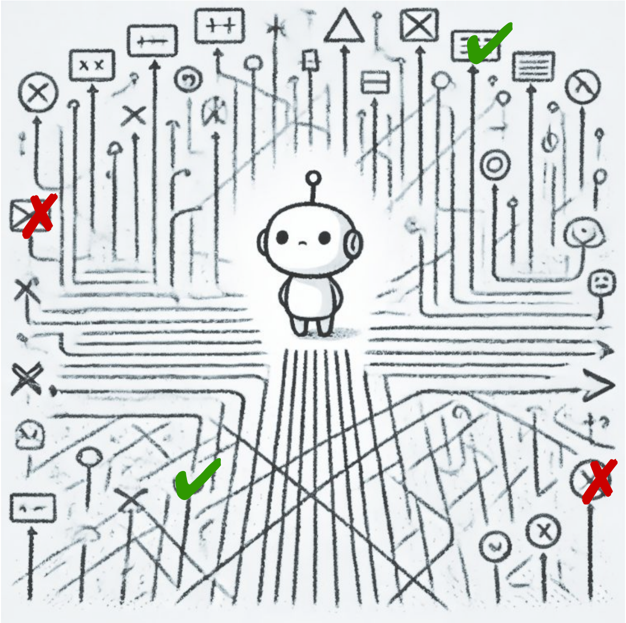
Reinforcement Learning:
How Models Learn to Reason

Abbie Petulante
DSI Postdoctoral Fellow

VANDERBILT UNIVERSITY®
Data Science Institute
Discovery through data.

Reinforcement Learning (RL)

- A **trial-and-error** learning process
  - an **agent**
  - interacts with an **environment**,
  - takes **actions**,
  - receives **rewards** for positive actions

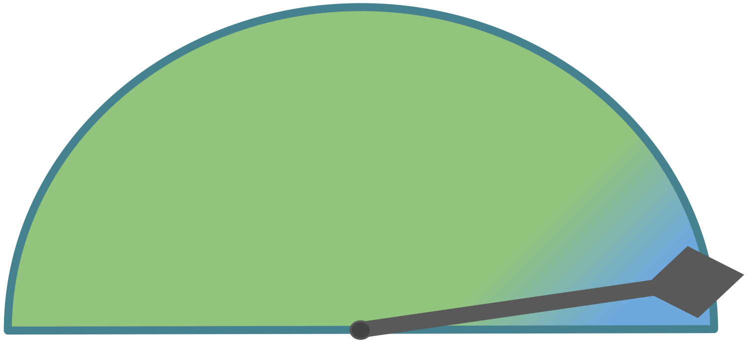


- used when we don't have **direct supervision** but *can* define a goal

Why in focus now?

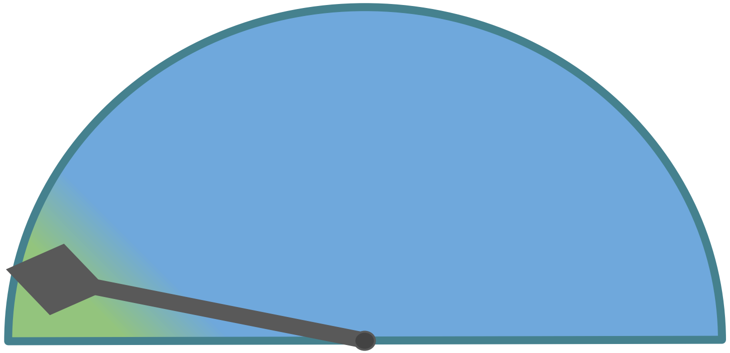
- **Because of reasoning models!**

Supervised Learning



Standard (SFT + RLHF at end)

Reinforcement Learning



Typical primary RL

Reasoning Responses

- *<think>*
  - Here is some reasoning through
  - Thoughts, step-by-steps
- *</think>*
- *<answer>*
  - Here is my final answer
  - Usually a summary of what was thought through
- *</answer>*

Why RL for Reasoning

- *<think>*
  - Here is some reasoning through
  - Thoughts, step-by-steps
- *</think>*

Hard to quantify as good, right, correct

- *<answer>*
  - Here is my final answer
  - Usually a summary of what was thought through
- *</answer>*

Might be easier to quantify, right vs wrong

An RL Crash Course

The Reward Function

- Rewards are a way of evaluating **goodness**
  - **“Goodness” ≠ “correctness”** *necessarily*
- **You might reward:**
  - Efficiency ➔ reach the answer in the few words
  - Coherence & Fluency ➔ for long dialogues
  - Preference & Style ➔ kindness, helpfulness, etc

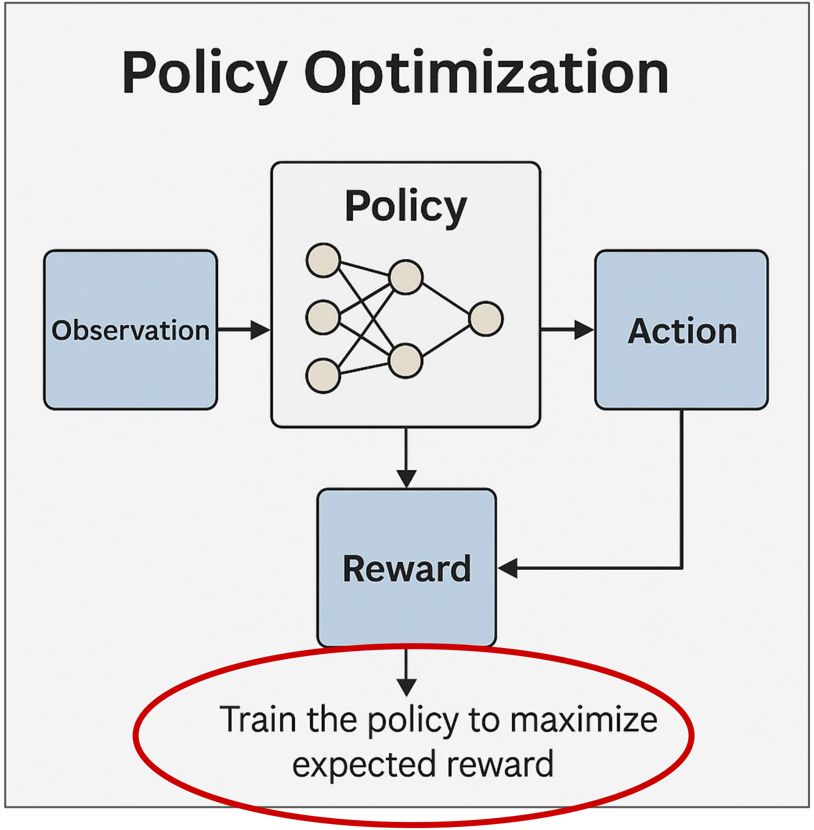
Policy Optimization

- A “policy” is a strategy to decide what action to take in a given situation
  - For LLMs, *the model* - it’s current weights and state!

- **Policy Optimization** refers to *how* you update the model to *get more rewards* next time.

Policy Optimization

- Policy optimization strategies ask:
  - How do I determine what responses are better?
  - How do I change my model to make more of those?
- Differ in how model updates are chosen and implemented



GRPO

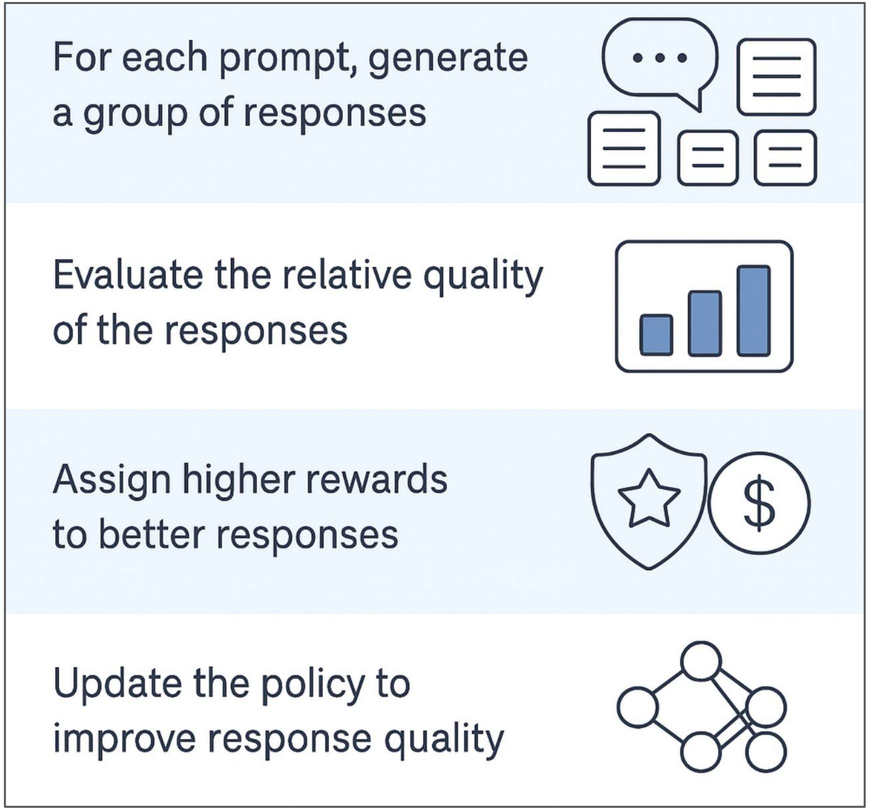
- **GRPO** (Group Relative Policy Optimization) is a policy optimization strategy designed for reasoning

Key Features:

- Compares groups of possible responses
- Encourages the model to prefer responses that are better than others *in the group*

GRPO for Reasoning

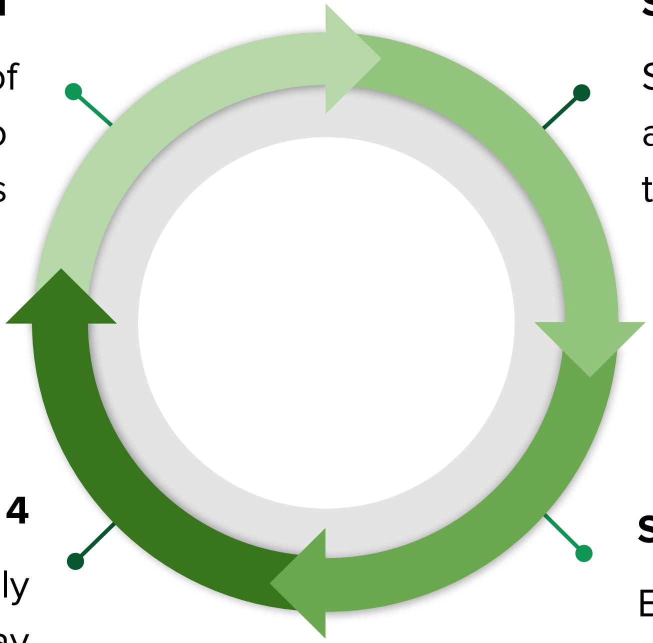
- By focusing on relative performance within groups, GRPO helps models develop better reasoning strategies
  - Comparison of different "approaches" to a problem
  - Aligns with how humans favor responses (seeing many options)



The RL Training Loop

Step 1

Take the current version of the model. You want to see how good it is



Step 2

Sample across many different actions that the model is likely to take

Step 3

Evaluate the rewards that all of those potential paths would receive

Step 4

Based on all rewards of all likely paths, move the model in a way that gives *more rewards*

Imagine You’re an LLM…

“Traditional” LLM Learning

- The model is trained to predict what comes next:

“I want to go to ______”

“Traditional” LLM Learning

- The model is trained to predict what comes next:

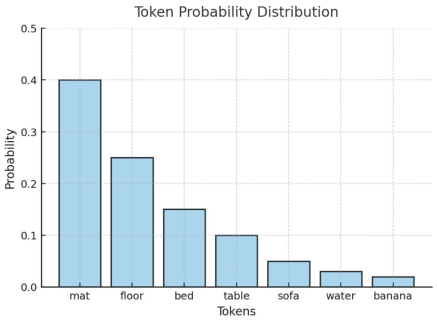
“I want to go to ______”

bed?

sleep?

the?

see?



Probably, pretty high probabilities for all

RL

- The model is trained to predict what comes next:

“I want to go to ______”

bed?

sleep?

the?

see?

All of these choices are *reasonable*…but which are *better*

Which do people prefer?

RL

- The model is trained to predict what comes next:

"I want to go to ______"

~~bed.~~

~~sleep.~~

The movies later today because Ghostbusters just came out and I'm so excited to see it.

See the cherry blossoms bloom, I've always thought that must be the most marvelous sight.

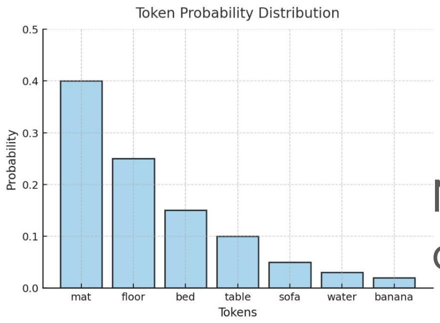
RL

- The model is trained to predict what comes next:

"I want to go to ______"

~~bed.~~

~~sleep.~~



Not wrong at any step, just not optimized for our preference of long answers

RL for Reasoning

Next-token self-supervised learning:

$$2x + 3 = 9$$

$$x + \_\_\_$$

RL for Reasoning

Next-token self-supervised learning:

$$2x + 3 = 9$$
$$x + \_\_\_\_$$

You have access to the original equation

Fill onward just based on memorization, not on solving and working backwards

Uses what you've memorized, what typically follows x=?

Evaluate on whether the blank is filled exactly how was expected

RL for Reasoning

Next-token self-supervised learning:

$$\begin{aligned} 2x + 3 &= 9 \\ x + &\underline{\qquad} \end{aligned}$$

Reinforcement Learning:

$$\begin{aligned} 2x + 3 &= 9 \\ x + 3 &= 9 \\ x &= 6 \end{aligned}$$

$$\begin{aligned} 2x + 3 &= 9 \\ 2x &= 6 \\ x &= 3 \end{aligned}$$

RL for Reasoning

Next-token self-supervised learning:

$$
\begin{aligned}
2x + 3 &= 9 \\
x + {} & \underline{\qquad}
\end{aligned}
$$

Reinforcement Learning:

$$
\begin{aligned}
2x + 3 &= 9 \\
x + 3 &= 9 \\
x &= 6 \quad ✗
\end{aligned}
\qquad\qquad
\begin{aligned}
2x + 3 &= 9 \\
2x &= 6 \\
x &= 3 \quad ✔
\end{aligned}
$$

You can see both potential responses in their entirety

And evaluate both separately and on multiple criteria

Are they correct? Are they logical? Are they concise?

RL for Reasoning

Next-token self-supervised learning:

$$
\begin{aligned}
2x + 3 &= 9 \\
x + &\underline{\quad\quad}
\end{aligned}
$$

Reinforcement Learning:

$$
\begin{aligned}
2x + 3 &= 9 \\
x + 3 &= 9 \quad ✗ \\
x &= 6
\end{aligned}
$$

$$
\begin{aligned}
2x + 3 &= 9 \\
✓\ 2x &= 6 \\
x &= 3
\end{aligned}
$$

Per-token rewards give us more info and help avoid bad paths in the future

An RL Dataset

Unlike vast web-scale datasets for next-token prediction, RL-trained models:

- Generate their own outputs and **learn by self-evaluating**
- Use a ***reward model*** to score the outputs

The reward model is often optimizing, from the start:

- Correctness, efficiency, logical coherence, step-by-step accuracy

Traditional vs. RL

	Traditional Supervised Learning	RL for Reasoning
Learning Task	Next-token prediction	Optimizing reasoning paths
Optimization	Minimize token loss (cross-entropy)	Maximize total reward over reasoning steps
Training Data	Fixed datasets of human-written text	Self-generated reasoning sequences
Weaknesses	Struggles with multi-step logic	Requires complex reward models
Key Benefit	Fluent, general-purpose text generation	Structured, efficient reasoning

The Drawbacks

- **Challenges with using primary RL**

    - **Exploration problems** – Models must try inefficient reasoning paths before finding the best.
    - **Computational cost** – must simulate many responses.
    - **Sparse rewards** can delay feedback (dense rewards solve)
    - **Reward hacking** – poorly designed rewards have exploitable shortcuts

A Brief History of RL

1950's

Early RL concepts

Rooted in behaviorist psychology (reward-driven learning)

1989

Q-Learning

A foundational RL algorithm introduced by Chris Watkins popularizes the field

1990's

RL in Practice

Fueled by this, it becomes commonplace in robotics, control systems, and game-playing AI

2013-2016

RL Breakthroughs

Deep reinforcement learning rapidly expands capabilities (Deep Q-Networks, AlphaGo)

A Brief History of RL for LLM's

2020

OpenAI Releases GPT3

It's impressive, but **lacks fine-grained control and alignment**, leading to unpredictable responses.

Jan 2022

RLHF Revolutionizes Alignment

OpenAI releases InstructGPT, which uses RL ***from human feedback,*** significantly improving response helpfulness and safety

Nov 2022

OpenAI Releases ChatGPT

Aligned with RLHF, the model is broadly useful and *changes the world*

2023

RL Use Expands

RL is no longer just about safety but actively **improving logical and task-specific performance**.

Why in focus now?

- Until recently, RL was an “accessory” to LLM training
    - Supervised learning teaches language
    - Instruction fine-tuning teaches directions
    - RLHF teaches the model to “be nice”
- RL was mainly used for **alignment**, **user satisfaction**, **safety** as RLHF
- Now, used as a first-step in training, to teach **reasoning, abstract thought**

2024: A Paradigm Shift

- **DeepSeek R1 (Late 2024)**: *trained* using **primarily RL** rather than supervised fine-tuning.

Supervised Learning

Core Mechanism: Next-token prediction → memorizing patterns of massive datasets

Lacks global planning—each token is predicted **locally**, without assessing the long-term impact of decisions.

Reinforcement Learning

Core Mechanism: Reward-guided optimization → learning from outcomes

Evaluates sequences of actions holistically, refining its reasoning rather than just token probabilities.