000 001

002

# **Reinforcement Learning Teachers of Test Time Scaling**

Anonymous Authors<sup>1</sup>

### Abstract

Training reasoning language models (LMs) with reinforcement learning (RL) for one-hot correctness inherently relies on the LM being able to explore and solve its task with some chance at initialization. Furthermore, a key use case of reason-015 ing LMs is to act as teachers for distilling new students and cold-starting future RL iterations rather than being deployed themselves. From these con-018 siderations, we introduce a new framework that 019 avoids RL's exploration challenge by training a 020 new class of Reinforcement-Learned Teachers (RLTs) focused on yielding the most effective downstream distillation. RLTs are prompted with both the question and solution to each problem, and tasked to simply "connect-the-dots" with de-025 tailed explanations tailored for their students. We train RLTs with dense rewards obtained by feed-027 ing each explanation to the student and testing its 028 understanding of the problem's solution. In prac-029 tice, the raw outputs of a 7B RLT provide higher 030 final performance on competition and graduatelevel tasks than existing distillation and coldstarting pipelines that collect and postprocess the reasoning traces of orders of magnitude larger 034 LMs. Furthermore, RLTs maintain their effec-035 tiveness when training larger students and when applied zero-shot to out-of-distribution tasks, unlocking new levels of efficiency and re-usability for the RL reasoning framework. 039

#### 1. Introduction

041

043

045

046

047

049

050

051

052

053

054

Exploration is one of the critical challenges in reinforcement learning (RL) and has been a core focus of its literature (Bellemare et al., 2016; Salimans & Chen, 2018; Ecoffet et al., 2019). Sparse rewards cannot yield any learning signal unless the agent is already capable of solving the



*Figure 1.* RLTs provide better student distillation and RL coldstarts than orders of magnitude larger LMs across competition and graduate-level tasks (AIME, MATH, GPQA).

given task at initialization. With the rise of RL for openended reasoning (RL reasoning) inducing a new form of language model (LM) scaling beyond prompt-engineering and search (Kojima et al., 2022; Snell et al., 2024), exploration has re-emerged as a key challenge. A canonical motivation for RL is the potential to bootstrap from partial solutions guided by the reward function and learn entirely new tasks from scratch. However, the nature of one-hot correctness rewards used in the RL reasoning framework fails to provide a dense form of guidance, focusing instead on reinforcing correct responses in the initial model's pool of pass-at-k attempts - without true extrapolation beyond the LM's initial latent abilities (Yue et al., 2025). As a result, mostly large, already-capable models have been shown to improve consistently beyond cheaper and simpler supervised optimization (Guo et al., 2025).

Due to this fundamental limitation, coupled with RL's training instability, distillation has emerged as another ubiquitous component of current reasoning paradigms. In this case, the test-time role of LMs trained with RL is to act as a *teacher* providing instructive reasoning traces for a *student* to solve new problems. This teacher-student paradigm is widely adopted both to train smaller, less-capable models (Muennighoff et al., 2025; Xu et al., 2025) and even to cold-start future RL iterations for better final convergence with the teacher's own initial checkpoint acting as the student (Guo et al., 2025; Shen et al., 2025). However, the problemsolving skills reinforced by correctness-based rewards have been shown not to be entirely aligned with the goal of downstream distillation (Li et al., 2025a). To account for this

<sup>&</sup>lt;sup>1</sup>Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

mismatch, current pipelines significantly rely on heuristicdriven postprocessing of the teacher's outputs for effective
student transfer (Li et al., 2025a).

058 Based on these considerations, we propose a framework that 059 avoids RL's exploration challenge with a new class of spe-060 cialized Reinforcement-Learned Teachers (RLTs) trained 061 specifically to yield effective downstream distillation. Our 062 main intuition is simple: the ability of real-world teachers 063 is not measured by whether they can come up on their own 064 with complex theorems, proofs, or answers from scratch. 065 Instead, what matters is their ability to make use of read-066 ily available solutions and devise instructive explanations 067 for their students. Thus, we depart from the traditional RL 068 reasoning framework, tasking a model to first think and 069 then come up with a new solution for the first time. In-070 stead, RLTs are tasked with the easier problem of providing an effective explanation with the problem's solution al-072 ready given within their prompt. We train RLTs with dense rewards using the student's log probabilities to assess its understanding of each problem's ground-truth solution from 075 the teacher's explanations, and the interpretability of the 076 logical leaps in the explanations themselves. 077

078 By distilling students from the raw outputs of a lightweight 079 RLT with 7B parameters, we demonstrate significantly higher performance than using existing pipelines relying on 081 reasoning LMs with orders of magnitude more parameters 082 (Figure 1). We show our framework provides superior bene-083 fits even when distilling the RLT's explanation to train larger 32B students and to cold-start traditional RL optimization. Furthermore, we showcase how RLTs can be transferred 086 zero-shot to new domains and still produce effective distil-087 lation datasets that yield yet better final students than direct 088 RL with access to the task's reward. Overall, these results 089 highlight the potential of our new method for overcoming 090 the large costs of RL by focusing on stronger, smaller, and 091 highly reusable specialized teachers. In summary, Our main 092 contributions are threefold:

- We introduce the RLT framework, tackling exploration with a simpler dense-reward that aligns the objective of RL training to providing effective distillation.
  We show how distilling the raw outputs of a 7R PLT
  - We show how distilling the raw outputs of a 7B RLT outperforms training students trained on postprocessed reasoning traces from orders of magnitude larger LMs.
  - We demonstrate that RLTs also allow for better cold-starts for traditional RL, distillation to larger students, and even zero-shot transfer to new reasoning domains.

#### 2. Inducing reasoning in language models

#### 2.1. Reinforcement learning

097

098

099

100

104

105

106

107

109

The RL post-training recipe for inducing reasoning behavior was recently popularized by the DeepSeek R1 line of work (Wang et al., 2023; Shao et al., 2024; Guo et al., 2025). By fine-tuning on a dataset of questions  $D = \{q_1, \ldots, q_N\}$ with verifiable solutions  $\{s_1, \ldots, s_N\}$ , Guo et al. (2025) show effective "reasoning" behavior emerges out of a 671Bparameter LM (DeepSeek-AI, 2024), significantly pushing its performance on challenging math and coding tasks. Their training is conducted with GRPO (Shao et al., 2024), an online RL algorithm that foregoes the use of a critic model with a simple Monte-Carlo value estimate. GRPO prompts the LM  $\pi_{\theta}$  to produce a set of G >> 1 "grouped" outputs  $o_1, \ldots o_G$  for each sampled question  $q \in D$ , optimizing:

$$J(\theta) = \mathbb{E}_{q \sim D, \{o\}_1^G \sim \pi_\theta(\cdot|q)} \left[ \frac{1}{G} \sum_{i=1}^G \left( A_i - \beta \mathbb{D}_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}}) \right) \right].$$
(1)

Here, the "advantages"  $A_i$  are obtained by normalizing each output's reward  $r_i$  within each group:

$$A_{i} = \frac{r_{i} - \text{mean}(\{r_{1}, \dots, r_{G}\})}{\text{std}(\{r_{1}, \dots, r_{G}\})}.$$
(2)

A key component of their design is a system prompt that asks the LM to decompose each generated output  $o_i$  into two separate formatted sections separated by <think> and <solution> tags, denoted  $t_{o_i}$  and  $s_{o_i}$ . This structure is forced by assigning rewards  $r_i = -1$  to unformatted completions,  $r_i = -0.5$  to wrong but formatted completions, and  $r_i = 1$  only to correct and formatted completions. Training with this strategy, Guo et al. (2025) show the LM's completion length gradually grows with reflection, verification, and self-correction steps emerging, mirroring human chain-of-thoughts.

#### 2.2. Supervised distillation

Supervised distillation is another critical step used to train recent reasoning models to complement RL's shortcomings. For any online RL objective like Equation 1 to avoid collapse, the model must already possess a non-trivial chance of producing correct responses with non-zero gradients at initialization. This defining property makes the RL objective much less applicable than cross-entropy objectives that always include the correct response's information in the model's gradients. As a consequence of this dichotomy, distilling the reasoning traces of large RL-trained LMs with supervised learning is not only cheaper but has also been shown to be notably more effective than performing RL itself for inducing reasoning in smaller, less-capable models (Guo et al., 2025; Muennighoff et al., 2025; Ye et al., 2025; Li et al., 2025a; Xu et al., 2025). Furthermore, RL appears prone to instabilities and output degradation, especially during extended training sessions. Due to this second limitation, DeepSeek R1 and several other models (Guo et al., 2025; Shen et al., 2025) perform RL training over multiple iterations. This is done by using the RL-trained models at the end of each intermediate iteration only to, once again, collect distillation datasets used for "cold-starting"

0	RL reasoning format	RL teaching format		
.1	System	System		
2	Your role as an assistant involves thoroughly exploring questions	Your role as a teacher involves thoroughly exploring questions and		
3	through a systematic long thinking process	solutions with systematic long explanations		
ŀ	User	User		
Ď	Return your final response within . The domain of the function $Sa(x) = xA + Ax^2 + AS$ is $S(0)$ \infty) $S(0)$ what is the range?	Return your final response within $boxed$ . The domain of the function $Sa(x) = xA + 4xA^2 + 4S$ is $S(0)$ infty). What is the range?		
-	$function \varphi(x) = x + x + x + x + y + z + y + z = \varphi(x) + y + z + y + z = y + y + z = y + y + z = y + y + z = y + y + z = y + y + z = y$	<pre> vinction \$q(x) = x 4 + 4x 2 + 45 is \$(0, (inty)\$). What is the range:</pre>		
	Expected completion	the range of \( q(x) \) is \(\boxed{[4, \infty)}\).		
	Assistant	<pre></pre>		
	< begin_of_thought >	Expected completion		
)	<pre>ckay, so i need to into the range of the function <math>q(x) = x^2 + 4x^2 + 4</math>,</pre> <pre></pre>	Assistant		
)	<pre>&lt; begin_of_solution &gt;</pre>	< begin_of_explanation >		
, L	the range of $\langle q(x) \rangle$ is $\langle bxed{[4, infty]} \rangle$ .	Okay, so I need to find the range of the function $q(x) = x^4 + 4x^2 + 4$ ,		
		Clend_or_expranacion/>		

Figure 2. Left: RL format asking an LM to think and solve hard problems from scratch. Right: RLT format asking an LM to produce instructive step-by-step explanations given access to the solutions.

125 their original initial checkpoint and obtain a stronger initialization point for the next RL iteration. 126

122

123

124

145

146

147

148

149

164

127 Constructing a dataset of distillation prompts  $D_{SD}$  = 128  $\{d_1,\ldots,d_N\}$  involves using the RL-trained LM  $\pi_{\theta}$  with 129 its reasoning system prompt to answer a corpus of verifi-130 able questions, which can be chosen with several heuris-131 tics (Muennighoff et al., 2025; Ye et al., 2025; Li et al., 132 2025a; Xu et al., 2025). The LM's output reasoning traces 133 for each question  $o \sim \pi_{\theta}(\cdot|q)$  are then filtered by compar-134 ing them with the ground-truth solutions to ensure their 135 correctness. Commonly, these reasoning traces are also 136 post-processed via additional "manual" steps of refinements, 137 such as asking other closed-sourced LMs to remove gram-138 matical issues and refactor the reasoning steps into a nicer, 139 consistent format. In fact, Li et al. (2025a) even argues that 140 the structure and format of the thinking data is a critical 141 component to make weaker models actually understand and 142 learn how to reason from distillation, potentially even more 143 important than correctness itself. 144

#### 3. Reinforcement learning teachers

#### 3.1. The implications of training teacher models as students

150 We distinguish two separate training and inference "roles" that can be performed by LMs in the modern reasoning 151 152 framework. As detailed in Section 2, after RL training, LMs 153  $\pi_{\theta}$  are often not deployed themselves but rather used to 154 obtain reasoning distillation datasets for fine-tuning weaker 155 models and cold-starting future RL iterations. Thus, these 156 models can be effectively seen as teachers, providing ex-157 **planations** for future **student** models  $\pi_s$  to learn from.

158 This teacher-student paradigm highlights a potential mis-159 match between the objective used for RL training and the 160 teacher's test-time role. In traditional settings, teachers are 161 trained with sparse correctness rewards to improve their 162 ability to solve hard problems from scratch. This objective 163

not only precludes the applicability of RL training for tasks beyond the base model's original capabilities, due to its inherent exploration challenge, but is also not aligned with the teacher's actual end goal: producing reasoning traces from which students  $\pi_s$  can learn the necessary skills to derive correct solutions themselves. Based on these considerations, we propose a different training framework for RL reasoning models to be deployed as teachers that avoids RL's exploration challenge and breaks this objective mismatch. Our framework comprises a much easier task formulation, a dense reward objective, and a carefully designed training recipe, allowing us to learn a new class of specialized Reinforcement Learned Teachers (RLTs).

#### 3.2. Aligning the task of teacher models

In the traditional RL paradigm, the solution  $s_i$  to each problem is never explicitly provided to the model and is only employed for checking the correctness of the corresponding solutions within the LM's completions  $s_{\alpha_i}$ . Precluding direct access or information about the solutions aligns training with the test-time objective of solving entirely new test problems from scratch, but is precisely what makes exploration challenging, as the model receives no gradients until its first successful attempt. Our key observation, however, is that the test-time "teaching" objective of producing effective distillation datasets  $D_{SD}$  for questions with known solutions, can be greatly facilitated by explicitly providing access to such solutions - as is the case for real-world teachers, who can rely on access to readily available solutions and, thus, focus entirely on how instructive their explanations are for students.

To this end, as illustrated in Figure 2, RLTs are prompted with a new formatting style, providing both the question and solution to each problem as inputs, and are tasked to produce instructive step-by-step explanations, connecting the dots between the two. We design our prompts to allow direct reuse of the teacher's outputs for student distillation while keeping the task natural, appending the solution to-



*Figure 3.* The tokens from the RLT's explanations are copied intothe student format to measure its understanding with our reward.

190 kens  $s_i$  and tags to the RLT's system prompt and input 191 question before generating each completion. At test-time, 192 constructing the corresponding question completions for the 193 student distillation datasets  $d_i \in D_{\text{SD}}$  is then as simple as 194 extracting the think tokens from the teacher's outputs by 195 replacing surrounding explanation with think tags 196 and appending back the solutions  $s_i$ .

#### **3.3. Evaluating the quality of explanations**

199 The reward function to train RLTs is made of two terms 200 to incentivize explanations  $o_i$  that lead the student  $\pi_s$  to 201 recover correct solutions  $s_i$  and are also logical continua-202 tions from questions alone under the student's perspective. In particular, following the procedure from the previous 204 subsection, for each completion  $o_i$  from the teacher  $\pi_{\theta}$ , we 205 extract the think tokens  $t_{o_i}$  and format the corresponding 206 student distillation prompt  $d_i$  by prepending the question  $q_i$ and appending the ground-truth solution  $s_i$ . As illustrated in Figure 3, each distillation prompt is then fed as input to the 209 student model to obtain a set of per-token log probabilities, 210 which are processed into our two reward terms as follows: 211

212<br/>213i.  $r_i^{SS}$ : quantifying the student  $\pi_s$  understanding of the<br/>solutions  $s_i$  given the question  $q_i$  and think tokens  $t_{o_i}$ <br/>in context. This first reward term is computed with<br/>the student's log probabilities over the solution tokens,<br/>reduced with both average and minimum operations:

217

218

219

$$r^{\rm SS}(o_i, s_i, q_i) = \arg\{\log \pi_s^{s_i}\} + \alpha \min\{\log \pi_s^{s_i}\},$$
  
where  $\pi_s^{s_i} = \pi_s(s_i \mid t_{o_i}.q_i).$  (3)

ii.  $r_i^{\text{KL}}$ : quantifying whether the think tokens  $t_{o_i}$  themselves are interpretable logical continuations from the student's perspective as compared with the teacher's. This second reward term is computed with the KL divergence over the same think tokens between the teacher's distribution (under the RLT's format with both  $q_i$  and  $s_i$  in context) and the student's (with only the question  $q_i$  in context), reduced with both average and maximum operations:

$$r^{\mathrm{KL}}(o_{i}, s_{i}, q_{i}) = \operatorname{avg}\left\{\mathbb{D}_{\mathrm{KL}}\left(\pi_{\theta}^{t_{o_{i}}} \| \pi_{s}^{t_{o_{i}}}\right)\right\} + \alpha \max\left\{\mathbb{D}_{\mathrm{KL}}\left(\pi_{\theta}^{t_{o_{i}}} \| \pi_{s}^{t_{o_{i}}}\right)\right\},$$
where  $\pi_{s}^{t_{o_{i}}} = \pi_{s}(t_{o_{i}} \mid q_{i}), \quad \pi_{\theta}^{t_{o_{i}}} = \pi_{\theta}(t_{o_{i}} \mid s_{i}, q_{i}).$  (4)

Finally, the RLT rewards are obtained by combining these two terms with a weighting coefficient  $\lambda$ :

$$r_i^{\text{RLT}} = r^{\text{SS}}(o_i, s_i, q_i) - \lambda r^{\text{KL}}(o_i, s_i, q_i)$$
(5)

Each term in our reward function serves a precise purpose. First, optimizing  $r^{SS}$  will produce explanations containing think tokens  $t_{o_i}$  that maximize the student's likelihood of reaching the correct solution  $s_i$ . However, this term alone does not differentiate between explanations that guide the student step-by-step and those that increase the solution's likelihood without a logical path that can be learned from. An extreme instance of the latter would be an explanation simply repeating the solution tokens to increase likelihood, failing to provide general examples of reasoning methods that can be applied when approaching new problems. Thus, introducing  $r^{\text{KL}}$  fills precisely this gap, aligning the teacher's distribution toward the student's such that each think token in the output explanations cannot have too low probability when formatted in the distillation prompt  $d_i$  with only the question  $q_i$  and the previous think tokens in context. Intuitively, introducing this term regularizes for each step in the logical path traced by the teacher's explanation to still make sense in the "student's mind" given only its prior understanding and the question itself. Additionally, combining the average with min/max reductions ensures the rewards do not forego any individual token, regardless of the solution length or the number of think tokens in the teacher's explanations. For instance, their omission could bias  $r^{SS}$ based on the length of the solutions or lead the teacher to prefer long explanations only to reduce the influence on  $r^{\text{KL}}$ of hard but necessary individual logical steps. For further discussion, we refer to Appendix D, where we empirically analyze and validate all these design choices.

#### 3.4. The RLT training paradigm

The RLT framework can be used with any RL algorithm (e.g., (Schulman et al., 2017; Ahmadian et al., 2024)) with minimal modifications to the LM's conditioning and reward, as described in the above subsections. In this work, we employ the simple GRPO recipe detailed in Section 2. where

the advantage is computed using the RLT reward function from Equation 5. Unlike for correctness-based rewards, our learning signal is inherently dense, providing informative rankings to the RLT's output even before achieving any task expertise. This fundamental difference greatly facilitates our optimization, akin to how heuristically shaped rewards enabled RL agents to learn entirely new behaviors for videogames and robotics tasks (Mnih et al., 2016; 2013).

#### 4. Experiments

229

230

231

274

#### 4.1. Training, distillation, and evaluation

232 We train RLTs on the set of questions and solutions se-233 lected by Li et al. (2025a) based on their level of challenge. 234 This dataset comprises less than 17K math and coding prob-235 lems originally used for distilling filtered and post-processed 236 reasoning traces collected from QwQ (Team, 2025b) and 237 DeepSeek R1 (Guo et al., 2025). In contrast, the RLTs 238 we consider are orders-of-magnitude smaller models, all 239 trained starting from the Qwen2.5-7B-Instruct LM (Hui 240 et al., 2024). We precede our RL phase with a short super-241 vised fine-tuning phase to familiarize RLTs with their new 242 system prompt and input format using the open reasoning 243 dataset released by Labs (2025). During RL, we compute 244 the reward for the RLT explanations using another small 245 Qwen-7B model as the student. We train our main models 246 for 125 steps, less than a single epoch, with a batch size of 247 1024, a constant learning rate of  $1 \times 10^{-6}$ , and a group size 248 of 64. We note that we were also able to train RLTs with a 249 smaller batch size of 256 and more steps for faster prelimi-250 nary experimentation with only slightly inferior results. 251

252 We collect our distillation dataset with the learned RLTs 253 using the same full set of 17K question-solution pairs from 254 training. With the new reasoning traces, we then proceed to 255 fine-tune our students either on this full data or a randomly 256 sampled 1K subset, equating the distillation budget and 257 following the same recipes as our baselines (Muennighoff 258 et al., 2025; Li et al., 2025a). Unlike previous RL distillation 259 pipelines, we do not apply extra postprocessing refinements to improve the quality of the RLT's reasoning traces, directly 261 using our model's raw outputs for student fine-tuning. We refer to Appendices A and B for further details regarding 263 our training and distillation phases. hyperparameters. 264

Following prior work (Muennighoff et al., 2025; Labs, 2025; 265 Li et al., 2025a), our main evaluation considers three popular 266 and challenging tasks from the literature: AIME24 (of Amer-267 ica, 2024), the set of problems used for the American Invitational Mathematics Examination. MATH 500 (Hendrycks 269 et al., 2021), the set of problems selected by (Lightman 270 et al., 2023) from the canonical competition math bench-271 mark. GPQA Diamond (Rein et al., 2024), the set of di-272 amond difficulty problems on natural science topics from 273

the Graduate-level Google-proof Q&A benchmark. We report the completion accuracy of each of our students using Lighteval (Fourrier et al., 2023). When available, we use baseline results reported in prior work, which we found close to our early reproduction attempts. In Appendix C, we extend the experiments in this section by evaluating our models on additional tasks and settings.

#### 4.2. Test-time reasoning across teachers and students

Our main experiments focus on grounding the effectiveness of RLTs to obtain instructive reasoning traces beyond traditional distillation pipelines. As described in Section 4.1, to construct the student distillation dataset, we use the same starting question-solution pairs as our recent state-of-the-art baselines (Li et al., 2025a; Labs, 2025), with each sample only differing in terms of its reasoning trace. While RLTs could be inexpensively applied to provide explanations of larger corpora, this consistency serves to remove potential confounding factors, other than the quality of the reasoning traces, biasing our experiments and comparisons. For the same reason, we do not retune any hyperparameters for the distillation phase, training students following the same procedure as our baselines based on data size.

We compare the RLTs' explanations with prior approaches, evaluating students fine-tuned on both our full 17K distillation samples and its 1K subset. Our recent baselines all follow a similar recipe of distilling datasets obtained by generating reasoning traces with expensive reasoning models or API calls and postprocessing them with closed-source LMs: s1 (Muennighoff et al., 2025) using traces from Gemini Flash Thinking (Google, 2024), Sky-T1 (Li et al., 2025a) using traces from QwQ (Team, 2025b), and Bespoke (Labs, 2025) using traces from DeepSeek R1 (Guo et al., 2025). Since the Bespoke baseline obtained state-of-the-art results with our same question-solution corpus, we extend its evaluation with new results distilling its processed R1 traces only for our same 1K questions subset, equating its number of datapoints with the other s1 baseline.

As shown in Table 1, the raw output explanations of our small 7B parameter RLT outperform all the considered datadistillation pipelines involving teachers with orders of magnitude more parameters and additional ad-hoc postprocessing steps. We also find the effectiveness of the RLT traces stays consistent across different data sizes, in contrast to the R1 traces from the Bespoke pipeline that appear significantly less effective when subsampled. Furthermore, even when distilling a Qwen-32B student, much larger than our 7B teacher, our RLT still outperforms all prior methods for both data sizes with considerable margins. We believe this result, in particular, shows how our framework could allow overcoming the current prohibitive costs of RL reasoning: shifting the burden of expensive RL procedures to small

Large Language Models to Diffusion Finetuning

277	Model	Data size	AIME 2024	<b>MATH 500</b>	GPQA Diamond	Overall
278	OwO-32B	N.A.	50.00	90.60	54.50	65.03
279	DeepSeek-R1	800K+	79.80	97.30	71.50	82.87
280	Qwen2.5-7B-Instruct	N.A.	10.00	74.20	33.30	39.17
201	Bespoke-7B-1K	1K	13.30	80.00	33.80	42.37
283	RLT-7B-1K (Ours)	1K	20.00	80.40	41.90	47.43
284	Bespoke-7B	17K	20.00	82.00	37.80	46.60
285	RLT-7B (Ours)	17K	23.30	82.80	42.40	49.50
286	Qwen2.5-32B-Instruct	N.A.	26.70	84.00	49.00	53.23
287	s1-32B	1K	50.00	92.60	56.60	66.40
288	s1-32B + budget forcing	1K	56.70	93.00	59.60	69.77
289	Bespoke-32B-1K	1K	46.70	92.60	57.50	65.60
290 201	RLT-32B-1K (Ours)	1K	60.00	94.00	60.10	71.37
291	Sky-T1-32B	17K	43.30	82.40	56.80	60.83
293	Bespoke-32B	17K	63.30	93.00	58.10	71.47
294	RLT-32B (Ours)	17K	66.70	93.40	59.60	73.23
295						

teachers, unable to effectively solve problems from scratch 296 but highly specialized in the simpler task of producing ef-297 fective explanations for large, more powerful students.

#### 4.3. RLTs to cold-start RL

275

299

300

329

301 Our next set of experiments focuses on evaluating the effec-302 tiveness of RLTs in providing cold-start data for traditional 303 RL. For this new RL phase, we use our same GRPO imple-304 mentation with the standard student format and correctness-305 based rewards described in Section 2. As compared to the 306 RLT framework, we find that using a larger batch size of 307 1024 is significantly more beneficial to better cope with the 308 increased variance and reward sparsity of traditional RL. We 309 train for a full epoch on the recent RL dataset from Li et al. 310 (2025b) collected by analyzing and selecting an effective 311 subset of the competition math data based on the correlation 312 of individual samples with overall improvement. 313

314 We compare performing this new RL phase on the Qwen-315 7B model cold-started from the reasoning traces of our 7B 316 RLT and the postprocessed R1 traces from the Bespoke 317 pipeline, our strongest distillation baseline. Moreover, we 318 also compare a 7B parameters baseline teacher trained with 319 traditional RL as done in prior work (Shen et al., 2025; 320 Guo et al., 2025): effectively performing RL twice on the Qwen model and collecting a dataset at the end of the first iteration to cold-start the second. To construct the cold-323 starting dataset for this last baseline, we consider either 324 taking the model's raw output traces, as done with RLTs, 325 or postprocessing them with additional refinements using GPT4.1-mini (Achiam et al., 2023) and following a very 327 similar strategy to the other R1 and QwQ traditional distilla-328 tion pipelines (Li et al., 2025a; Team, 2025a).

As shown in Table 2, the reasoning traces from our 7B parameter RLT again display superior cold-starting effectiveness compared to all of our baselines. The performance gap is exceedingly noticeable with the cold-starting approaches that are also using a 7B teacher trained with traditional RL. In fact, only after improving the format and structure of the traces from these RL-trained teachers with GPT postprocessing, we were able to observe any improvements from the original Qwen-7B results. While our RLT was itself trained from the same 7B model, it again demonstrates superior cold-starting even when compared to postprocessed R1 pipelines. Overall, we find these results to be compelling evidence indicating that RLTs have the potential to unlock new key avenues to democratize the RL reasoning framework beyond the current reliance on prohibitively large and closed-source LMs.

#### 4.4. Out-of-domain zero-shot transfer

Unlike problem-solving from scratch, we posit that providing effective explanations to given solutions is a much less task-specific skill. Thus, in this subsection, we evaluate how well RLTs can be applied to construct datasets and distill new specialized students in out-of-distribution domains, without any expensive RL retraining. In particular, we focus on the canonical countdown task (Gandhi et al., 2024), asking student models to combine a set of numbers to equal a given target using basic arithmetic operations. We train and test our models on distinct datasets of 16K and 1K automatically-generated question and solution pairs. We compare zero-shot transferring our RLT with transferring the RLT-7B student and the Bespoke-7B baselines from the previous subsections. To ground our results, we also con-

Large Language Models to Diffusion Finetuning

Table 2. RLTs and prior distillation pipelines for cold-starting traditional RL.						
Model	Data size	AIME 2024	<b>MATH 500</b>	<b>GPQA Diamond</b>	Overall	
Qwen2.5-7B-Instruct	N.A.	10.00	74.20	33.30	39.17	
Bespoke-7B	17K	20.00	82.00	37.80	46.60	
RLT-7B (Ours)	17K	23.30	82.80	42.40	49.50	
RL no cold-start	N.A.	13.30	74.20	34.80	40.77	
RL cold-start (raw) + RL	17K	10.00	71.00	34.80	38.60	
RL cold-start (GPT) + RL	17K	16.70	78.20	36.90	43.93	
Bespoke-7B + RL	17K	16.70	82.80	45.40	48.30	
RLT-7B + RL (Ours)	17K	26.70	84.00	40.90	50.53	



357 358 corpus and RL on the countdown task. Right: Performance after training on different distillation datasets ranked by the RLT reward. 359

360 sider performing RL on the countdown task itself (CD RL), 361 training both from the Qwen-7B model and the cold-started 362 Bespoke-7B baseline as described in Section 4.3.

333

335

343

345

346 347

348

349

350

351

352

353

354

355

383

384

363 As shown in the left bar plot of Figure 4, applying RLT 364 distillation zero-shot remarkably achieves even higher performance than direct RL on the countdown task. Interestingly, direct RL appears to provide only marginally better 367 scores than using students distilled from our original set of reasoning questions that do not include any examples of 369 countdown problems (50.8 vs. 49.2). Furthermore, we find 370 there is stark overlap of over 98.5% in the final sets of solved 371 problems between direct RL and the RL-free Bespoke-7B baseline. We find these results in line with prior analysis (Li 373 et al., 2025a; Yue et al., 2025), providing further evidence 374 that the exploration challenge of traditional RL might make 375 most of its benefits come from steering the base model's 376 distribution toward long-context generation. In contrast, 377 by simplifying the task and foregoing sparse rewards, our 378 RLT appears much more effective - providing countdown-379 specific traces for students to learn new knowledge and solve 380 unseen questions, yielding higher improvements than direct 381 RL even without any teacher training in this new domain. 382

#### 4.5. Explanation reward analysis

To analyze the design of the RLT reward function, we start by examining the relationship between the traces' rewards and the effectiveness of student distillation. In particular, we use our RLT's checkpoint right before RL training to generate 16 completions for each question-solution pair in our data. We then score all completions with our reward and divide them into groups based on their relative rank for each prompt. Thus, we obtain 16 datasets with different reasoning traces for each question, which we use to train 16 new 7B students from Qwen. As illustrated in the right bar plot of Figure 4, ordering student performance by the respective dataset rank shows a clear correlation between the two, with a Pearson coefficient over 0.89, validating the efficacy of the RLT objective. Additionally, the highest ranked traces of our 7B teacher before any RL remarkably already yield 90% the performance gains of our baseline R1 distillation pipeline (Labs, 2025), showing how even small models already possess latent teaching skills unlocked by our new reward and simplified task formulation.

We also inspect qualitative examples chosen by selecting samples where the reward of the RLT explanations is particularly improved from the baseline R1 distillation pipeline.

385	5 R1 postprocessed reasoning		RLT explanation		
386	< begin_of_thought >		<pre>&lt; begin_of_explanation &gt;</pre>		
387	[]		[]		
388	Just to check, if I use <b>a calculator</b>		First, the unit circle definition		
389	[]		[]		
390	Another angle (no pun intended): using the derivative		Alternatively, using calculus. The derivative of		
391	[]		[]		
392	Another way to check: maybe using complex numbers or other		Alternatively, use complex numbers. Let me try that. Express the two		
393	identities, <b>but I think the method is solid</b> the final answer is		angles as complex numbers on the unit circle		
394	(leng_or_engenel)		(lena_or_orbianacion))		

*Figure 5.* Contents from post-processed R1 traces (Labs, 2025) with lower rewards than the corresponding RLT explanations.

396 As shown in Figure 5, we find that the R1 traces with low 397 rewards often try to rely on external tools, such as calcu-398 lators, and employ language patterns likely idiosyncratic 399 to the training data of the DeepSeek-V3 LM, as sentences 400 with brief humorous comments (DeepSeek-AI, 2024). In-401 stead, the corresponding RLT explanations appear much 402 more grounded and even manage to add new verification 403 steps not considered by R1 to check the final solution. In 404 Appendix D, we provide additional examples showcasing 405 further qualitative differences of our framework with R1 406 traces and also specific failure cases from training RLTs 407 without proper balance between each reward component, 408 such as repetitions and overly-long explanations. 409

#### 5. Related work

410

411

412 Inspired by the unprecedented abilities of the OpenAI o1 413 model (Jaech et al., 2024), there has been a resurgence of RL 414 approaches aimed at inducing a new kind of open-ended rea-415 soning to scale test-time compute. The work from Guo et al. 416 (2025) was another milestone in this new domain, providing 417 a first openly detailed example of what is possible with large 418 models and RL. Other follow-ups considered smaller LMs 419 and ways to decrease optimization costs via approaches such 420 as explicit task breakdown (Shen et al., 2025), exploration 421 strategies (Hou et al., 2025), new RL objectives (Liu et al., 422 2025), and cold-start data scale (Xu et al., 2025). However, 423 it is still an open question if RL on small models can go 424 beyond cheaper supervised alternatives (Guo et al., 2025) 425 and induce new skills beyond the pretraining corpus (Yue 426 et al., 2025). In contrast to this work, RLTs break the tra-427 ditional framework of maximizing one-hot accuracy with 428 verifiable rewards - turning the task on its head by feeding 429 the model the correct solution as input and avoiding RL's 430 inherent exploration challenge. 431

A large part of the recent test-time scaling literature considering smaller LMs has focused on inducing reasoning
with "teacher-student" supervised distillation (Hinton et al.,
2015), a widely validated technique in traditional LM development (Hui et al., 2024; Shing et al., 2025). This approach's popularity to induce LM reasoning dates earlier
than the RL paradigm, with older methods harnessing veri-

fiers and prompting for self-improvement (Zelikman et al., 2022; 2024). By following a common structure of generation, filtering correct responses, and postprocessing them, modern RL-based distillation has seen significant advances mostly driven by more capable teachers (Labs, 2025; Li et al., 2025a) and carefully curating targeted datasets (Muennighoff et al., 2025; Ye et al., 2025). However, the effectiveness of current distillation pipelines was shown to be closely tied to the properties of the student itself (Li et al., 2025c; Xu et al., 2025), and their ability to induce actual generalization remains unclear (Chu et al., 2025). Unlike these traditional distillation pipelines, the RLT framework does not rely on verifiers for filtering, directly optimizes the teacher for downstream distillation, and does not require any post-processing, allowing direct transfer of reasoning capabilities to arbitrary tasks and even larger student models.

#### 6. Discussion and extensions

This work introduced a new class of Reinforcement-Learned Teachers trained with a simpler dense-reward task that inputs both each problem's question and solution, and optimizes the LM to provide instructive reasoning traces for distillation as outputs. Empirically, students trained or cold-started from the raw outputs of a 7B RLT obtain higher performance than prior distillation pipelines using orders-of-magnitude larger LMs. Furthermore, RLTs maintain their effectiveness even for distilling much larger students, and when providing reasoning traces for out-of-distribution tasks beyond their training corpus. Nonetheless, our work has only begun to study the design space of our new framework, with many exciting directions yet to be explored. One example is training RLTs and their students in tandem, allowing the teacher's explanations to adapt to the student's learning dynamics live. Pushing this further, the same model could even take both roles, iterating RL with our task formulation, providing access to each problem's solution, to obtain instructive reasoning traces, and self-distillation (Zelikman et al., 2022) to revise its own explanation and learn how to solve questions from scratch - unifying the open-endedness of RL with the stability of supervised optimization.

#### 440 Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

#### References

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

463

- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.
- Ahmadian, A., Cremer, C., Gallé, M., Fadaee, M., Kreutzer, J., Pietquin, O., Üstün, A., and Hooker, S. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.
- Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T.,
  Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.
- Chu, T., Zhai, Y., Yang, J., Tong, S., Xie, S., Schuurmans,
  D., Le, Q. V., Levine, S., and Ma, Y. Sft memorizes, rl
  generalizes: A comparative study of foundation model
  post-training. *arXiv preprint arXiv:2501.17161*, 2025.
- 468
  469
  470
  470
  DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.
- Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., and Clune, J. Go-explore: a new approach for hardexploration problems. *arXiv preprint arXiv:1901.10995*, 2019.
- Fourrier, C., Habib, N., Kydlíček, H., Wolf, T., and Tunstall, L. Lighteval: A lightweight framework for llm evaluation, 2023. URL https://github.com/ huggingface/lighteval.
- 481 Gandhi, K., Lee, D., Grand, G., Liu, M., Cheng, W., Sharma,
  482 A., and Goodman, N. D. Stream of search (sos): Learning
  483 to search in language. *arXiv preprint arXiv:2404.03683*,
  484 2024.
- 485 486 487 487 488 488 489 489 Google. Gemini 2.0 flash thinking mode (gemini-2.0flash-thinking-exp-1219), December 2024. URL https://cloud.google.com/vertex-ai/ generative-ai/docs/thinking-mode.
- Gorbatovski, A., Shaposhnikov, B., Malakhov, A., Surnachev, N., Aksenov, Y., Maksimov, I., Balagansky, N., and Gavrilov, D. Learn your reference model for real good alignment. *arXiv preprint arXiv:2404.09656*, 2024.

- Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. Olympiadbench: A challenging benchmark for promoting agi with olympiadlevel bilingual multimodal scientific problems. *arXiv* preprint arXiv:2402.14008, 2024.
- Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.
- Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- Hou, Z., Lv, X., Lu, R., Zhang, J., Li, Y., Yao, Z., Li, J., Tang, J., and Dong, Y. Advancing language model reasoning through reinforcement learning and inference scaling. *arXiv preprint arXiv:2501.11651*, 2025.
- Hui, B., Yang, J., Cui, Z., Yang, J., Liu, D., Zhang, L., Liu, T., Zhang, J., Yu, B., Lu, K., et al. Qwen2. 5-coder technical report. arXiv preprint arXiv:2409.12186, 2024.
- Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. arXiv preprint arXiv:2412.16720, 2024.
- Jain, N., Han, K., Gu, A., Li, W.-D., Yan, F., Zhang, T., Wang, S., Solar-Lezama, A., Sen, K., and Stoica, I. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.
- Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS* 29th Symposium on Operating Systems Principles, 2023.
- Labs, B. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. www.bespokelabs.ai/blog/bespokestratos-the-unreasonable-effectiveness-of-reasoningdistillation, 2025. Accessed: 2025-01-22.
- Li, D., Cao, S., Griggs, T., Liu, S., Mo, X., Tang, E., Hegde, S., Hakhamaneshi, K., Patil, S. G., Zaharia, M., et al.

495 Llms can easily learn to reason from demonstrations 496 structure, not content, is what matters! arXiv preprint 497 arXiv:2502.07374, 2025a.

498

518

520

521

522

523

524

525

526

527

528 529

530

531

532

533

534

535

536

537

538

539

540

- Li, X., Zou, H., and Liu, P. Limr: Less is more for rl scaling. 499 arXiv preprint arXiv:2502.11886, 2025b. 500
- 501 Li, Y., Yue, X., Xu, Z., Jiang, F., Niu, L., Lin, B. Y., Ra-502 masubramanian, B., and Poovendran, R. Small models 503 struggle to learn from strong reasoners. arXiv preprint 504 arXiv:2502.12143, 2025c. 505
- 506 Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, 507 B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and 508 Cobbe, K. Let's verify step by step. arXiv preprint 509 arXiv:2305.20050, 2023. 510
- 511 Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, 512 W. S., and Lin, M. Understanding r1-zero-like training: 513 A critical perspective, 2025. URL https://arxiv. 514 org/abs/2503.20783. 515
- 516 Loshchilov, I. Decoupled weight decay regularization. arXiv 517 preprint arXiv:1711.05101, 2017.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., 519 Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602, 2013.
  - Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In International conference on machine learning, pp. 1928-1937. PmLR, 2016.
  - Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling. arXiv preprint arXiv:2501.19393, 2025.
  - of America, M. A. Aime, February 2024. URL https: //artofproblemsolving.com/wiki/index. php/AIME\_Problems\_and\_Solutions/.
  - Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. In First Conference on Language Modeling, 2024.
- 542 Salimans, T. and Chen, R. Learning montezuma's re-543 venge from a single demonstration. arXiv preprint 544 arXiv:1812.03381, 2018. 545
- 546 Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and 547 Klimov, O. Proximal policy optimization algorithms. 548 arXiv preprint arXiv:1707.06347, 2017. 549

- Shao, Z., Wang, P., Zhu, O., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024.
- Shen, M., Zeng, G., Qi, Z., Hong, Z.-W., Chen, Z., Lu, W., Wornell, G., Das, S., Cox, D., and Gan, C. Satori: Reinforcement learning with chain-of-action-thought enhances llm reasoning via autoregressive search. arXiv preprint arXiv:2502.02508, 2025.
- Shing, M., Misaki, K., Bao, H., Yokoi, S., and Akiba, T. TAID: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview. net/forum?id=cqsw28DuMW.
- Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm testtime compute optimally can be more effective than scaling model parameters. arXiv preprint arXiv:2408.03314, 2024.
- Team, N. Sky-t1: Train your own o1 preview model within \$450. https://novasky-ai.github.io/posts/sky-t1, 2025a. Accessed: 2025-01-09.
- Team, Q. Qwq-32b: Embracing the power of reinforcement learning, March 2025b. URL https://gwenlm. github.io/blog/qwq-32b/.
- von Werra, L., Belkada, Y., Tunstall, L., Beeching, E., Thrush, T., Lambert, N., Huang, S., Rasul, K., and Gallouédec, Q. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.
- Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. arXiv preprint arXiv:2312.08935, 2023.
- Xu, H., Wu, X., Wang, W., Li, Z., Zheng, D., Chen, B., Hu, Y., Kang, S., Ji, J., Zhang, Y., et al. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? arXiv preprint arXiv:2501.11284, 2025.
- Ye, Y., Huang, Z., Xiao, Y., Chern, E., Xia, S., and Liu, P. Limo: Less is more for reasoning. arXiv preprint arXiv:2502.03387, 2025.
- Yue, Y., Chen, Z., Lu, R., Zhao, A., Wang, Z., Yue, Y., Song, S., and Huang, G. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?, 2025.
- Zelikman, E., Wu, Y., Mu, J., and Goodman, N. D. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

550	Zelikman, E., Harik, G., Shao, Y., Jayasiri, V., Haber, N.,
551	and Goodman, N. D. Quiet-star: Language models can
552	teach themselves to think before speaking, 2024. URL
553	https://arxiv.org/abs/2403.09629.
554	
555	
556	
557	
558	
559	
560	
561	
562	
563	
564	
565	
566	
567	
568	
569	
570	
571	
572	
5/3	
574	
576	
570	
578	
579	
580	
581	
582	
583	
584	
585	
586	
587	
588	
589	
590	
591	
592 503	
504	
505	
596	
597	
598	
599	
600	
601	
602	
603	
604	

# 605 **A. Implementation details**

608

658

659

# 607 A.1. RLT training phase and reward

609	Table 3. Hyperparameter listing for	the RLT training optimization and reward.
610	Hyperparameter name	Value
611		value
612	<b>RLT training</b>	
614	Fine-tuned model	Owen2 5-7B-instruct (Hui et al. 2024)
615	Number of training steps	125
616	Ratch size	1024
617	L corring rote	1024 $1 \times 10^{-6}$
618	Learning rate decay	$1 \times 10^{-4}$
619	Learning rate decay	
620	Final learning rate	$1 \times 10^{-5}$
621	Weight decay	0
622	Optimizer	AdamW (Loshchilov, 2017)
623	Adam beta1	0.9
624	Adam beta2	0.999
625	Adam epsilon	1e-8
626	Warmup steps	0
627	Maximum gradient norm	1.0
628	Maximum generation context size	16384
629	Generation temperature	0.7
630	Generation top- <i>p</i>	1.0
631	Generation top-k	No
632	Generation min- <i>n</i>	0.0
633	Generation repetition penalty	1.0
625	GRPO group size	64
636	GRPO B	0.04
637	<b>Reference model sync. steps</b>	32
638	Reference model sync. mixun	0.0
639	Dtype	0.9 hfloat16
640	Dtype Credient checke sinting	biloat io
641	Gradient checkpointing	true
642	<b>RLT reward</b>	
643	Student model	Owen2.5-7B-instruct (Hui et al., 2024)
644	λ	3
645	O C	0.01
646	a Format penalty	1
647	Format penany	-1
648		

649 Our experiments are conducted on a single compute node comprising 8 Nvidia Hopper H100 GPUs, 1.8 TB of memory, and 650 208 Intel Xeon Platinum 8481C CPUs. Due to the efficiency of training with our small 7B models, we note this setup is 651 significantly less resource-intensive than prior RL and even SFT work, often relying on multi-node settings (Muennighoff 652 et al., 2025; Li et al., 2025a). As described in Section 4, for its new RL phase, we train our Reinforcement Learned 653 Teachers on the set of questions and solutions selected by Li et al.  $(2025a)^1$  which is available under an Apache 2.0 License, 654 comprising less than 17K math and coding problems. All our new teachers are trained from a Qwen2.5-7B-Instruct LM (Hui 655 et al., 2024) also available under an Apache 2.0 License, together with the other models from the Qwen family. Before 656 RL, we shortly fine-tune on pre-collected samples using example traces from Labs (2025) formatted using our new system 657

<sup>&</sup>lt;sup>1</sup>https://huggingface.co/datasets/bespokelabs/Bespoke-Stratos-17k

prompt and tags. This short, inexpensive phase is conducted with the same distillation hyperparameters used for the 1K data 661 subset following Muennighoff et al. (2025), detailed in Appendix B, but with double the number of epochs, and serves to 662 quickly familiarize our teacher with the new RLT input format. We also use the output checkpoint at the end of this phase 663 for our correlation analysis experiment conducted in Section 4.5. For the same reason and also to limit the requirements for 664 reproducibility, our lightweight Qwen2.5-7B-instruct (Hui et al., 2024) student model used to compute the RLT reward function is initialized with the checkpoint provided by Labs  $(2025)^2$  available under an Apache 2.0 License, which is already 665 666 familiar with their system prompt without needing further training. However, we found that the RLT reward is robust to the 667 specific LM choice for the student, yielding numerically close values when using our own Qwen student finetuned on only 668 1K samples.

669 We use simple coefficients to regulate the terms in the RLT rewards, with  $\lambda = 3$  to scale  $r^{\text{KL}}$  and  $\alpha = 0.01$  to scale both the 670 min term in  $r^{SS}$  and the max term in  $r^{KL}$ . Our choice was based on making each individual component of the RLT rewards 671 have approximately the same expected magnitude over the model's initial output completions. As we observed the overall 672 rankings to be quite robust after testing small changes to these initial choices, we did not find extensive sweeps necessary. 673 We also add a -1 penalty to any completion that does not use the explanation tags or that exceeds our maximum generation 674 context length to limit training time and disincentivize overly long and expensive reasoning traces. For our training runs, we 675 make use of a custom GRPO implementation with the specifications from Shao et al. (2024), extending the TRL library (von 676 Werra et al., 2020) with faster distributed VLLM generation (Kwon et al., 2023). Our RL phase is short, comprising only 677 125 steps, less than a single epoch, with a batch size of 1024, an AdamW optimizer (Loshchilov, 2017) with a constant 678 learning rate of  $1 \times 10^{-6}$ , and a group size of 64. We synchronize the reference model every 32 steps as popularized by 679 (Gorbatovski et al., 2024) with a mixup ratio of 0.9. For the post-cold-start RL phase employed in Section 4.3, we use our 680 same GRPO implementation with the standard student format and correctness-based rewards described in Section 2. The 681 only difference is that we train for one epoch in total on the very small LIMR (Li et al., 2025b) dataset, containing less than 682 1K samples. We provide a full list of hyperparameters to ensure reproducibility in Table 3. All main libraries used in our 683 implementation are again available under an Apache 2.0 Licence. 684

#### 685 686 A.2. Traditional reasoning and RLT formats

687 As detailed in Section 3.2 and illustrated in Figure 2, RLTs are prompted with a new formatting style that provides both 688 the question and solution to each problem as inputs. Then, their system prompt instructs to output instructive, detailed 689 step-by-step explanations, connecting the dots between the two. In contrast, traditional formats used for reasoning datasets 690 employed for RL and distillation forego any information about each problem's solution and task the model to solve each 691 problem from scratch. We provide a specific comparison between the two, contrasting the system prompts used for traditional 692 RL and distillation from Li et al. (2025a) in Figure A.2, and our own new RLT input format in Figure A.2. As shown, 693 our prompting format's design strives to make minimal changes to the prompts used for traditional reasoning frameworks, 694 replacing surrounding explanation with think tags, and simply appending the solution tokens after the user's provided 695 input question, to allow our teachers to make full use of this information before generating each completion. This design 696 also practically avoids the need to re-prompt our teacher multiple times for each question and manually check the answers to 697 filter out incorrect solutions. 698

# B. Student distillation

#### **B.1. Reasoning traces generation**

As detailed in Sections 3 and 4, we collect reasoning traces for each sample in the student distillation dataset by feeding our RLTs both the problem's question and its solution as input. Moreover, for our cold-starting experiments in Section 4.3, we also collect reasoning traces by feeding the RL-trained Qwen models each question and postprocessing each output with GPT4.1 (Achiam et al., 2023) similarly to Li et al. (2025a). We provide the hyperparameters used to collect these traces across all our datasets and settings in Table 4. In particular, we use standard generation hyperparameters for reasoning Qwen-based models (Li et al., 2025a), that are aligned with the online generation parameters during our new RL phases, as detailed in Appendix A. The only main difference is that we allow for a longer maximum context size to avoid collecting incomplete traces for downstream distillation, which is possible thanks to the fact that at test time we are not subject to the same training-time memory constraints dictated by backpropagating through long traces.

712

699

<sup>&</sup>lt;sup>2</sup>https://huggingface.co/bespokelabs/Bespoke-Stratos-7B

<sup>713</sup> 714

715		
716	Table 4. Hyperparameter listing for the RLI g	eneration pipel
717	Hyperparameter name	Value
718		<u> </u>
719	Distillation data generation sett	ing
720	Maximum generation context size	32764
721	Generation temperature	0.7
722	Generation temperature	0.7
723	Generation top- $p$	1.0
724	Generation top-k	No
725	Generation min-p	0.0
726	Generation repetition penalty	1.0
727	Generation dtype	hfloat16
728	Generation dtype	Ulloat10

The RLT distillation datasets used for training all of our 7B students were collected by generating a single completion for each question-solution pair, directly placing it in the student format. After preliminary experiments, we also found our 32B students can be particularly sensitive to cropped reasoning traces that exceed the 16384 maximum context length specified in the SFT hyperparameters from Li et al. (2025a), which they purposefully limited for computational efficiency. Thus, to avoid this mismatch potentially affecting our results, we simply collected up to 16 reasoning traces for each question-solution pair and selected anyone below 16384, otherwise resorting to  $r^{SS}$  for selection.

#### **B.2. Student distillation specifications**

729 730 731

732

733

734

735

736

737 738

739

758

740 Our main experiments from Section 4 use the RLT traces collected either from the full set of training questions from Li et al. 741 (2025a) or its randomly selected 1K data subset. For the student distillation phases of these experiments, we re-use the same 742 hyperparameters from Li et al. (2025a) and Muennighoff et al. (2025), for the full data and 1K subset, respectively. The 743 purpose of not re-tuning based on our new data is to attempt to isolate the reasoning traces used for student distillation as the 744 only degree of variation in our comparison between RLTs with traditional reasoning distillation pipelines. Furthermore, in 745 our experiments in Section 4.4, we also compare transferring students learned on our original set of datapoints and direct RL 746 with transferring the RLTs themselves zero-shot to the countdown task (Gandhi et al., 2024). For these experiment, we use a 747 set of 16K automatically-generated countdown question and solution pairs with 3 or 4 numbers and, after obtaining the 748 corresponding RLT traces, distill our students using the hyperparameters from Li et al. (2025a) once again, which we found 749 to work well in practice without further tuning, given the two similar dataset sizes. 750

We provide a full list of the hyperparameters used for all our distillation experiments in each setting in Table 5, where we 751 highlight the key differences across the three. As detailed, the considered approaches mostly differ in terms of the number 752 of epochs, the batch size, and the optimizer parameters. These differences reflect the total number of samples and relative 753 variance in each data batch. In line with the findings from the relative prior works (Muennighoff et al., 2025; Li et al., 2025a; 754 Team, 2025a), we note that training on these small datasets was very inexpensive and could be completed within hours. We 755 note that the learning rate of our reinforcement learning phases matches the final learning rate during our SFT optimization, 756 a simple choice which we found to work well in practice. 757

#### **B.3. Student RL details** 759

760 For our cold-starting experiments in Section 4.3, we also implement and perform a phase of traditional RL training optimizing 761 the model with correctness-based rewards under the "student's perspective." This phase is done atop a Qwen2.5-7B-Instruct, 762 the cold-started Bespoke-7B, and the RLT-7B models using our same GRPO implementation on the open-source LIMR 763 dataset (Li et al., 2025b). We re-use most of the hyperparameters from the RLT training phase with a batch size of 1024, 764 something particularly important to cope with the increased variance and reward sparsity of traditional RL. Additionally, 765 we also conduct traditional RL as a baseline for our out-of-distribution transfer experiments in Section 4.4 that directly 766 optimizes correctness on the countdown task starting again from the Qwen2.5-7B-Instruct model, and the cold-started 767 Bespoke-7B models. For this particular task, we found using a larger batch size to be not strictly necessary, as for RLT training, and obtained better results optimizing the model for more steps (250 total) with a batch size of 256. We provide a 769

Hyperparameter name	Full fine-tuning	1K subset fine-tuning	Countdown fine-tuning
Student training			
Distilled model	Qwen2	2.5-7B/32B-instruct (Hui et al.,	2024)
Number of training samples	16710	1000	16000
Number of epochs	3.0	5.0	3.0
Batch size	96	16	96
Learning rate	$1 \times 10^{-5}$	$1 \times 10^{-5}$	$1 \times 10^{-5}$
Learning rate decay	Cosine	Cosine	Cosine
Final learning rate	$1 \times 10^{-6}$	$1 \times 10^{-6}$	$1 \times 10^{-6}$
Weight decay	0	$1  imes 10^{-4}$	0
Optimizer	AdamW (Loshchilov, 2017)	AdamW (Loshchilov, 2017)	AdamW (Loshchilov, 2017
Adam beta1	0.9	0.9	0.9
Adam beta2	0.999	0.95	0.999
Adam epsilon	$1 \times 10^{-8}$	$1 \times 10^{-8}$	$1 \times 10^{-8}$
Warmup ratio	0.1	0.05	0.1
Maximum gradient norm	1.0	1.0	1.0
Dtype	bfloat16	bfloat16	bfloat16
Gradient checkpointing	true	true	true

full list of the distillation hyperparameters used for our RL phases on these tasks in Table 6, where we highlight the key differences between these two settings.

## B.4. Student evaluation

796 797

798

799 800

801

818 819

820

821 822

823

824

802 As described in Section 4.1, our main evaluation consists of three graduate and competition-level tasks on math and natural 803 science domains. In particular, these include AIME24 (of America, 2024), the set of problems used for the American 804 Invitational Mathematics Examination; MATH 500 (Hendrycks et al., 2021), the set of problems selected by (Lightman et al., 805 2023) from the canonical competition math benchmark; and GPQA Diamond (Rein et al., 2024), the set of diamond difficulty 806 problems on natural science topics from the Graduate-level Google-proof Q&A benchmark. Additionally, in Appendix C, 807 we also extend our set of experiments to include additional challenging coding and multilingual domains. In particular, we 808 consider LiveCodeBench (Jain et al., 2024), a contamination-free set of coding challenge problems continuously collected 809 from several prominent online hosting platforms; and OlympiadBench (He et al., 2024), a set of olympiad-level bilingual 810 problems in English and Chinese from past math and physics competitions. 811

We evaluate on all the above benchmarks using Lighteval (Fourrier et al., 2023), a library available under the MIT license. In all our results, we report the completion accuracy of each of our students for a single generated completion, as also reported in prior work (Muennighoff et al., 2025; Li et al., 2025a; Labs, 2025). Furthermore, we also ensure consistency by re-using the task implementation code, including the system prompt, provided by our baselines (Team, 2025a), which is available under an Apache 2.0 License. For the same reason, we do not modify any of the existing evaluation generation hyperparameters from the suggested settings used in their evaluation, which we report in Table 7.

# C. Additional experiments

## C.1. Coding and multilingual reasoning

We extend our main set of experiments from Section 4, focusing on graduate and competition-level tasks on math and natural science domains, comparing the effectiveness of the reasoning traces from our 7B parameter RLT with traditional

Hyperparameter name	LIMR data traditional RL	Countdown data traditional RL
Traditional RL student training		
Fine-tuned model	Qwen2.5-7B-instruct (Hui et a	1., 2024)/Bespokes-7B (Labs, 2025)
Number of training samples	1389	16000
Number of epochs	1.0	1.0
Number of training steps	86	250
Batch size	1024	256
Learning rate	$1 \times 10^{-6}$	$1 \times 10^{-6}$
Learning rate decay	Constant	Constant
Final learning rate	$1 \times 10^{-6}$	$1 \times 10^{-6}$
Weight decay	0	0
Optimizer	AdamW (Loshchilov, 2017)	AdamW (Loshchilov, 2017)
Adam beta1	0.9	0.9
Adam beta2	0.999	0.999
Adam epsilon	$1 \times 10^{-8}$	$1 \times 10^{-8}$
Warmup steps	0	0
Maximum gradient norm	1.0	1.0
Maximum generation context size	16384	16384
Generation temperature	0.7	0.7
Generation top-p	1.0	1.0
Generation top-k	No	No
Generation min-p	0.0	0.0
Generation repetition penalty	1.0	1.0
GRPO group size	64	64
GRPO $\beta$	0.04	0.04
Reference model sync. steps	32	32
Reference model sync. mixup	0.9	0.9
Dtype	bfloat16	bfloat16
Gradient checkpointing	true	true

864 865

866

867

868

869

870

distillation pipelines. We consider challenging coding and multilingual domains, which are less aligned with the training and distillation set of questions employed. In particular, these new benchmarks include LiveCodeBench (LCB) (Jain et al., 2024), a contamination-free set of coding challenge problems continuously collected from several prominent online hosting platforms across three difficulty categories; and OlympiadBench (He et al., 2024), a set of olympiad-level bilingual problems in English and Chinese from past math and physics competitions.

In Table 8, we provide a comparison of reported results from our baselines (Li et al., 2025a; Labs, 2025) and our fine-tuned student models using our 7B RLT traces. We also re-collected the performance on OlympiadBench of the Qwen2.5-7B-Instruct model and the Bespoke-7B baseline fine-tuned from the postprocessed R1 traces, as they were omitted in the reference results from prior work. For LiveCodeBench, we report both the performance on the "hard" difficulty set of problems and the average weighted performance. This was obtained by weighting the performances of the models in the "easy," "medium," and "hard" sets of LiveCodeBench problems, by the relative number of problems in each.

We find the performance of RLT distilled students on these new tasks to be consistent with the performance from our Section 4 experiments. In particular, the overall performance exceeds the performance of the baseline distillation pipelines

Large Language Models to Diffusion Finetuning

880 _					
881 Ta	able 7. Student evaluation hyperparameter	r listing for the RLT gen	eration pipeline, matching t	he hyperparameters from Li et al	. (2025a).

Hyperparameter name	Value
Student evaluation	
Maximum generation context size	32764
Generation temperature	0.7
Generation top-p	1.0
Generation top-k	No
Generation min-p	0.0
Generation repetition penalty	1.0
Generation dtype	bfloat16

Table 8. RLTs and prior distillation pipelines across model (7B and 32B) evaluated on coding and multilingual reasoning benchmarks.
 Overall is computed by taking the average between LCB-Average and OlympiadBench scores.

Model	Data size	LCB-Average	LCB-Hard	OlympiadBench	Overall
Qwen2.5-7B-Instruct	N.A.	31.88	3.30	35.90	33.89
Bespoke-7B	17K	36.10	1.60	43.30	39.70
RLT-7B	17K	34.63	3.30	46.10	40.37
Qwen2.5-32B-Instruct	N.A.	48.94	9.80	46.70	47.82
Sky-T1-32B	17K	57.94	17.90	57.30	57.62
Bespoke-32B	17K	71.06	26.20	60.30	65.68
RLT-32B	17K	71.24	32.50	64.00	67.62

using orders of magnitude larger models across student sizes. Furthermore, the RLT performance is also best across the
individual settings, with the sole exception of LCB-average only for the 7B model, where it comes as a close second.
However, as shown by the experiments in Section 4.4, by equating the pool of initial questions from which to perform
distillation, we believe these experiments could still be underplaying the true potential enabled by the zero-shot transferability
of RLTs. In particular, transferring RLTs, rather than their students, to construct reasoning traces to include more coding and
Chinese-written problems could allow downstream distillation to develop further domain-specific expertise and reasoning,
without the need to run expensive pipelines requiring large and closed-source models.

# D. Extended analysis

916

917 918

919

## D.1. RLT reward and R1 traces analysis

We analyze the empirical consequences of ablating the components in our RLT reward function, described in Section 3.3. In particular, we focus on the effects of removing two key components of our design. First, we examine ablating the think tokens KL reward  $r^{\text{KL}}$ , quantifying whether the think tokens  $t_{o_i}$  themselves are interpretable logical continuations from the student's perspective as compared with the teacher's. Second, we examine ablating the min/max reductions terms, which serve to ensure the rewards do not forego any individual token, avoiding introducing bias to the reward values based on the solution length or the number of think tokens in the teacher's explanations.

For these experiments, we train entirely new 7B RLTs following the hyperparameters from Table 3 but setting  $\lambda = 0$  (see Equation 5) or  $\alpha = 0$  (see Equations 4 and 3), respectively for the thought tokens KL reward and min/max reduction ablations. We then construct student distillation datasets, using the same full set of starting question-solution pairs as our recent state-of-the-art baselines (Li et al., 2025a; Labs, 2025) as considered in our main Section 4 experiments. We compare the generated datasets and students after distillation with these ablated teachers to our original 7B RLT trained with the full RLT reward and our strongest distillation baseline using postprocessed R1 reasoning traces (Li et al., 2025a; Labs, 2025).

P33 934 First, we focus on the effects of these ablations in terms of the traces' content as compared to our original RLT's traces and

Model	Data size	AIME 2024	<b>MATH 500</b>	<b>GPQA</b> Diamond	Overall
Qwen2.5-7B-Instruct	N.A.	10.00	74.20	33.30	39.17
Bespoke-7B (R1 traces)	17K	20.00	82.00	37.80	46.60
Full RLT reward	17K	23.30	82.80	42.40	49.50
No thought tokens KL term	17K	6.70	63.80	31.80	34.10
No min/max reduction	17K	23.30	79.00	40.00	47.43

Table 9. Teachers ablated of individual reward components compared with RLTs and prior distillation pipelines on our main set of challenging reasoning tasks.

946 the postprocessed R1 pipeline. As shown in Figure 8, the length of the reasoning traces is greatly affected by our ablations. 947 When looking at our original RLT with full rewards, the average length of the produced reasoning trace is 39% higher than 948 the original curated R1 traces in the Bespoke dataset (Labs, 2025). This is consistent with our analysis from Section 4.5 and 949 further highlighted by the full examples shown in Figures 9 and 10, showing how our 7B RLT often includes alternative 950 verification steps and approaches not considered by traditional pipelines that do not optimize directly for downstream 951 distillation. Furthermore, ablating each term in our reward leads to the concrete unwarranted effects described in Section 3.3, 952 allowing the RL optimization procedure to find "shortcuts" to maximize  $r^{\text{RLT}}$  that hurt the quality of the reasoning traces. 953 In particular, without the thought tokens KL reward  $r^{\rm KL}$ , our RLT cannot differentiate between explanations that guide 954 the student step-by-step and those that increase the solution's likelihood without a logical path that can be learned from. 955 Thus, as shown in Figure 11, this ablation leads to a teacher that only learns to repeat the solution tokens themselves in its 956 explanation to exploit the repetition tendency of pretrained student LMs, with the average length of its output reasoning 957 traces dramatically dropping. Moreover, in our second ablation of the min/max reward reduction terms, the rewards become 958 effectively biased by the length of the reasoning trace, leading the teacher to prefer long explanations only to reduce the 959 influence on  $r^{KL}$  of hard but necessary individual logical steps. As a consequence of this bias, the average number of think 960 tokens in the teacher's reasoning traces almost doubles from the postprocessed R1 traces and, as shown in Figure 12, their 961 content starts including many additional unnecessary steps that are just semantical repetitions of each other as learning 962 progresses. 963

Then, we also quantify and compare the effects that each of our ablations has on downstream student performance. As 964 965 shown in Table 9, the disruptive effects of ablating the thought tokens KL reward entirely reflect on the capabilities of the learned students, with their performance being lower than even the original Qwen-7B model they are fine-tuned from. This 966 result validates our reward design, showing how regularizing for the reasoning traces to be "natural" continuations from the 967 student's own perspective is of key importance for effective distillation. On the other hand, ablating the min/max reward 968 reduction terms produces a more moderate reduction in performance, with the new output traces of this 7B teacher still 969 remarkably outperforming our strongest baseline pipeline using the R1 LM with orders of magnitude more parameters. 970 However, we note that by preventing the increase in the lengths of the reasoning traces with the min/max reduction terms, 971 our full RLT reward also yields faster training, distillation dataset generation, and student fine-tuning, with non-trivial 972 benefits contributing to our framework's efficiency. 973

#### 975 **E. Limitations and future implications** 976

#### E.1. Limitations and unexplored directions

974 975

977

978 This work's purpose was to introduce a new class of Reinforcement-Learned Teachers designed to avoid the exploration 979 challenge of sparse rewards and align the optimization of RL-trained LMs with the test-time goal of downstream distillation. 980 However, there are still several limitations and improvements which we hope will be tackled in future extensions. First, to 981 output explanations, the RLT framework relies on access to the ground-truth solutions. Hence, when used with datasets and 982 domains where this information might not be available or not be practical to recover without querying LMs, small RLTs 983 might have to again rely on larger models, even though still to a lesser extent than prior distillation pipelines. Similarly, as 984 described in Section 4.1 and Appendix A, our current training recipe does not exclusively involve RL, with an initial phase 985 to familiarize our small model with its new teaching format and role, again relying on some level of initial access to pre-986 collected reasoning examples that can be used accordingly. We also note that RLT training makes use of an additional student 987 model for computing the rewards. In practice, however, we note the downsides from this were minimal, as simple parameter offloading removed any potential memory burden of having this model on GPU memory during backpropagation, and actual 989

Large Language Models to Diffusion Finetuning

training time was dominated (> 90%) by the costs of long-context autoregressive generation with our computational setup.

As described in Section 6, we also did not explore the potential of sharing the teacher and student roles with the same optimized model, nor concurrent training of the two. This leaves two open questions yet unanswered: whether there can be an effective transfer between the two roles, potentially improving training efficiency by harnessing the relatedness of their objectives, and whether RL can devise online curricula tailored to the student's specific learning dynamics. Furthermore, due to computational constraints, we had to limit the maximum context size during our RL-training and student distillation phases to 16384, half the maximum context available with Qwen-based models. For the same reason, the considered RLTs only comprised small, inexpensive 7B models and did not consider further scaling. Breaking these constraints, thus, also remains another outstanding immediate direction for future improvements and further pushing the capabilities of our framework. Lastly, this work did not explore increasing the number and breadth of starting question-solution pairs used for obtaining distillation datasets, beyond the ones considered in prior work that made use of much more expensive pipelines. This leaves the potential extent of another key feature of our method unexplored, as with our small 7B RLT, it could be much more feasible to cheaply collect data for student distillation that matches the level of scale and efficacy of closed-source state-of-the-art data sources such as the one from Guo et al. (2025). 

#### **E.2. Broader impact**

Our work introduced a new class of models that enable small LMs to generate better synthetic distillation datasets beyond prior, much more expensive pipelines. Rather than introducing a new application, this work's contribution was foundational in nature, thus, its broader implications are bound to the effect of improving the capabilities and democratizing the training of large language models. To this end, as the accessibility and capabilities of LMs improve, there is an increasing chance of misuse for potentially harmful goals, such as influencing public opinion or obtaining access to sensitive information. Moreover, with increasing demand, the carbon footprint of LMs and their potential social impact might become increasingly relevant. However, we believe these risks are currently offset by the potential upsides of advancing the AI field, such as empowering humanity to better face upcoming environmental and economic challenges.

Reace	mina i	nnut	format
Neasu	JIIII 2 I	արաւ	Iurmat

1053 System prompt

1051 1052

1054

#### 1055 <|im\_start|>system

Your role as an assistant involves thoroughly exploring questions through a 1056  $\rightarrow$  systematic long thinking process before providing the final precise and accurate solutions. This requires engaging in a comprehensive cycle of  $\hookrightarrow$ 1058 analysis, summarizing, exploration, reassessment, reflection, backtracing, 1059  $\hookrightarrow$ and iteration to develop well-considered thinking process. Please  $\hookrightarrow$ 1060 structure your response into two main sections: Thought and Solution. In 1061  $\hookrightarrow$ the Thought section, detail your reasoning process using the specified 1062  $\hookrightarrow$ format: <|begin\_of\_thought|> {thought with steps separated with '\n\n'} 1063  $\hookrightarrow$ <|end\_of\_thought|> Each step should include detailed considerations such as  $\hookrightarrow$ 1064 analisying questions, summarizing relevant findings, brainstorming new 1065  $\hookrightarrow$ ideas, verifying the accuracy of the current steps, refining any errors,  $\hookrightarrow$ 1066 and revisiting previous steps. In the Solution section, based on various 1067  $\hookrightarrow$ attempts, explorations, and reflections from the Thought section, 1068  $\hookrightarrow$ systematically present the final solution that you deem correct. The  $\hookrightarrow$ 1069 solution should remain a logical, accurate, concise expression style and  $\hookrightarrow$ detail necessary step needed to reach the conclusion, formatted as follows: 1071  $\hookrightarrow$ <|begin\_of\_solution|> {final formatted, precise, and clear solution}  $\hookrightarrow$ 1072 <|end\_of\_solution|> Now, try to solve the following question through the 1073  $\hookrightarrow$ above guidelines:</im\_end/> 1074  $\hookrightarrow$ 1075

Generation prefix

#### 1078 <|im start|>user 1079 Return your final response within \boxed{}. Positive integers \$a\$ and \$b\$ are $\rightarrow$ such that the graphs of y=ax+5 and y=3x+b intersect the xx-ax at 1081 $\hookrightarrow$ the same point. What is the sum of all possible \$x\$-coordinates of these 1082 $\rightarrow$ points of intersection? 1083 \$\textbf{(A)}\ {-20}\qquad\textbf{(B)}\ {-18}\qquad\textbf{(C)}\ 1084 $\rightarrow$ {-15}\qquad\textbf{(D)}\ {-12}\qquad\textbf{(E)}\ {-8}\$<|im\_end|> 1085 <|im start|>assistant 1086 <|begin\_of\_thought|> 1087

*Figure 6.* Reasoning input format employed in traditional RL and student distillation, using an example question and the system prompt from Li et al. (2025a), providing the model instructions first to present a step-by-step rationale and then the problem's solution, deriving it from scratch.

1093

1094

1088 1089

1076

1077

- 1095
- 1096 1097

1098

```
1100
1101
1102
1103
       RLT input format
1104
       System prompt
1105
1106
       <|im_start|>system
1107
       Your role as an assistant involves providing precise and accurate solutions
1108
        \rightarrow before providing detailed explanations with your full work showing your
1109
           systematic thinking process leading to each solution. Your explanations
        \hookrightarrow
1110
           should show how you engaged in a comprehensive cycle of analysis,
        \hookrightarrow
1111
           summarizing, exploration, reassessment, reflection, backtracing, and
        \hookrightarrow
1112
           iteration to develop well-considered thinking process. Please structure
        \hookrightarrow
1113
           your response into two main sections: Solution and Explanation. In the
        \hookrightarrow
1114
          Solution section, present your well-thought solution that accurately
        \hookrightarrow
1115
        \rightarrow answers the question. The solution should remain a logical, accurate,
1116
           concise expression style and detail necessary step needed to reach the
        \hookrightarrow
1117
           conclusion, formatted as follows: <|begin_of_solution|> {final formatted,
        \hookrightarrow
1118
          precise, and clear solution | < | end_of_solution | >. In the Explanation
        \hookrightarrow
1119
          section, comprehensively detail your reasoning process using the specified
        \hookrightarrow
1120
           format: <|begin_of_explanation|> {explanation with steps separated with
        \hookrightarrow
1121
           '\n\n'} <|end_of_explanation|> Each step should show detailed
       \hookrightarrow
1122
          considerations leading to your solutions such as analisying questions,
        \hookrightarrow
1123
        \hookrightarrow summarizing relevant findings, brainstorming new ideas, verifying the
1124
           accuracy of the current steps, refining any errors, and revisiting
        \hookrightarrow
1125
           previous steps. <|im_end|>
        \hookrightarrow
1126
1127
1128
       Generation prefix
1129
1130
       <|im start|>user
1131
       Return your final response within \boxed{}. Positive integers $a$ and $b$ are
1132
       \rightarrow such that the graphs of y=ax+5 and y=3x+b intersect the xx-ax at
1133
        \hookrightarrow the same point. What is the sum of all possible $x$-coordinates of these
1134
        → points of intersection?
1135
       \left( (A) \right) \left( -20 \right) \right) \left( (B) \right) \left( -18 \right) \right) 
1136
       \rightarrow {-15}\qquad\textbf((D)}\ {-12}\qquad\textbf((E)}\ {-8}$<\im_end|>
1137
       <|im_start|>assistant
1138
       <|begin_of_solution|>
1139
1140
       ... the sum of all possible (x )-coordinates of these points of
1141
        \rightarrow intersection is \(\boxed{E}\).
1142
1143
       <|end_of_solution|>
1144
1145
       <|begin of explanation|>
1146
1147
1148
1149
     Figure 7. RLT Input format using an example question from Li et al. (2025a), providing the model instructions to produce a step-by-step
1150
     explanation given the solution to each problem as input.
```



1210	
1211	
1212	
1213	
1214	
1214	
1215	
1210	
1217	
1218	Destargessed D1 reasoning distillation example
1219	Tostprocessed KT reasoning distination example
1220	Masked distillation input
1221	
1222	< im_start >system < im_start >system
1223	rour role as an assistant involves thoroughly exploring questions < im_start >user Generate an executable Puthon function generated from the given promot
1224	
1225	
1225	Distillation reasoning trace
1220	< begin_of_thought >
1227	Okay, let's see. The problem is to find the minimum number of sheets needed to print N pages double-sided. Each sheet can hold two pages. So, I need to
1228	-7 Figure out now to calculate this based on N. Hmm, let's think about some examples. Like the sample input is 5, output is 3. Because 5 divided by 2 is 2.5, which rounds up to 3. So maybe the general
1229	↔ formula is to divide N by 2 and round up.
1230	wait, but wait. Let me check another example. Suppose N is 4. Inen 4/2 is exactly 2 sheets. That makes sense. If N is 1, then 1/2 is 0.5, which rounds up to
1231	So the formula is simply ceiling (N $/$ 2). But how to compute that in Python.
1232	In Python, if you do integer division with //, but when you need to round up, you can do $(N + 1) // 2$ . Let's see. For N=5: $(5 + 1) // 2 = b // 2= 3$ . Correct. For $\rightarrow N=4$ : $(4+1)/2=5/2=2.5$ , integer division gives 2. Wait, no. Wait, $5//2$ in Python is 2. So $(4+1)//2$ is $5//2=2$ . Which is correct. For N=1: $(1+1)//2=1$ .
1233	So the function is straightforward. Take the input N, compute $(N + 1)//2$ , and output it.
1234	Testing the sample input: input 5. (5+1)//2=6//2=3. Correct.
1235	Another test case: N=1 + 1. N=2 + 1. N=3 +2. All correct.
1236	Read N from input, compute (N+1)//2, print the result.
1230	So the Python code would be:
1227	n = int(input()) print( (n +1) // 2)
1230	But wait, perhaps there's an edge case. Let's see. For N=0? But according to constraints, N is at least 1, so no problem.
1239	Yes, so that's the solution. The code is simple. Just read N, calculate (N +1)//2, print.
1240	<pre></pre>
1241	Distillation solution
1242	< begin_of_solution >
1243	To solve this problem, we need to determine the minimum number of sheets of paper required to print a document with N pages using double-sided printing,
1244	<pre>### Approach</pre>
1245	The key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of pages by 2 and rounding up to the nearest integer. This can be efficiently computed using integer arithmetic by adding 1 to the number of pages before performing integer division by 2. This approach works because it effectively rounds up the result of dividing N by 2.
1240	### Solution Code
1247	print((n + 1) // 2)
1248	### Explanation 1. **Reading Input**: The input is read as an integer N. which represents the number of pages.
1249	2. **Calculation**: The formula `(N + 1) // 2` is used to compute the minimum number of sheets. Adding 1 to N before division ensures that any remainder from → the division results in rounding up to the next whole number.
1250	<ol> <li>**Output**: The result is printed, which is the minimum number of sheets needed.</li> <li>This approach efficiently computes the result in constant time. ((1) making it very efficient even for the upper constraint limit.</li> </ol>
1251	<pre></pre>
1252	
1253	
1254	Figure 9. Full formatted distillation data sample using a postprocessed DeepSeek R1 reasoning trace from the Bespoke dataset (Labs,
1255	2025).
1256	
1257	
1258	
1250	
1207	
1200	
1261	
1262	
1263	

1265	
1266	
1267	
1268	
1269	
1270	
1271	
1271	
1272	
1273	Dem 7D DI T d'atillation anomale
1274	Kaw /B KL1 distillation example
1275	Masked distillation input
1270	
1277	system system
1278	<pre>//im_start/&gt;user Generate an executable Python function generated from the given prompt</pre>
1279	<pre> assistant</pre>
1280	Distillation reasoning trace
1281	Dominition reasoning wave
1282	< begin_of_thought >
1283	Okay, let's see. The problem is to find the minimum number of sheets needed to print N pages double-sided. Each sheet can hold two pages. So, I need to
1284	Hmm, let's think about some examples. Like the sample input is 5, output is 3. Because 5 divided by 2 is 2.5, which rounds up to 3. So maybe the general
1285	Wait, but wait. Let me check another example. Suppose N is 4. Then 4/2 is exactly 2 sheets. That makes sense. If N is 1, then 1/2 is 0.5, which rounds up to
1286	$\hookrightarrow$ 1. So that's correct. So the formula is simply ceiling (N / 2). But how to compute that in Puthon
1287	In Python, if you do integer division with //, but when you need to round up, you can do (N + 1) // 2. Let's see. For N=5: (5 +1)//2 =6//2=3. Correct. For
1288	→ N=4: (4+1)/2=5/2=2.5, integer division gives 2. wait, no. wait, 5//2 in Fython is 2. So (4+1)//2 is 5//2=2. Which is correct. For N=1: (1+1)//2=1. → Correct. So yes, the formula is (N +1) // 2.
1289	So the function is straightforward. Take the input N, compute (N +1)//2, and output it. Testing the sample input: input 5. (5+1)//2=6//2=3. Correct.
1290	Another test case: N=1 + 1. N=2 + 1. N=3 +2. All correct.
1291	So the code would be something like:
1292	Read N from input, compute (N+1)//2, print the result. So the Python code would be:
1293	n = int (input ()) $nrint (n + 1) / (2)$
1294	But wait, perhaps there's an edge case. Let's see. For N=0? But according to constraints, N is at least 1, so no problem.
1295	Yes, so that's the solution. The code is simple. Just read N, calculate (N +1)//2, print.
1296	<pre></pre>
1207	Distillation
1208	< begin_of_solution >
1200	To solve this problem, we need to determine the minimum number of sheets of paper required to print a document with N pages using double-sided printing,
1200	### Approach
1201	→ pages by 2 and rounding up to the nearest integer. This can be efficiently computed using integer arithmetic by adding 1 to the number of pages before → performing integer division by 2. This approach works because it effectively rounds up the result of dividing N by 2.
1202	### Solution Code
1202	n = <sup>f</sup> int(input()) print((n + 1) // 2)
1303	### Explanation
1304	1. **Reading Input**: The input is read as an integer N, which represents the number of pages. 2. **Calculation**: The formula `(N + 1) // 2` is used to compute the minimum number of sheets. Adding 1 to N before division ensures that any remainder from the division ensures that any remainder from
1305	3. **Output**: The result is printed, which is the minimum number of sheets needed.
1306	This approach efficiently computes the result in constant time, O(1), making it very efficient even for the upper constraint limit.
1307	
1308	
1309	Figure 10. Full formatted distillation data sample using the raw reasoning trace from our 7B RLT model.
1310	
1311	
1312	
1313	
1314	
1315	
1316	
1317	

1320	
1321	
1322	
1323	
1324	
1325	
1326	
1327	
1328	
1329	
1330	
1331	Thought tokens KL reward ablation distillation example
1332	Masked distillation input
1333	
1334	< im_start >system < im_start >system
1335	Your role as an assistant involves thoroughly exploring questions
1336	<pre></pre>
1337	Dictillation reasoning trace
1338	Distination (Casoning Case
1339	< begin_of_thought >
1340	To solve this problem, we need to determine the minimum number of sheets of paper required to print a document with N pages using double-sided printing, $\leftrightarrow$ where each sheet can hold two pages.
1341	<pre>### Approach The key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of The key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of the key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the number of the key is determined by dividing the key is determined by dividing the key is determined by dividing the key is determined by dividin</pre>
1342	⇒ pages by 2 and rounding up to the mearest integer. Inis can be efficiently computed using integer arithmetic by adding 1 to the number of pages before performing integer division by 2. This approach works because it effectively rounds up the result of dividing N by 2.
1343	<pre>### Solution Code '''python n = int(input())</pre>
1344	print((n + 1) // 2)
1345	<pre>### Explanation 1. **Reading Input*: The input is read as an integer N, which represents the number of pages. 2. exclamulation The formula (N+1) // 2 is used to compute the minimum number of sheets. Adding 1 to N before division ensures that any remainder from </pre>
1346	→ the division results in rounding up to the next whole number. 3. **output**: The result is printed, which is the minimum number of sheets needed.
1347	This approach efficiently computes the result in constant time, O(1), making it very efficient even for the upper constraint limit.
1348	<pre></pre>
1349	Distillation solution
1350	< begin_of_solution >
1351	To solve this problem, we need to determine the minimum number of sheets of paper required to print a document with N pages using double-sided printing, $\leftrightarrow$ where each sheet can hold two pages.
1352 1353	<pre>### Approach The key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number of</pre>
1354	→ performing integer division by 2. This approach works because it effectively rounds up the result of dividing N by 2. ### Solution Code
1355	<pre>``python n = int (input()) print((n + 1) // 2)</pre>
1356	### Explanation
1357	<pre>1. **Reading input**: The input is read as an integer N, which represents the number of pages. 2. **Calculation*: The formula (N + 1) // 2 is used to compute the minimum number of sheets. Adding 1 to N before division ensures that any remainder from</pre>
1358	5. **output**: The result is printed, which is the minimum number of sneets needed. This approach efficiently computes the result in constant time, O(1), making it very efficient even for the upper constraint limit.
1359	<pre><!--end_of_solution --><!--im_end --></pre>
1360	
1361	

Figure 11. Full formatted distillation data sample using a reasoning trace collected after training our teacher ablated from the thought tokens KL reward  $r^{KL}$  in the RLT rewards defined in Equation 5.

1	
	Min/max reduction reward ablation distillation example
	Distillation reasoning trace
	<pre></pre> (begin_of_thought)> Other I made to call this problem where Takabashi wante to print N pages double sided with each sheet helding two pages. The task is to find the minimum
	Okay, I need to solve this problem where lakahashi wants to print w pages double-sided, with each sheet holding two pages. The task is to find the minimum of sheets required. Let's think through this step by step.
	So, the problem is about packing pages into sheets. Each sheet can have two pages. But since it's double-sided, maybe the way the pages are arranged after $\rightarrow$ how they fit. Wait, no, double-sided printing just means that two pages are printed on the same sheet, front and back. So for example, if I have page on the front and page 2 on the back, that's one sheet. But if there's an odd number of pages, the last sheet will have only one page. For example, if $j \in S$ , then the sample input shows 3 sheets. Let's see: 1-2 on first sheet, 3-4 on second, 5 on third. So each sheet except possibly the last can have
	- pages. Hmm, so the approach is to calculate how many pairs we can make. For N pages, the number of pairs is N//2. But if N is odd, there's an extra page. So the
	$\rightarrow$ total number of sneets would be (N // 2) + (1 11 N % Wait, but wait. Let's check another example. If N is 4. Then 4//2 is 2, and 4%
	Another example: N=3. $3//2$ is 1, 3%
	What about well life 17/2 is 0, 18 Wait, the problem says each sheet can hold two pages. So if there's an odd number of pages, the last sheet has one page. But the problem's sample input i
	→ which works out to 3 sheets. So maybe when N is odd, the formula is (N+1)//2. Wait, 5+1 is 6, 6//2 is 3. Which is correct. For N=3, 3+1=4, 4//2=2. → Correct. For N=1, 1+1=2, 2//2=1. So the formula is (N + 1) // 2. Let's test that. Wait, for N=1: (1+1)//2 = 1. Which is correct. For N=2: (2+1)/2=1.5, integer division is 1. Which is correct. For N=3: (3+1)/2=2. Correct. N=4: 5//2=2.54
	→ correct. N=5: 3. Correct. So this formula seems to work. Wait, but wait. Let's see. For N=1, if we use (N+1)//2, that's (1+1)//2=1. So yes. So the formula is (N +1)//2. But wait, in the problem statement, the
	→ sample input 5 gives output 3, which is (5+1)//2=3. So that's correct. So the formula is (N +1)//2. Because adding 1 gives even number of pages, whic → can be divided by 2, and then integer division by 2 gives the correct number of sheets. Wait, but let me think again. For N=1, the formula gives 1. So that's correct. For N=2, (2+1)//2=1.5→1. Which is correct. So yes, this formula works.
	Wait, but why would $(N+1)//2$ be the correct formula? Because when you have N pages, each sheet holds two pages, except possibly the last one. So the tota $\rightarrow$ number of sheets is the number of pairs (each pair is two pages) plus one if there's an odd number. So the number of pairs is N//2, and if N is odd, $\rightarrow$ there's an extra sheet. So the total is $(N/2) + (1 \text{ if } N \text{ is } N)$
	Yes. Because for N even, (N+1)/2 is (N/2) +0.5 → but integer division would be N/2. For example, N=4: (4+1)/2=2.5→2. Which is correct. For N=5: (5+1)/2=3 ↔ that's correct.
	So the formula is (N +1) $//$ 2.
	Wait, but wait. Let's test N=1. (1+1)/2=1. Correct. N=2: (2+1)/2=1.5+1. Correct. N=3: (3+1)/2=2. Correct. N=4: 2. Correct. N=5:3. Correct. So this formula works.
	So the problem reduces to calculating (N + 1) // 2.
	But wait, the input is N, which is between 1 and 100. So the function needs to read N from standard input, compute (N+1)//2, and output that. Wait, but wait. Let's take N=1. The sample input is 5, which gives output 3. For N=1, the output would be 1. But according to the problem statement, the $\rightarrow$ minimal number of sheets. So for N=1, you need 1 sheet. Because even though you can't fill a full sheet, you just have to use one sheet. So the form
	↔ works. So the code is straightforward. Read N. compute (N+1)//2. print that.
	Wait, but let me check another example. Let's say N=6. (6+1)//2=3.5+3. Wait, 6//2=3. Correct. So that's correct.
	Wait, but wait. Wait, no. Wait, (6+1)//2 is ///2=3. Wait, / divided by 2 is 3.5, integer division is 3. So (6+1)//2 is 3.5 +3. So yes. So 6 pages would $\leftrightarrow$ require 3 sheets. Each sheet has two pages. So 1-2, 3-4, 5-6. So three sheets. Correct.
	So the code is as simple as reading N, then compute the ceiling of N divided by 2. Which is $(N + 1)//2$ .
	Read N from input. Then compute (N +1) // 2. Print that result.
	Wait, but wait. Let's test N=1 again. (1+1)//2=1. Correct. Yes. So the code is straightforward.
	So the code would be:
	Read N as integer from input. Then print (N + 1) // 2. But wait, wait. Let's check N=1. (1+1)//2=1. Correct. For N=2: (2+1)//2=1. So 1 sheet. Which is correct. For N=3: 2 sheets. Correct. For N=4: 2. Correct.
	↔ N=5:3. Correct. For N=6:3. Correct. So the code is correct.
	Therefore, the Python function is:
	Read N from input, then compute (N +1) $//$ 2 and print it. So the code would be:
	n = int(input()) $print((n + 1) // 2)$
	Wait, but wait. Let me think again. For N=1, the answer is 1. For N=2, 1. So that's correct. So the code is correct.
	But wait, the problem says that N is the number of pages. So that's correct. So the code is as simple as that. I think that's it. So the answer is to compute (N +1) // 2.
	< end_of_thought >
	Distillation solution
	< begin_of_solution >
	Solve this problem, we need to determine the minimum number of sneets of paper required to print a document with w pages using double-sided printing, where each sheet can hold two pages.
	<pre>### Approach The key insight here is that each sheet of paper can hold two pages. Therefore, the minimum number of sheets required is determined by dividing the number pages by 2 and rounding up to the nearest integer. This can be efficiently computed using integer arithmetic by adding 1 to the number of pages befor performing integer division by 2. This approach works because it effectively rounds up the result of dividing N by 2.</pre>
	<pre>### Solution Code</pre>
	### Explanation
	1. **Reading Input**: The input is read as an integer N, which represents the number of pages. 2. **Calculation*: The formula `(N + 1) // 2' is used to compute the minimum number of sheets. Adding 1 to N before division ensures that any remainder → the division results in rounding up to the next whole number. 3. **Output**: The result is printed, which is the minimum number of sheets needed.
	This approach efficiently computes the result in constant time, O(1), making it very efficient even for the upper constraint limit.
1	< end_of_solution >< im_end >