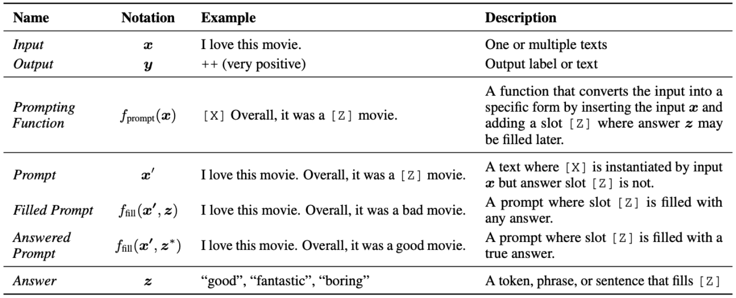
0. NLP Basics: how to represent language

- Rule-based, One-hot vectors
- Bag-of-Words: TF-IDF, LDA ...
- Language Model:
  - A language model is a machine learning model that predicts upcoming words.
  - Probability of a sentence
    - $P(W) = P(w_1, w_2, \ldots)$
  - Probability of next token
    - $P(w_k) = P(w_1, w_2, \ldots, w_{k-1})$
  - N-gram:
    language represented as a set of conditional probabilities
    $P(\text{I, saw, the, red, house}) \approx P(\text{I} \mid \langle s \rangle, \langle s \rangle) P(\text{saw} \mid \langle s \rangle, I) P(\text{the} \mid \text{I, saw}) P(\text{red} \mid \text{saw, the}) P(\text{house} \mid \text{the, red}) P(\langle /s \rangle \mid \text{red, h}$
- Static Word Embedding:
  word2vec
  assign each word a pre-defined vector
- contextual embedding (also language models):
  the vector of each token is calculated based on its context
  ususally Pretrained Language Models

0. Pretrained Language Model

- Paradigm: Pretrain + Fine-tune
  - 1. **PreTrain** a language model on large corpus with pretraining objectives.
    e.g. BERT: masked language model and next sentence prediction
  - 2. **Fine-tune** the model (usually later layers) on task specific dataset.
    e.g. text classification, named entity extraction
- ELMO, BERT, GPT

0. Prompt Learning



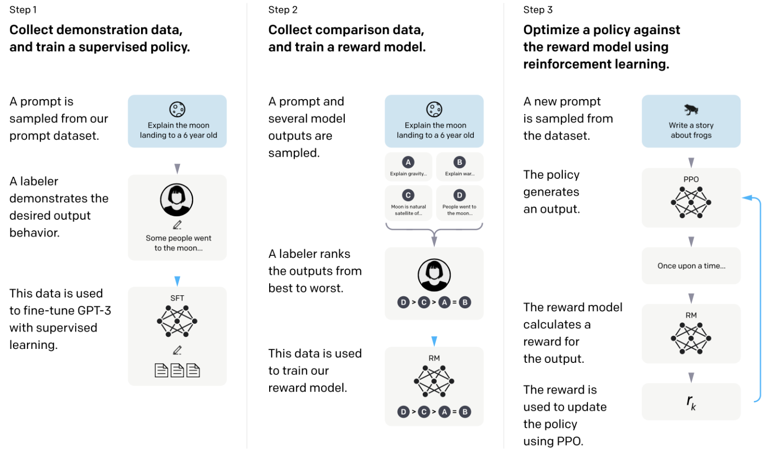
Name	Notation	Example	Description
Input	$\boldsymbol{x}$	I love this movie.	One or multiple texts
Output	$\boldsymbol{y}$	++ (very positive)	Output label or text
Prompting Function	$f_{\text{prompt}}(\boldsymbol{x})$	`[X]` Overall, it was a `[Z]` movie.	A function that converts the input into a specific form by inserting the input $\boldsymbol{x}$ and adding a slot `[Z]` where answer $\boldsymbol{z}$ may be filled later.
Prompt	$\boldsymbol{x}'$	I love this movie. Overall, it was a `[Z]` movie.	A text where `[X]` is instantiated by input $\boldsymbol{x}$ but answer slot `[Z]` is not.
Filled Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z})$	I love this movie. Overall, it was a bad movie.	A prompt where slot `[Z]` is filled with any answer.
Answered Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z}^*)$	I love this movie. Overall, it was a good movie.	A prompt where slot `[Z]` is filled with a true answer.
Answer	$\boldsymbol{z}$	"good", "fantastic", "boring"	A token, phrase, or sentence that fills `[Z]`

Table 2: Terminology and notation of prompting methods. $\boldsymbol{z}^*$ represents answers that correspond to true output $\boldsymbol{y}^*$.

- Traditional Supervised Learning
  - Take an **input** $\boldsymbol{x}$, and predict an **output** $\boldsymbol{y}$ based on a model $P(\boldsymbol{y}|\boldsymbol{x};\theta)$. ($\theta$ is parameters of the model)
- Prompt-based Learning
  - Learn a LM that models the probability $P(\boldsymbol{x};\theta)$ of text $\boldsymbol{x}$ itself and use this probability to predict $\boldsymbol{y}$ [1]

[1] Liu, et al. (2021)

0. InstructGPT and RLHF



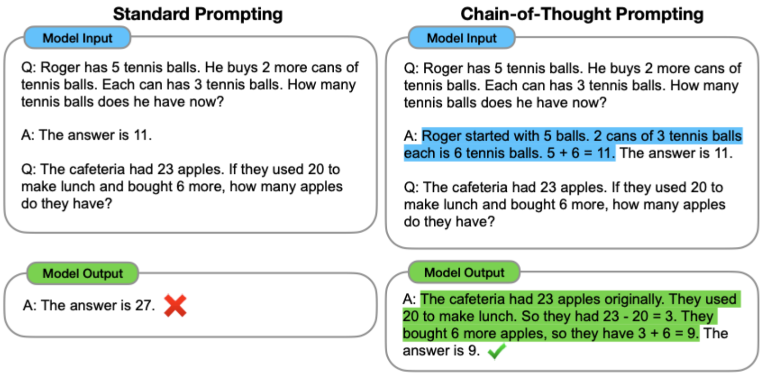
Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune GPT-3 with supervised learning.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A Explain gravity...
B Explain war...
C Moon is natural satellite of...
D People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

2

[2] Ouyang, et al. (2022)

0. Chain-of-Thoughts prompting



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

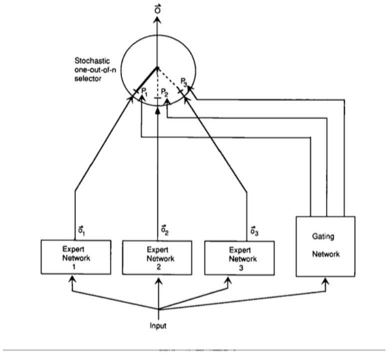
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

First used in inference [3], later used in fine-tune stage.

[3]Wei, et al. (2022)

0. Mixture of Experts



- use a gating network to decide expert that handles the input

0. Deepseek-V3's DeepSeekMoE

- DeepseekMOE

$$
\begin{aligned}
\mathbf{h}_t' &= \mathbf{u}_t + \sum_{i=1}^{N_s} \mathrm{FFN}_i^{(s)}\left(\mathbf{u}_t\right) + \sum_{i=1}^{N_r} g_{i,t}\, \mathrm{FFN}_i^{(r)}\left(\mathbf{u}_t\right), \\
g_{i,t} &= \frac{g_{i,t}'}{\sum_{j=1}^{N_r} g_{j,t}'}, \\
g_{i,t}' &= \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise}, \end{cases} \\
s_{i,t} &= \mathrm{Sigmoid}\left({\mathbf{u}_t}^T \mathbf{e}_i\right),
\end{aligned}
$$

  - only 37B out of 671B parameters are activated
  - uses finer-grained experts and isolates some experts as shared ones

$\mathbf{u}_t$: the FFN input of the t-th token
$\mathbf{h}'_t$: FFN output
N_s and N_r: the numbers of shared experts and routed experts, respectively;
$\mathrm{FFN}_i^{(s)}(\cdot)$ and $\mathrm{FFN}_i^{(r)}(\cdot)$: the i-th shared expert and the i-th routed expert, respectively;
K_r: the number of activated routed experts;
$g_{i,t}$: gating value for the i-th expert;
$s_{i,t}$: token-to-expert affinity;
$\mathbf{e}_i$: centroid vector of the i-th routed expert;
$\mathrm{Topk}(\cdot, K)$: the set comprising K highest scores among the affinity scores calculated for the t-th token and all routed experts.

3.1 Deepseek-Math: Group Relative Policy Optimization

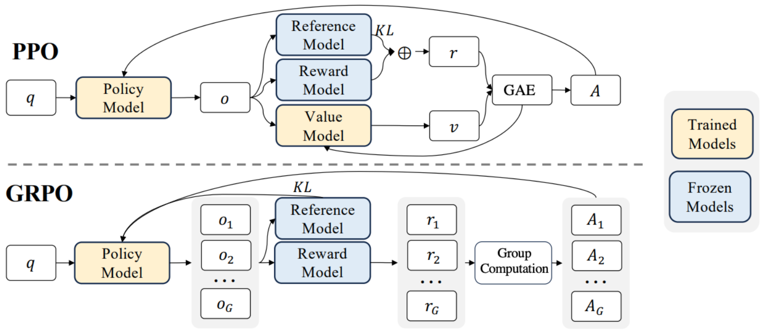


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

- Reduce computation burden
- Avoid complication of training value function

0. Deepseek-R1 Basics

Deepseek-V3:

- Base model

Deepseek-R1-Zero:

- Self-verification
- Reflection
- Generating long CoTs
    - Challenges: poor readability, language mixing

Deepseek-R1:

- improved reasoning patterns
- aligning with human preferences

1. Contributions

- Post-training
  - Reasoning capabilities of LLMs can be incentivized purely through RL, without the need for SFT (Supervised Fine-tuning)
  -
- Distillation
  - Reasoning patterns of larger models can be distilled into smaller models.

2. Approaches

Previous Work:

- Large amounts of supervised data to enhance model performance

Deepseek-R1-Zero:

- Pure reinforcement learning
- no supervised data

Deepseek-R1:

- cold-start data before RL
- multi-stage training

3. Deepseek-R1-Zero

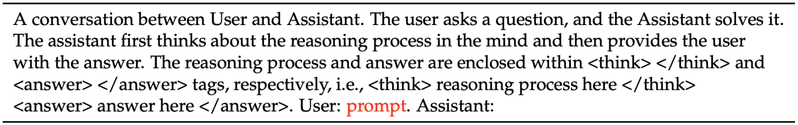
- Approach: Pure Reinforcement Learning
  - develop reasoning capabilities without any supervised data
  - self-evolution through a pure RL process
  - employ GRPO as RL framework
- Base model: Deepseek-V3

3.2 Deepseek-R1-Zero - Reward Modeling

Rule-based reward system:

- Accuracy rewards:
  - Evaluates whether the response is correct
- Format rewards:
  - Enforce the model to put thinking process between '<think>' and '</think>'
- no outcome or process neural reward model
  - avoid reward hacking
  - less training resource
  - simplicity of training pipeline

3.3 Deepseek-R1-Zero - Training Template



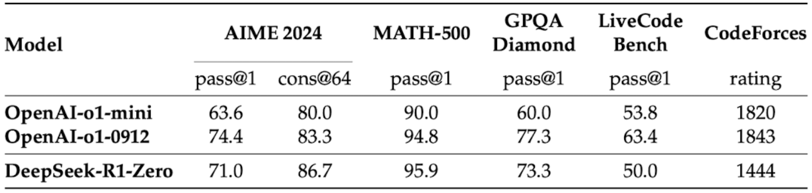
A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>. User: prompt. Assistant:

Table 1 | Template for DeepSeek-R1-Zero. prompt will be replaced with the specific reasoning question during training.

- straightforward template
  - produce a reasoning process followed by the final answer
- limit constraints, avoid content-specific biases
  - no mandating reflective reasoning
  - no promoting particular problem-solving strategies

3.4 Deepseek-R1-Zero - Performance and Self-evolution

Performance



Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	1820
OpenAI-o1-0912	74.4	83.3	94.8	77.3	63.4	1843
DeepSeek-R1-Zero	71.0	86.7	95.9	73.3	50.0	1444

Table 2 | Comparison of DeepSeek-R1-Zero and OpenAI o1 models on reasoning-related benchmarks.

- Robust reasoning capabilities without the need of any supervised fine-tuning data
- Models have ability to learn and generalize effectively through RL alone
- Can be further augmented through the application of majority voting

3.4 Deepseek-R1-Zero - Performance and Self-evolution

Self-evolution 1: R1-Zero's improvement through RL

- Intrinsic development within the model
  - naturally acquires the ability to solve increasingly complex reasoning tasksk by leveraging extended test-time computation
- Emergence as the test-time computation increases
  - Reflection: revisits and reevaluates its previous steps
  - Exploration of alternative appraches to problem-solving arise spontaneously
- Aha moment at an intermediante version of the model
  - Learns to allocate more thinking time to a problem by reevaluating initial approach
    - growing reasoning abilities
    - reinforcement learning can lead to unexpected and sophisticated outcomes
  - Insight:
    - explicitly teaching - No
    - provide it with right incentives - Yes

3.4 Deepseek-R1-Zero - Performance and Self-evolution

Self-evolution 2: thinking time increases as training step increases.

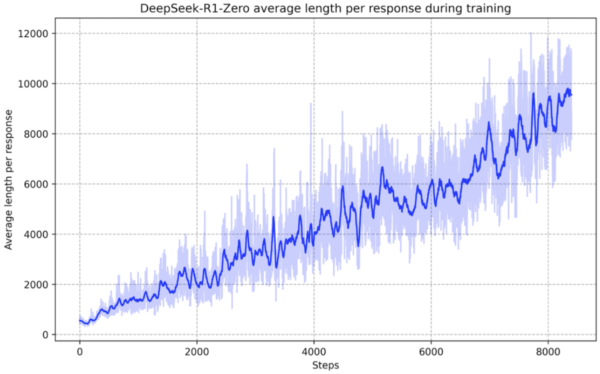


Figure 3 | The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time.

4. Deepseek-R1

- Challenges faced by Deepseek-R1-Zero:
  - poor readability
  - language mixing
- Deepseek-R1's Approach: SFT data with Reinforcement Learning
  - small amount of cold-start data
  - multi-stage training pipeline
- Base model: Deepseek-V3

4.1 Deepseek-R1 - Cold Start

Small amount of long CoT data to prevent initial unstable phase

- Collecting cold-start data (focus on reasoning capabilities)
  - Deepseek-R1-Zero's outputs
  - refining the results through post-processing by human annotators
- Deepseek-V3-base as the starting point
- Advantages of Cold Start:
  - Readability:
    - Readable pattern with a summary at the end of each response for filtering out according to user-friendliness
    - |special_token|<reasoning_process>|special_token|<summary>
  - Potential:
    - iterative training is better for reasoning models

4.2 Deepseek-R1 - Reasoning-oriented Reinforcement Learning: after fine tuning V3

- Language consistency reward during RL training:
  - Define: proportion of target language words in the CoT.
  - Mitigates CoT's language mixing.
- Final reward (directly sum):
  - Accuracy of reasoning tasks
  - Language consistency reward

When converge on reasoning tasks, this checkpoint is used to generate SFT data for later phases.

4.3 Deepseek-R1 - Rejection Sampling and Supervised Fine-Tuning: after the RL

For general capabilities other than reasoning.

- Reasoning data (600k)
  - filtered out low quality data from the previous checkpoint
  - Additional data: ground-truth and model predictions judged by Deepseek-V3.
- Non-reasoning data (200k)
  - Deepseek-V3's SFT data
  - CoT generated by Deepseek-V3's prompted answers
  - No CoT for simple questions.

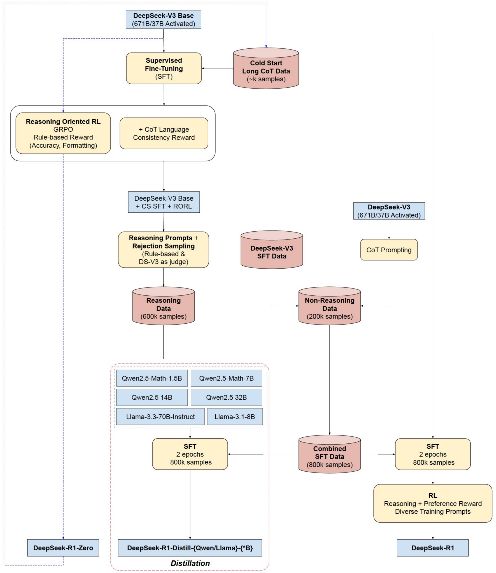
4.4 Deepseek-R1 - Further Training: 2-stage Framework

Retrain the DeepSeek-V3-Base model before the next RL stage.

Second RL stage:

- Helpfulness and harmlessness
- Refine reasoning capabilities

4.5 R1 Training Flow Chart

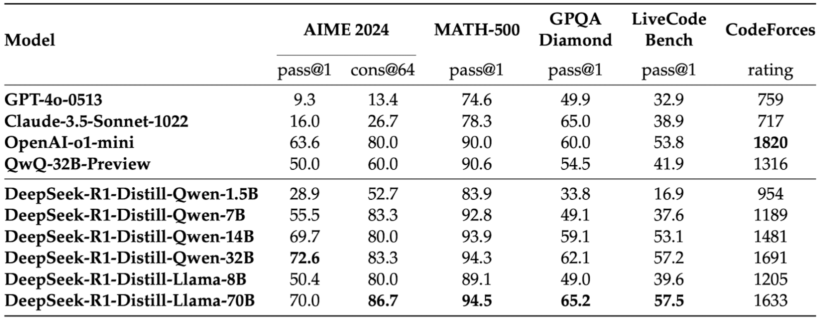


DeepSeek-V3 Base (671B/37B Activated)
Supervised Fine-Tuning (SFT)
Cold Start Long CoT Data (~k samples)
Reasoning Oriented RL GRPO Rule-based Reward (Accuracy, Formatting)
+ CoT Language Consistency Reward
DeepSeek-V3 Base + CS SFT + RORL
DeepSeek-V3 (671B/37B Activated)
Reasoning Prompts + Rejection Sampling (Rule-based & DS-V3 as judge)
DeepSeek-V3 SFT Data
CoT Prompting
Reasoning Data (600k samples)
Non-Reasoning Data (200k samples)
Qwen2.5-Math-1.5B
Qwen2.5-Math-7B
Qwen2.5 14B
Qwen2.5 32B
Llama-3.3-70B-Instruct
Llama-3.1-8B
SFT 2 epochs 800k samples
Combined SFT Data (800k samples)
SFT 2 epochs 800k samples
RL Reasoning + Preference Reward Diverse Training Prompts
DeepSeek-R1-Zero
DeepSeek-R1-Distill-{Qwen/Llama}-{*B}
DeepSeek-R1
Distillation

https://x.com/SirrahChan/status/1881488738473357753

5. Distillation

Data: previous 800k samples in R1 training.



Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	**1820**
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9	1316
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481
DeepSeek-R1-Distill-Qwen-32B	**72.6**	83.3	94.3	62.1	57.2	1691
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205
DeepSeek-R1-Distill-Llama-70B	70.0	**86.7**	**94.5**	**65.2**	**57.5**	1633

Table 5 | Comparison of DeepSeek-R1 distilled models and other comparable models on reasoning-related benchmarks.

Applying RL will further boost distilled model performance

Conclusion:

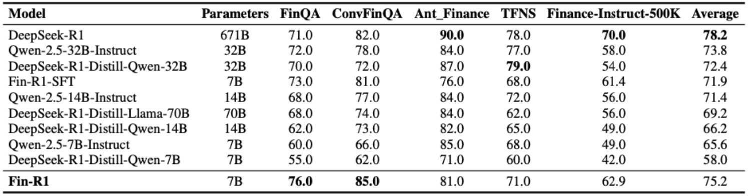
1. Distillation more powerful model into smaller models >= Smaller models relying on RL
2. Distillation requires more powerful base models

5.1 Distillation - Fin-R1

Biult financial reasoning dataset with Distilled data from deepseek-R1

Use LLM (Qwen 72B) to judge the thinking process

Applied Deepseek-R1's training Framework onto distilled data



Model	**Parameters**	**FinQA**	**ConvFinQA**	**Ant_Finance**	**TFNS**	**Finance-Instruct-500K**	**Average**
DeepSeek-R1	671B	71.0	82.0	**90.0**	78.0	**70.0**	**78.2**
Qwen-2.5-32B-Instruct	32B	72.0	78.0	84.0	77.0	58.0	73.8
DeepSeek-R1-Distill-Qwen-32B	32B	70.0	72.0	87.0	**79.0**	54.0	72.4
Fin-R1-SFT	7B	73.0	81.0	76.0	68.0	61.4	71.9
Qwen-2.5-14B-Instruct	14B	68.0	77.0	84.0	72.0	56.0	71.4
DeepSeek-R1-Distill-Llama-70B	70B	68.0	74.0	84.0	62.0	56.0	69.2
DeepSeek-R1-Distill-Qwen-14B	14B	62.0	73.0	82.0	65.0	49.0	66.2
Qwen-2.5-7B-Instruct	7B	60.0	66.0	85.0	68.0	49.0	65.6
DeepSeek-R1-Distill-Qwen-7B	7B	55.0	62.0	71.0	60.0	42.0	58.0
Fin-R1	7B	**76.0**	**85.0**	81.0	71.0	62.9	75.2