7</b>	<b>66.8 ± 1.4</b>	<b>7.7 ± 2.7</b>	<b>26.3 ± 2.0</b>	<b>28.2 ± 0.8</b>	<b>42.7</b>

Table 2: Pass@1 performance with standard deviation on machine translation benchmarks for both English-to-German (EN→DE) and German-to-English (DE→EN) directions. GPT-4.1-nano and Qwen2.5-3B-Instruct were evaluated with the same system and user prompts as used for our method.

Method	Trained Model	EN → DE				DE → EN				
		BLEU	CometKiwi	XComet	Avg.	BLEU	CometKiwi	XComet	Avg.	
Others	Google translator	-	32.39	80.20	92.00	68.20	32.95	81.00	92.90	68.95
	GPT4.1-nano	-	31.89	83.90	96.00	70.60	37.14	83.70	97.00	72.61
	Qwen2.5-3B-Instruct	-	23.28	78.25	91.26	64.26	32.75	82.66	95.70	70.37
Llama3.2-3B	Backbone only	-	23.19 ± 0.12	77.72 ± 0.12	90.97 ± 0.12	63.96	30.65 ± 0.16	81.27 ± 0.05	94.53 ± 0.05	68.82
	Backbone + 1B	-	24.10 ± 0.04	78.70 ± 0.05	91.47 ± 0.05	64.76	31.82 ± 0.08	81.75 ± 0.02	94.74 ± 0.03	69.44
	GRPO LoRA	3B	24.18 ± 0.08	78.93 ± 0.05	91.97 ± 0.05	65.03	31.39 ± 0.07	81.73 ± 0.03	94.95 ± 0.03	69.36
	GRPO Full	3B	25.26 ± 0.07	80.13 ± 0.06	92.88 ± 0.06	66.09	35.52 ± 0.13	80.43 ± 0.27	92.35 ± 0.41	69.43
	<b>Ours</b>	1B	<b>27.03 ± 0.07</b>	<b>81.43 ± 0.02</b>	<b>93.70 ± 0.03</b>	<b>67.39</b>	<b>37.21 ± 0.08</b>	<b>82.82 ± 0.02</b>	<b>95.89 ± 0.03</b>	<b>71.97</b>

implicit reasoning based on domain-specific knowledge and semantic understanding. These tasks provide complementary benchmarks for assessing our decomposed reasoning module.

**Models and Baselines.** Our experiments utilize models from Qwen2.5 (Yang et al., 2024a) and Llama3.2 (Grattafiori et al., 2024). We train the reasoning module  $\pi_r$  alongside a 3B-scale backbone  $\pi_b$  in two settings: (1)  $\pi_b$  as Qwen2.5-3B with  $\pi_r$  as Qwen2.5-1.5B, and (2)  $\pi_b$  as Llama3.2-3B with  $\pi_r$  as Llama3.2-1B. For the transferability experiment, we use Qwen2.5-14B and Llama3.1-8B as backbone models. Unless stated otherwise, all models are Instruct variants. For baselines, we apply GRPO to fine-tune the entire backbone model (e.g., the 3B-scale  $\pi_b$  model) directly on each task using the same predefined rewards. Baseline tuning is implemented through either full model tuning or LoRA-based fine-tuning.

**Dataset and Reward Formulation.** For mathematical reasoning tasks, we use the Math-12k (Lightman et al., 2023; Liu et al., 2025b) and GSM8k (Cobbe et al., 2021) datasets. The reward  $r(x, y)$  is determined by the rule-based correctness of the final answer, following the methodology in Dr.GRPO (Liu et al., 2025b). For machine translation, we use the IWSLT 2017 dataset (Cettolo et al., 2017) for English-to-German and German-to-English translation, employing BLEU (Papineni et al., 2002) and CometKiwi (Rei et al., 2022) as reward during training. Following MT-R1-Zero (Feng et al., 2025), we combine lexical and semantic metrics for balanced training.

**Evaluation Metrics.** We evaluate mathematical reasoning using pass@1 accuracy on five benchmarks: GSM8k, MATH-500 (Lightman et al., 2023), AIME24<sup>1</sup>, Minerva (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). GSM8k is tested with a model trained by its own training split; the others are evaluated with a model trained by Math-12K. For the translation task, we report BLEU, CometKiwi, and XComet-XL (Guerreiro et al., 2024) scores. We set the temperature to 0.6 and evaluated performance across 10 different seeds. Further details on training hyperparameters, optimization settings, evaluation details are provided in Appendix B, and prompt templates are presented in Appendix E.

<sup>1</sup><https://huggingface.co/datasets/AI-MO/aimo-validation-aimo>

324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377

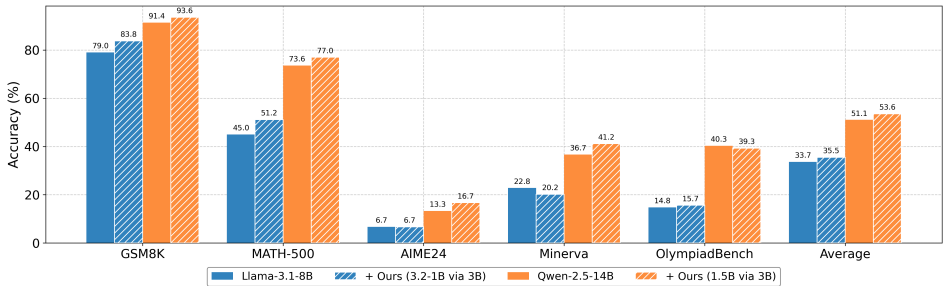


Figure 2: **Effectiveness of Reasoning module Transfer.** Results demonstrate that a trained reasoning module can improve performance when integrated with larger backbone models across diverse mathematical reasoning tasks.

### 5.2 ENHANCING REASONING ABILITY

As shown in Table 1, our framework consistently achieves substantial improvements over GRPO baselines across both in-distribution and out-of-distribution benchmarks. Specifically, UniR achieves an average score of 33.3 and 42.7, surpassing GRPO Full (25.3, 40.3) and GRPO LoRA (25.6, 38.5) with both model families, respectively. In machine translation, as shown in Table 2, UniR also outperforms both GRPO baselines in both translation directions. For EN→DE, UniR improves the BLEU score by +1.77 over GRPO Full, while for DE→EN it improves BLEU by +1.69. Beyond BLEU, semantic metrics such as CometKiwi and XComet also exhibit significant improvements. These results indicate that UniR effectively generalizes beyond explicit reasoning in solving mathematical problems, extending its efficacy to implicit reasoning tasks such as translation.

To isolate the contribution of our reward-driven alignment, we compare UniR against ablative baselines (e.g., Backbone + 1B). This represents a naive ensemble where the logits of the frozen backbone are directly summed with those of a pre-trained reasoning module without any of the reward optimization proposed by UniR. As shown in Tables 1 and 2, merely combining pre-trained model outputs performs worse than the UniR method. This highlights the need for our reward alignment process, which is crucial for transforming the reasoning module into an effective guide.

### 5.3 TRANSFERABILITY AND COMPOSABILITY

**Transferability to Larger Backbone LLMs.** Figure 2 illustrates the transferability of our reasoning module  $\pi_r$ , originally trained on Math-12k and GSM8k datasets with a 3B-scale backbone. We evaluate its effectiveness when deployed to guide larger frozen backbone models with 8B and 14B parameters. Across most benchmarks, the transferred  $\pi_r$  improves average reasoning performance (35.5 and 53.6) compared to the respective backbone baselines (33.7 and 51.1), even without additional fine-tuning. This suggests that  $\pi_r$  is not narrowly specialized to its original backbone but instead functions as a generalizable reasoning augmentation module to enhance the reasoning capabilities of stronger models at inference time in a resource-efficient framework.

**Combining Specialized Reasoning Modules.** To investigate the advantages of integrating multiple specialized reasoning capabilities enabled by UniR, we design a composite task: translating German math problems from the MGSM dataset (Shi et al., 2022) into English. We utilize two distinct reasoning modules:  $\pi_r^{\text{Math}}$ , a module specialized for mathematical problem-solving on the GSM8k dataset, and  $\pi_r^{\text{Trans}}$ , a module trained for German-to-English translation. These modules are combined with a frozen base LLM ( $\pi_b$ ) through a weighted summation of their logits:

$$\log \pi_b + \alpha \log \pi_r^{\text{Math}} + (1 - \alpha) \log \pi_r^{\text{Trans}}, \quad (10)$$

where  $\alpha \in [0, 1]$  is a coefficient that balances the influence between both modules. We employ GPT-4.1-nano to evaluate translation quality and the accuracy of the generated output. Detailed configurations for this experiment are provided in Appendix E.3.

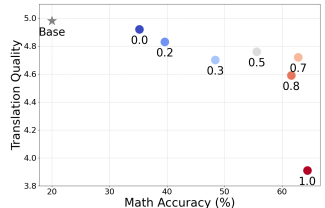


Figure 3: **Performance on a German-to-English Math problem-solving task.** The numbers in the figure indicate the value of  $\alpha$  and “Base” denotes backbone-only method.

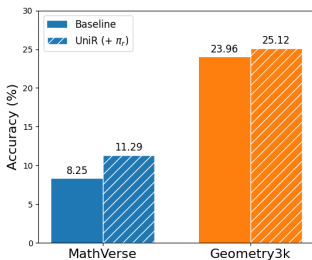


Figure 4: UniR Performance on Visual Math Reasoning.

As depicted in Figure 3, the backbone-only model (Base) achieves a high translation quality but yields poor math accuracy since it only perform translation. When the translation module is added ( $\alpha = 0$ ), math accuracy improves even though the module was trained only for translation. This phenomenon highlights that the  $\pi_r^{\text{Trans}}$  modules’ guidance steers the backbone reasoning states into a English-understanding mode. Once the backbone is controlled to think in coherent English, its own latent mathematical ability suppressed by the German input is enhanced, causing the accuracy jump. Beyond this baseline clarification, we observe that increasing the weight  $\alpha$  generally leads to improved accuracy in solving the mathematical problems, while a higher weight for the translation module tends to improve translation quality. This highlights a crucial capability of UniR: the ability to combine and balance multiple specialized reasoning modules. This modularity allows for precise control over the policy model’s behavior, leading to improved performance on multi-faceted tasks. We illustrate the example of the responses in Figure 18.

#### 5.4 ADDITIONAL EXPERIMENTS: EXTENSION AND ANALYSIS

**Extension to Vision Language Models.** To investigate whether our UniR framework can transfer reasoning ability in multimodal contexts, we extend it by employing a Vision Language Model (VLM) as the backbone. Specifically, we utilize Qwen2.5-VL-3B-Instruct (Bai et al., 2025) as the frozen backbone model  $\pi_b$ . This VLM is guided by a 0.5B Qwen-based reasoning module  $\pi_r$ , which is trained on the text-based GSM8k dataset. For evaluation, we use the “Vision Intensive” questions in MathVerse (Zhang et al., 2024) and Geometry3k (Lu et al., 2021) benchmarks, which feature mathematical problems accompanied by visual diagrams, thus requiring both visual understanding and mathematical reasoning. Evaluation protocols and system prompts follow those used in the GSM8k experiments. An example is depicted in Figure 20.

As shown in Figure 4, UniR significantly improves performance over the baseline VLM on both benchmarks. These enhancements are particularly noteworthy because the reasoning module  $\pi_r^{\text{Math}}$  was trained exclusively on text-based mathematical problems, with no access to visual information. This aligns with prior work (Liu et al., 2025a), which shows that text-only post-training can improve multimodal reasoning capabilities. Unlike monolithic post-training, UniR achieves this through a plug-and-play module that guides frozen VLMs, providing a flexible and efficient way to extend reasoning to multimodal domains.

**Generalization to Medical Domain.** To further assess the generalizability of our method, we conduct additional experiments in the medical domain using the MIMIC-IV dataset (Johnson et al., 2020). Access to the dataset was obtained through PhysioNet (Goldberger et al., 2000) under the appropriate data use agreements. We evaluate UniR on two primary tasks: (1) Readmission Prediction and (2) Length of Stay (LOS) Prediction. In the readmission prediction task, the model predicts whether a patient will be readmitted within 15 days of discharge (binary classification). For LOS prediction, we train UniR to categorize the patient’s ICU stay length into four classes: short stay ( $\leq 1$  day), moderate stay (2–7 days), extended stay (8–14 days), and long-term stay ( $> 14$  days). Detailed settings are provided in Appendix B.3.1.

We use Qwen2.5-3B as the backbone model and Qwen2.5-0.5B as reasoning modules. Table 3 summarizes the results. For the readmission task, UniR improves accuracy by 13.45 points and F1-score by 0.9 points over the baseline. For the LOS task, UniR achieves an accuracy of 67.77% and an F1-score of 16.25%, representing a dramatic improvement over the baseline of 9.64% and 5.92%, respectively. Moreover, we evaluate composability of our framework in the medical setting. While this composition decreases accuracy compared to using each module individually, it consistently

Table 3: Evaluation of UniR on ICU readmission, length of stay (LOS), and mortality prediction.

Task	Method	Reasoning module	Acc.	F1
Readmission	Backbone only	-	39.46	25.87
	UniR	Readmission	<b>52.91</b>	26.77
	UniR	Readmission + LOS	44.88	<b>28.79</b>
Length of stay (LOS)	Backbone only	-	9.64	5.92
	UniR	LOS	<b>67.77</b>	16.25
	UniR	Readmission + LOS	61.75	<b>18.32</b>
Mortality	Backbone only	-	39.76	28.27
	UniR	Readmission + LOS	<b>63.65</b>	<b>35.18</b>

Table 4: Comparison on math problems in Japanese (JP) and Spanish (ES).

Method	Accuracy (%)		Translation Quality $\uparrow$	
	JP	ES	JP	ES
Backbone	42.0	62.4	4.63	4.98
Ours	50.4	66.8	4.64	4.75

Table 5: Evaluation of using a small pre-trained reasoning model on math reasoning within the UniR framework.

Backbone Model $\pi_b$	Reasoning Module $\pi_r$	Train with UniR	Avg.
DeepSeek 1.5B	-	X	32.8
Qwen2.5 3B	DeepSeek 1.5B	X	43.7
Qwen2.5 3B	DeepSeek 1.5B	O	46.9

Table 6: Comparison of inference speed with UniR acceleration.

Method	Sec./Token $\downarrow$
Backbone only	0.0262
UniR (naive)	0.0414
UniR + CoS	0.0279

improves F1-scores. Furthermore, the combined inference generalizes to the unseen mortality prediction task, which is designed for predicting mortality in binary classification, yielding substantial gains over the baseline without task-specific training. These results suggest that complementary knowledge from different tasks can be integrated, which is beneficial in medical reasoning as their combination provides a richer basis for risk estimation, extending even to mortality. The example is shown in Figure 21.

**Generalization across Low-resource Languages.** We investigate whether the UniR math reasoning module  $\pi_r^{Math}$ , trained solely on English math problems, can generalize to low-resource languages, Japanese and Spanish, which are underrepresented in pre-training corpora. Each task is tested on MGSM, with accuracy and translation quality, which measure fluency and consistency in each language using the GPT-4.1-nano model. As shown in Table 4, UniR consistently improves math reasoning accuracy even when both inputs and outputs are entirely in low-resource languages, highlighting its generalization capability beyond English. See Appendix B.3.2 for details.

**Using Pre-trained Reasoning Model Directly as Guidance.** We investigate the effectiveness of using a pre-trained small reasoning model as the reasoning module  $\pi_r$  within the UniR framework. Specifically, we use DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), a distilled 1.5B model trained via supervised fine-tuning on reasoning data. We evaluate its performance with and without further UniR optimization. Table 5 as an average across five mathematical benchmarks (see Table 12 for the full breakdown), shows that integrating a pre-trained reasoning model into UniR improves performance in mathematical reasoning, even without additional training. We believe this is due to the theoretical alignment between Thm. 1 and recent findings (Wulfmeier et al., 2024; Li et al., 2025) which show that models trained via next-token prediction implicitly learn soft Q-functions and encode endogenous reward signals. Further fine-tuning through UniR enhances reasoning capabilities beyond the simple combination. This highlights the potential of reusing pre-trained small reasoning models within UniR fine-tuning for improving reasoning performance.

**Enhancing Inference Efficiency with Speculative Decoding.** Integrating additional reasoning modules can increase per-token inference time. As shown in Table 6, integrating UniR naively increases per-token inference time compared to the backbone alone. To mitigate this, we integrate UniR with a recent speculative decoding method, specifically CoS (Fu et al., 2025), which enables collaborative inference by leveraging multiple model logits in parallel. This approach effectively reduces the overhead, bringing inference speed close to that of the backbone model. See Appendix B.3.3 for detailed experimental setup.

**Analysis of Reward Decomposition.** To empirically validate our reward modeling (Eq. 4), we analyze the summed log-probabilities ( $\sum_t \log \pi_r$ ) in generated responses on the GSM8k test set. As detailed in Appendix C.1 and shown in its accompanying figure, our analysis reveals a strong positive correlation: correct responses ( $r = 1$ ) are assigned significantly higher log-probabilities than incorrect ones ( $r = 0$ ). This effect is amplified by our training method. While the pre-trained module exhibits moderate separation, the distinction becomes far more pronounced after UniR training, confirming that our framework effectively trains the module to internalize the reward signal.

## 6 CONCLUSION

In this paper, we present Universal Reasoner, a novel plug-and-play reasoning module that directly optimizes predefined rewards by focusing solely on a compact reasoning module while keeping the larger backbone model frozen, enabling efficient training. Our approach exhibits strong transferability; the reasoning module can effectively guide backbone models of various sizes without requiring further fine-tuning. Furthermore, UniR facilitates the combination of modules specialized for different tasks. UniR demonstrates generalization not only across modalities but also across different

domains, highlighting its potential to integrate multiple domain-specific modules and achieve synergistic improvements across diverse tasks.

**Limitations.** While UniR is designed to be architecture-agnostic, its application requires tokenizer alignment. The performance of our framework is inherently bounded by the capabilities of the frozen backbone model  $\pi_b$ , as  $\pi_r$  primarily guides its existing representations. While additive composition is grounded in multi-objective optimization, the potential for negative interference between modules with conflicting reward signals remains an area for future theoretical investigation.

## REFERENCES

- Abbas Abdolmaleki, Sandy Huang, Leonard Hasenclever, Michael Neunert, Francis Song, Martina Zambelli, Murilo Martins, Nicolas Heess, Raia Hadsell, and Martin Riedmiller. A distributional view on multi-objective policy optimization. In *International conference on machine learning*, pp. 11–22. PMLR, 2020.
- Daman Arora and Andrea Zanette. Training language models to reason efficiently. *arXiv preprint arXiv:2502.04463*, 2025. URL <https://arxiv.org/abs/2502.04463>.
- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- Kush Bhatia, Avanika Narayan, Christopher M De Sa, and Christopher Ré. Tart: A plug-and-play transformer module for task-agnostic reasoning. *Advances in Neural Information Processing Systems*, 36:9751–9788, 2023.
- Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Jan Niehues, Sebastian Stüker, Katsutho Sudoh, Koichiro Yoshino, and Christian Federmann. Overview of the iwslt 2017 evaluation campaign. In *Proceedings of the 14th International Workshop on Spoken Language Translation*, pp. 2–14, 2017.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL <https://arxiv.org/abs/2501.12948>.
- Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin Vechev. Controlled text generation via language model arithmetic. *arXiv preprint arXiv:2311.14479*, 2023.
- Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, et al. Agent ai: Surveying the horizons of multimodal interaction. *arXiv preprint arXiv:2401.03568*, 2024.
- Zhaopeng Feng, Shaosheng Cao, Jiahao Ren, Jiayuan Su, Ruizhe Chen, Yan Zhang, Zhe Xu, Yao Hu, Jian Wu, and Zuozhu Liu. Mt-r1-zero: Advancing llm-based machine translation via r1-zero-like reinforcement learning. *arXiv preprint arXiv:2504.10160*, 2025.
- Jiale Fu, Yuchu Jiang, Junkai Chen, Jiaming Fan, Xin Geng, and Xu Yang. Fast large language model collaborative decoding via speculation. *arXiv preprint arXiv:2502.01662*, 2025.
- Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

- 540 Nuno M Guerreiro, Ricardo Rei, Daan van Stigt, Luisa Coheur, Pierre Colombo, and André FT  
541 Martins. xcomet: Transparent machine translation evaluation through fine-grained error detection.  
542 *Transactions of the Association for Computational Linguistics*, 12:979–995, 2024.
- 543  
544 Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with  
545 deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361.  
546 PMLR, 2017.
- 547 Seungwook Han, Idan Shenfeld, Akash Srivastava, Yoon Kim, and Pulkit Agrawal. Value  
548 augmented sampling for language model alignment and personalization. *arXiv preprint*  
549 *arXiv:2405.06639*, 2024.
- 550  
551 Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu,  
552 Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for  
553 promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint*  
554 *arXiv:2402.14008*, 2024.
- 555 Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuezhi Li, Shean Wang, Lu Wang,  
556 Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- 557  
558 Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum.  
559 Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base  
560 model, 2025. URL <https://arxiv.org/abs/2503.24290>.
- 561 Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark.  
562 Mimic-iv. *PhysioNet*. Available online at: <https://physionet.org/content/mimiciv/1.0/>(accessed  
563 August 23, 2021), pp. 49–55, 2020.
- 564  
565 Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative  
566 decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.
- 567 Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ra-  
568 masesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative  
569 reasoning problems with language models. *Advances in Neural Information Processing Systems*,  
570 35:3843–3857, 2022.
- 571  
572 Yi-Chen Li, Tian Xu, Yang Yu, Xuqin Zhang, Xiong-Hui Chen, Zhongxiang Ling, Ningjing Chao,  
573 Lei Yuan, and Zhi-Hua Zhou. Generalist reward models: Found inside large language models.  
574 *arXiv preprint arXiv:2506.23235*, 2025.
- 575 Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. Rain: Your language  
576 models can align themselves without finetuning. *arXiv preprint arXiv:2309.07124*, 2023.
- 577  
578 Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan  
579 Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. In *The Twelfth*  
580 *International Conference on Learning Representations*, 2023.
- 581 Qianchu Liu, Sheng Zhang, Guanghui Qin, Timothy Ossowski, Yu Gu, Ying Jin, Sid Kiblawi, Sam  
582 Preston, Mu Wei, Paul Vozila, et al. X-reasoner: Towards generalizable reasoning across modal-  
583 ities and domains. *arXiv preprint arXiv:2505.03981*, 2025a.
- 584  
585 Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee,  
586 and Min Lin. Understanding rl-zero-like training: A critical perspective. *arXiv preprint*  
587 *arXiv:2503.20783*, 2025b.
- 588 Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu.  
589 Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning.  
590 *arXiv preprint arXiv:2105.04165*, 2021.
- 591  
592 Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng  
593 Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, et al. Controlled decoding from  
language models. *arXiv preprint arXiv:2310.17022*, 2023.

- 594 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke  
595 Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time  
596 scaling, 2025. URL <https://arxiv.org/abs/2501.19393>.
- 597 Open Thoughts. Open thoughts. <https://open-thoughts.ai>, 2025. Accessed: 2025-05-05.
- 598  
599 OpenAI. Learning to reason with llms. [https://openai.com/index/  
600 learning-to-reason-with-llms/](https://openai.com/index/learning-to-reason-with-llms/), September 2024. Accessed: 2025-05-05.
- 601  
602 Bo Pang, Hanze Dong, Jiacheng Xu, Silvio Savarese, Yingbo Zhou, and Caiming Xiong.  
603 Bolt: Bootstrap long chain-of-thought in language models without distillation. *arXiv preprint  
604 arXiv:2502.03860*, 2025.
- 605 Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic  
606 evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association  
607 for Computational Linguistics*, pp. 311–318, 2002.
- 608 Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea  
609 Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-  
610 seventh Conference on Neural Information Processing Systems*, 2023. URL [https://arxiv.  
611 org/abs/2305.18290](https://arxiv.org/abs/2305.18290).
- 612 Ricardo Rei, Marcos Treviso, Nuno M Guerreiro, Chrysoula Zerva, Ana C Farinha, Chris-  
613 tine Maroti, José GC De Souza, Taisiya Glushkova, Duarte M Alves, Alon Lavie, et al.  
614 Cometkiwi: Ist-unbabel 2022 submission for the quality estimation shared task. *arXiv preprint  
615 arXiv:2209.06243*, 2022.
- 616  
617 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy  
618 optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- 619 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,  
620 Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical  
621 reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- 622  
623 Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi,  
624 Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Lan-  
625 guage models are multilingual chain-of-thought reasoners, 2022.
- 626 Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu.  
627 Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains. *arXiv  
628 preprint arXiv:2503.23829*, 2025a.
- 629 Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu.  
630 Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains. *arXiv  
631 preprint arXiv:2503.23829*, 2025b.
- 632  
633 Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun  
634 Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1.5: Scaling reinforcement learning with  
635 llms. *arXiv preprint arXiv:2501.12599*, 2025.
- 636 Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tom-  
637 maso Biancalani. Inference-time alignment in diffusion models with reward-guided generation:  
638 Tutorial and review. *arXiv preprint arXiv:2501.09685*, 2025.
- 639 Xi Wang, Laurence Aitchison, and Maja Rudolph. Lora ensembles for large language model fine-  
640 tuning. *arXiv preprint arXiv:2310.00035*, 2023.
- 641  
642 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny  
643 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in  
644 neural information processing systems*, 35:24824–24837, 2022.
- 645 Markus Wulfmeier, Michael Bloesch, Nino Vieillard, Arun Ahuja, Jorg Bornschein, Sandy Huang,  
646 Artem Sokolov, Matt Barnes, Guillaume Desjardins, Alex Bewley, et al. Imitating language via  
647 scalable inverse reinforcement learning. *Advances in Neural Information Processing Systems*, 37:  
90714–90735, 2024.

- 648 Yuancheng Xu, Udari Madhushani Sehwag, Alec Koppel, Sicheng Zhu, Bang An, Furong Huang,  
649 and Sumitra Ganesh. Genarm: Reward guided generation with autoregressive reward model for  
650 test-time alignment. *arXiv preprint arXiv:2410.08193*, 2024.
- 651 An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li,  
652 Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint*  
653 *arXiv:2412.15115*, 2024a.
- 654 An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jian-  
655 hong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical  
656 expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.
- 657 Shentao Yang, Shujian Zhang, Congying Xia, Yihao Feng, Caiming Xiong, and Mingyuan Zhou.  
658 Preference-grounded token-level guidance for language model fine-tuning. *Advances in Neural*  
659 *Information Processing Systems*, 36:24466–24496, 2023.
- 660 Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more  
661 for reasoning, 2025. URL <https://arxiv.org/abs/2502.03387>.
- 662 Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong  
663 Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at  
664 scale. *arXiv preprint arXiv:2503.14476*, 2025.
- 665 Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. Lora-fa: Memory-efficient  
666 low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2308.03303*,  
667 2023.
- 668 Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou,  
669 Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the  
670 diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186.  
671 Springer, 2024.
- 672 Jiaying Zhao, Xihan Wei, and Liefeng Bo. R1-omni: Explainable omni-multimodal emotion recog-  
673 nition with reinforcement learning. *arXiv preprint arXiv:2503.05379*, 2025.
- 674 Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse  
675 reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.
- 676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701

## 702 A THEORETICAL JUSTIFICATION

### 703 A.1 REWARD PARAMETERIZATION

704 To justify the reward parameterization in Eq. (4), which decomposes the trajectory-level reward  
705 function into a token-level sum, we first formally prove that this parameterization is equivalent and  
706 expressive for any reward function within our optimization objective.

707 **Lemma 2** (Theorem 3 in Xu et al. (2024)). *For any trajectory-level reward function  $r(x, y)$  and  
708 any  $\beta > 0$ , there exists an equivalent reward function  $\hat{r}(x, y)$  and a probability distribution  $\pi_r(y|x)$   
709 such that  $\hat{r}(x, y) = \beta \log \pi_r(y|x)$ , where  $r(x, y)$  and  $\hat{r}(x, y)$  induce the identical optimal policy  $\pi_{\theta}^*$   
710 for the KL-regularized objective in Eq. (1).*

711 *Proof.* As shown in Eq. (2), the unique optimal policy  $\pi_{\theta}^*$  for the KL-regularized objective has the  
712 closed-form solution. Following Theorem 3 in Xu et al. (2024), we define two reward functions  
713  $r(x, y)$  and  $\hat{r}(x, y)$  as equivalent if they induce the same optimal policy. This holds if they differ  
714 only by a function  $f(x)$  that is independent of the trajectory  $y$ :

$$715 \hat{r}(x, y) = r(x, y) - f(x). \quad (11)$$

716 This is because the  $f(x)$  term is absorbed into the partition function  $Z(x)$ .

717 Our goal is to find a function  $f(x)$  that allows  $\hat{r}(x, y)$  to be written in the form  $\beta \log \pi_r(y|x)$  for  
718 some valid probability distribution  $\pi_r(y|x)$ . Let us define a probability distribution  $\pi_r(y|x)$  using  
719 the given reward function  $r(x, y)$ :

$$720 \pi_r(y|x) := \frac{\exp(r(x, y)/\beta)}{\sum_{y'} \exp(r(x, y')/\beta)} \quad (12)$$

721 By construction,  $\sum_y \pi_r(y|x) = 1$ , so  $\pi_r$  is a valid probability distribution.

722 Let us define our new reward function  $\hat{r}(x, y)$  using this  $\pi_r(y|x)$ , as required by the lemma:

$$723 \hat{r}(x, y) := \beta \log \pi_r(y|x). \quad (13)$$

724 Substitute the definition of  $\pi_r(y|x)$  into this equation:

$$725 \hat{r}(x, y) = \beta \log \left( \frac{\exp(r(x, y)/\beta)}{\sum_{y'} \exp(r(x, y')/\beta)} \right) \quad (14)$$

$$726 = \beta \left[ \log \left( \exp \left( \frac{r(x, y)}{\beta} \right) \right) - \log \left( \sum_{y'} \exp \left( \frac{r(x, y')}{\beta} \right) \right) \right] \quad (15)$$

$$727 = r(x, y) - \beta \log \left( \sum_{y'} \exp \left( \frac{r(x, y')}{\beta} \right) \right) \quad (16)$$

$$728 = r(x, y) - f(x), \quad (17)$$

729 where  $f(x) = \beta \log \left( \sum_{y'} \exp(r(x, y')/\beta) \right)$ . Therefore,  $\hat{r}(x, y)$  and  $r(x, y)$  are equivalent, as they  
730 induce the identical optimal policy  $\pi_{\theta}^*$ . This proves that any reward function  $r(x, y)$  has an equivalent  
731 counterpart  $\hat{r}(x, y)$  that can be represented as a scaled log-probability of a distribution  $\pi_r$ .  
732 □

733 **Corollary 2.1.** *The trajectory-level reward  $\hat{r}(x, y)$  proven in Lemma 2 can be decomposed into a  
734 sum of token-level log-probabilities.*

735 *Proof.* By the chain rule of probability, trajectory-level distribution  $\pi_r(y|x)$  can be decomposed as:

$$736 \log \pi_r(y|x) = \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t}). \quad (18)$$

756 Substituting this into the result from Lemma 2:  
757

$$758 \hat{r}(x, y) = \beta \log \pi_r(y|x) = \beta \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t}). \quad (19)$$

760 Dividing by  $\beta$ , we arrive at the form used in our paper Eq. (4):  
761

$$762 \frac{1}{\beta} \hat{r}(x, y) = \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t}). \quad (20)$$

765 This formally justifies that our parameterization in Eq. (4) is not a limiting assumption, but a fully  
766 general way to represent any reward function for our objective.  $\square$   
767

## 768 A.2 INTERPRETATION OF $\pi_r$ AT THE OPTIMUM

769 In this section, we derive the token-level structure of the optimal policy  $\pi_\theta^*$  and use it to motivate the  
770 design of our guided policy  $\pi_\theta$ .  
771

772 From Lemma 2 and Corollary 2.1, we substitute the equivalent reward  $r(x, y)$  (omitting the  $\hat{\cdot}$  for  
773 clarity) its token-decomposed log-probability from Eq. (2):  
774

$$775 \pi_\theta^*(y|x) \propto \pi_b(y|x) \exp \left( \sum_{t=1}^{|y|} \log \pi_r(y_t|x, y_{<t}) \right) \quad (21)$$

$$778 = \pi_b(y|x) \prod_{t=1}^{|y|} \pi_r(y_t|x, y_{<t}) \quad (22)$$

$$781 = \prod_{t=1}^{|y|} \pi_b(y_t|x, y_{<t}) \pi_r(y_t|x, y_{<t}) \quad (23)$$

784 By definition, the trajectory-level policy  $\pi_\theta^*(y|x)$  is also a product of its own token-level conditional  
785 policies:  $\pi_\theta^*(y|x) = \prod_t \pi_\theta^*(y_t|x, y_{<t})$ . By matching the terms inside the product for each step  $t$ , we  
786 derive the token-level optimal policy defined in Eq. (3).  
787

788 Motivated by this derivation, we define our trainable policy  $\pi_\theta$  to have this exact mathematical  
789 structure, but replacing the ideal  $\pi_r$  with our learnable reasoning module  $\pi_r(\cdot; \phi)$  in Eq. (5).

790 **Theorem 3.** Let  $\pi_\theta^*$  be the unique optimal policy for the KL-regularized objective Eq. (1) with  
791 reward  $r(x, y)$ . The token-level distribution of  $\pi_\theta^*$  is given by Eq. (3):  
792

$$793 \log \pi_\theta^*(y_t|x, y_{<t}) = \log \pi_b(y_t|x, y_{<t}) + \frac{1}{\beta} Q^*(y_t|x, y_{<t}) - \log Z'(x, y_{<t}), \quad (24)$$

794 where  $Q^*(y_t|x, y_{<t})$  is the optimal soft Q-function for the reward  $r(x, y)$  and  $Z'$  is the per-step  
795 partition function. As the KL-regularized objective drives  $\pi_\theta \rightarrow \pi_\theta^*$ , we can formally interpret the  
796 trained reasoning module’s output as the scaled optimal soft Q-function:  
797

$$798 \log \pi_r(y_t|x, y_{<t}) = \frac{1}{\beta} Q^*(y_t|x, y_{<t}) \quad (25)$$

800 *Proof.* The objective in Eq. (1) has a unique optimal solution  $\pi_\theta^*$ . Let us assume the optimization is  
801 successful and  $\pi_\theta$  has converged to  $\pi_\theta^*$ :  
802

$$803 \log \pi_\theta(y_t|x, y_{<t}; \phi) = \log \pi_\theta^*(y_t|x, y_{<t}). \quad (26)$$

804 Now, substitute the definitions from Eq. (3) and Eq. (5) into this equality. Then, we get:  
805

$$806 \log \pi_r(y_t|x, y_{<t}; \phi) = \frac{1}{\beta} Q^*(y_t|x, y_{<t}). \quad (27)$$

808 This proves that our reasoning module  $\pi_r(\cdot; \phi)$  is learning to approximate the scaled optimal soft  
809 Q-function via the KL-regularized optimization.  $\square$

## B EXPERIMENTAL DETAILS

### B.1 HYPERPARAMETER SETUP

The detailed hyperparameter settings are provided in Table 7. For the fair comparison, we used the same hyperparameters across all methods, including the baseline model, GRPO with full fine-tuning, GRPO with LoRA, and simple logit-sum. Specifically, we set the KL divergence coefficient  $\beta = 0.04$  for GRPO, following the default value in the Hugging Face TRL library<sup>2</sup>. However, in our UniR framework, the reasoning module  $\pi_r$  is often initialized from a general pre-trained small model, which may not possess strong task-specific reasoning capabilities. Regularizing  $\pi_\theta$  towards  $\pi_b$  which is heavily influenced by the initial, potentially  $\pi_{r,old}$  can impede the learning of high-reward reasoning paths. Therefore, we omit such an explicit KL regularization by  $\beta_{KL} = 0$ , allowing  $\pi_r$  more freedom to explore and learn effective reasoning strategies driven by the predefined rewards. This choice aligns with findings in related works such as DAPO (Yu et al., 2025), which also found benefits in adjusting or omitting certain regularization components.

Among the reward configurations, *rule\_based\_accuracy* assigns a reward of 1 if the numeric value between the `<answer>` and `</answer>` tags matches the ground truth and 0 otherwise. The *boxed\_reward* evaluates the value enclosed within the `\boxed{ }` format, assigning a reward of 1 if it matches the correct answer and 0 otherwise. Similarly the *tag\_based\_reward* evaluates the value enclosed within the answer tag. For the implementation of *boxed\_reward* and *tag\_based\_reward* we adopted the official code from the DR.GRPO repository (Liu et al., 2025b)<sup>3</sup>. For BLEU, we used the default implementation of BLEU scoring provided by SacreBLEU<sup>4</sup>.

Table 7: Hyperparameter setups.

Parameter	GSM8K	Math-12K-Qwen	Math-12K-Llama	Translation
<i>Training Configuration</i>				
gradient_checkpointing	true	true	true	true
gradient_checkpointing_kwargs	use_reentrant: false	use_reentrant: false	use_reentrant: false	use_reentrant: false
learning_rate	1.0e-06	1.0e-06	1.0e-06	1.0e-06
lr_scheduler_type	cosine with min_lr: 0.1	cosine with min_lr: 0.1	cosine with min_lr: 0.1	cosine with min_lr: 0.1
warmup_ratio	0.1	0.1	0.1	0.1
max_steps	1000	1000	1000	10000
effective_batch_size	64	32	32	32
max_prompt_length	512	512	512	512
max_completion_length	1024	2048	2048	1024
num_generations	8	8	8	8
temperature	0.7	0.7	0.7	0.7
$\alpha$	1.0	1.0	1.0	1.0
<i>Evaluation Settings</i>				
max_prompt_length	512	512	512	512
max_completion_length	1024	2048	2048	250
temperature	0.6	0.6	0.6	0.6
$\alpha$	1.0	1.0	1.0	1.0
<i>Reward Configuration</i>				
reward_funcs	rule_based_accuracy	boxed_reward	tag_based_reward	BLEU, cometkiwi
reward_weights	1.0	1.0	1.0	1.0, 1.0

### B.2 EVALUATION DETAIL

For evaluation, BLEU and accuracy were computed using the same reward logic applied during training for each dataset. Translation baselines included outputs from Google Translator, implemented via the `GoogleTranslator` interface of the `deep-translator` library<sup>5</sup>. GPT-4.1-nano was also used for comparison, with decoding performed at a sampling temperature of 0.7. We employ CometKiw<sup>6</sup> and XComet-XL<sup>7</sup> scores to evaluate translation quality. For the main results in Table 1, 2, and 9, decoding was performed with a sampling temperature of 0.6. We report the

<sup>2</sup><https://github.com/huggingface/trl>

<sup>3</sup><https://github.com/sail-sg/understand-r1-zero>

<sup>4</sup><https://github.com/mjpost/sacrebleu>

<sup>5</sup><https://github.com/nidhaloff/deep-translator>

<sup>6</sup><https://huggingface.co/Unbabel/XCOMET-XL>

<sup>7</sup><https://huggingface.co/Unbabel/wmt22-CometKiw-da>

mean and standard deviation over ten runs with different seeds (1-10). All other experiments were conducted deterministically using greedy decoding `do_sample=False`.

### B.3 ADDITIONAL EXPERIMENTAL SETUP

#### B.3.1 MEDICAL REASONING

Table 8: Hyperparameter setup for training readmission and length of stay tasks.

Parameter	Value
<i>Training Configuration</i>	
max_steps	1000
max_prompt_length	3072
max_completion_length	1024
num_generations	4
<i>Reward Configuration</i>	
reward_funcs	accuracy
reward_weights	1.0

The detailed hyperparameter settings are summarized in Table 8. A key difference from previous tasks was the use of longer prompt and completion lengths to accommodate the more extensive data in the medical domain. We provide the prompt in Figure 15. The patient information contains a Patient ID and information on conditions, procedures, and medications for each visit. When composing the two task modules, we set the weight for each module to  $\alpha_i = 1$ , as specified in Eq. (6). All other prompts and parameters remain the same as when a single module was used. For label construction, we map all categorical values to numeric labels. In the readmission task, patients readmitted within 15 days are labeled 1, while others are labeled 0. For the length of stay prediction task, we use four categories: 0 ( $\leq 1$  day), 1 (2–7 days), 2 (8–14 days), and 3 ( $> 14$  days). Finally, in the mortality prediction task, patients who died are labeled 1, and survivors are labeled 0. For reward design, the model receives a reward of 1 if the numeric value extracted between the `<answer>` and `</answer>` tags matched the ground truth, and 0 otherwise.

#### B.3.2 LOW-RESOURCE LANGUAGE

We follow the experimental setting described in Appendix E.3. For prompts, we adapt those shown in Figure 13 and Figure 14 by replacing the language tokens ‘German’ and ‘English’ with low-resource languages such as ‘Japanese’ and ‘Spanish’. For each language, we test each model on math problems written and answered in the original language using MGSM dataset.

#### B.3.3 INFERENCE WITH SPECULATIVE DECODING

Collaborative decoding is a technique that improves generation by combining outputs, such as probability distributions or logits, from multiple models at each step. Building on Speculative Decoding (Leviathan et al., 2023), Collaborative decoding via Speculation (CoS) (Fu et al., 2025) theoretically guarantees that the inference speed is never slower than standard collaborative decoding. Since our UniR framework utilizes the summation of logits, we seamlessly apply the CoS approach to enhance inference speed. We employ the Llama model family and the reasoning module  $\pi_r^{Math}$  trained on GSM8k dataset. To evaluate inference speed, we generate 100 tokens and measure the average token generation time across the entire GSM8k test benchmark.

## C ADDITIONAL EXPERIMENTS

### C.1 ANALYSIS OF THE REWARD DECOMPOSITION

To empirically validate our core assumption that the trajectory-level reward  $r(x, y)$  can be expressed as the sum of log-probabilities from the reasoning module  $\pi_r$  (Eq. 4), we evaluate our Llama-based model on the GSM8k test set. For each generated response, we compute  $\sum_t \log \pi_r(y_t | x, y_{<t})$  using the reasoning module and group the responses according to whether the final answer is correct ( $r = 1$ ) or incorrect ( $r = 0$ ). To examine the effect of training, we compare outputs from the pre-trained module and the UniR-trained module.

As shown in Figure 5, correct responses consistently achieve higher summed log-probabilities than incorrect ones in both settings. This correlation provides empirical evidence that the reasoning module internalizes the external reward signal by assigning higher probability to trajectories that lead to correct outcomes. Moreover, while the pre-trained module already shows moderate separation between correct and incorrect responses, this separation becomes substantially more pronounced

after UniR training, indicating that our framework strengthens the module’s ability to distinguish successful reasoning paths from unsuccessful ones.

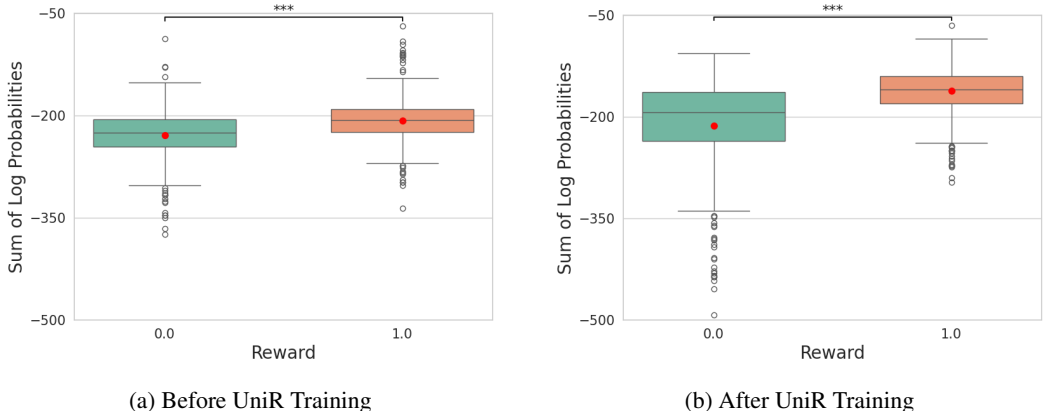


Figure 5: Summed log-probability distributions from the reasoning module ( $\pi_r$ ) on the GSM8k test set. (a) Before training, correct ( $r = 1$ ) and incorrect ( $r = 0$ ) responses exhibit moderate separation. (b) After UniR training, the separation becomes more pronounced, with correct responses receiving significantly higher probabilities. Red dots indicate means and asterisks denote statistical significance ( $p < 0.001$ ).

### C.2 IMPACT OF REASONING MODEL SIZE

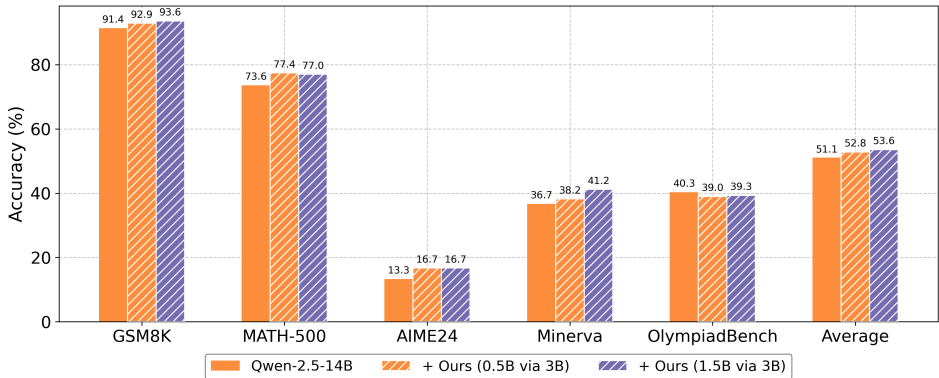


Figure 6: Transferability of the 0.5B  $\pi_r$  and 1.5B  $\pi_r$  reasoning modules when combined with a 14B frozen backbone model.

Table 9: Zero-shot Pass@1 performance on mathematical benchmarks comparing reasoning modules of different sizes (0.5B vs. 1.5B) trained with a 3B backbone.

Method	Trained Model	In-distribution		Out-of-distribution			Avg.	
		GSM8K	MATH-500	AIME24	Minerva	OlympiadBench		
Qwen2.5 3B	Backbone only	-	74.4 ± 1.0	44.2 ± 1.3	6.3 ± 3.7	19.2 ± 1.7	24.6 ± 0.9	33.7
	Backbone + 0.5B	-	70.9 ± 0.9	43.1 ± 1.2	6.0 ± 3.1	16.5 ± 1.9	22.5 ± 0.9	31.8
	<b>Ours</b>	0.5B	81.1 ± 0.4	62.2 ± 1.4	5.0 ± 3.2	24.7 ± 2.0	26.3 ± 4.8	39.9
	<b>Ours</b>	1.5B	84.5 ± 0.7	66.8 ± 1.4	7.7 ± 2.7	26.3 ± 2.0	28.2 ± 0.8	42.7

We conducted an ablation study to investigate the impact of reasoning module size, training a Qwen2.5-0.5B reasoning module  $\pi_r$  with a 3B backbone. As shown in Table 9, this smaller module achieved improved performance over the backbone alone, though it achieves lower performance

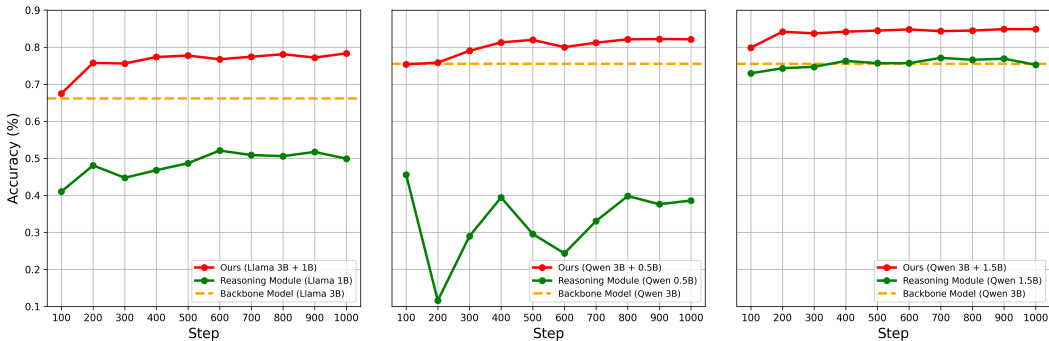


Figure 7: Performance comparison on the GSM8K dataset between standalone reasoning modules ( $\pi_r$ , green) and our UniR framework (red), evaluated against their respective backbone models (orange). (Left)  $\pi_r$  module using Llama-3.2-1B with a Llama-3.2-3B backbone. (Middle)  $\pi_r$  module using Qwen2.5-0.5B with a Qwen2.5-3B backbone. (Right)  $\pi_r$  module using Qwen2.5-1.5B with a Qwen2.5-3B backbone.

than the larger 1.5B module. Figure 6 further illustrates the transferability of the 0.5B reasoning module  $\pi_r$  to a 14B frozen backbone model  $\pi_b$ . The transferred 0.5B  $\pi_r$  led to an improved average performance (52.8) compared to the baseline  $\pi_r$  (51.1), showing that a module trained in a small-scale setting can generalize effectively to a larger model. Importantly, these results highlight that our main contribution is to demonstrate the ability to enhance the performance of larger models with comparatively small reasoning modules while prioritizing efficiency.

### C.3 ANALYSIS OF REASONING MODULE DURING TRAINING

We evaluated the performance of trained  $\pi_r$  reasoning modules, as depicted in Figure 7. This figure tracks the GSM8k dataset accuracy throughout the training process of reasoning modules designed for two different backbone architectures. The results indicate that the reasoning modules, when used in isolation, exhibit suboptimal performance. However, when these same reasoning modules are combined with their respective backbone models within our proposed framework, the resulting performance surpasses that of the backbone model alone. This highlights the efficacy of our learned reasoning modules in providing valuable guidance when integrated with a larger language model, leading to enhanced reasoning capabilities in the combined system.

### C.4 GUIDANCE EFFECT OF $\pi_r$ ON BACKBONE INFERENCE.

Examining the reasoning capabilities of the  $\pi_r$  and  $\pi_b$  modules in isolation highlights both their individual limitations and the synergy achieved when combined. As shown in Figure 19, the standalone  $\pi_r$  module fails to generate coherent and relevant reasoning for mathematical tasks. Likewise, the large backbone model alone struggles to reach correct conclusions on complex tasks. However, when the reasoning module is paired with a stronger backbone, it provides valuable guidance to the backbone’s inference process, resulting in significantly improved reasoning performance.

### C.5 COMPUTATIONAL EFFICIENCY IN TRAINING

**VRAM Usage.** Due to its structural decomposability, our framework substantially reduces VRAM usage by avoiding gradient updates to the large backbone model. While LoRA also reduces memory usage by restricting updates to a small subset of parameters, it still incurs substantial activation memory costs since the full computation graph must be stored for backpropagation (Zhang et al., 2023). In contrast, our approach updates only the lightweight reasoning module, drastically minimizing activation memory requirements and enabling significantly larger batch sizes or sequence lengths under the same hardware constraints.

We examine peak memory usage as a function of the number of tokens processed per step. Input length was fixed to 250 tokens (plus up to 5 output tokens), and batch size was gradually increased

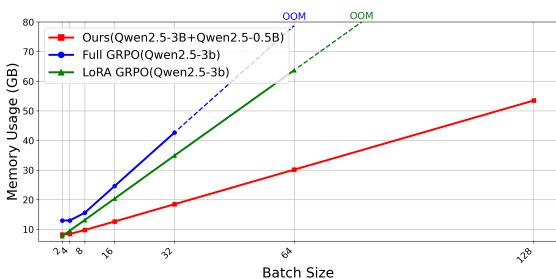


Figure 8: VRAM usage versus batch size under an 80GB constraint. Our method scales to batch size 128, while full fine-tuning and LoRA are limited, demonstrating memory efficiency for large batch.

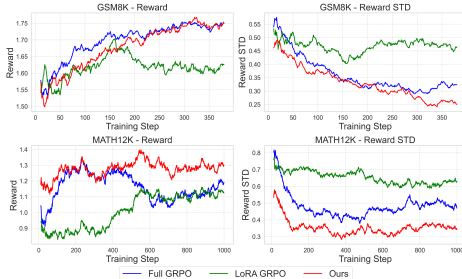


Figure 9: Training dynamics. We visualize the mean (Left) and the standard deviation (Right) of reward on the GSM8k (Top) and Math-12k (Bottom) during training.

Table 10: Comparison of training throughput and inference FLOPs.

Method	Training throughput (Sec. / step) ↓	Inference FLOPs (GFLOPs) ↓
Backbone	-	6.17
GRPO LoRA	4.67	6.17
GRPO Full	7.87	6.17
Ours	6.29	7.16

under an 80GB VRAM limit. As shown in Figure 8, UniR supports batch sizes up to 128, whereas LoRA and full fine-tuning are limited to 64 and 32, respectively. Although sequence length was fixed in this experiment, activation memory generally scales with both batch size and sequence length, making UniR more flexible for large-scale training. Moreover, UniR remains compatible with efficiency techniques such as quantization, caching, and adapter-based methods (e.g., LoRA applied to the reasoning module itself).

**Training Throughput and Inference FLOPs.** To further demonstrate the efficiency of UniR, we report training throughput and inference FLOPs, computed as  $FLOPs = num_{layers} \times (FLOPs_{Attn} + FLOPs_{MLP}) + 2 \times d_{model} \times vocab_{size}$ , for UniR and GRPO variants under identical hardware using the Qwen model. As shown in Table 10, GRPO Full is slower than the other methods since it requires backpropagation through the entire 3B model in training. Although UniR is slightly slower than GRPO LoRA—mainly due to inference latency during reward computation—this bottleneck can be mitigated via Collaborative Speculation (CoS), as discussed in Sec. 5.4.

### C.6 REWARD STABILIZATION AND TRAINING DYNAMICS

**Reward variance.** A notable empirical finding during training is that UniR not only achieves higher rewards compared to GRPO but also exhibits a smaller reward variance, as shown in Figure 9. Since the advantage calculation in GRPO normalizes the unbiased advantage  $r_i - \text{mean}\{r_1, \dots, r_G\}$  by its standard deviation (Eq. (7)), lower reward variance amplifies the magnitude of the effective advantage  $A_i$ . This means even small deviations from the mean reward translate into strong learning signals.

**Gradient norm.** We further analyze the gradient norm dynamics of UniR and GRPO Full over training steps on GSM8K and Math-500 using Qwen model. Specifically, we compute the average gradient norm every 100 training steps and report the maximum value within each 100-step window in parentheses in Table 11. These results demonstrate that UniR avoids the gradient explosions observed in GRPO Full, highlighting its training stability across both datasets.

These dynamics relate to the “Question-level difficulty bias” identified by (Liu et al., 2025b), where highly consistent rewards induce strong learning directions and thus lead to unstable policy updates. Indeed, GRPO often suffers from training instabilities when encountering policy collapse that can result in overfitting on *too easy* or *too hard* samples. By maintaining a fixed backbone model  $\pi_b$ ,

Table 11: Gradient norm dynamics of GRPO Full and UniR on GSM8K and Math-500. We report average gradient norms every 100 training steps, with maximum values within each window shown in parentheses. Bold values indicate maximum gradient norms exceeding 5.

Dataset	Method	100 steps	200 steps	300 steps	400 steps	500 steps
GSM8K	GRPO Full	1.44 ( <b>27.6</b> )	0.85 (2.74)	0.74 (2.18)	2.14 ( <b>84.6</b> )	93.9 ( <b>3888.2</b> )
	Ours	0.86 (1.50)	0.74 (1.30)	0.63 (1.19)	0.70 (1.38)	0.63 (1.19)
Math-500	GRPO Full	12.5 ( <b>798.6</b> )	21.1 ( <b>1958.2</b> )	1.18 ( <b>42.5</b> )	0.69 (3.77)	0.77 ( <b>7.11</b> )
	Ours	0.69 (2.22)	0.51 (1.32)	0.48 (0.96)	0.42 (1.31)	0.47 (1.10)

Table 12: Evaluation of using a small pre-trained reasoning model within the UniR framework. ‘DeepSeek 1.5B’ and ‘Qwen 3B’ represents DeepSeek-R1-Distill-Qwen-1.5B and Qwen2.5-3b-Instruct model, respectively.

Backbone	Reasoning Module	Train with UniR	In-distribution		Out-of-distribution			Avg.
			GSM8K	MATH-500	AIME24	Minerva	OlympiadBench	
DeepSeek 1.5B	-	X	71.7	51.6	0.0	17.7	23.3	32.8
Qwen2.5 3B	DeepSeek 1.5B	X	76.4	69.6	10.0	27.6	35.0	43.7
Qwen2.5 3B	DeepSeek 1.5B	O	83.8	73.4	10.0	30.1	37.3	46.9

UniR avoids such collapses and produces consistent responses with lower reward variance. This structural stability allows UniR to turn strong learning signals into constructive updates, rather than unstable divergence.

### C.7 DETAILED RESULTS OF USING A PRE-TRAINED MODEL AS A REASONING MODULE

This section presents the full results for using a pre-trained reasoning model as guidance. Table 12 breaks down the performance for each benchmark, showing that this approach improves scores even without UniR fine-tuning, with further gains achieved after optimization.

## D DECLARATION OF LLM USAGE

Our core methodology involves utilizing pre-trained LLMs in two key roles: (1) as a frozen backbone/reference model ( $\pi_b$ ) whose generation process is guided, and (2) potentially as the architecture for the smaller, trainable reasoning policy ( $\pi_r$ ) itself. The interaction between these LLM components, specifically the combination of their logits to form a guided policy, is a central and original part of our proposed method. We also use LLMs as evaluation methods, as described in the paper. In addition, LLMs were used during the writing stage only for language polishing and expression refinement. They were not used for research ideation, methodological design, or initial draft writing. All conceptual contributions and scientific content are solely the responsibility of the authors.

## E PROMPT TEMPLATE

For all models in the Llama family, we modified the default chat template by removing the knowledge cutoff and the generation time, as these fields vary depending on the time of execution and thus hinder reproducibility, and are also irrelevant to the task. For models in the Qwen family, however, we retained the default chat template as provided.

### E.1 MATH REASONING TASK.

The prompt template for mathematical problems follows the style of DeepSeek-R1 (DeepSeek-AI, 2025; Liu et al., 2025b). Our system prompts were designed to encourage the model to articulate its reasoning process before providing a final answer, illustrated in Figure 10, 11. Specifically, for the Math-12k dataset using Qwen model in Figure 11, we instructed the model to output its answer

1134 within a boxed LaTeX environment. This format was adopted to ensure that various valid answer  
1135 formats were correctly recognized and evaluated.

## 1137 E.2 TRANSLATION TASK.

1138  
1139 For translation, inspired by the findings of MT-R1-Zero (Feng et al., 2025), which show that train-  
1140 ing with and without explicit reasoning yields comparable performance, we use direct translation  
1141 prompts without explicit reasoning instructions detailed in Figure 12.

## 1143 E.3 COMPOSITIONAL TASK.

1144  
1145 We instructed the model to solve math problems written in German and to perform reasoning and  
1146 solving in English. The reasoning process was placed within `<think>` tags, and the final answer was  
1147 enclosed within `<answer>` tags, as illustrated in Figure 13. Using the prompt shown in Figure 14,  
1148 we evaluated the responses with GPT-4.1-nano-2025-04-14 on two dimensions: translation quality  
1149 and math accuracy. Translation quality measures how well the response was translated into English,  
1150 while math accuracy assesses accuracy using ground truth for a given problem.

1151 While rule-based evaluation is often used for math accuracy due to the structured format of answers,  
1152 we chose to use an LLM-based evaluator. This is because the math module is explicitly trained to  
1153 follow a specific output format, whereas the translation module is not. Therefore, using a rule-based  
1154 approach would unfairly favor the math side. To ensure a fair comparison between translation and  
1155 problem-solving accuracy, we opt for GPT-4.1-nano as an impartial judge capable of understanding  
1156 semantic correctness beyond strict formatting.

## 1157 E.4 MEDICAL TASK.

1158  
1159 For our medical tasks shown in Figure 15, we designed the system prompt to explicitly instruct the  
1160 model to produce a single numeric label, which must be enclosed within `<answer>` and `</answer>`  
1161 tags. The user prompt provides the specific context for each prediction task. This clear explanation  
1162 helps the model understand the task and the correct output format, reducing the chance of an incor-  
1163 rect or uninterpretable response.

### 1164 The prompt for GSM8K

1165  
1166 **System Prompt:** A conversation between User and Assistant. The user asks a question, and the  
1167 Assistant solves it. The assistant first thinks about the reasoning process and answer are enclosed  
1168 within `<think>` `</think>` and `<answer>` `</answer>` tags, respectively. Your response should be in  
1169 the following format: `<think>` `\n` Your reasoning here `\n` `</think>` `\n` `<answer>` `\n` answer here  
1170 `\n` `</answer>`. The reasoning process Note that respond by English, NOT use other languages.

1171  
1172 **User Prompt:** `{question}`

1173  
1174 Figure 10: For GSM8K, the prompt specifies a reasoning-then-answer format, where the model  
1175 is instructed to articulate its thought process within `<think>` tags, followed by the final answer  
1176 enclosed in `<answer>` tags. *question* denotes the question of the benchmark.

## 1179 F QUALITATIVE EXAMPLES

### 1180 F.1 EXAMPLE OF MATH TASK

1181  
1182 Based on Table 1, fine-tuning with GRPO demonstrates improved performance over the base model;  
1183 however, it also exhibits a tendency to generate longer outputs, which is alongside the results ob-  
1184 served in (Yu et al., 2025; Team et al., 2025; Arora & Zanette, 2025). In contrast, our proposed  
1185 method not only surpasses the performance of both the base model and the GRPO-tuned model  
1186 but also achieves this with a lower number of tokens to reach the final answer. Figure 16 contains  
1187 detailed examples that clearly show that our model is able to reach the correct answer in fewer steps.

1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241

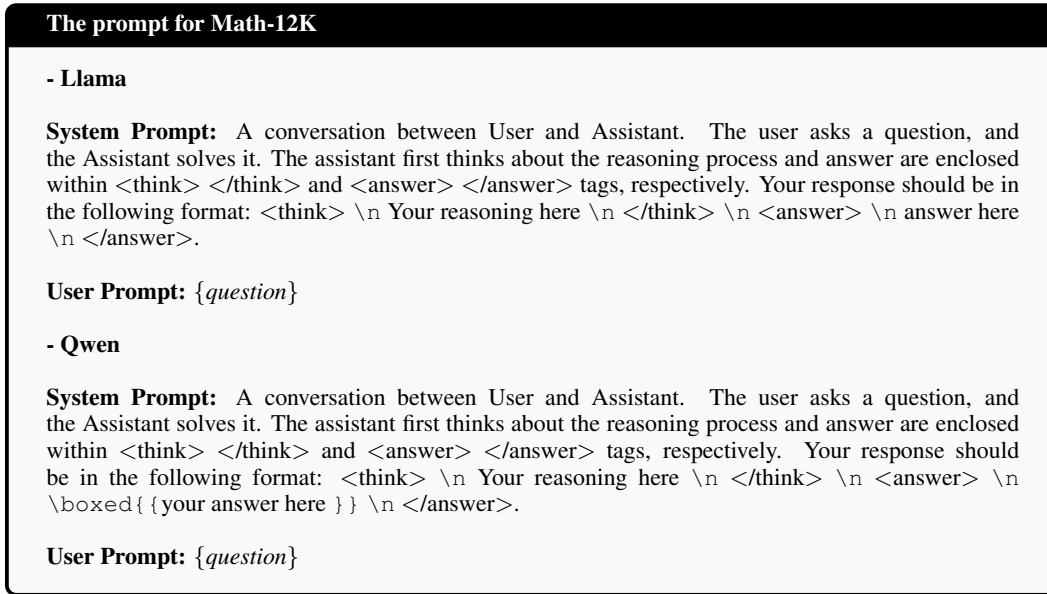


Figure 11: In the Math-12K prompt format used with the Llama and Qwen models, answers follow a reasoning-then-answer structure using `<think>` and `<answer>` tags. With Llama, we observed frequent failures to generate valid `\boxed{{}}` expressions, so we chose to omit this syntax. In contrast, when using Qwen, we explicitly enclose the final answer within `\boxed{{}}`, prompting the model to express its answer in a boxed format.

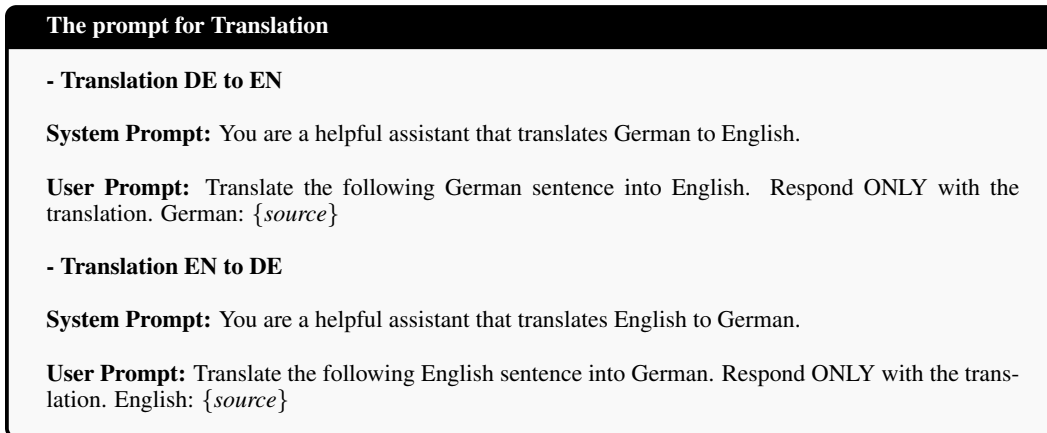


Figure 12: Prompt used for Translation German-to-English and English-to-German. `source` indicates the source sentences to be translated.

## F.2 EXAMPLE OF TRANSLATION TASK

As shown in Figure 17, our method generates more fluent and faithful translations in both directions. Compared to the base model and GRPO, our DE→EN translation correctly uses the natural phrase “What about...”, aligning with the tone of the original. On the EN→DE side, only our method preserves the correct compound adjective “künstlich intelligenten”, which is grammatically accurate and semantically precise. In contrast, baseline and GRPO variants either use awkward phrasing (“artificial intelligent system”) or deviate from standard German compound structures. This demonstrates the effectiveness of our method in producing higher-quality, context-aware translations.

1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295

**The prompt for German math questions into English**

**System Prompt:** You are a helpful assistant that translates German math questions into English and solves them. The user asks a math question in German. You first think about the reasoning process and answer. Both should be in English. Your response should be enclosed within `<think>` `</think>` and `<answer>` `</answer>` tags, respectively. Your response should be in the following format: `<think>` `\n` Your reasoning here `\n` `</think>` `\n` `<answer>` `\n` answer here `\n` `</answer>`.

**User Prompt:** German Question: `{question}`

Figure 13: System prompt used for German to English Math task: specifies the reasoning-then-answer format with `<think>` and `<answer>` tags. `question` indicates the math question in German.

**The prompt for scoring German math questions into English**

**- Evaluating Translation Quality**

**System Prompt:** You are a linguistic evaluator. Your job is to assess how well the model’s response is in English and whether it avoids using German. Score only the language usage, not the math correctness.

**User Prompt:** Here is a model’s response to a German math question. Please evaluate how well it used English instead of German. Assign a score between 1 and 5:  
 - 1: Mostly or entirely in German  
 - 3: Mixed, partly German, partly English  
 - 5: Fully in English, fluent and natural  
 Response: `{response}` `\n` End your response with only the score after ‘#####’. Example: ##### 4

**- Evaluating Accuracy**

**System Prompt:** You are a correctness evaluator. You will assess the factual accuracy of a math answer. You will return: `\n` AnswerCorrect: `<True or False>`

**User Prompt:** Here is a model’s response to a German math question.  
 Ground Truth Answer: `{gt_answer}` `\n` `{response}` `\n` Evaluate: `\n` - AnswerCorrect: `<True or False>`

Figure 14: Prompt used for scoring German-to-English Math task: Translation quality and Math Accuracy. `response` and `gt_answer` represent the model output and the ground truth of each math question, respectively.

### F.3 EXAMPLE OF COMBINING SPECIALIZED MODULES

Figure 18 presents an example of the response to a German math-solving in English task. We compare the results from the UniR framework using the math-specific reasoning module ( $\pi_r^{Math}$ ), the DE→EN translation-specific reasoning module ( $\pi_r^{Trans}$ ), and their combination. This example demonstrates that using only the math module fails to generate a response in English, while using only the translation module results in an incorrect solution. In contrast, incorporating both modules achieves both high translation quality and mathematical accuracy.

### F.4 EXAMPLE OF $\pi_r$ REASONING PERFORMANCE

Figure 19 illustrates a specific instance where the backbone model and the reasoning module fail to adequately comprehend the problem’s context. In contrast, our combined approach demonstrates a correct and coherent reasoning process for the same problem, highlighting the synergistic benefit of integrating the specialized reasoning module with the backbone model for improved contextual understanding and accurate inference.

1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349

**The prompt for Medical Task**

**System Prompt:** A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process and answer are enclosed within `<think>` `</think>` and `<answer>` `</answer>` tags, respectively. Your response should be in the following format: `<think>` `\n` Your reasoning here `\n` `</think>` `\n` `<answer>` `\n` *{single number answer}* `\n` `</answer>`.

**- Readmission Prediction**

**User Prompt:** Based on the patient’s conditions, procedures, and medications, is the patient likely to be readmitted to the hospital within 15 days of discharge? (Answer with 1 for yes and 0 for no) `\n` *{patient information}*

**- Length of Stay Prediction**

**User Prompt:** Based on the patient’s conditions, procedures, and medications, predict the patient’s ICU length of stay by categorizing it into one of the following 10 classes:`\n`0: ICU stay shorter than 1 day`\n`1: ICU stay lasting 1 to 7 days`\n`2: ICU stay longer than 7 days but less than or equal to 14 days`\n`3: ICU stay longer than 14 days`\n`Return a single number (0–3) indicating the predicted length of stay category`\n` *{patient information}*

**- Mortality Prediction**

**User Prompt:** Based on the patient’s conditions, procedures, and medications, predict whether the patient is likely to die during a subsequent hospital visit. (Answer with 1 for yes and 0 for no) `\n` *{patient information}*

Figure 15: The model was instructed to output a single numeric label within `<answer>` tags. Every user prompt included a clear explanation of the numeric labels.

## F.5 EXAMPLE OF FAILURE CASES

The primary cause of failure was the misinterpretation of semantic context. As illustrated in Figure 22, the first case shows the model overlooking the constraint “each kid,” resulting in an incorrect distribution of the remaining marshmallows. In the second case, the model misinterprets the phrase “3-piece place setting.” UniR incorrectly treats “3-piece” as meaning three of each item, whereas the intended meaning is that one set consists of three distinct items. These cases reflect the fundamental challenge of accurately translating semantic nuances into arithmetic operations.

1350

1351

1352 **Prompt:** Hannah’s city is having a big display of fireworks for the 4th of July. They’re going to set off 15  
 1353 boxes of 20 fireworks each. Hannah’s house is at the right angle to see 40% of the city’s fireworks. Hannah  
 1354 will also set off 3 boxes of 5 fireworks each in her backyard. How many fireworks will Hannah see in total?

1355

**Backbone Model:**

1356

1357

1358

1359

1360

1361

1362

1363

**GRPO:**

1364

1365

1366

1367

1368

1369

1370

1371

**Ours:**

1372

1373

1374

1375

1376

1377

1378 Figure 16: Chain-of-thought comparison on a GSM8k example: while the Backbone only and GRPO  
 1379 models both arrive at the correct answer with lengthy explanations, our approach produces the same  
 1380 result with a much more concise reasoning process.

1381

1382

1383

1384

1385

1386

**Source Prompt:**

1387

1388

EN : What about the artificially intelligent system in the movie "Her," Samantha?  
 DE : Was ist mit dem künstlichen intelligenten System im Film "Her", Samantha?

1389

**Backbone Model:**

1390

1391

1392

DE → EN: What’s with the artificial intelligent system in the movie "Her", Samantha?  
 EN → DE: Was ist mit dem künstlichen Intelligenzsystem im Film "Her", Samantha?

1393

**GRPO:**

1394

1395

DE → EN: What’s with the artificial intelligence system in the movie "Her"?  
 EN → DE: Was ist mit dem künstlichen Intelligenzsystem in dem Film "Her", Samantha?

1396

**Ours:**

1397

1398

DE → EN: What about the artificial intelligence system in the movie "Her," Samantha?  
 EN → DE: Was ist mit dem künstlich intelligenten System im Film "Her", Samantha?

1399

1400

1401

1402

1403

Figure 17: Example of bidirectional translation (German (DE) and English (EN)). Our approach produces more fluent and grammatically correct outputs in both directions, preserving natural phrasing and accurate compound structures.

1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457

**Prompt:**

Billy schält und schneidet Kartoffeln als Vorbereitung für die Zubereitung des Kartoffelsalats für seine große Grillparty anlässlich eines Familientreffens. Es ist ein großes Ereignis, also muss er 60 Kartoffeln schaffen. Die Kartoffeln haben ungefähr die gleiche Größe, also braucht es ungefähr gleich viel Zeit, um jede Einzelne zu schälen und zu schneiden. Billy braucht ungefähr eineinhalb Minuten, um eine Kartoffel zu schälen, aber nur ungefähr 5 Sekunden, um sie klein zu schneiden. Wie lange braucht Billy, um die Kartoffelzubereitung abzuschließen?

**Backbone Model +  $\pi_r^{Math}$ :**

<think> Zunächst müssen wir die Gesamtzeit berechnen, die Billy benötigt, um die Kartoffeln zu schälen und zu schneiden. Da es 60 Kartoffeln gibt und jede Kartoffel ungefähr die gleiche Größe hat, können wir die Zeit pro Kartoffel berechnen. Die Zeit, die Billy benötigt, um eine Kartoffel zu schälen, beträgt ungefähr 1,5 Minuten, also  $1,5 * 60 = 90$  Sekunden. Die Zeit, die Billy benötigt, um eine Kartoffel zu schneiden, beträgt ungefähr 5 Sekunden. Um die Gesamtzeit zu berechnen, müssen wir die Zeit pro Kartoffel für die Schälen und Schneiden addieren. Die Gesamtzeit beträgt also  $90 + 5 = 95$  Sekunden. Da es 60 Kartoffeln gibt, müssen wir die Gesamtzeit mit 60 multiplizieren, um die Gesamtzeit für die Vorbereitung aller Kartoffeln zu erhalten. Die Gesamtzeit beträgt also  $60 * 95 = 5700$  Sekunden. Um die Zeit in Minuten umzurechnen, können wir die Sekunden durch 60 teilen. Die Gesamtzeit beträgt also  $5700 / 60 = 95$  Minuten. </think> <answer> 95 Minuten </answer> </answer>

**Backbone Model +  $\pi_r^{Trans}$ :**

<think> To solve this problem, we need to calculate the total time it takes Billy to prepare the carrots for his party. Since he needs to prepare 60 carrots, and the time it takes to prepare one carrot is 1.5 minutes, we can multiply the number of carrots by the time it takes to prepare one carrot. However, we also need to consider the time it takes to cut the carrots into smaller pieces, which is 5 seconds per carrot. So, we need to add the time it takes to cut the carrots to the total time. First, let's calculate the time it takes to prepare one carrot: 1.5 minutes = 90 seconds. Now, let's calculate the time it takes to cut one carrot: 5 seconds. Since Billy needs to cut 60 carrots, we can multiply the time it takes to cut one carrot by 60: 5 seconds \* 60 = 300 seconds. Now, let's add the time it takes to prepare one carrot to the time it takes to cut one carrot: 90 seconds + 300 seconds = 390 seconds. Finally, let's convert the total time from seconds to minutes: 390 seconds / 60 = 6.5 minutes. So, Billy needs 6.5 minutes to prepare the carrots for his party. </think> <answer> Billy needs 6.5 minutes to prepare the carrots for his party.

**Backbone Model +  $0.5\pi_r^{Math} + 0.5\pi_r^{Trans}$ :**

<think> First, let's calculate the total time it takes to peel and chop one potato. It takes Billy 1.5 minutes to peel a potato, and 5 seconds to chop it. To convert the time to seconds, we multiply 1.5 minutes by 60 seconds per minute, which equals 90 seconds. Now, we add the time it takes to peel and chop one potato, which is 90 seconds + 5 seconds = 95 seconds. Since Billy needs to peel and chop 60 potatoes, we multiply the time it takes to peel and chop one potato by 60, which equals 95 seconds \* 60 = 5700 seconds. To convert this to minutes, we divide by 60, which equals 5700 seconds / 60 = 95 minutes. Therefore, Billy needs 95 minutes to peel and chop all 60 potatoes. </think> <answer> 95 minutes </answer>

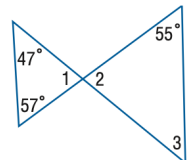
Figure 18: We illustrate outputs for a German math problem solved in English. (Top) Guidance with only the math module solves correctly but responds in German; (Middle) while using only a translation module ensures English output but fails at math. (Bottom) Their combination achieves both target language and correct solution.

1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511

	Backbone Model $\pi_b$ :	Reasoning Module $\pi_r$ :	Ours $\pi_b + \pi_r$ :
Prompt:	Tanya is teaching at school. She earns \$15 for every hour and an additional \$5 per day if she teaches more than 3 classes. On Monday she teaches 4 classes for 5 hours, and on Wednesday 2 classes for 2 hours. How much did Tanya earn for these two days of teaching?		
	<p>&lt;think&gt; First, let's calculate the earnings for Monday. Tanya teaches 4 classes for 5 hours each. Her base pay for teaching is \$15 per hour. So, her earnings from teaching alone on Monday would be <math>4 \text{ classes} * 5 \text{ hours/class} * \\$15/\text{hour}</math>....&lt;/think&gt;&lt;answer&gt; For Monday, Tanya's earnings from teaching are <math>4 * 5 * 15 = \\$300</math>. Since she teaches more than 3 classes, she earns an additional \$5, so her total for Monday is <math>\\$300 + \\$5 = \\$305</math>. For Wednesday, Tanya's earnings from teaching are <math>2 * 2 * 15 = \\$60</math>....The answer is <b>\$365</b>.&lt;/answer&gt;</p>	<p>&lt;think&gt; &lt;answer&gt; &lt;mark&gt; &lt;ul&gt; &lt;li&gt;On Monday, she earns <math>\\$15/\text{hour} * 4 \text{ classes} * 5 \text{ hours} = \\$300</math>&lt;/li&gt; &lt;li&gt; On Monday, she earns <math>\\$15/\text{hour} * 2 \text{ classes} * 2 \text{ hours} = \\$60</math>&lt;/li&gt; &lt;li&gt; On Wednesday, she earns <math>\\$15/\text{hour} * 2 \text{ classes} * 2 \text{ hours} = \\$60</math>&lt;/li&gt; &lt;li&gt; On Wednesday, she earns <math>\\$15/\text{hour} * 2 \text{ classes} * 2 \text{ hours} = \\$60</math>&lt;/li&gt; &lt;/ul&gt;&lt;li&gt; On Monday, she earns <math>\\$15/\text{hour} * 4 \text{ classes} * 5 \text{ hours} = \\$300</math>&lt;/li&gt; &lt;li&gt; On Monday, she earns <math>\\$15/\text{hour} * 2 \text{ classes} * 2 \text{ hours} = \\$60</math>&lt;/li&gt; &lt;li&gt; On Wednesday, she earns <math>\\$15/\text{hour} * 2 \text{ classes} * 2 \text{ hours} = \\$60</math>....&lt;/li&gt; &lt;/ul&gt;</p>	<p>&lt;think&gt; First, let's calculate the earnings for Monday. Tanya teaches 4 classes for 5 hours each, so she earns <math>\\$15 \text{ per hour} * 5 \text{ hours} = \\$75</math>. Additionally, since she teaches more than 3 classes, she earns an extra \$5 per day, which is \$5. So, her total earnings for Monday are <math>\\$75 + \\$5 = \\$80</math>. Next, ... Now, we add the earnings from both days to get the total earnings for the two days: <math>\\$80 \text{ (Monday)} + \\$30 \text{ (Wednesday)} = \\$110</math>.&lt;/think&gt;&lt;answer&gt; <b>\$110</b>&lt;/answer&gt;</p>

Figure 19: Reasoning Performance of the  $\pi_r$  Module. (Left) The backbone model  $\pi_b$  and (Middle) the standalone reasoning module  $\pi_r$  produce **incorrect, repetitive and logically flawed reasoning**. (Right) When combined, it generates **coherent reasoning and arrive at the correct solution**, showing the effectiveness of the modular guidance.

Q: Find angle 2



Qwen2.5-VL-3B-Instruct

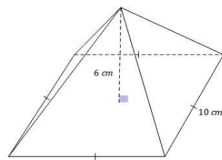
<think> In the given triangle, we have two angles labeled as 47° and 57°. The sum of the angles in a triangle is always 180°. Therefore, we can find the measure of angle 1 by subtracting the sum of the other two angles from 180°. Once we have the measure of angle 1, we can find the measure of angle 2 by subtracting the measure of angle 1 from 180°.</think><answer> To find the measure of angle 1, we subtract the sum of the other two angles from 180°:  $180^\circ - (47^\circ + 57^\circ) = 76^\circ$ . Therefore, the measure of angle 1 is 76°. To find the measure of angle 2, we subtract the measure of angle 1 from 180°:  $180^\circ - 76^\circ = 104^\circ$ . Therefore, the measure of angle 2 is **104°**.</answer>

UniR (+ Qwen2.5-0.5B-Instruct)

<think> The given figure is a triangle with angles labeled as follows:  $\angle 1 = 47^\circ$ ,  $\angle 2 = 55^\circ$ , and  $\angle 3 = 57^\circ$ . The sum of the angles in a triangle is  $180^\circ$ . Therefore, we can find  $\angle 2$  by subtracting the sum of the other two angles from  $180^\circ$ :  $\angle 2 = 180^\circ - (\angle 1 + \angle 3) = 180^\circ - (47^\circ + 57^\circ) = 180^\circ - 104^\circ = 76^\circ$ .</think><answer> **76°**</answer>

(a) Geometry3k

Q: Given the following solid. The length of the base is 10cm. Find the volume of the solid.



Qwen2.5-VL-3B-Instruct

<think> To find the volume of the solid, we need to determine the area of the base and then multiply it by the height of the solid. The base of the solid is a triangle with a base of 10 cm and a height of 6 cm. The area of a triangle is given by the formula:  $\text{Area} = \frac{1}{2} \times \text{base} \times \text{height}$ . Substituting the given values:  $\text{Area} = \frac{1}{2} \times 10 \times 6 = 30 \text{ cm}^2$ . The height of the solid is the perpendicular distance from the base to the apex. Since the solid is a pyramid, the height is the same as the height of the triangle, which is 6 cm. The volume of a pyramid is given by the formula:  $\text{Volume} = \frac{1}{3} \times \text{Base Area} \times \text{Height}$ . Substituting the values we have:  $\text{Volume} = \frac{1}{3} \times 30 \times 6 = 60 \text{ cm}^3$ .</think><answer> The volume of the solid is **60 cm³**.</answer>

UniR (+ Qwen2.5-0.5B-Instruct)

<think> To find the volume of the solid, we need to determine the area of the base and the height of the solid. The base is a rectangle with a length of 10 cm and a width of 10 cm, so the area of the base is  $10 \times 10 = 100 \text{ cm}^2$ . The height of the solid is the perpendicular distance from the base to the apex, which is given as 6 cm. The volume of a pyramid is given by the formula  $V = \frac{1}{3} \times \text{Base Area} \times \text{Height}$ . Substituting the values, we get  $V = \frac{1}{3} \times 100 \times 6 = 200 \text{ cm}^3$ .</think><answer> **200**</answer>

(b) MathVerse

Figure 20: Illustrative examples of responses from the baseline VLM and our UniR-extended framework on tasks from the (Left) Geometry3k and (Right) MathVerse benchmarks.

1512  
1513  
1514  
1515  
1516

---

**Patient Information:**  
1517 Patient ID: \*\*\*\*  
1518 Visit 0:  
1519 Conditions:  
1520 1. other complications of birth; puerperium affecting management of mother  
1521 2. early or threatened labor  
1522 3. other pregnancy and delivery including normal  
1523 ...  
1524 Procedures:  
1525 1. cesarean section  
1526 Medications:  
1527 1. vitamin b 12 and folic acid  
1528 2. antiinfectives and antiseptics, excl. combinations with corticosteroids  
1529 3. lipid modifying agents, plain  
1530 ...

---

**Backbone Model:**  
1531 <think> The patient has a history of complications related to pregnancy and delivery, which could  
1532 indicate a high risk of readmission. The patient also has a history of mycoses, which could be  
1533 a sign of an underlying health issue that might require further medical attention. The patient has  
1534 also undergone a cesarean section, which is a significant procedure that could lead to complications.  
1535 The patient is on a variety of medications, including corticosteroids, antacids, and analgesics, which  
1536 could indicate ongoing health issues. However, **the list of medications does not include any specific  
medications that are typically used for readmission prevention.** </think> <answer> 0 </answer>

---

**Backbone Model +  $\pi_r^{Read}$ :**  
1538 <think> To determine if the patient is likely to be readmitted to the hospital within 15 days of discharge, I need  
1539 to consider several factors including the patient's current health status, the procedures and medications they are  
1540 undergoing, and the conditions they are currently experiencing. The patient's conditions, such as complications  
1541 of pregnancy, early or threatened labor, and other complications of birth, suggest that they are in a high-risk  
1542 category for readmission. The procedures, such as cesarean section, also indicate a high-risk situation. The  
1543 medications, including antiinfectives and antiseptics, and lipid modifying agents, suggest that the patient is at  
1544 risk for infections and other complications. The patient's current medications, such as vitamin B12 and folic  
1545 acid, and iron antianemic preparations, suggest that they are at risk for anemia. The patient's current medica-  
1546 tions, such as corticosteroids for systemic use, and other analgesics and antipyretics, suggest that they are at  
1547 risk for side effects and complications. The patient's current medications, such as antipruritics, antihistamines,  
1548 and throat preparations, suggest that they are at risk for allergic reactions and other side effects. The patient's  
1549 current medications, such as digestive enzymes, suggest that they are at risk for digestive issues. </think>  
<answer> 0 </answer>

---

**Backbone Model +  $\pi_r^{Read}$  +  $\pi_r^{LOS}$ :**  
1550 <think> The patient is likely to be readmitted to the hospital within 15 days of discharge due to the presence  
1551 of multiple complications and procedures, including cesarean section, which can lead to postoperative compli-  
1552 cations. The patient's conditions and medications, such as vitamin B12 and folic acid, antiinfectives, and lipid  
1553 modifying agents, are also relevant to the patient's health status. The patient's history of early or threatened la-  
1554 bor, mycoses, esophageal disorders, and headache, including migraine, also suggest a high risk of readmission.  
1555 Additionally, the patient's use of corticosteroids for systemic use, plain, and other analgesics and antipyretics in  
1556 atc, as well as the use of beta-lactam antibacterials, penicillins, and other therapeutic products, further **increases  
the likelihood of readmission.**</think> <answer> 1 </answer>

---

1558 Figure 21: Example of Readmission Prediction. (Top) the output of the backbone model; (Middle)  
1559 the backbone model with the Readmission module; (Bottom) and the backbone model with both the  
1560 Readmission and Length of Stay modules. The model with both specialized modules successfully  
1561 identifies risk factors and provides an accurate final prediction.  
1562  
1563  
1564  
1565

1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619

**Prompt:**

John and DeSean bought a bag of marshmallows to make s'mores together. The bag has 35 marshmallows. Each S'more uses one marshmallow. If John makes 9 S'mores, DeSean makes 9 S'mores, and they dropped 3 marshmallows on the ground, how many S'mores can each kid have with the marshmallows left in the bag?

**Ours:**

<think> First, let's determine the total number of marshmallows used by John and DeSean. Each S'more uses one marshmallow, so together they made  $9 + 9 = 18$  S'mores. They started with 35 marshmallows and dropped 3 on the ground, so they have  $35 - 3 = 32$  marshmallows left. Next, we need to find out how many marshmallows are left after they've made their S'mores. Since they used 18 marshmallows, the remaining marshmallows are  $32 - 18 = 14$ . Now, we need to determine how many S'mores each can make with the remaining marshmallows. **Since each S'more uses one marshmallow, each can make 14 S'mores with the remaining marshmallows. Therefore, John and DeSean can each make 14 S'mores with the marshmallows left in the bag.** </think> <answer> 14 </answer>

**Prompt:**

Avery needs to buy a 3 piece place setting (dinner & salad plate and a bowl) for her holiday dinner. She's having 12 people over for dinner. If the dinner plates cost \$6.00 each and bowls each cost \$5.00 and the salad plates cost \$4.00, how much will she spend on place settings?

**Ours:**

<think> To determine the total cost of the place settings, we need to calculate the cost for each type of plate and bowl and then sum them up. 1. Dinner Plates: - **Avery needs 3 dinner plates. - Each dinner plate costs \$6.00. - Total cost for dinner plates =  $3 * \$6.00 = \$18.00$ .** 2. Salad Plates: - **Avery needs 3 salad plates. - Each salad plate costs \$4.00. - Total cost for salad plates =  $3 * \$4.00 = \$12.00$ .** 3. Bowls: - **Avery needs 3 bowls. - Each bowl costs \$5.00. - Total cost for bowls =  $3 * \$5.00 = \$15.00$ .** Now, we sum up the costs of all the place settings: - **Total cost = Cost of dinner plates + Cost of salad plates + Cost of bowls- Total cost =  $\$18.00 + \$12.00 + \$15.00 = \$45.00$ . Therefore, Avery will spend \$45.00 on place settings.** </think> <answer> \$45.00 </answer>

Figure 22: Examples of Failure Cases