Advanced Methods in Text Analytics

Reasoning LLMs

What is Test-Time Compute?

- Quite a hot topic in LLM research
  - Lots of papers in recent months (late 2024, early 2025)
  - Brand new direction, many things unclear
- **Test-time compute**: computations done at inference time
  - Specifically for LLMs, **extra computations** done at inference time.
- In the literature, it is a **synomyn for reasoning LLMs**
  - To avoid generating confusion, we also treat them as synonyms here
  - But we will discuss this relation in detail
- Today, we'll:
  - Build some **intuition about** what we mean by **reasoning**
  - **Discuss** what we mean by **reasoning LLMs**
  - Discuss the methods behind **DeepSeek-R1**
- **Disclaimer:** this is about *reasoning LLMs*, not about reasoning in general
  - We are not discussing whether LLMs reason the way humans reason
  - In fact, there is evidence they don't reason, e.g. see here, here or here

Outline

1. Reasoning LLMs
2. Overview of DeepSeek-R1

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Reasoning LLMs

A Computational Perspective (1)

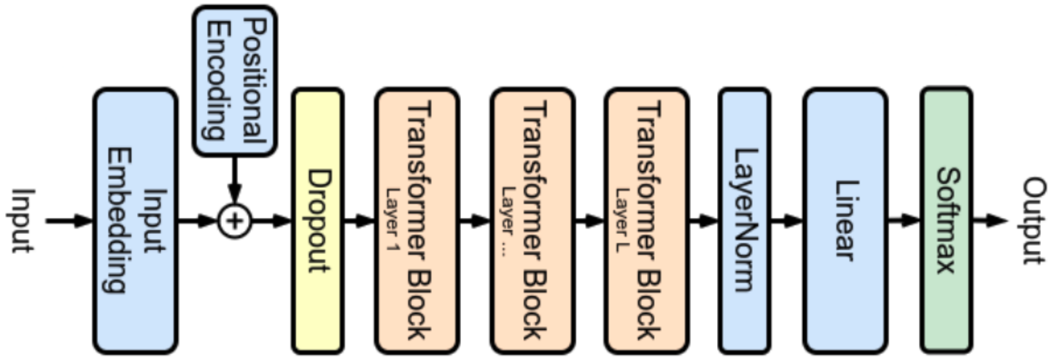
- **Computational costs** are usually a function of *both*:
  - (i) resources used
  - (ii) task at hand
- (i) Given fixed task, **cost changes depending on resources used**
  - E.g. a solution may be $O(n^2)$ when using two for loops, but become $O(n)$ when using a single for loop and a dictionary
  - Similarly, runtime costs increase massively when using a CPU instead of a GPU to train a model
- (ii) Given fixed resources, **cost changes depending on task**
  - While this depends on resources used, some problems known to have optimal solutions in polynomial time, whereas others don't
  - Generally, some tasks are easier than others, think runtime/memory costs
- Let's build some **intuition with a more concrete example**
  - Should be useful to illustrate the **motivation behind reasoning LLMs**

A Computational Perspective (2)

- Imagine we use the best LLM on earth in the following way:
  - **Input:** ask model a yes-or-no question
  - **Output:** predict answer by decoding a single token from the model
- We can think of decoding the output as we please
  - E.g. constrained decoding on the set of tokens {yes, no}.
  - Clarify in prompt that answer should only be Yes or No (works well with largest models)
- We **then ask the model the following questions**:
  - Is Berlin the capital of Germany?
  - Is 2048 the square of 64?
  - Does *P* equal *NP*?
- **Computational resources available** for model to answer each question?

A Computational Perspective (3)

- Compute resources for each question: **a single forward pass**


Input
Input Embedding
Positional Encoding
Dropout
Transformer Block Layer 1
Transformer Block Layer ...
Transformer Block Layer L
LayerNorm
Linear
Softmax
Output

[Image source](#)

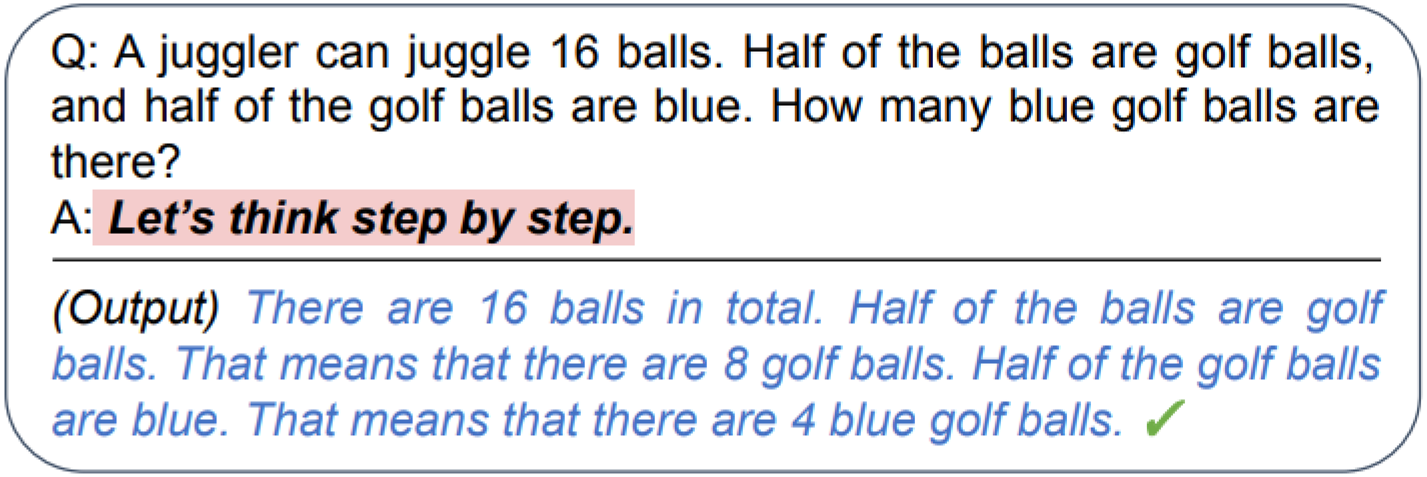
- In other words, resources are fixed, and the same, for different tasks
  - Even from residual stream perspective, we have an **upper bound on amount of compute** the model has available
- This is **essentially how LLMs are often used**
  - Same resources for many different tasks, often a single forward pass
  - **Claim was:** if model/training data is large enough, this will work

What are Reasoning LLMs? (1)

- **Reasoning:** yet another vague term
  - As vague as "artificial intelligence" or "data science"
  - Has taken many forms throughout decades of research, e.g. traversing a graph, solving a logical problem, etc.
- **Reasoning LLMs**: producing an answer to a task in **multiple steps**
  - No agreed upon definition in the literature, definition often missing
  - But this generally describes the process that all papers are studying
- **Multiple steps** can mean different things for different methods
  - Generating multiple answers and picking most likely one (i.e. beam search)
  - Generating multiple answers and using a verifier to pick best answer (verifier can be another model trained for this task, as in Cobbe et al.)
- Generally, **multiple steps = extra computation at inference time**
  - But note the pattern: **generating more tokens than usual**
  - I.e. more forward passes --> more computational resources

What are Reasoning LLMs? (2)

- A classic example of LLM reasoning: **chain-of-thought (CoT) prompting**



Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

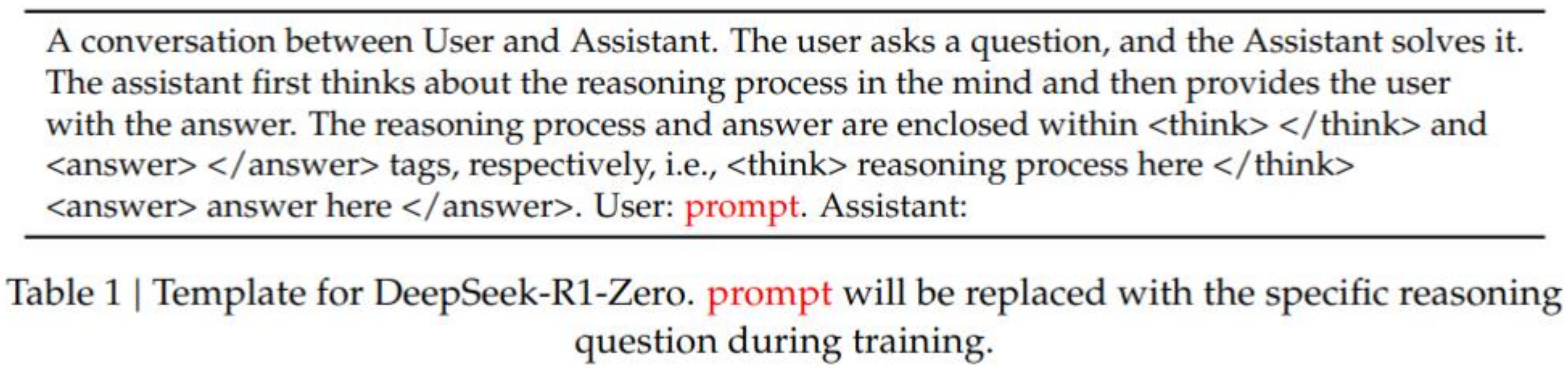
(Output) *There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls.* ✓

Kojima et al. 2022

- **Language modeling interpretation:** autoregressively producing an explicit reasoning path increases probability of correct answer
- **Computational interpretation:** model iteratively uses indefinite amount of compute (multiple forward passes) to produce an answer
- Generally, **can be seen as a much more involved form of decoding**
    - Note that **LLM agents** can **also** be seen as extra computation at test time

What are Reasoning LLMs? (3)

- Motivated by CoT, latest trend in mainstream models (not necessarily a research trend): **thinking phase during inference**
  - Models "allowed" to think before they arrive at an answer
  - Thinking phase usually denoted by thinking tokens or thinking tags



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.

Image source

- "Ok, we understand what reasoning LLMs are, but..."
  - **What can we accomplish with them?**
  - They are **more costly during inference**, so when should I use them?

LLMs vs Reasoning LLMs

- Designed for more complex problems
  - No need for less reasoning-intensive tasks, e.g. knowledge-based QA
  - Generally better at coding/math and similar type of problems
- Multiple studies show CoT reasoning LLMS better on complex tasks
  - E.g. Wei et al. (2022), Suzgun et al. (2022), Saparov et al. (2023)
- More recently, some studies focus on the difference between the cost of pre-training vs the cost at inference time
  - E.g. Snell et al. (2024) show that optimal test-time scaling a small model can be more less expensive than pre-training a much larger model
  - Improving on Snell et al., Liu et al. (2025) show that with test-time scaling, a 1B model can achieve the performance of a 405B model
- Computational perspective more concrete than reasoning perspective
  - Unclear if human-like reasoning taking place, e.g. Mirzadeh et al. (2024)
- "I am convinced, reasoning LLMs are useful in some settings. Now..."
  - **How do we build reasoning LLMs?**

How to Construct Reasoning LLMs

- Two main ways:
  - **Post-training** methods, i.e. more training but now on reasoning tasks
  - **Test-time** methods, i.e. only extra computations at inference time
- Naturally, the starting point is a "base" LLM
  - In this sense, reasoning LLMs are a form of LLM specialization
  - Same as RAG, coding assistants, etc.
- **Examples of test-time methods**
  - Muenninghoff et al. (2025) use a WAIT token to force the model to double check its reasoning path (requires training this new token)
  - Geiping et al. (2025) add an RNN-like component that the model can use at inference time to reason iteratively (i.e. reasoning process more obscure)
- **Examples of post-training methods**
  - Many, mostly based on reinforcement learning (RL)
  - Arguably latest breakthrough comes from DeepSeek-R1

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Overview of DeepSeek-R1

The DeepSeek-R1 Family

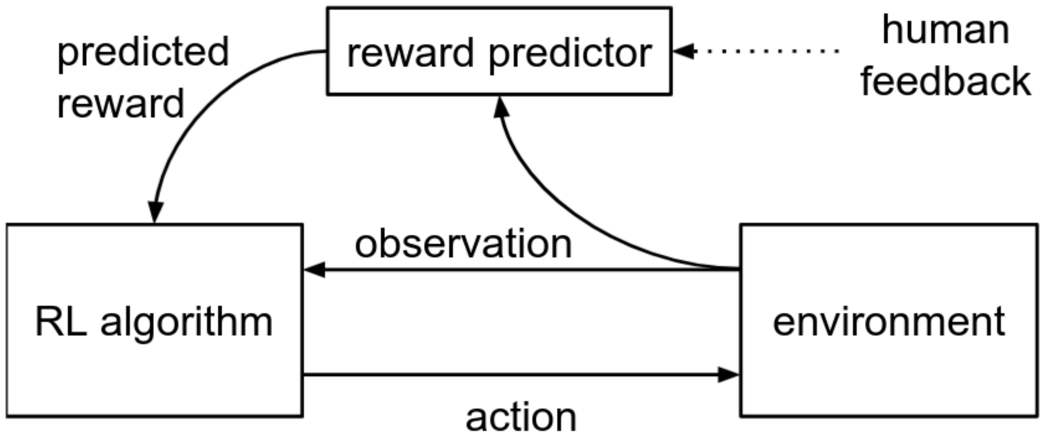
- All based on DeepSeek-V3 base model
  - A mixture-of-experts model with 671B parameters
  - Many innovations, e.g. multi-head latent attention from DeepSeek V2
  - Lots of great engineering to scale with *relatively limited resources*
- Today's focus: the reasoning models build on top of DeepSeek-V3
  - Known as the DeepSeek-R1 family
- **DeepSeek-R1-Zero**
  - Achieves reasoning with "pure RL" (RL = reinforcement learning)
- **DeepSeek-R1**
  - Their flagship reasoning model built on top of DeepSeek-R1-Zero
- **DeepSeek-R1-Distill**
  - Smaller reasoning models distilled from DeepSeek-R1
  - Based on small versions of Llama3 and Qwen2.5

Post-Training to Achieve Reasoning

- RL is a common tool for post-training LLMs
  - Main example: RLHF
  - Also used in subsequent variants like RLAIF
- **RL also proposed as post-training to induce reasoning in models**
  - E.g. Zelikman et al. (2022) were inspired by CoT
  - More recently, OpenAI (2024) for developing their "o" reasoning models
- High-level **overview of components in RL**
  - An **agent** interacting with an **environment** over sequence of steps T
  - At time step $t \in T$, agent receives **observation** $o \in O$ from environment, sends **action** $a_t \in A$ to environment based on o
  - Environment provides **scalar reward** $y_t = r(a_t)$ where r is **reward function**
  - **Goal in RL:** learn **policy** $p: O \rightarrow A$ that predicts set of actions $a_i \in A$ for $i \in T$ that maximizes rewards over T (in deep RL, r and p are deep networks)
- Key to success: **choice of reward function, challenging** choice
  - E.g. train robot to do scrambled eggs (function for good scrambled eggs?)

Refresher: RLHF (1)

- Christiano et al. 2017 proposed the following:
  - **Learn reward function** ***r:*** *A x O* **-->** $\mathbb{R}$ **that provides rewards aligned with human preferences** while **simultaneously training** a successful **policy**



- Such a method should **allow solving tasks defined by desired behavior**
  - "By desired behavior" --> **agents can be taught by non-experts**
  - E.g. no need to be a chef to provide feedback on scrambled eggs
- This **approach now referred to as RLHF**
  - **InstructGPT**: recent work that used RLHF to *align* LLMs

Refresher: RLHF (2)

- Ouyang et al. 2022 **used RLHF** to **fine-tune GPT-3** to **follow broad class of written instructions** with outputs **aligned with human preference**
- Process broken into **three steps**:
  - **1. Collect examples for several generative tasks, train supervised policy**
    - **Example of task:** "List five ideas for how to regain enthusiasm for my career"
    - **Human labelers** provided **examples of desired outputs**
    - Recall **policy**: function that provides actions given observations
    - In the context of LLMs, actions are generating different text
    - So, **"train policy"** here was **fine-tuning GPT-3 with examples from labelers**
  - **2. Collect comparison data, train reward model to mimic humans**
    - Given tasks and model from Step 1, **generate multiple outputs for each task**
    - **Human labelers choose preferred output** from given pairs of outputs for tasks
    - Recall **reward model**: function that maps actions and observations to reward
    - Reward model: **copy of Step 1 model that produces scalar** (LM head replaced)
  - **3. Optimize policy (Step 1) against reward model (Step 2) using PPO**
    - **Proximal policy optimization (PPO)** (Schulman et al. 2017)**:** RL objective

DeepSeek-R1-Zero (1)

- **RLHF steps 1 and 2** designed to get **reward model that mimics humans**
  - For this, a target signal is required (e.g. from humans or models)
  - These steps often **referred to as supervised fine-tuning (SFT) stage**
  - Getting these labels can be expensive
- For inducing reasoning, **target signal is reasoning tasks**
- The "zero" in DeepSeek-R1-**Zero** comes from skipping the SFT stage
  - Instead, they went directly to the RL objective (Step 3)
  - So, what did they used as reward then?
- Reward models were deterministic out-of-the-box systems
  - **Accuracy rewards:** LeetCode-style compiler for coding tasks, rule-based verifier for math tasks
  - **Format rewards:** enforce "thinking process" between think tags
- **RL Objective**: Group Relative Policy Optimization (GRPO)
  - Yet another innovation from the DeepSeek team
  - Arguably the main innovation

GRPO

- Their RL objective to maximize:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$

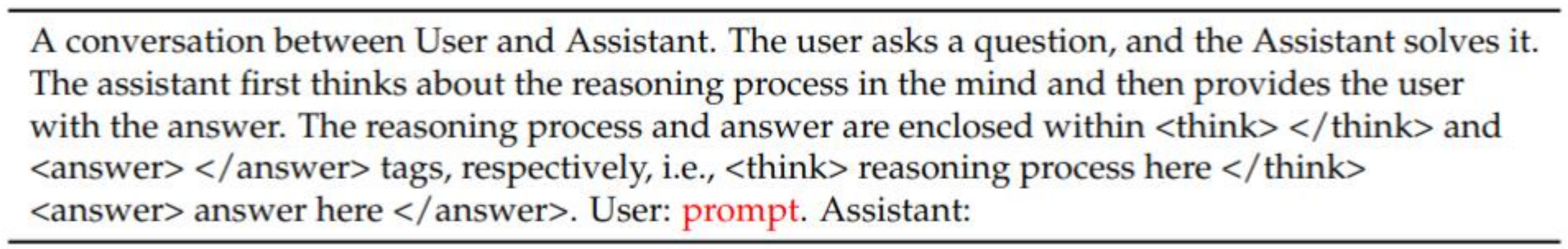
$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right),$$

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

- Roughly:
  - π_θ is policy model ($\pi_{\theta\ old}$ is same model in last iteration)
  - A_i is standardized reward
  - Term normalizing π with $\pi_{\theta\ old}$ is main RL objective
  - Other two terms (clip function, KL divergence) act as regularization
  - Clip prevents large change from last update (epsilon is hyperparameter)
  - KL divergence is between trained model and base model (starting LLM)

DeepSeek-R1-Zero (2)

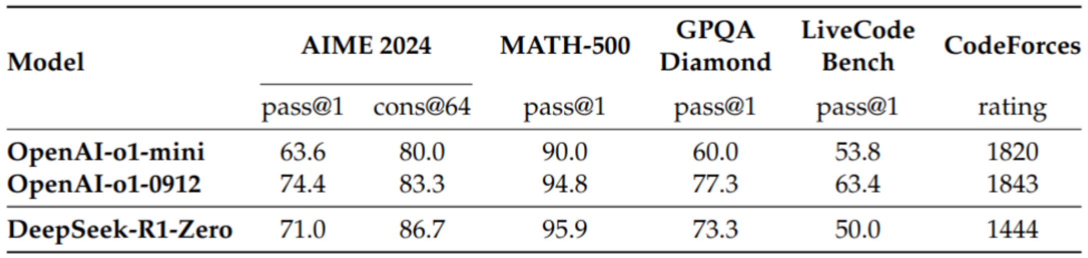
- They used the following training template during this process:



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.

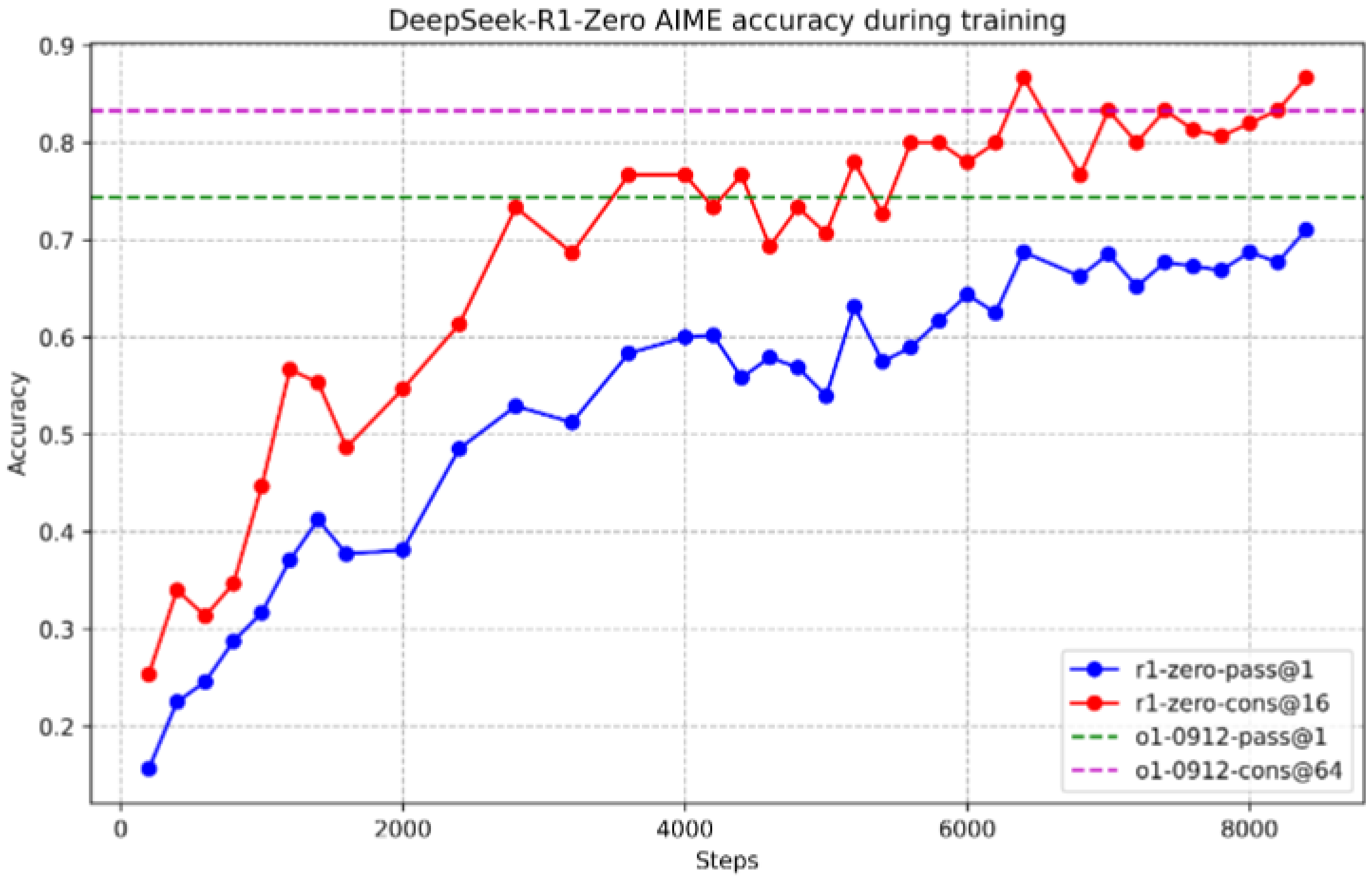
- Note **this is the only way the model was encouraged to think**
  - Typically, target signal used to induce multi-step reasoning
- They compared against two versions of OpenAI's o1 models



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

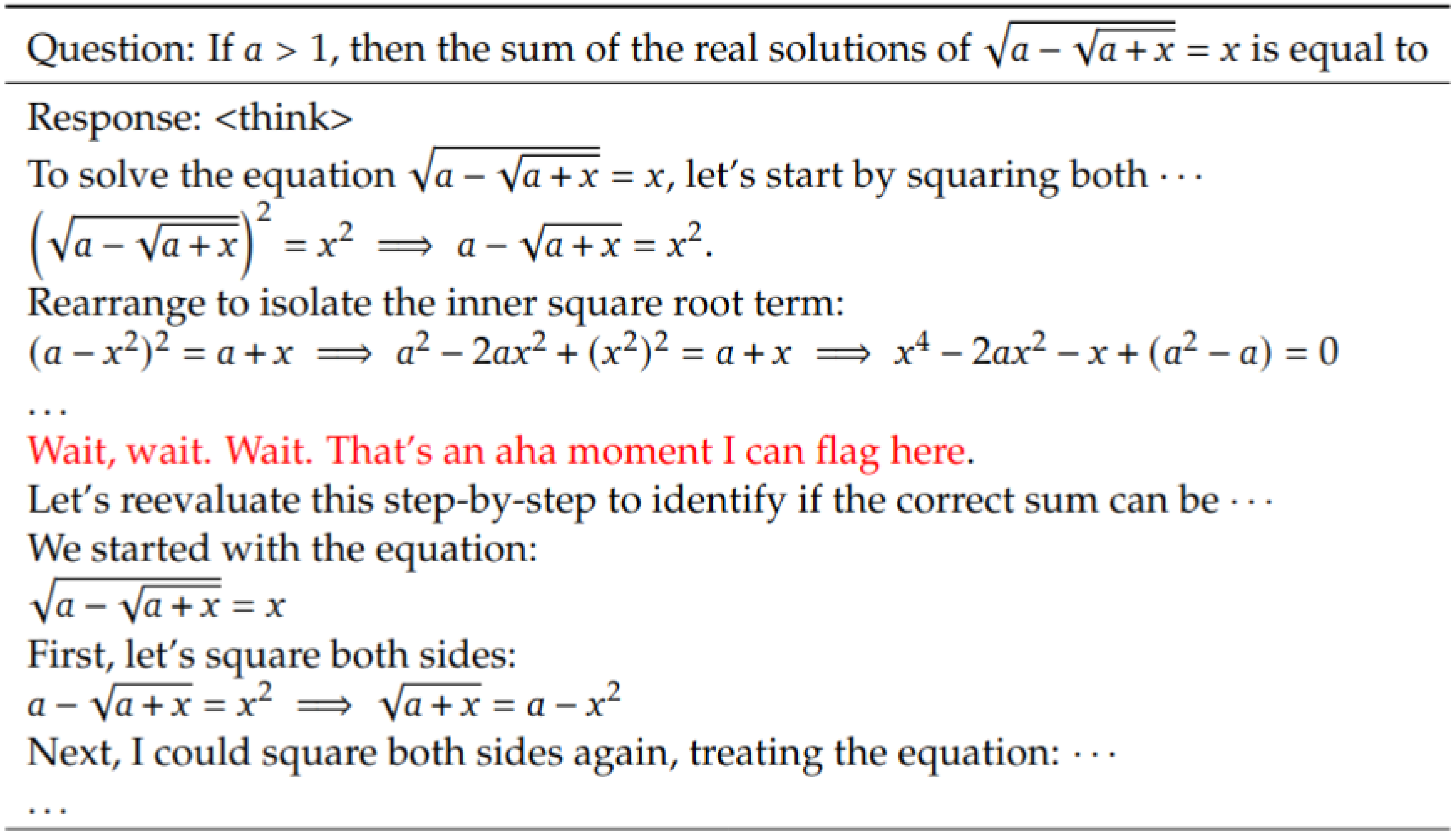
DeepSeek-R1-Zero (3)

- Perhaps more importantly, the process to achieve these results



DeepSeek-R1-Zero (4)

- Reasoning "development" was quite visible at times

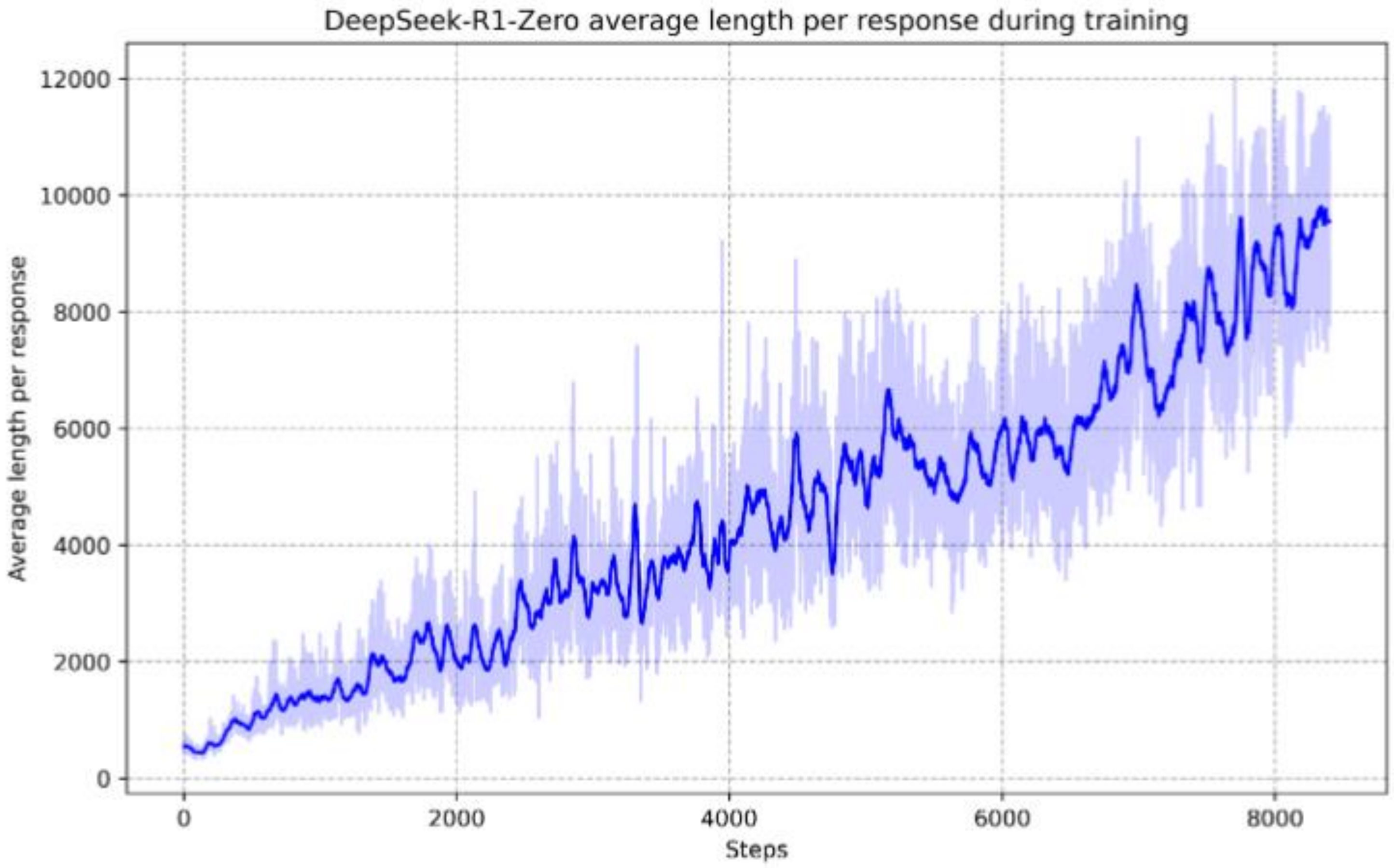


Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a + x}} = x$ is equal to

Response: <think>
To solve the equation $\sqrt{a - \sqrt{a + x}} = x$, let's start by squaring both $\cdots$
$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2.$
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 $\cdots$
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: $\cdots$
...

DeepSeek-R1-Zero (5)

- Their results "naturally" provide support for the computational motivation behind reasoning LLMs

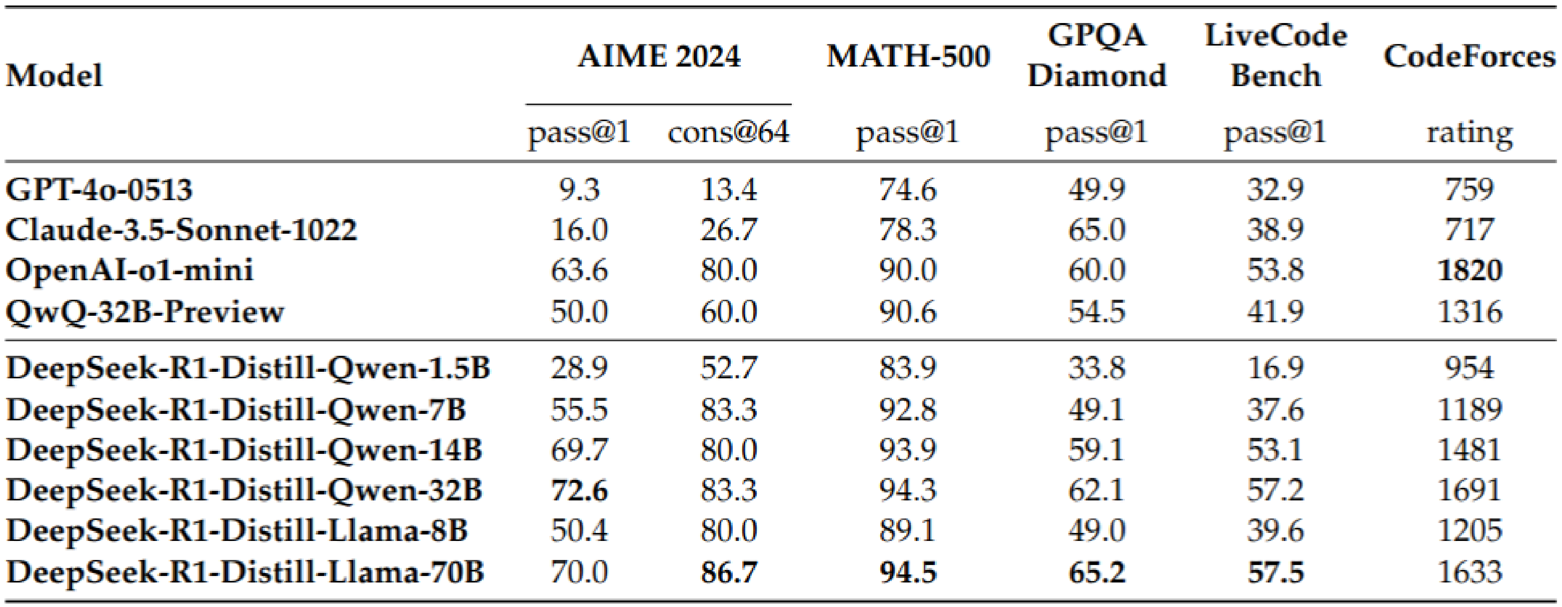


DeepSeek-R1

- DeepSeek-R1-Zero still had issues
  - Poor readability
  - Code switching
- **DeepSeek-R1 goes back to using an SFT signal**
  - R1-Zero model was used as part of this signal
  - But here post-training process is back to normal
- Results were as you would expect from such technical reports
- Arguably more interesting, their distillation models

DeepSeek-R1-Distill

- They used DeekSeek-R1 (671B) to generate a tuning signal
  - **Input:** reasoning task
  - **Output:** what DeepSeek-R1 generated
- They tuned smaller Llama and Qwen versions on this signal



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.

Summary

- **Reasoning LLMs**: producing more tokens at inference time
  - **Language perspective**: explicit reasoning path leads to correct answers more often
  - **Compute perspective**: generating more tokens means more compute resources to reach the correct answer
- Multiple methods for building reasoning LLMs
  - **Post-training** methods, i.e. more training but now on reasoning tasks
  - **Test-time** methods, i.e. only extra computations at inference time
- DeepSeek-R1
  - Recent success of post-training to build reasoning LLM
  - DeepSeek-R1-Zero based on pure RL, no commonly used supervised signal
  - DeepSeek-R1 uses supervision on top of R1
  - DeepSeek-R1-Distill proves strong reasoning LLMs can be built with smaller models using large model as supervision signal

References

- Understanding Reasoning LLMs by Sebastian Raschka
  - February 5th, 2025
- The State of LLM Reasoning Models by Sebastian Raschka
  - March 8th, 2025
- References linked in corresponding slides