

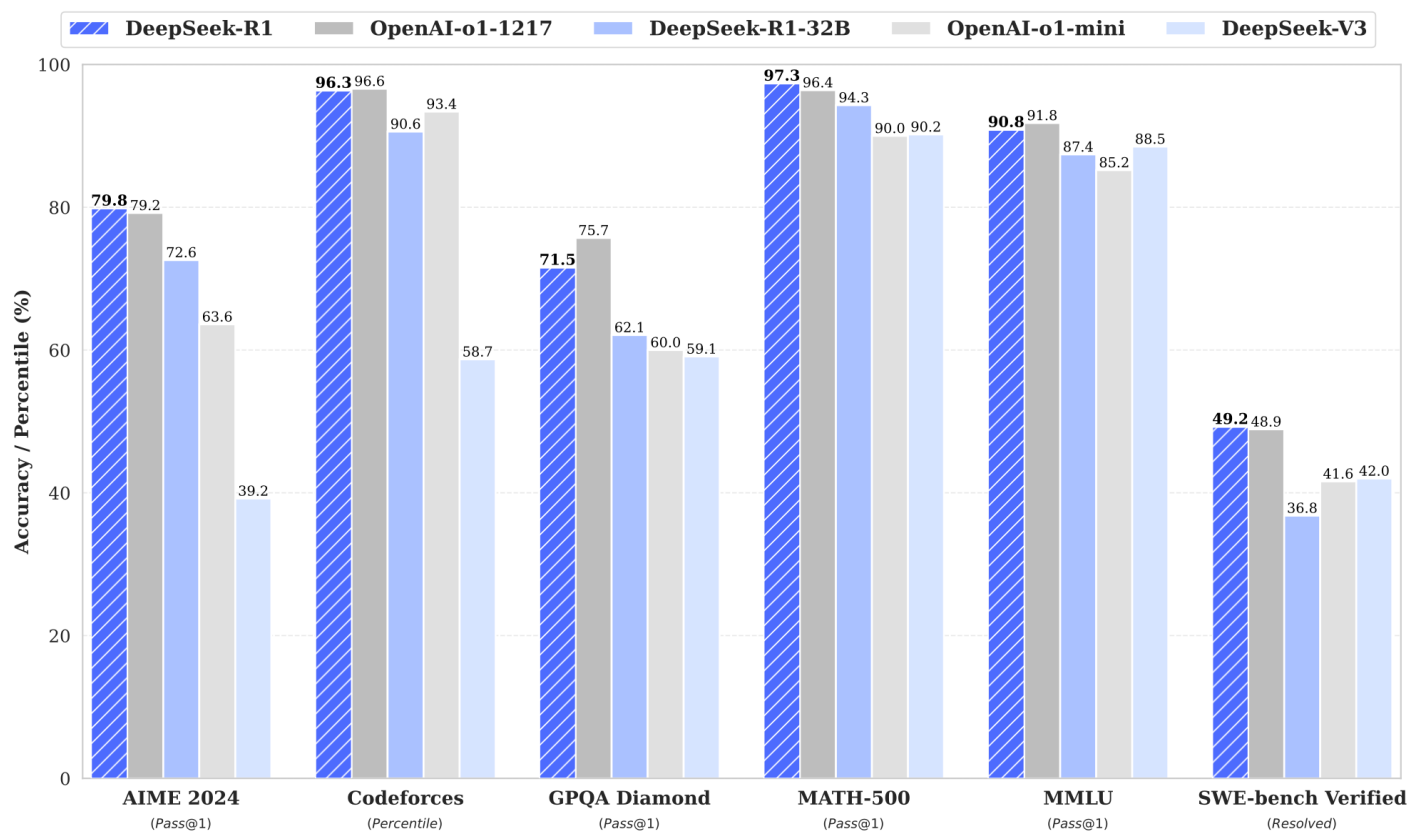
DeepSeek-R1: Insights

Original Paper: <https://arxiv.org/pdf/2501.12948>

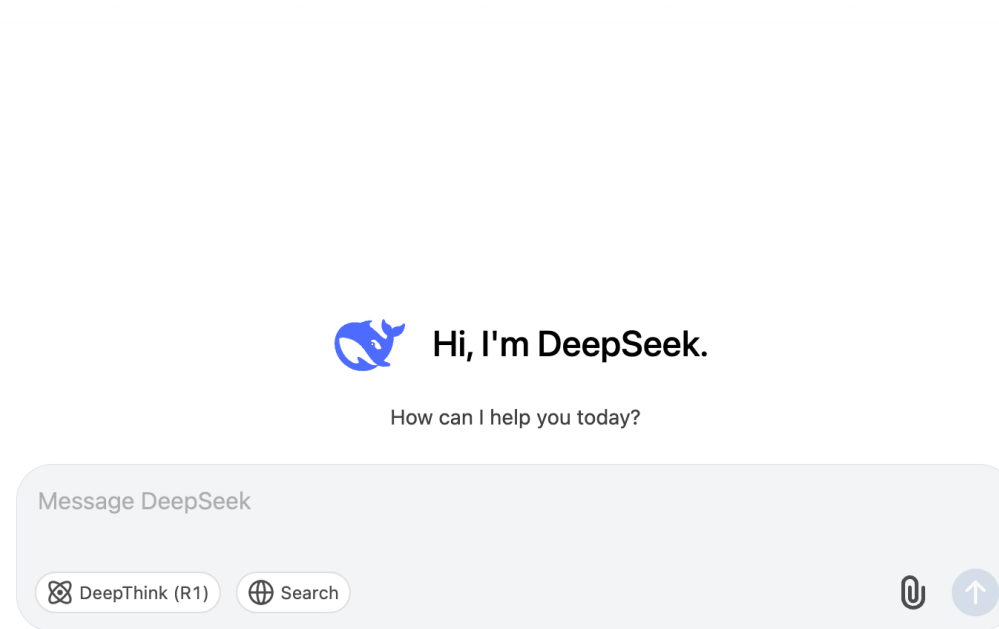
Presented by:

John Tan Chong Min

Performance - On par with o1



Web Usage (Free as of 13 Feb 2025)



chat.deepseek.com

Structured Output Usage (with AgentJo)

```
def llm(system_prompt: str, user_prompt: str) -> str:
    ''' Here, we use OpenAI for illustration, you can change it to your own LLM '''
    # ensure your LLM imports are all within this function
    from openai import OpenAI

    # define your own LLM here
    client = OpenAI(api_key = os.environ["DEEPSEEK_API_KEY"], base_url = "https://api.deepseek.com")
    response = client.chat.completions.create(
        model='deepseek-chat',
        temperature = 0,
        messages=[
            {"role": "system", "content": system_prompt},
            {"role": "user", "content": user_prompt}
        ]
    )
    return response.choices[0].message.content
```

```
from agentjo import *

res = strict_json(system_prompt = "Generate 1 main category and 3 sub-categories for topic",
                  user_prompt = "DeepSeek",
                  output_format = {"Main Category": "Suitable for 12 year olds, type: str",
                                   "Sub-categories": "type: list"},
                  llm = llm)

print(res)
```

```
{'Main Category': 'Technology and Innovation', 'Sub-categories': ['Artificial Intelligence', 'Machine Learning', 'Data Science']}
```

<https://github.com/tanchongmin/agentjo>

Agentic Usage (with AgentJo)

```
# Define a function for appointment
def make_appointment(person_name: str, start_time: str, end_time: str):
    '''Makes an appointment with person_name from start_time to end_time. Time is in xxxxH'''
    return f"Appointment made with {person_name} from {start_time} to {end_time}"

# Create the Agent
agent = Agent(
    agent_name = "Personal Secretary",
    agent_description = "You are a friendly assistant.",
    llm = llm,
).assign_functions([make_appointment])

result = agent.run("Make me an appointment with Paul at 3pm, Sally at 4pm. 1 hour each")
```

Observation: No subtasks have been completed yet. The task requires making two appointments: one with Paul at 3pm and another with Sally at 4pm, each for 1 hour.

Thoughts: To complete the task, I need to schedule the first appointment with Paul at 3pm for 1 hour. Once that is done, I can proceed to schedule the second appointment with Sally at 4pm for 1 hour.

Subtask identified: Schedule an appointment with Paul from 1500H to 1600H.

Calling function make_appointment with parameters {'person_name': 'Paul', 'start_time': '1500H', 'end_time': '1600H'}

> {'output_1': 'Appointment made with Paul from 1500H to 1600H'}

Observation: An appointment with Paul has been successfully made from 1500H to 1600H.

Thoughts: The next step is to make an appointment with Sally at 4pm for 1 hour. This can be done using the make_appointment function.

Subtask identified: Make an appointment with Sally from 1600H to 1700H using the make_appointment function.

Calling function make_appointment with parameters {'person_name': 'Sally', 'start_time': '1600H', 'end_time': '1700H'}

> {'output_1': 'Appointment made with Sally from 1600H to 1700H'}

Observation: Appointments have been successfully made with Paul from 1500H to 1600H and with Sally from 1600H to 1700H.

Thoughts: The assigned task of making appointments with Paul and Sally has been fully completed. No further actions are required.

Subtask identified: End Task

Task completed successfully!

<https://github.com/tanchongmin/agentjo>

Overview and Key Points

Overview

- DeepSeek-V3-Base as the base model and employ Group Relative Policy Optimisation (GRPO) as the Reinforcement Learning (RL) framework to improve model performance in reasoning
- Incorporates a small amount of cold-start data
- Applies RL to the base model to be trained model
- Generate trajectories from trained model, filter good ones, distill to a new model and repeat RL process again

Key Insight: Learning from Competition

- Some trajectories are good
 - Learn from them
- Some trajectories are bad
 - Do not repeat them
- Use an unbiased selection process: Ground truth comparison (Math), Verify with compiler (code)



Group Relative Policy Optimisation

- Group-learning could be viewed as a form of competition-based learning – choose the best few trajectories to learn from that are above average, and try not to repeat the worse few trajectories
- Can we do this for fine-tuning without RL too?

Group Relative Policy Optimization In order to save the training costs of RL, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question q , GRPO samples a group of outputs $\{o_1, o_2, \dots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model π_{θ} by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G} \sum_{i=1}^G \left(\min \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{old}}(o_i|q)} A_i, \text{clip} \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1 - \varepsilon, 1 + \varepsilon \right) A_i \right) - \beta \mathbb{D}_{KL}(\pi_{\theta} || \pi_{ref}) \right), \quad (1)$$

$$\mathbb{D}_{KL}(\pi_{\theta} || \pi_{ref}) = \frac{\pi_{ref}(o_i|q)}{\pi_{\theta}(o_i|q)} - \log \frac{\pi_{ref}(o_i|q)}{\pi_{\theta}(o_i|q)} - 1, \quad (2)$$

where ε and β are hyper-parameters, and A_i is the advantage, computed using a group of rewards $\{r_1, r_2, \dots, r_G\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})}. \quad (3)$$

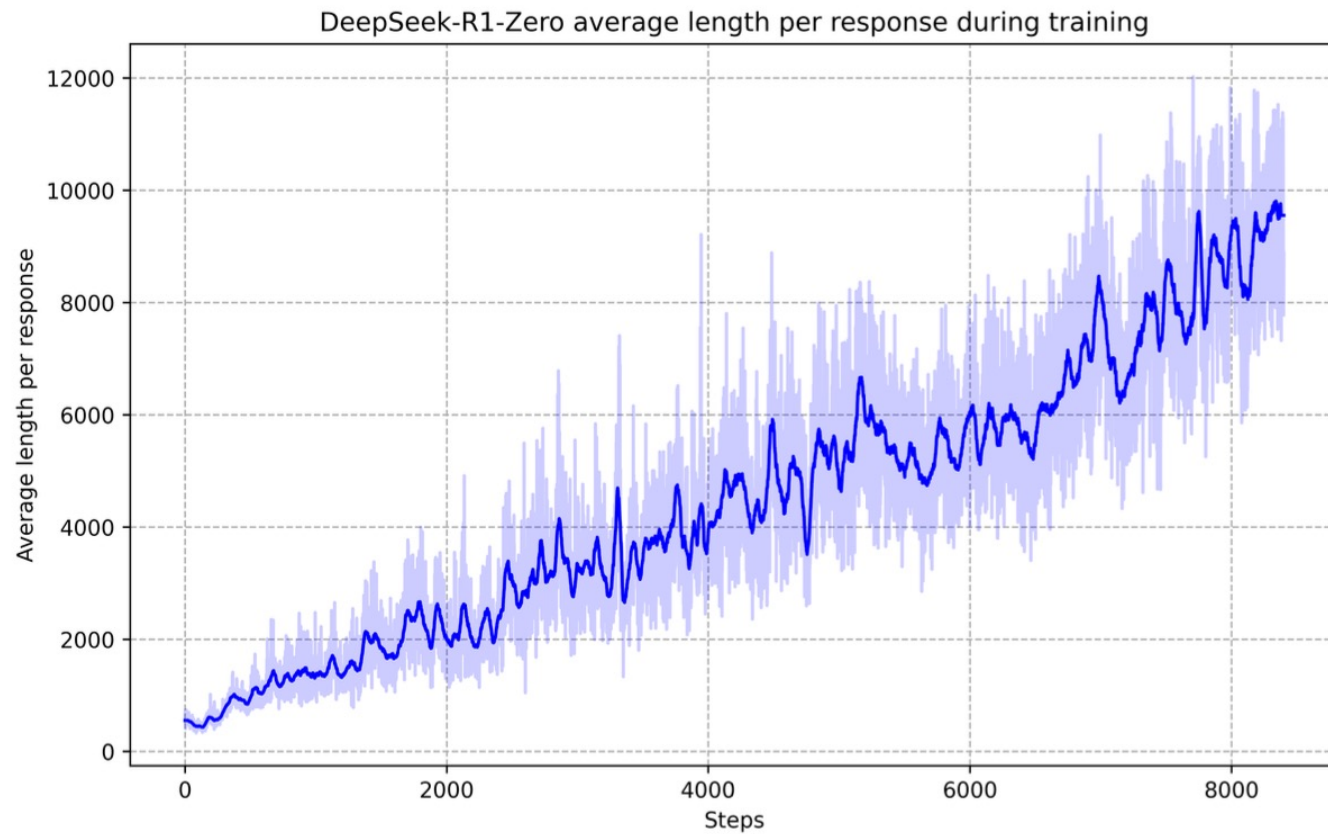
Prompt for Deepseek

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.

Question: Why can't we just do Chain of Thought using JSON output fields – Thought, then Answer

Model answers are longer after RL training



Wait.. Wait... as a form of reflection

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a+x}} = x$ is equal to

Response: <think>

To solve the equation $\sqrt{a - \sqrt{a+x}} = x$, let's start by squaring both ...

$$\left(\sqrt{a - \sqrt{a+x}}\right)^2 = x^2 \implies a - \sqrt{a+x} = x^2.$$

Rearrange to isolate the inner square root term:

$$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$$

...

Wait, wait. Wait. That's an aha moment I can flag here.

Let's reevaluate this step-by-step to identify if the correct sum can be ...

We started with the equation:

$$\sqrt{a - \sqrt{a+x}} = x$$

First, let's square both sides:

$$a - \sqrt{a+x} = x^2 \implies \sqrt{a+x} = a - x^2$$

Next, I could square both sides again, treating the equation: ...

...

How it is trained

Cold Start, RL, SFT to next model

Phase 1: Cold Start

- To prevent the early unstable cold start phase of RL training from the base model, for DeepSeek-R1 we **construct and collect a small amount of long Chain-of-Thought (CoT) data** to fine-tune the model as the initial RL actor
- **Question to Ponder: How much of DeepSeek-R1's performance is from distillation from other models (perhaps OpenAI o1 model)?**

Phase 2: Reasoning-oriented RL

- Use RL to train model in reasoning-intensive tasks such as coding, mathematics, science, and logic reasoning, which involve well-defined problems with clear solutions
- During the training process, we observe that CoT often exhibits **language mixing**, particularly when RL prompts involve multiple languages
- To mitigate the issue of language mixing, we introduce a language consistency reward during RL training, which is calculated as the proportion of target language words in the CoT
- We combine the **accuracy of reasoning tasks** and the **reward for language consistency** by directly summing them to form the final reward.

Phase 3: Rejection Sampling and Supervised Fine-Tuning

- When reasoning-oriented RL converges, we utilize the resulting checkpoint to **collect SFT (Supervised Fine-Tuning) data for the subsequent round**
- Additionally, because the model output is sometimes chaotic and difficult to read, we have filtered out chain-of-thought with mixed languages, long paragraphs, and code blocks.
- Iterative Model Training

Food for Thought

Diverse Trajectory Sampling

- Having more diverse trajectories to select the better ones is important.
- How can these trajectories be sampled diversely enough from just the model itself?
- Should we use more structured prompts to coax behaviour?

Is RL really needed?

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCodeBench
	pass@1	cons@64	pass@1	pass@1	pass@1
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9
DeepSeek-R1-Zero-Qwen-32B	47.0	60.0	91.6	55.0	40.2
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2

Distilled

- Distillation: Just supervised fine-tuning (SFT) without RL
- RL is not as good as distillation from a bigger model
- Instead of RL, selecting the right trajectories to fine-tune / pre-train on via distillation may be good enough

Process Reward Models are flimsy

- Intermediate rewards, such as Process Reward Models, are not ideal as:
 - It is challenging to explicitly define a fine-grain step in general reasoning
 - Determining whether the current intermediate step is correct is a challenging task
 - It inevitably leads to reward hacking
- For that reason, methods that require intermediate rewards such as Monte Carlo Tree Search are not ideal

Goal-Directed Neural Network

Extracted from: Learning, Fast and Slow (John and Mehul, 2023)

- At each time step, learns from:
 - **Previous states replay**
 - **Future states replay (only if lookahead trajectory found)**
- Intuition:
 - If we have a trajectory $A \rightarrow B \rightarrow C$
 - We know $A \rightarrow B$, $A \rightarrow C$, $B \rightarrow C$
 - Maximise learning from experience/lookahead

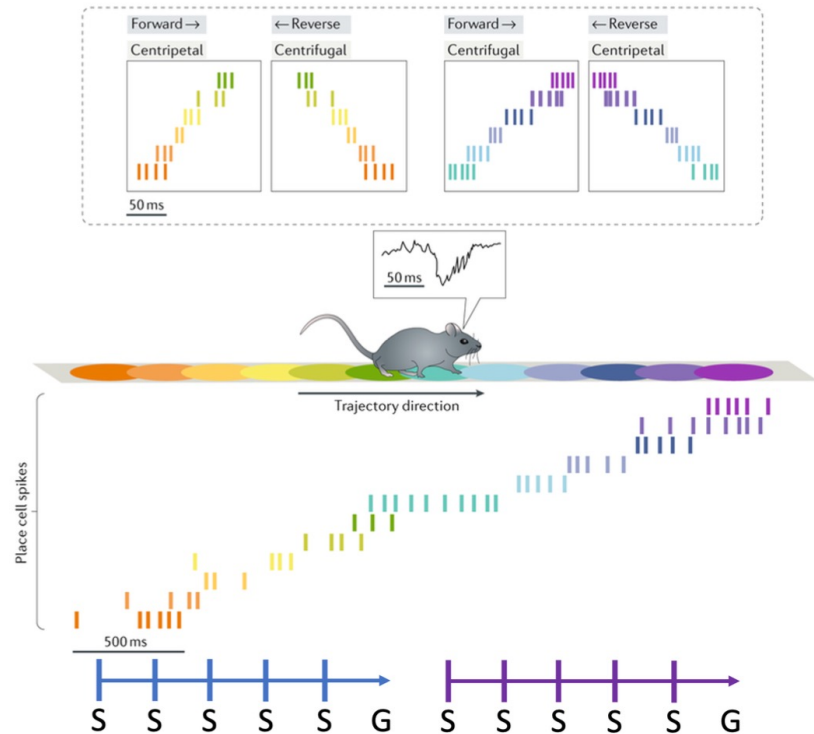
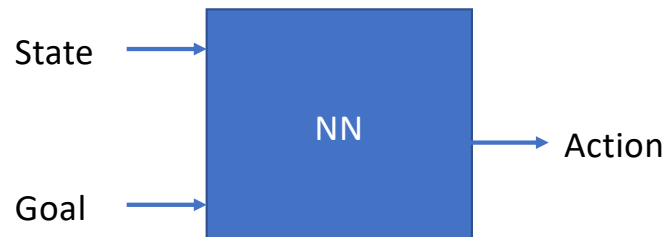


Figure extracted from Joo, H. R., & Frank, L. M. (2018). The hippocampal sharp wave-ripple in memory retrieval for immediate use and consolidation. *Nature reviews. Neuroscience*, 19(12), 744–757. <https://doi.org/10.1038/s41583-018-0077-1>

Proposal: Goal-Directed Fine-tuning in LLMs (Varying Start State)

- Prompt:
 - Question: Calculate $1 + 2 + 3$
- Completion:
 - Firstly, I would calculate $1+2 = 3$
 - Then, I would calculate $3+3 = 6$
 - The answer is 6

- Prompt:
 - Question: Calculate $1 + 2 + 3$
 - Firstly, I would calculate $1+2 = 3$
- Completion:
 - Then, I would calculate $3+3 = 6$
 - The answer is 6

- Prompt:
 - Question: Calculate $1 + 2 + 3$
 - Firstly, I would calculate $1+2 = 3$
 - Then, I would calculate $3+3 = 6$
- Completion:
 - The answer is 6



Start State shifts rightwards

Proposal: Goal-Directed Fine-tuning in LLMs (Varying Goal State)

- Prompt:
 - Question: Calculate $1 + 2$
 - Firstly, I would calculate $1+2 = 3$
- Completion:
 - The answer is 3

- Prompt:
 - Question: Calculate $1 + 2 + 3$
- Completion:
 - Firstly, I would calculate $1+2 = 3$
 - Then, I would calculate $3+3 = 6$
 - The answer is 6



Goal State shifts rightwards

Proposal: Goal-Directed Fine-tuning in LLMs (Changing Goal State to fit Trajectory)

- Prompt:
 - Question: Calculate $1 + 2 + 3$
- Completion:
 - Firstly, I would calculate $1+2 = 3$
 - Then, I would calculate $3+2 = 5$
 - The answer is 5



Re-define
Goal State according
to what is done

- Prompt:
 - Question: Calculate $1 + 2 + 2$
- Completion:
 - Firstly, I would calculate $1+2 = 3$
 - Then, I would calculate $3+2 = 5$
 - The answer is 5

Question to Ponder

- How much of DeepSeek-R1's performance is from distillation from models such as OpenAI o1 models?
- Can we apply the competition method of learning to learn from better trajectories, and not repeat worse trajectories without RL?
- Can we learn from competition if we do not have any ground truth?
- Can we scale up competition to include diverse agents rather than just trajectories sampled from one?
- Can we learn from goal-directed intermediate states as well?