

# Learning and Reasoning reading group # 1

Presenter: Irtaza Khalid  
Topic: Deepseek R1 spills the post-training beans

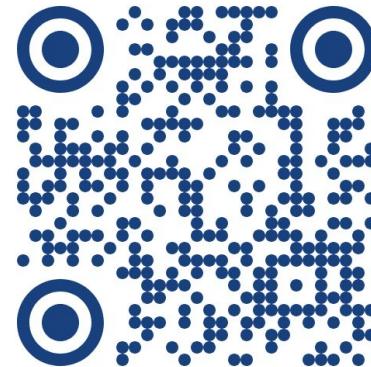
# Long term goals of the group

1. What we want this to be
  - a. Open to all at the section level (KRR+NLP+Theoretical ML+others if they don't do this already?)
  - b. with paper contributions from one volunteering or chosen attendee from a previous session
2. The topic space is anything within machine learning and reasoning
  - a. We encourage generality to capture most people's interests
  - b. Examples (ML): latest model developments: Modern BERT, diffusion, Scaling Laws etc.
3. The dates and times can be fixed via a poll:
  - a. For now, let's assume this will occur **every few weeks on Tuesday at 11 am?**
4. If you want to add more people to the seminar, do let us know!

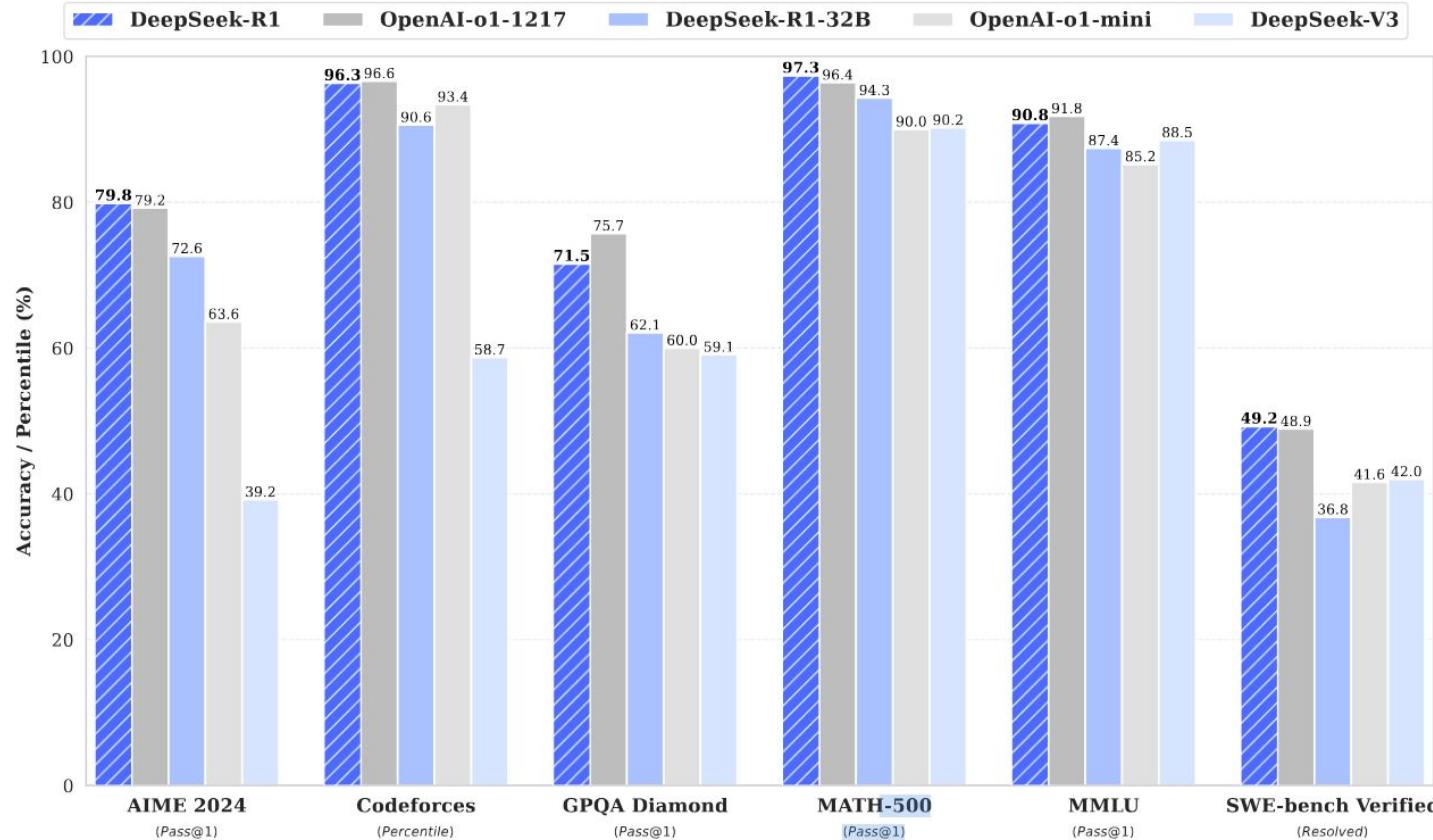
## DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI

# Deep seek R1



[Paper link](#)



# Brief summary

- Recipe
  1. Get a strong pretrained base model (their DS-v3 model)
  2. Cold start SFT
  3. Do GRPO (Group relative policy optimization) on DS-v3
    - a. using math and coding data
    - b. and a rules-based reward function
    - c. No SFT to elicit reasoning: only RL is enough
    - d. CoT thinking tokens: <think> <\think>
  4. R1-0 to R1: SFT to make the thinking CoT more human readable using “high quality language focussed” data (writing, QA etc.)
  5. Another RL round (unclear) using “prompts from all scenarios”
- Distillation from large to smaller models is better than ab initio post-training
- Open source model weights but not training data (likely due to copyright issues)



A shallow dive (granular picture)

# RL post 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.

- Two types of rewards:
  - Accuracy of response
  - Correctness of Format: e.g. thinking in <think> <\think>
  - No process rewards (reward hacking) or MCTS or anything fancy!

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.

Majority vote

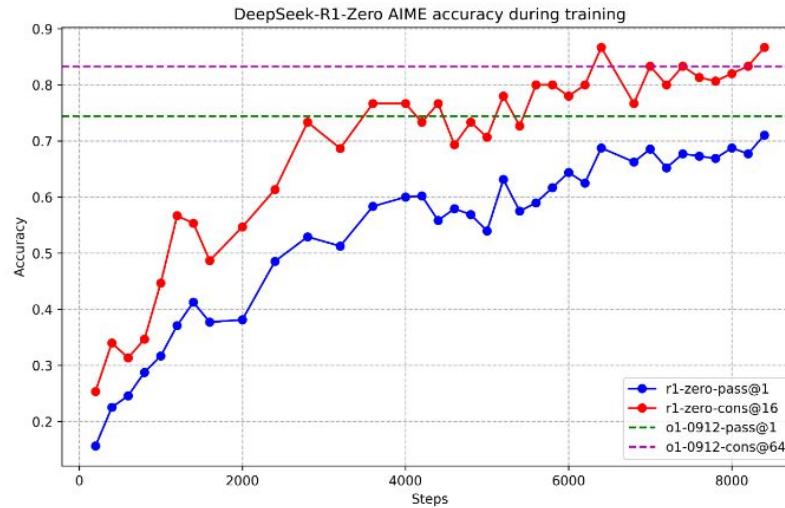
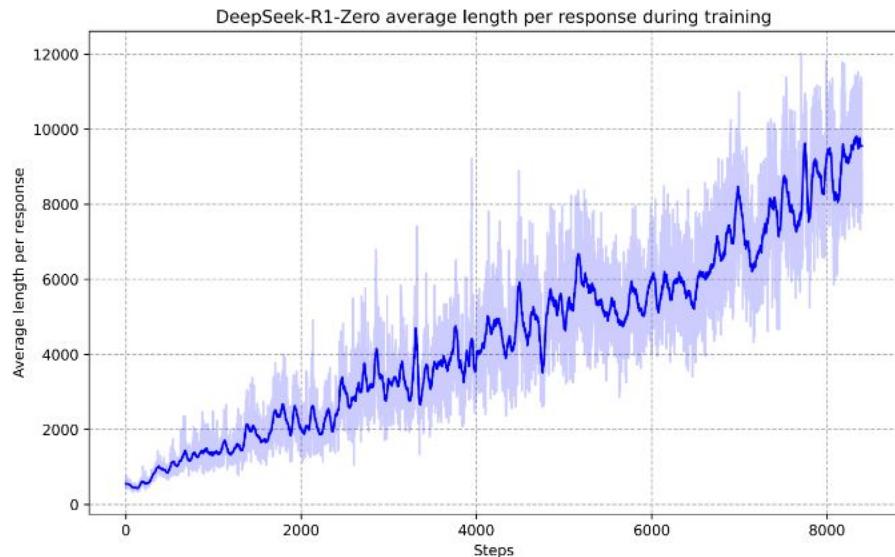


Figure 2 | AIME accuracy of DeepSeek-R1-Zero during training. For each question, we sample 16 responses and calculate the overall average accuracy to ensure a stable evaluation.

# R1-0 learns to think for longer and performance goes up

learned behaviors:

1. Leveraging test-time compute to improve final task performance
2. Aha moments:
  - a. questioning itself after a stream of reasoning trace a few times leads to “step change realizations”
  - b. These occur uncontrollably though
3. Mixed languages and incoherent CoT
4. Fix with R1 using SFT on reflection prompts with long CoT data (model generated) and refined by human annotator



# RL Primer

- Definitions:

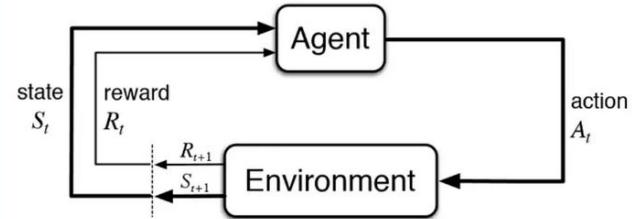
- Markov (memoryless) decision problem (MDP)
- with States ( $S_t$ ), Next states ( $S_{t+1}$ ), Actions ( $A_t$  for the transition) and
- a reward  $R_t(S_{t+1}|S_t, A_t)$
- Policy model or actor that acts  $A_t \sim \pi_\theta(S_t)$
- and the critic or value function that estimates the expected reward of states and actions.
- From [my thesis](#) (~pp 50)

The agent's ( $\pi$ ) goal is to maximize the discounted cumulative rewards called the returns,

$$\eta(\pi) := \mathbb{E}_{\pi,P} \left[ \sum_{k=0}^{\infty} \gamma^k r_k \right], \quad (2.44) \quad \text{Dynamic programming / Bellman update}$$

The value function  $V_\pi(s)$  is the expected returns following  $\pi$  starting from some state  $s$ . Mathematically,

$$V_\pi(s) = \mathbb{E}_{\pi,P}[R_j|s_j = s] = \sum_{s',r,a} \pi(a|s)P(s',r|s,a) (r + \gamma V_\pi(s')). \quad (2.46)$$



# Continued... Overarching RL strategy

1. *Policy evaluation* (Prediction): Given some policy  $\pi$ , we compute the value function ( $Q$  or  $V$ ). These form our prediction of the returns and inform the decision-making of the behavior policy.
2. *Policy improvement* (Control): Given a value function, we compute the optimal policy  $\pi$ , e.g., by acting greedily w.r.t. the value function, where the agent takes the action with the highest predicted value at each state.

$$V^*(s) = \max_{\pi} V_{\pi}(s) = \max_a \sum_{s',r} P(s',r|s,a) (r + \gamma V^*(s')) . \quad (2.49)$$

The companion argmax policy for above is the greedy policy in step 2

# Example training loop (just for fun)

---

**Algorithm 1:** Reinforcement learning loop

---

Initialize empty dataset  $\mathcal{D}$ , parametrized random policy  $\pi_\theta$ ,  $k \leftarrow 0$   
Observe initial state  $s_0$   
**while**  $k < T/\Delta t$  **do**  
    Execute  $\mathbf{a}_k \leftarrow \pi_\theta(\cdot | s_k)$   
    Observe  $s_{k+1}, r_k \leftarrow \mathcal{E}(s_k, \mathbf{a}_k)$   
    Store  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_k, s_{k+1}, \mathbf{a}_k, r_k)\}$   
     $k \leftarrow k + 1$   
// if require update: perform model-free update of parameters  
(e.g. policy  $\pi_\theta$ )

---

# GRPO

- Basically PPO (if familiar, if not it's here: [Schulman 2017](#)) but with 2 major points to note
  - No value function: estimate the value using a group-wise mean reward over the outputs of the older policy
  - ~~Reward model is also an LLM~~
  - Reward model is verifiable:
    - rule based (boxed math answers can be verified)
    - Compiler tests against predefined test cases for code
-

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  $\pi_\theta$  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  $\varepsilon$  and  $\beta$  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:

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

### 3.1. DeepSeek-R1 Evaluation

Benchmark (Metric)	Claude-3.5-Sonnet-1022	GPT-4o 0513	DeepSeek V3	OpenAI o1-mini	OpenAI o1-1217	DeepSeek R1
Architecture	-	-	MoE	-	-	MoE
# Activated Params	-	-	37B	-	-	37B
# Total Params	-	-	671B	-	-	671B
MMLU (Pass@1)	88.3	87.2	88.5	85.2	<b>91.8</b>	90.8
MMLU-Redux (EM)	88.9	88.0	89.1	86.7	-	<b>92.9</b>
MMLU-Pro (EM)	78.0	72.6	75.9	80.3	-	<b>84.0</b>
DROP (3-shot F1)	88.3	83.7	91.6	83.9	90.2	<b>92.2</b>
English	IF-Eval (Prompt Strict)	<b>86.5</b>	84.3	86.1	84.8	-
	GPQA Diamond (Pass@1)	65.0	49.9	59.1	60.0	<b>75.7</b>
	SimpleQA (Correct)	28.4	38.2	24.9	7.0	<b>47.0</b>
	FRAMES (Acc.)	72.5	80.5	73.3	76.9	-
	AlpacaEval2.0 (LC-winrate)	52.0	51.1	70.0	57.8	-
	ArenaHard (GPT-4-1106)	85.2	80.4	85.5	92.0	-
	LiveCodeBench (Pass@1-COT)	38.9	32.9	36.2	53.8	63.4
	Codeforces (Percentile)	20.3	23.6	58.7	93.4	<b>96.6</b>
	Codeforces (Rating)	717	759	1134	1820	<b>2061</b>
	SWE Verified (Resolved)	<b>50.8</b>	38.8	42.0	41.6	48.9
Code	Aider-Polyglot (Acc.)	45.3	16.0	49.6	32.9	<b>61.7</b>
	AIME 2024 (Pass@1)	16.0	9.3	39.2	63.6	<b>79.2</b>
	MATH-500 (Pass@1)	78.3	74.6	90.2	90.0	<b>96.4</b>
	CNMO 2024 (Pass@1)	13.1	10.8	43.2	67.6	-
	CLUEWSC (EM)	85.4	87.9	90.9	89.9	-
Chinese	C-Eval (EM)	76.7	76.0	86.5	68.9	-
	C-SimpleQA (Correct)	55.4	58.7	<b>68.0</b>	40.3	-
						<b>63.7</b>

Outside of the RL post training domain, there are relatively small gains

Table 4 | Comparison between DeepSeek-R1 and other representative models.

# Distillation makes small models competitive with prev SoTA (GPT-4o)

## 3.2. Distilled Model Evaluation

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.

A Qwen 14B is pretty competitive compared to a Llama 70b and better than(!) GPT-4o and significantly smaller than both models

These are limited evals on coding/math tho

# Distillation works better than ab initio GRPO

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	<b>72.6</b>	<b>83.3</b>	<b>94.3</b>	<b>62.1</b>	<b>57.2</b>

Table 6 | Comparison of distilled and RL Models on Reasoning-Related Benchmarks.

# Summary

1. A simple post-training approach using memory efficient RL can unlock reasoning in verifiable domains via CoT tokens
2. Distillation of reasoning ability from large to small works better than ab initio RL post training on small models
3. A strong base model and stable training “meta” strategies are crucial for unlocking downstream abilities
4. Interesting learned behavior emerges (self-reflection/ error correction, “ahas”) during RL training that is absent in more supervised training patterns.
  - a. can also simply append “Wait” to a model’s outputs to get super-long CoTs...



## Shallow dive 2: DS-v3 innovations

# Deepseek V3 innovations [out of scope or for next time?]

1. Flash Multihead latent attention (recently open sourced kernels on github [here](#))
2. Mixed precision FP8 training
  - a. DualPipe vs. ZeroBubble (distributed/parallelism innovation)
  - b. FP32 accumulation
3. Multi-token prediction (new variant) similar to speculative decoding
4. MoE
5. Auxiliary-loss-free load balancing
  - a. (just a dynamic bias hyperparameter to the router output?)
6. Distillation from R1 (learned verification+reflection) back into the base model
7. Data curation (not complete)

# FP8 training with increased precision accumulation

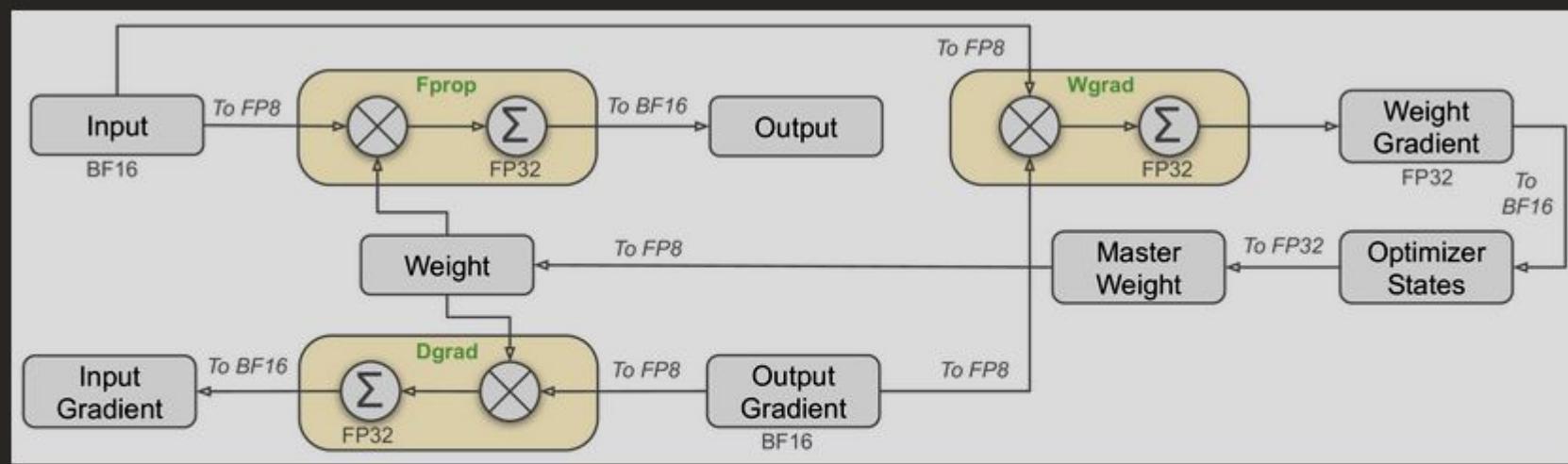


Figure 6: The overall mixed precision framework with FP8 data format. For clarification, only the Linear operator is illustrated.

# DualPipe: constant compute-to-comm ratio over experts

Overlap comms and compute with 1 forward and backward chunk. Goes beyond Zerobubble (Qi 2023)

## 3.2.1 DualPipe and Computation-Communication Overlap



Figure 4: Overlapping strategy for a pair of individual forward and backward chunks (the boundaries of the transformer blocks are not aligned). Orange denotes forward, green denotes "backward for input", blue denotes "backward for weights", purple denotes PP communication, and red denotes barriers. Both all-to-all and PP communication can be fully hidden.