Reinforcement Learning in LLMS

GRPO — Deepseek-R1

[*] Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." arXiv preprint arXiv:2501.12948 (2025).

InstructGPT

- Making language models bigger does not inherently make them better at following a user's intent.
- Guard-rails —
  - Outputs can be untruthful, toxic, or simply not helpful to the user.
- Fine-tune GPT-3 using supervised learning.
  - Aka, from human feedback.

[*] Ouyang, Long, et al. "Training language models to follow instructions with human feedback." Advances in neural information processing systems 35 (2022): 27730-27744.

InstructGPT

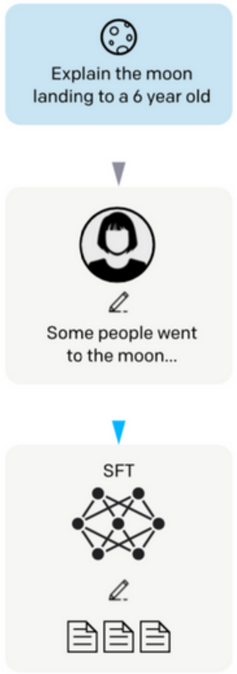
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



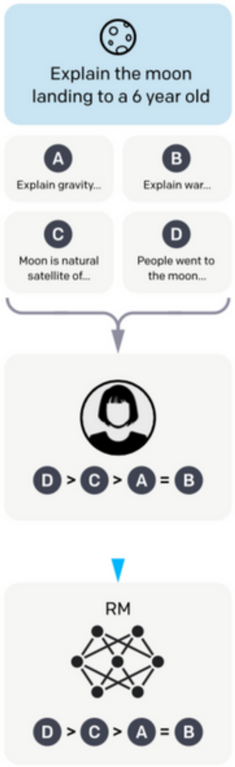
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

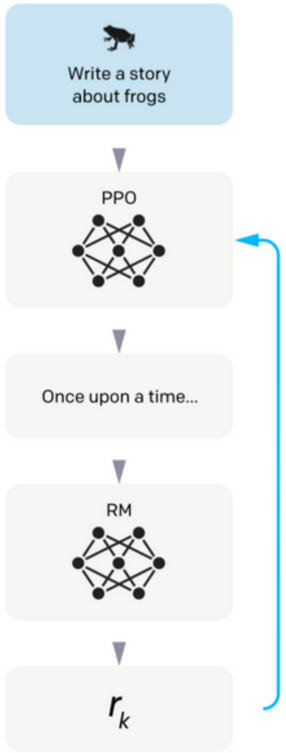
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



InstructGPT

Supervised Fine Tuning (SFT)

```
{
    "prompt": "write me a song about an ox plowing a field of data",
    "response": "Surely! Here is a song about an ox plowing a field of data:\nOx go
plow..."
}
```

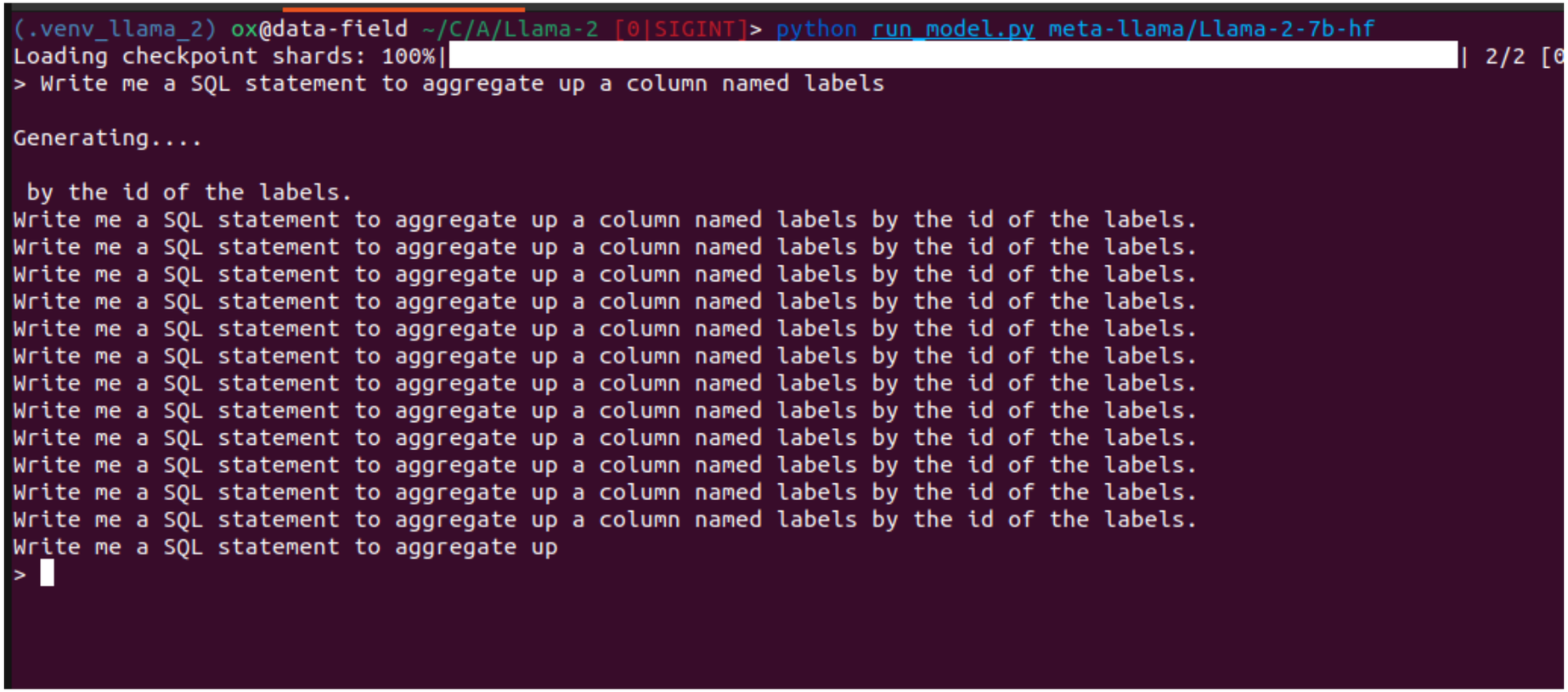
Reward Model (RM)

```
{
    "prompt": "write me a song about an ox plowing a field of data",
    "responses": [
        {
            "text": "Ox go plow",
            "ranking": 3
        },
        {
             "text": "You got it! Ox go plow, make no sound, data so wow.",
             "ranking": 1
        },
        {
            "text": "Sure! Ox go plow",
            "ranking": 2
        }
    ]
}
```

Reinforcement Learning - Proximal Policy Optimization (PPO)

```
{
    "prompt": "write me a song about an ox"
    "response": "You got it! Ox go plow, make no sound, data so wow.",
    "expected_reward": 1.0,
    "predicted_reward": 0.8,
}
```

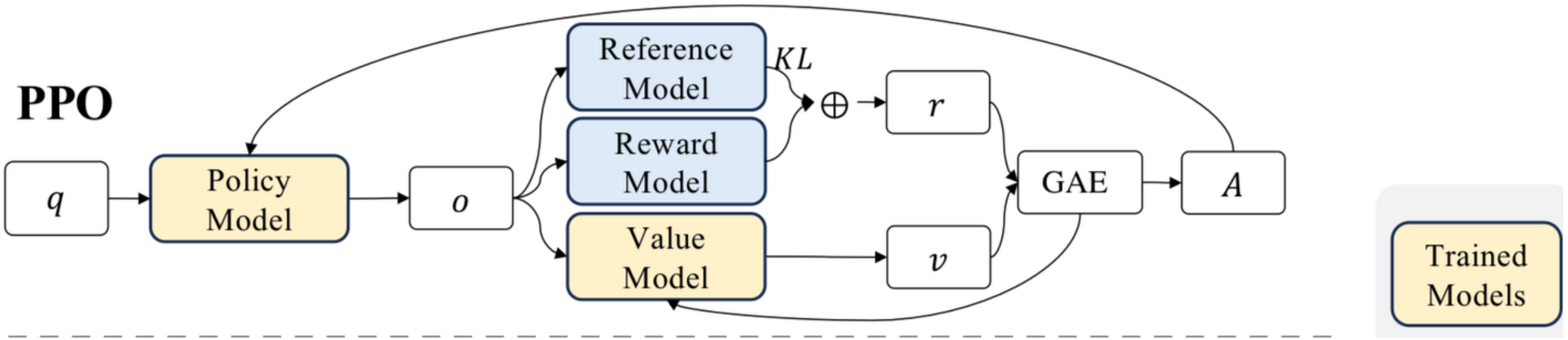
InstructGPT



InstructGPT

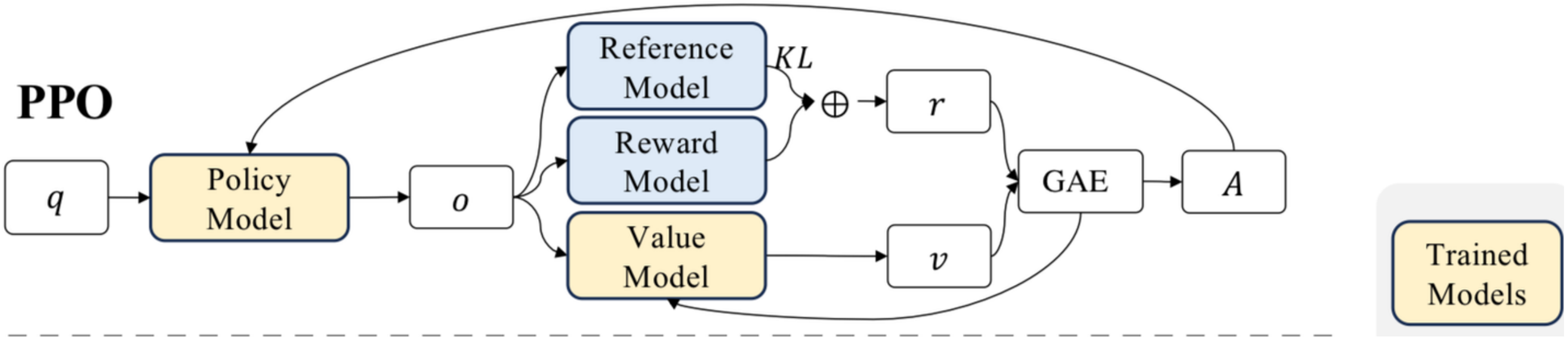
```
(.venv_llama_2) ox@data-field ~/C/A/Llama-2 [0|SIGINT]> python run_model.py meta-llama/Llama-2-7b-chat-hf
Loading checkpoint shards: 100%|████████████████████████████████████████████| 2/2 [00:04<00:00,  2.42s/it]
> Write me a SQL statement to aggregate up a column named labels

Generating....

 and count the number of unique labels.

For example, if the table has the following data:

| id | labels |
| --- | --- |
| 1  | apple |
| 1  | banana |
| 2  | orange |
| 3  | apple |
| 3  | orange |

The SQL statement should return:

| count |
| --- |
| 2 |

The count of unique labels is 2, because there are 2 unique labels: "apple" and "orange".

Please let me know if you need more information.
>
```

InstructGPT



The Problem: PPO can be pretty expensive to train.

Optimize



Memory Usage in PPO:

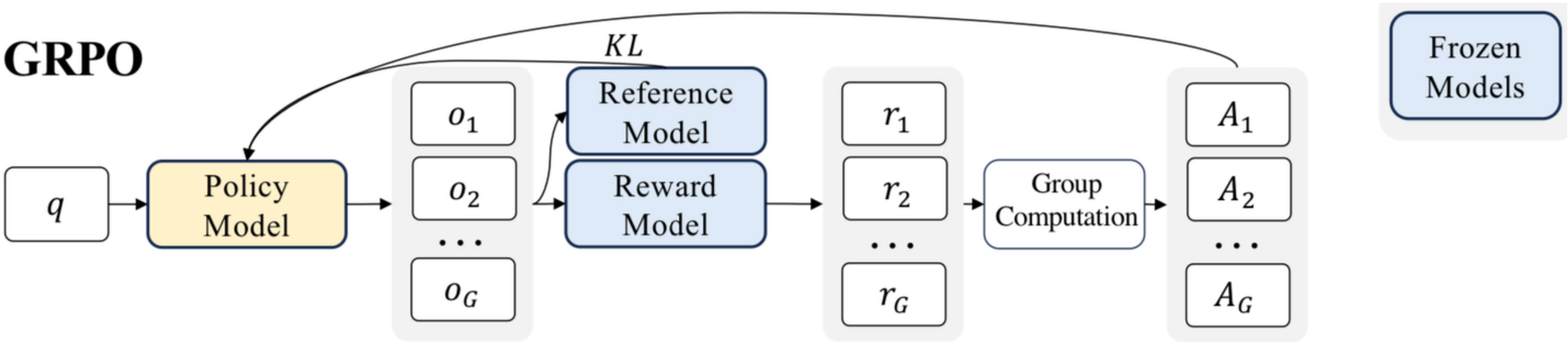
PPO requires significant memory due to backpropagation through both policy and value model parameters.

Optimize

DROP THE VALUE MODEL

IT’S CLEANER

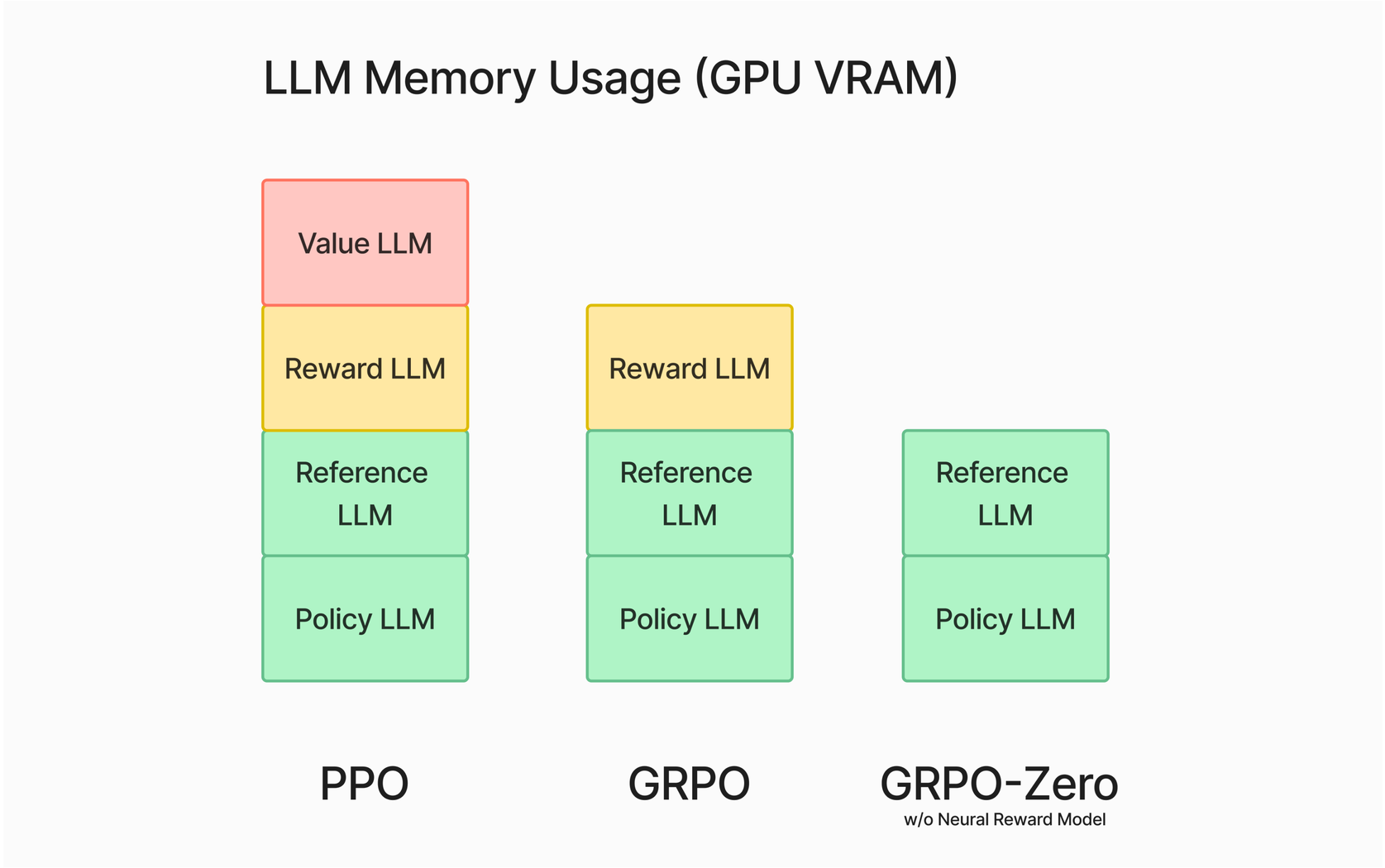
Optimize



Memory Usage in GRPO:

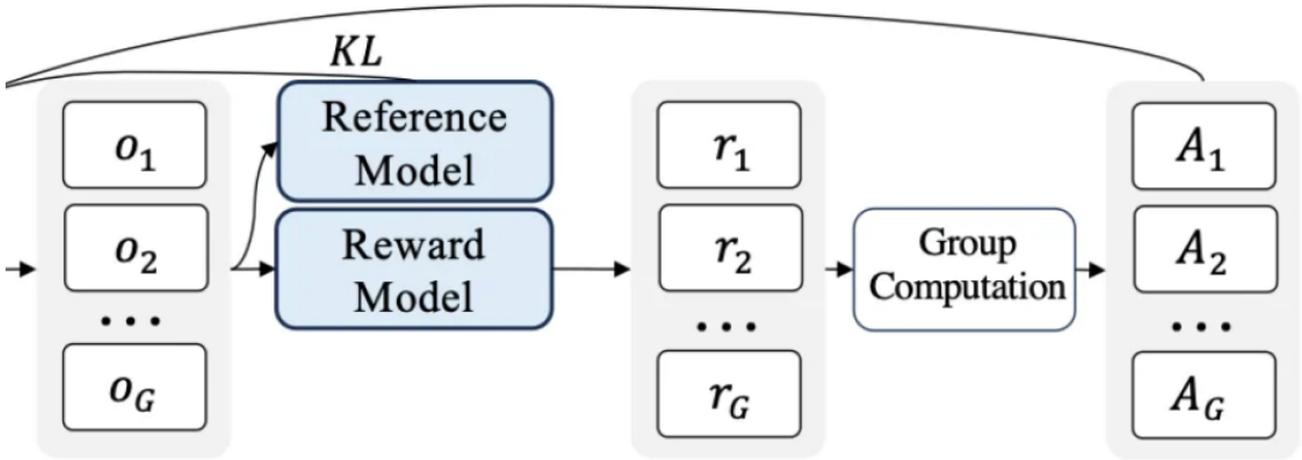
GRPO eliminates the value model, reducing memory usage compared to PPO.

Optimize


LLM Memory Usage (GPU VRAM)
Value LLM
Reward LLM
Reference LLM
Policy LLM
Reward LLM
Reference LLM
Policy LLM
Reference LLM
Policy LLM
PPO
GRPO
GRPO-Zero
w/o Neural Reward Model

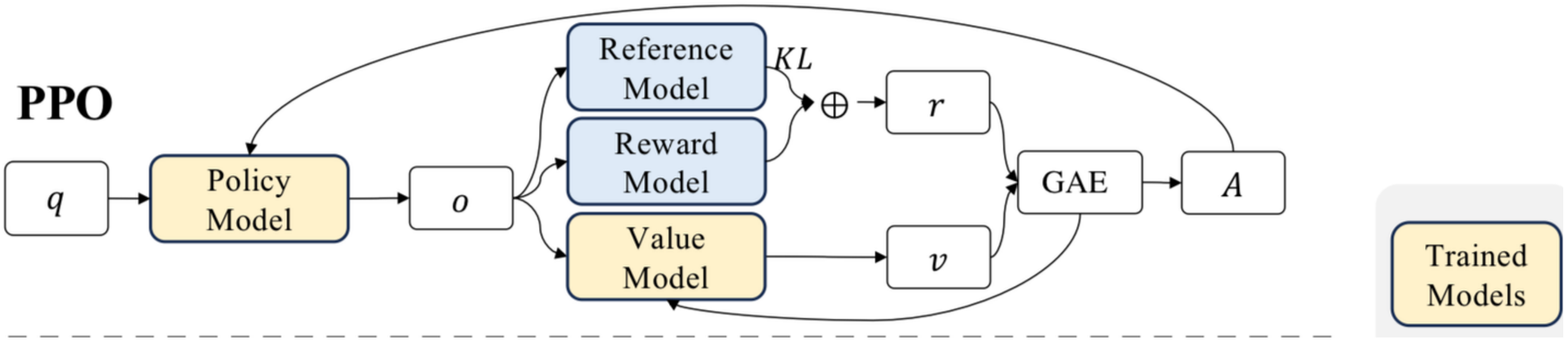
Group Relative Advantages

GRPO



- This helps give direction to update the original LLM's weights —
  - If the advantage is high, you want to encourage the model to keep doing the same actions.
  - If it is low, you want to encourage the model to try something different.

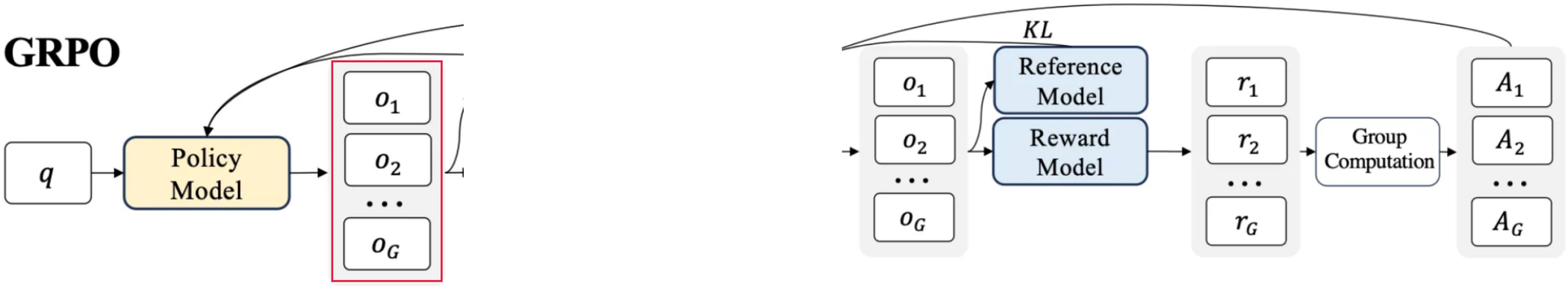
InstructGPT



The Problem: PPO can be pretty expensive to train.

Group Relative Advantages

PPO v/s GRPO



- So how does GRPO remove the need for this?
  - The first trick is that instead of generating one output per query, GRPO starts by generating multiple outputs.

Intuition Building

GRPO

- If the question posed is a mathematical problem.

- Eg:
    - Mr. Curtis has 325 chickens on his farm where 28 are roosters and the rest are hens. Twenty hens do not lay eggs while the rest of the hens do. How many egg-laying hens does Mr. Curtis have on his farm?

Intuition Building

GRPO

- Eg:
  - Mr. Curtis has 325 chickens on his farm where 28 are roosters and the rest are hens. Twenty hens do not lay eggs while the rest of the hens do. How many egg-laying hens does Mr. Curtis have on his farm?

question	answer	response	model_ar	is_correct
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, let's find out how many hens there are. The total n...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>You need to subtract the 20 hens that do not lay eggs fr...	305	false
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, we need to find the total number of hens. We know...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>Your observation is incorrect. 28 is more than half of 32...	297	false

question (str)

Mr. Curtis has 325 chickens on his farm where 28 are roosters and the rest are hens. Twenty hens do not lay eggs while the rest of the hens do. How many egg-laying hens does Mr. Curtis have on his farm?

Reward Model

GRPO

question	answer	response	model_ar	is_correct
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, let's find out how many hens there are. The total n...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>You need to subtract the 20 hens that do not lay eggs fr...	305	false
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, we need to find the total number of hens. We know...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>Your observation is incorrect. 28 is more than half of 32...	297	false

question (str)

Mr. Curtis has 325 chickens on his farm where 28 are roosters and the rest are hens. Twenty hens do not lay eggs while the rest of the hens do. How many egg-laying hens does Mr. Curtis have on his farm?

Correct Output

<reasoning>First, let's find out how many hens there are. The total number of chickens is 325, and 28 are roosters. So, the number of hens is 325 - 28 = 297. Of these 297 hens, 20 do not lay eggs, so the number of egg-laying hens is 297 - 20 = 277.</reasoning>

<answer>277</answer>

Incorrect Output

<reasoning>You need to subtract the 20 hens that do not lay eggs from the total number of hens to find the number of egg-laying hens. So, the number of egg-laying hens is 325 - 20 = 305.</reasoning>

<answer>305</answer>

Reward Model

GRPO

question	answer	response	model_ar	is_correct
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, let's find out how many hens there are. The total n...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>You need to subtract the 20 hens that do not lay eggs fr...	305	false
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>First, we need to find the total number of hens. We know...	277	true
Mr. Curtis has 325 chickens on his farm where ...	277	<reasoning>Your observation is incorrect. 28 is more than half of 32...	297	false

question (str)

Mr. Curtis has 325 chickens on his farm where 28 are roosters and the rest are hens. Twenty hens do not lay eggs while the rest of the hens do. How many egg-laying hens does Mr. Curtis have on his farm?

- Rewards may be given for:
  - Formatting — (1.0 points)
  - Answer — (0.0 points)
  - Consistency — (0.5 points)
- Higher when the response is positive and lower when the response is negative

Calculating Advantage

GRPO

- Once we have our set of rewards (r) given our outputs,
  - GRPO calculates our “advantage” (A) by simply looking at the mean and standard deviation of all the rewards.
- $\hat{A}_{i,t} = \tilde{r}_i = \frac{-mean(r)}{std(r)}$
- It helps normalize arbitrary values to more a more learnable positive or negative signal.
- Reward the good outputs, and penalize the bad ones in this batch.

Comparison to PPO

GRPO

- $\hat{A}_{i,t} = \tilde{r}_i = \frac{-mean(r)}{std(r)}$
- This is pretty similar to what the value model was originally trying to do:
  - i.e. Estimate what our reward will be given a response.

KL-Divergence

Final Piece of GRPO

- $-\beta \mathbb{D}_{KL}[\pi_\theta || \pi_{ref}]$
- This is why we have been keeping around a “reference model” during the training.
- The idea is that we do not want to drift too far from the original model.
- For each token, we want to make sure the new predictions do not drift too far from the original ones.

Final GRPO Equation

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{q,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|}\sum_{t=1}^{|o_i|} \min\left(r_t(\theta)\hat{A}_{i,t}, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t} \right) \right] - \beta D_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}).$$

- **Supervised Fine Tuning (SFT)**
  - Cold start the training with high quality data
  - A couple thousand examples verified by humans
- **Reinforcement Learning w/ GRPO**
  - Train the model to have reasoning traces <reasoning></reasoning>
  - Deterministic rewards for formatting, consistency, and correctness

- **Supervised Fine Tuning (SFT)**
  - Generate 800k Synthetic SFT data points and reject and filter
  - LLM As A Judge to filter incorrect responses
- **Reinforcement Learning w/ GRPO**
  - Align the model to be helpful and harmless

Final GRPO Equation

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 $\{o_1, o_2, \cdots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model π_θ by maximizing the following objective:

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

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

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

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

$$A_i = \frac{r_i - mean(\{r_1, r_2, \cdots, r_G\})}{std(\{r_1, r_2, \cdots, r_G\})}. \tag{3}$$

Deepseek-R1-Zero

The Reward Signals

- They are literally using regex and string matching for reward signals.
- They argue that this helps with “reward hacking” and simplifies the whole training pipeline.

Regex based Reward signaling

```python
def format_reward_func(completions, **kwargs) -> list[float]:
  """Reward function that checks if the completion has a specific format."""

  pattern = r"^<reasoning>\n.*?\n</reasoning>\n<answer>\n.*?\n</answer>\n$"
  responses = [completion[0]["content"] for completion in completions]
  matches = [re.match(pattern, r) for r in responses]
  return [0.5 if match else 0.0 for match in matches]
```