

Vinija's AI Notes

[Back to vinija.ai](#)

Primers • DeepSeek R1

- [Introduction](#)
- [Architectural Foundation](#)
 - [Multi-Token Prediction](#)
- [Training Pipeline: from Pre-Training to Reasoning](#)
 - [Stage 1: Cold Start with Supervised Fine-Tuning \(SFT\)](#)
 - [Stage 2: Reinforcement Learning \(RL\)](#)
 - [Rewards](#)
- [Group Relative Policy Optimization \(GRPO\)](#)
 - [How GRPO Works](#)
 - [Step-by-Step Breakdown](#)
 - [GRPO vs. PPO vs. DPO](#)
 - [Tabular Comparison](#)
- [Emergent Reasoning Behaviors](#)
- [Distillation: Reasoning in Compact Models](#)
- [Results](#)
 - [References](#)

Introduction

- DeepSeek-R1 represents a landmark in reasoning-capable large language models (LLMs). Released under an MIT license, this model rivals closed-source giants like OpenAI's o1 series while pioneering a reinforcement learning (RL)-driven framework for reasoning tasks. It introduces Group Relative Policy Optimization (GRPO), which replaces traditional methods like PPO, making training both efficient and scalable. Furthermore, DeepSeek-R1 demonstrates how reasoning capabilities emerge naturally through RL without relying on massive supervised fine-tuning (SFT).
- DeepSeek-R1 redefines open-source AI, proving that reasoning capabilities can emerge from RL alone. Through innovations like GRPO, FP8 quantization, and emergent CoT reasoning, it rivals closed-source models while fostering transparency and accessibility. As the research community builds upon these innovations, DeepSeek-R1 signals a shift towards efficient, reasoning-driven AI accessible to all.
- This primer explores its architecture, multi-stage training pipeline, GRPO mechanics, and emergent reasoning behaviors, alongside how distillation propagates reasoning capabilities to smaller models.

Architectural Foundation



- DeepSeek-R1 builds upon the DeepSeek-V3-Base model, leveraging architectural innovations to optimize training and inference. Its technical foundation integrates:

1. Mixture of Experts (MoE):

- In MoE, each Transformer block activates only a subset ($1/k$) of parameters, significantly reducing computational costs. For DeepSeek's 600B-parameter model, only ~37B parameters are activated per token during inference, achieving 80% computational savings.

2. Multihead Latent Attention (MLA):

- MLA reduces inference latency by projecting the Key-Query-Value (KQV) matrices into a latent space of lower dimensions. This decreases attention complexity and enables the reuse of these projections across tokens during inference.

3. FP8 Quantization:

- DeepSeek-R1 employs 8-bit floating-point quantization (FP8), reducing memory usage by 75% compared to FP32 while maintaining numerical stability. This innovation allows training to scale efficiently on 2048 H800 GPUs at a fraction of the cost traditionally required for models of this scale.

4. Multi-Token Prediction (MTP):

- Unlike traditional autoregressive decoding, which generates one token at a time, MTP predicts multiple tokens simultaneously. This improves inference efficiency, particularly for long-context reasoning tasks.

Multi-Token Prediction

- DeepSeek-R1's Multi-Token Prediction (MTP) is a crucial innovation aimed at enhancing the efficiency of both training and inference. While most traditional LLMs generate one token at a time in an autoregressive manner, MTP allows DeepSeek-R1 to predict **multiple tokens per step**. Here's how it works and why it's impactful:

- **How Multi-Token Prediction Works:**

1. Parallel Decoding of Tokens:

- Instead of producing a single token at every decoding step, DeepSeek-R1's architecture predicts a sequence of tokens in parallel.
- The model uses an autoregressive base but extends it by parallelizing token prediction within the same context window. For example, instead of decoding "The cat is" one token at a time, it might predict the next 2-3 tokens ("The cat is") simultaneously.

2. Token Sampling and Re-ranking:

- The multi-token outputs are often sampled from a probabilistic distribution of the next tokens given the context.
- This requires the model to maintain high contextual awareness since predicting multiple tokens increases the chances of compounding errors. DeepSeek-R1 mitigates this by re-ranking outputs to ensure coherence and consistency across the token sequence.

3. Prediction Horizon:

- A dynamic prediction horizon is employed, meaning that the number of tokens generated simultaneously may vary depending on the confidence of the model. For example, simpler or deterministic tasks (e.g., generating code snippets or completing structured sentences) might involve longer multi-token predictions, while ambiguous tasks might still rely on single-token predictions.

4. Intermediate Attention Updates:

- DeepSeek-R1's Multihead Latent Attention (MLA) plays a significant role here. MLA projects the KQV (Key, Query, Value) attention mechanism into a lower-dimensional latent space. This enables efficient caching and reusing of intermediate computations, which is critical when predicting multiple tokens at once without recalculating all attention matrices.

5. Decoding Optimization with MTP:

- The mixture-of-experts (MoE) architecture further supports MTP by assigning specialized expert layers to handle different segments of token prediction. This specialization ensures that errors in one part of the sequence do not propagate to others.

◦ Why Multi-Token Prediction is Important:

1. Efficiency Gains:

- Predicting multiple tokens reduces the number of decoding steps required to generate a full output, leading to significant speedups, especially for long-text or multi-turn interactions.

2. Longer-Context Generation:

- DeepSeek-R1's MTP excels in reasoning tasks requiring long chains of thought (CoT). By generating multiple reasoning steps simultaneously, it reduces the cognitive overhead of managing intermediate outputs.

3. Emergent Self-Correction:

- The model's reasoning architecture incorporates feedback loops (like reflection and self-correction) that allow it to detect and adjust errors within multi-token outputs.

4. Improved Coherence:

- By treating multiple tokens as a single prediction unit, the model can better maintain linguistic and semantic coherence across generated sequences. This is particularly valuable for structured tasks like coding, mathematical reasoning, or document generation.

◦ Challenges Addressed:

1. Error Propagation:

- One of the primary concerns with MTP is that predicting multiple tokens increases the risk of cascading errors. DeepSeek-R1 counters this with advanced reward mechanisms during reinforcement learning, emphasizing accuracy and consistency within multi-token outputs.

2. Scaling Attention Mechanisms:

- Predicting multiple tokens demands more efficient attention mechanisms to handle larger contexts. DeepSeek-R1's MLA reduces computational overhead and memory bottlenecks, ensuring scalability.

3. Balancing Token Granularity:

- MTP dynamically adapts to the complexity of tasks. For example, in reasoning tasks with deterministic steps (like arithmetic), longer token predictions are feasible. For creative tasks, shorter predictions are favored to maintain quality.

◦ Key Applications of MTP in DeepSeek-R1:

1. Reasoning Chains:

- MTP enables the model to generate multiple reasoning steps at once, ensuring fluid and logical progression in math proofs, coding solutions, and logical arguments.

2. Code Generation:

- In tasks like Python scripting or SQL generation, MTP allows DeepSeek-R1 to output complete function definitions or query blocks, significantly speeding up iterative development.

3. Document Completion:

- For long-form writing or multi-paragraph completions, MTP ensures continuity and thematic consistency across sentences or paragraphs.

Training Pipeline: from Pre-Training to Reasoning

- DeepSeek-R1 employs a multi-stage pipeline meticulously designed to maximize its reasoning capabilities while minimizing computational costs. This process consists of distinct stages, each guided by task-specific loss functions and reward mechanisms.

Stage 1: Cold Start with Supervised Fine-Tuning (SFT)

- DeepSeek-R1 begins its journey by fine-tuning the V3-Base model with high-quality Chain-of-Thought (CoT) examples. These examples are carefully curated using few-shot prompting, manual annotation, and refinement of DeepSeek-R1-Zero outputs.
- **Comparison to Cold Start in Recommender Systems:**
 - In recommender systems, the “cold start problem” refers to the challenge of providing accurate recommendations for new users or items with limited historical data. The focus is on mitigating data sparsity by learning user preferences or item properties.
 - In contrast, DeepSeek-R1’s cold start addresses the challenge of initializing a large language model with structured reasoning and readability. By fine-tuning on curated data, the model develops a foundational understanding of chain-of-thought reasoning, overcoming instability observed in RL-only training setups.
- **Advantages of Cold Start:**
 - **Readability:** DeepSeek-R1-Zero struggled with poor readability and language mixing. In contrast, the cold-start phase imposes a structured output format:

```
<reasoning_process> CoT explanation </reasoning_process>
<summary> Final Answer </summary>
```

- **Alignment:** Cold start data introduces **human priors**, accelerating convergence and improving performance on reasoning-intensive tasks.
- DeepSeek-R1 begins its journey by fine-tuning the V3-Base model with high-quality **Chain-of-Thought (CoT)** examples. These examples are carefully curated using **few-shot prompting**, manual annotation, and refinement of DeepSeek-R1-Zero outputs.
- **Loss Function for SFT:**
 - The model is fine-tuned using a supervised cross-entropy loss:

$$L_{\text{SFT}} = - \sum_{i=1}^n \log P_{\theta}(o_i | q, \{o_1, \dots, o_{i-1}\}),$$

- where:

- o_i : the i -th token in the output sequence,
- q : the input query,

- o_1, \dots, o_{i-1} : previously generated tokens.

Stage 2: Reinforcement Learning (RL)

- Reinforcement learning is the backbone of DeepSeek-R1's reasoning evolution. The model learns from rewards rather than curated datasets, enabling self-improvement over thousands of iterations.

Rewards

- DeepSeek-R1 uses two primary reward functions:
 1. **Accuracy Rewards:**
 - Evaluate the correctness of deterministic tasks, such as math problems and code-generation outputs. For example:
 - In math, the model's final answer is verified against a ground truth.
 - For coding, unit tests evaluate the validity of generated code solutions.
 2. **Format Rewards:**
 - Encourage consistent reasoning structures by rewarding outputs that adhere to the specified CoT format. For example:

```
<reasoning_process> Step-by-step explanation </reasoning_process>
<answer> Final Output </answer>
```

Group Relative Policy Optimization (GRPO)

- Group Relative Policy Optimization (GRPO), introduced in [DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models](#), is the centerpiece of DeepSeek-R1's RL framework, introducing a simplified and efficient alternative to traditional policy optimization methods like PPO and DPO. Here, we explore how GRPO works, why it differs, and its key advantages over other approaches.

How GRPO Works

- The GRPO objective is:

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

Step-by-Step Breakdown

1. Likelihood Ratio ρ_i :

- The likelihood ratio measures how much more likely the new policy π_{θ} is to produce an output o_i compared to the old policy $\pi_{\theta_{\text{old}}}$: $\rho_i = \frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)}$.

2. Advantage Function A_i :

- The advantage function evaluates how much better an output o_i is compared to the average outputs in the group: $A_i = \frac{r_i - \text{mean}(r_1, \dots, r_G)}{\text{std}(r_1, \dots, r_G)}$,

where r_i is the reward associated with the i -th output.

3. Clipping Mechanism:

- Clipping ensures that the policy updates remain stable by restricting the likelihood ratio within the range $[1 - \epsilon, 1 + \epsilon]$: $\text{clip}(\rho_i, 1 - \epsilon, 1 + \epsilon)$.
- This prevents overly aggressive updates that could destabilize training.

4. KL Divergence Penalty:

- The KL divergence term ensures the new policy remains close to a reference policy:

$$-\beta D_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{ref}})$$

- where, β controls how much the KL term influences the overall objective.

GRPO vs. PPO vs. DPO

- Here is a comparison of Group Relative Policy Optimization (GRPO), Proximal Policy Optimization (PPO), and Direct Preference Optimization (DPO):

1. Proximal Policy Optimization (PPO):

- **Critic Model Usage:** PPO relies on an actor-critic architecture where a critic model estimates the baseline for reward computation, which increases memory and computational requirements.
- **Stabilization Techniques:** PPO uses a clipping mechanism in the optimization process to ensure stable policy updates.
- **Complexity:** The requirement of training both the policy and value models makes PPO resource-intensive and computationally heavier.

2. Direct Preference Optimization (DPO):

- **Human Feedback Integration:** DPO optimizes policies based on pairwise preferences from human feedback, which are directly incorporated into the model.
- **Simplicity:** Unlike PPO, DPO does not require a separate value model, which simplifies training.
- **Challenges:** The reliance on human feedback can introduce inconsistencies and biases into the reward model, affecting the robustness of the training process.

3. Group Relative Policy Optimization (GRPO):

- **Critic-Free Design:** GRPO eliminates the need for a critic model by using group-level rewards for baseline estimation, drastically reducing memory and computational overhead.
- **Efficiency:** GRPO uses group comparisons of outputs to compute relative rewards, simplifying the reinforcement learning process while maintaining strong performance.
- **Emergent Behaviors:** The algorithm encourages emergent behaviors such as self-reflection, potentially enhancing general reasoning and task diversity during training.
- **Versatility:** GRPO shows significant improvements in both in-domain and out-of-domain tasks, particularly for models focused on mathematical reasoning.

Tabular Comparison



Feature/Aspect	PPO	DPO	GRPO
Critic Model Requirement	Yes	No	No
Reward Source	Estimated baseline from value model	Human feedback with pairwise preferences	Group-level rewards via relative comparisons
Memory Efficiency	High memory usage	Moderate memory usage	Low memory usage
Computational Overhead	High (training policy and value models)	Moderate	Low
Human Feedback Dependency	Indirect (through value model and optimization)	Direct	None
Emergent Behavior	Not explicitly encouraged	Limited	Encouraged during training
Task Adaptability	Requires adjustments for specific tasks	Moderate adaptability	High adaptability across multi-domain tasks
Strengths	Stable policy updates, robust optimization	Simpler optimization process	Highly efficient, encourages emergent behavior
Weaknesses	Resource-intensive	Susceptible to biases in human feedback	Dependent on effective group comparison metrics

Emergent Reasoning Behaviors

- During training, DeepSeek-R1 developed remarkable reasoning patterns:
 - Reflection:** Revisiting and revising intermediate steps.
 - Self-Correction:** Identifying and fixing errors in real-time.
 - Aha Moments:** Pausing and reevaluating to discover new solutions.
- For example:
 - Solving $x^2 - 5x + 6 = 0$, the model might initially propose incorrect factors, pause to reflect, and ultimately derive $x = 2$ and $x = 3$.
 - The table below from the original paper, we can see where R1 has it's “aha” moment and re-evaluates the solution:

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

Table 3 | An interesting “aha moment” of an intermediate version of DeepSeek-R1-Zero. The model learns to rethink using an anthropomorphic tone. This is also an aha moment for us, allowing us to witness the power and beauty of reinforcement learning.

- DeepSeek-R1's reasoning capabilities were distilled into smaller models (e.g., Qwen-7B, Llama-8B), achieving state-of-the-art performance:
 - **Teacher-Student Paradigm:** Outputs from DeepSeek-R1 trained smaller models with minimal computational overhead.
 - **Efficiency:** Distilled models retained reasoning capabilities while outperforming larger, non-reasoning models like GPT-4o.
- The table below shows how distilled R1 models and how they compare on reasoning related benchmarks.

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.

Results

- The figure below from the [original paper](#), shows the performance of DeepSeek R1 being at par with or outperforming OpenAI's models at several benchmarks.

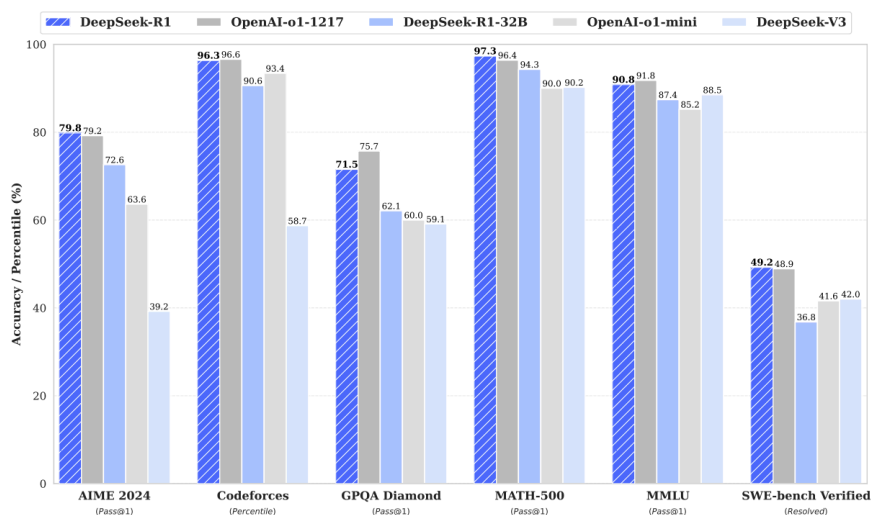


Figure 1 | Benchmark performance of DeepSeek-R1.

References

- [DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning](#)
- [DeepSeek-R1: A Pure RL-based Reasoning Model](#)



www.vinija.ai