Post-training LLMs: Smarter Algorithms & Rewards

Kirill Tyshchuk
ex-Yandex, ex-PerplexityAI
Incoming Research Engineer, DeepMind

Skoltech
Skolkovo Institute of Science and Technology

Yandex

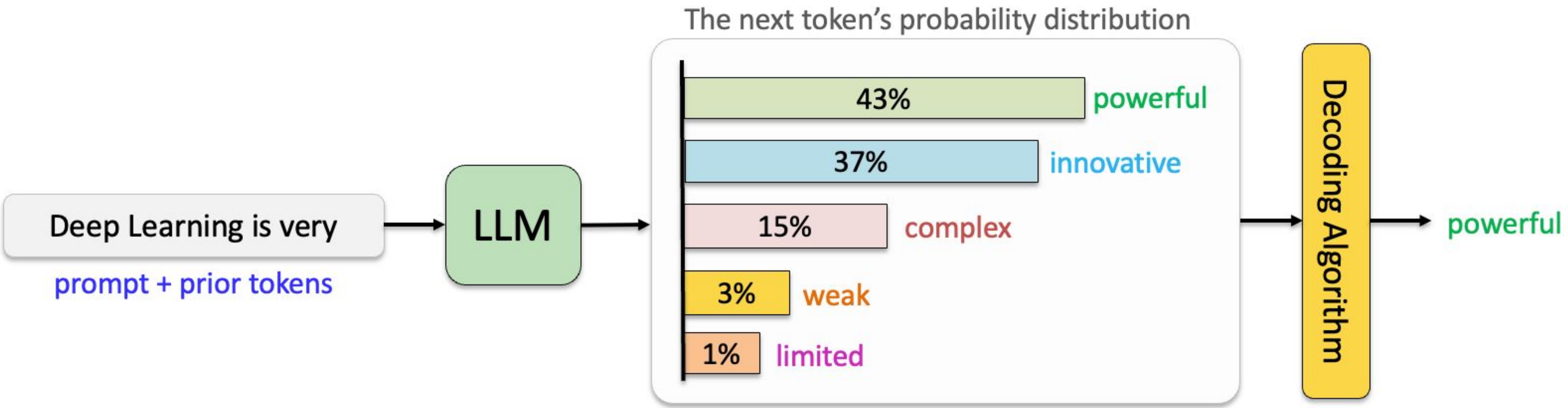
perplexity

Plan

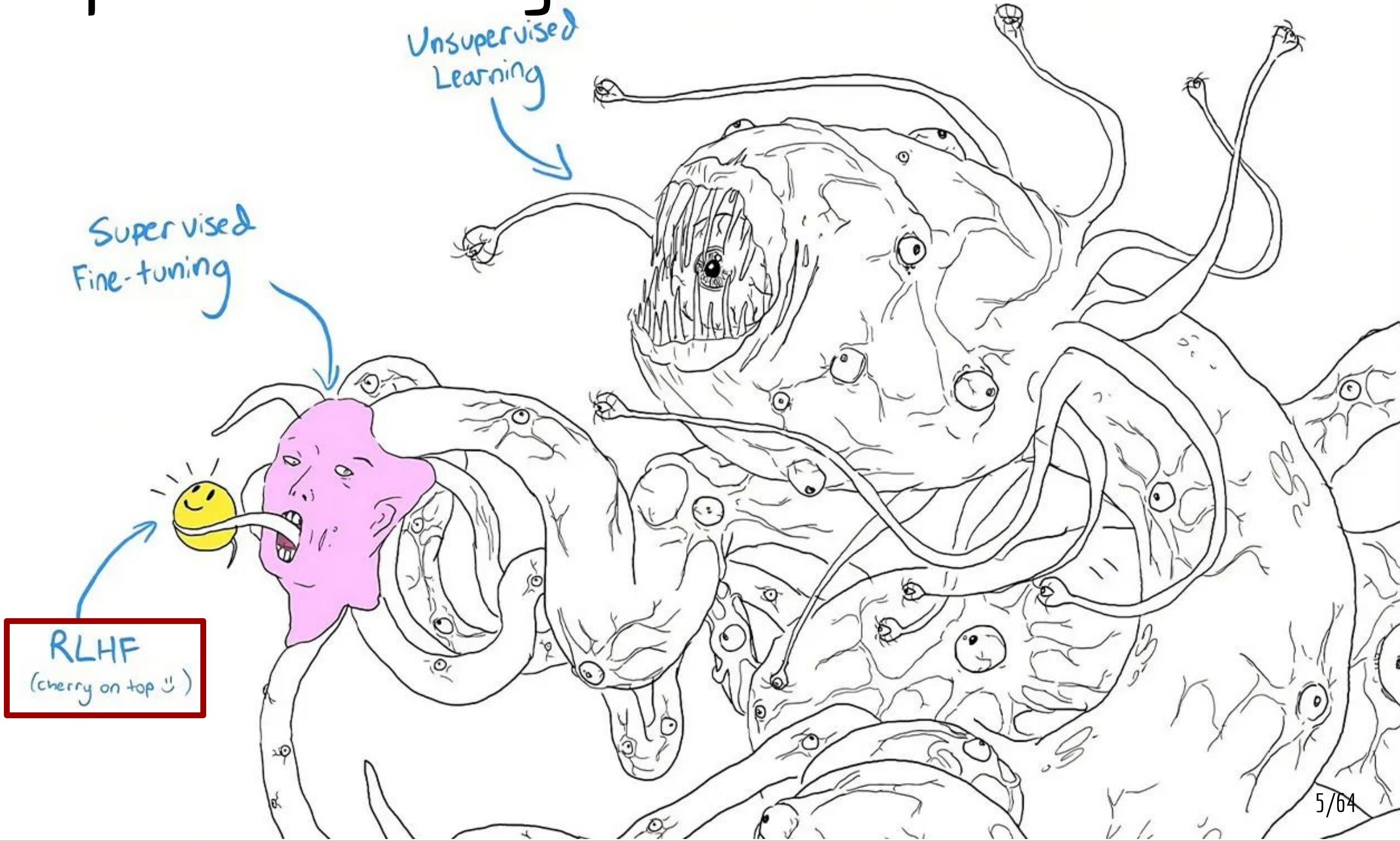
- Intro
  - LLM, pre-train, SFT
  - RL and RLHF
  - Reward modelling
- RLHF
  - Rejection sampling
  - PPO (KL, GAE)
  - DPO
  - RLOO, CGPO
  - Verifiable rewards
  - GRPO

Intro

Intro: LLMs


The next token's probability distribution
Deep Learning is very
prompt + prior tokens
LLM
43%
powerful
37%
innovative
15%
complex
3%
weak
1%
limited
Decoding Algorithm
powerful

Intro: stages of LLM training



Intro: pretraining

- Gather A LOT of text from the internet
- Train an LLM to predict the next word

- ➕ Cheap data
- ➖ Expensive large-scale training
- ➖ Don't adhere to instructions well
- ➖ Have to "trick" or fine-tune the model for specific tasks

What is the capital of France?
What is France's largest city?
What is France's population?
What is the currency of France?

Intro: Supervised Fine-Tuning (SFT)

- Collect examples written by humans
- Teach the LLM the output format and basic skills

$$\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w}$$

- ➕ High-quality data
- ➖ Expensive to collect data
- ➖ Expensive to change data
- ➖ Can’t directly penalize unwanted behavior
- ➖ LLM’s outputs won’t be better than its training data

What is the capital of France?
The capital of France is Paris.

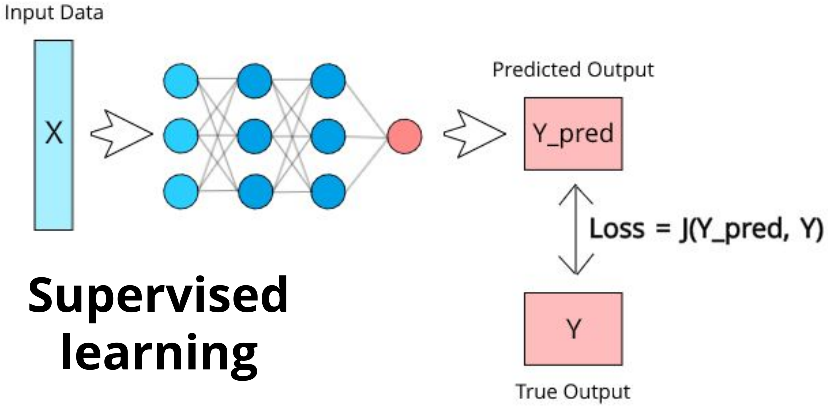
Questions?

RLHF

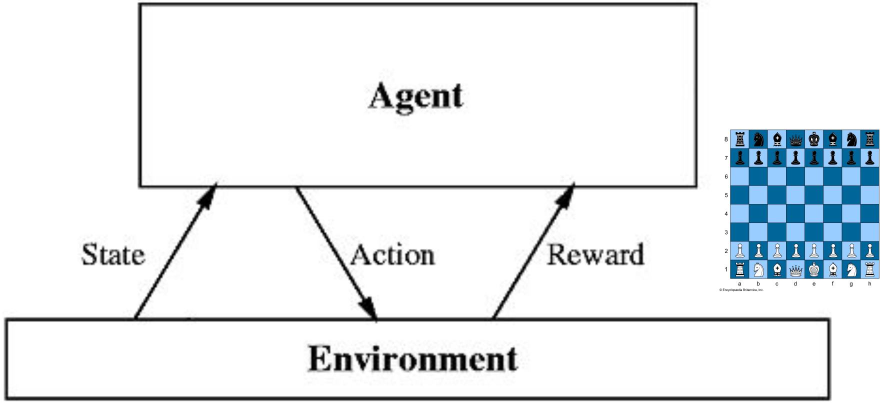
Intro: Reinforcement Learning (RL)

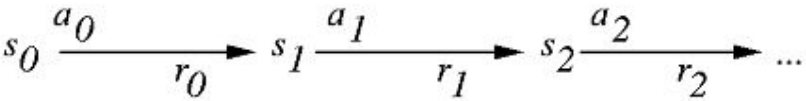
- Environment, actions, reward

- ➕ Chains of stochastic actions
- ➕ Non-differentiable reward
- ➖ Training is unstable



Supervised learning





Reinforcement learning

RLHF for LLMs

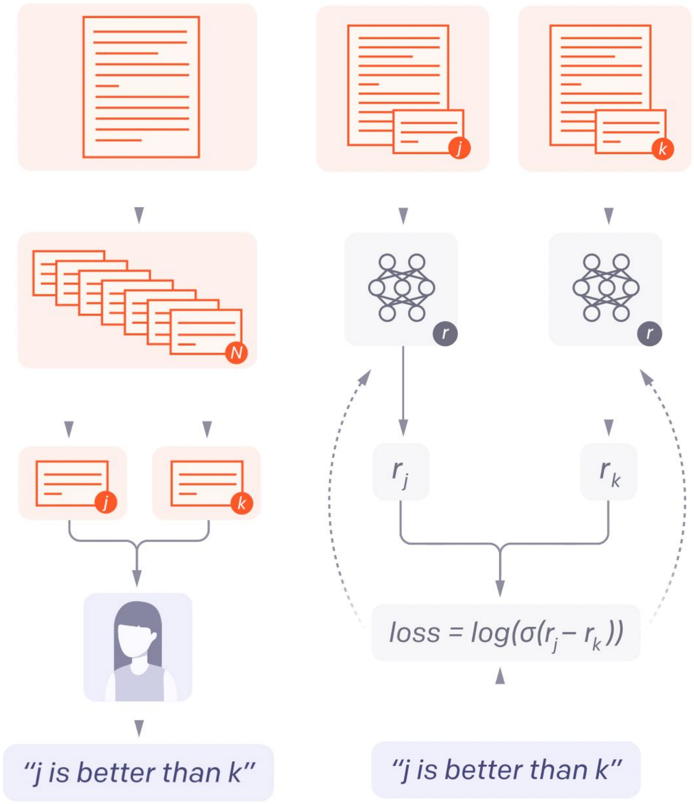
- Actions = tokens or the whole LLM answer
- Reward = how good the answer is

- ➕ Align the AI with human values
- ➕ Judging is easier than demonstrating
- ➕ Online learning and exploration (e.g. CoT)
- ➖ Still not scalable enough

RLHF: reward modelling

- Collect **pairwise** data from humans
- Train a reward model as approximation

- ➕ Scalable, fast inference
- ➕ Captures more nuance
- ➕ Removes calibration problem
- ➖ Reward's absolute value is meaningless
- ➖ Optimizing imperfect rewards
  leads to overfitting / goodharting



RLHF: reward modelling

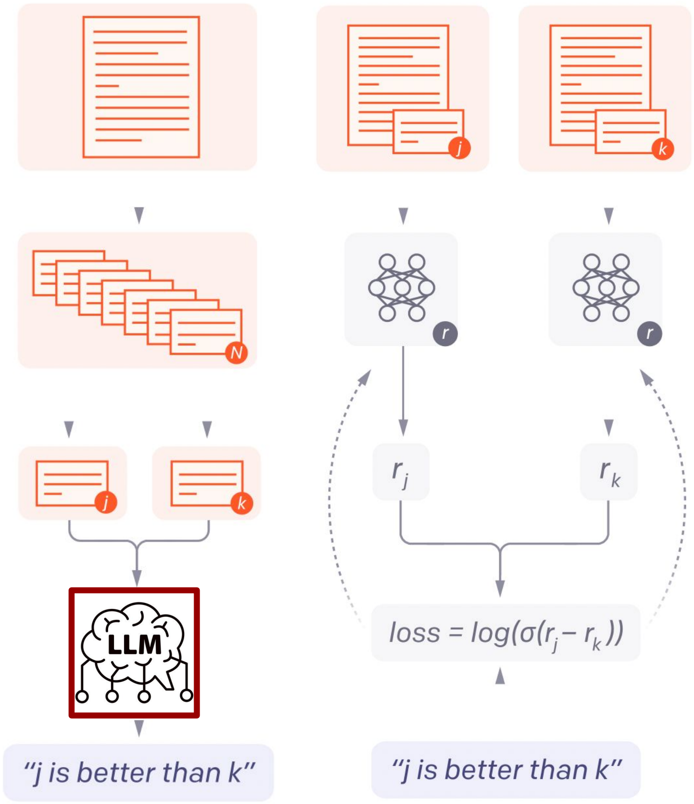
- Reward model training: loss from Bradley-Terry model

$$\mathcal{L}(\psi) = \log \sigma(r(x, y_w) - r(x, y_l))$$

RLAIF: reward modelling

- Collect data from a **frontier AI**
- Train a reward model as **distillation**

- ➕ Much cheaper than human labels
- ➕ Faster setup and iterations
- ➖ Lower quality



Putting this together

- Base (reference) model: pretrain or SFT
- Reward: reward model and/or hardcoded functions
- RL algorithm: trains the LLM to maximize the reward without going too far from the base model or mode-collapsing

Putting this together

- Base (reference) model: pretrain or SFT
- Reward: reward model and/or hardcoded functions
- RL algorithm: trains the LLM to maximize the reward without going too far from the base model or mode-collapsing

Questions?

RLHF Algorithms

Rejection sampling (poor man's RL)

- Sample multiple completions per each prompt
- Pick the best
- Do SFT on those
- [Repeat]

$$\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w}$$

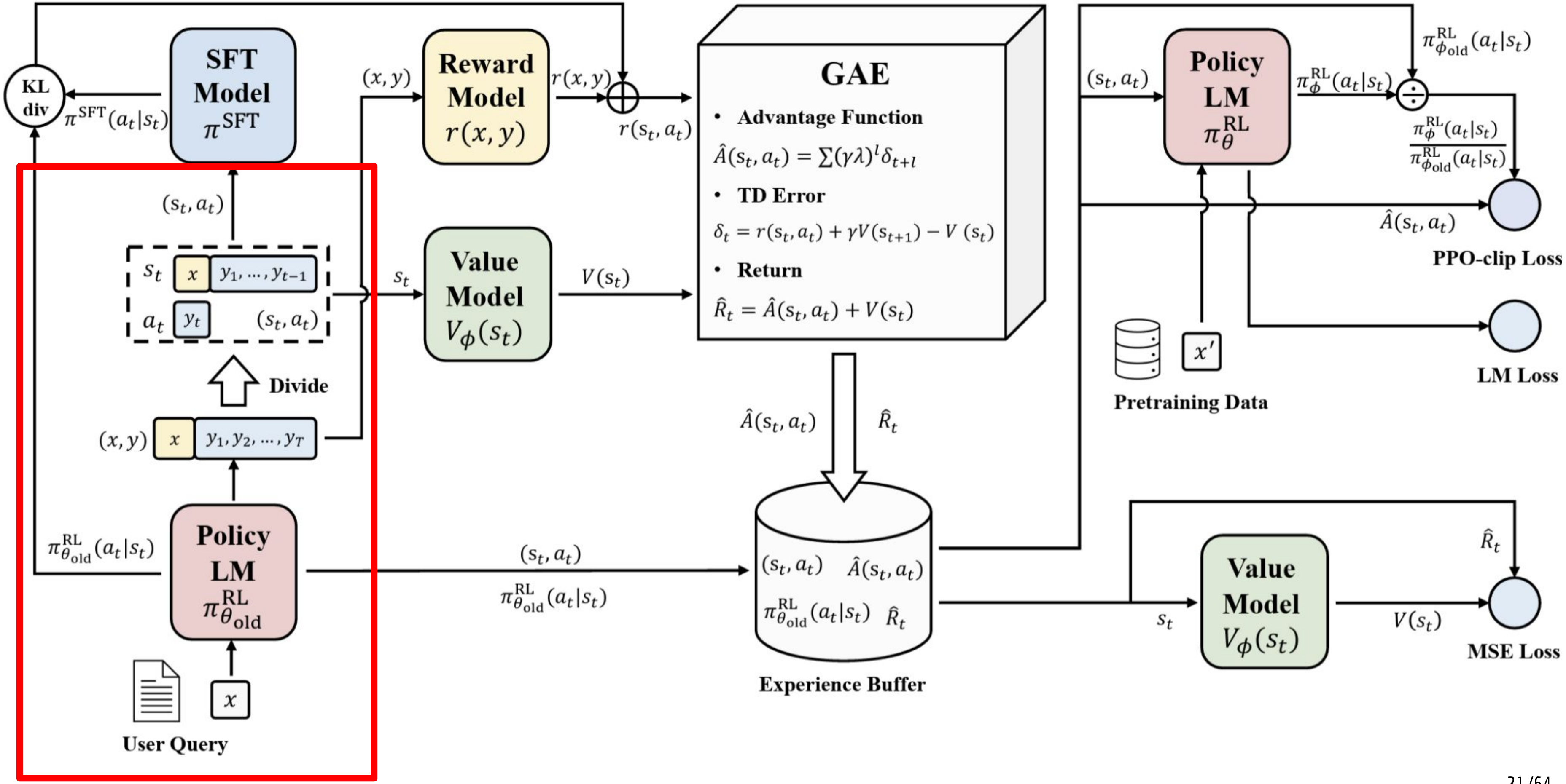
- ➕ Easy to implement
- ➕ Good sanity check for the reward
- ➖ Not very efficient/effective

PPO - Proximal Policy Optimization

- Components:
  - Policy model
  - Reference model
  - Reward
  - Value (critic) model
  - Duct tape
- Examples:
  - InstructGPT
  - ChatGPT
  - Llama 2

PPO - Proximal Policy Optimization

- Do several epochs
- Our current policy is $\pi_{\theta_{\text{old}}}$
- Step 1: sample generations


KL div
SFT Model π^{SFT}
$\pi^{SFT}(a_t|s_t)$
Reward Model $r(x,y)$
(x,y)
$r(x,y)$
$r(s_t,a_t)$
GAE
Advantage Function
$\hat{A}(s_t,a_t) = \sum(\gamma\lambda)^l \delta_{t+l}$
TD Error
$\delta_t = r(s_t,a_t) + \gamma V(s_{t+1}) - V(s_t)$
Return
$\hat{R}_t = \hat{A}(s_t,a_t) + V(s_t)$
Policy LM π_θ^{RL}
(s_t,a_t)
$\pi_\phi^{RL}(a_t|s_t)$
$\pi_{\phi_{old}}^{RL}(a_t|s_t)$
$\frac{\pi_\phi^{RL}(a_t|s_t)}{\pi_{\phi_{old}}^{RL}(a_t|s_t)}$
$\hat{A}(s_t,a_t)$
PPO-clip Loss
LM Loss
Pretraining Data
x'
s_t x $y_1, \dots, y_{t-1}$
a_t y_t
(s_t,a_t)
Divide
(x,y) x $y_1, y_2, \dots, y_T$
Value Model $V_\phi(s_t)$
s_t
$V(s_t)$
Policy LM $\pi_{\theta_{old}}^{RL}$
$\pi_{\theta_{old}}^{RL}(a_t|s_t)$
User Query
x
$\hat{A}(s_t,a_t)$ $\hat{R}_t$
(s_t,a_t) $\hat{A}(s_t,a_t)$
$\pi_{\theta_{old}}^{RL}(a_t|s_t)$ $\hat{R}_t$
Experience Buffer
Value Model $V_\phi(s_t)$
$\hat{R}_t$
$V(s_t)$
MSE Loss

PPO - Proximal Policy Optimization

- Step 2: construct the reward: reward model + regularization

$$r_{\text{total}} = r(x, y) - \eta \mathrm{KL}(\pi_\phi^{\mathrm{RL}}(y|x), \pi^{\mathrm{SFT}}(y|x))$$

Note on KL estimators

- Monte-Carlo estimator
- Difference of current and SFT logprobs

$$KL[q,p] = \sum_x q(x) \log \frac{q(x)}{p(x)} = E_{x \sim q}[\log \frac{q(x)}{p(x)}]$$

Note on KL estimators

- Monte-Carlo estimator
- Difference of current and SFT logprobs
- Can we do better?

$$KL[q,p] = \sum_x q(x) \log \frac{q(x)}{p(x)} = E_{x\sim q}[\log \frac{q(x)}{p(x)}]$$

$$\log \frac{q(x)}{p(x)} = -\log r$$

$$\frac{1}{2}(\log \frac{p(x)}{q(x)})^2 = \frac{1}{2}(\log r)^2$$

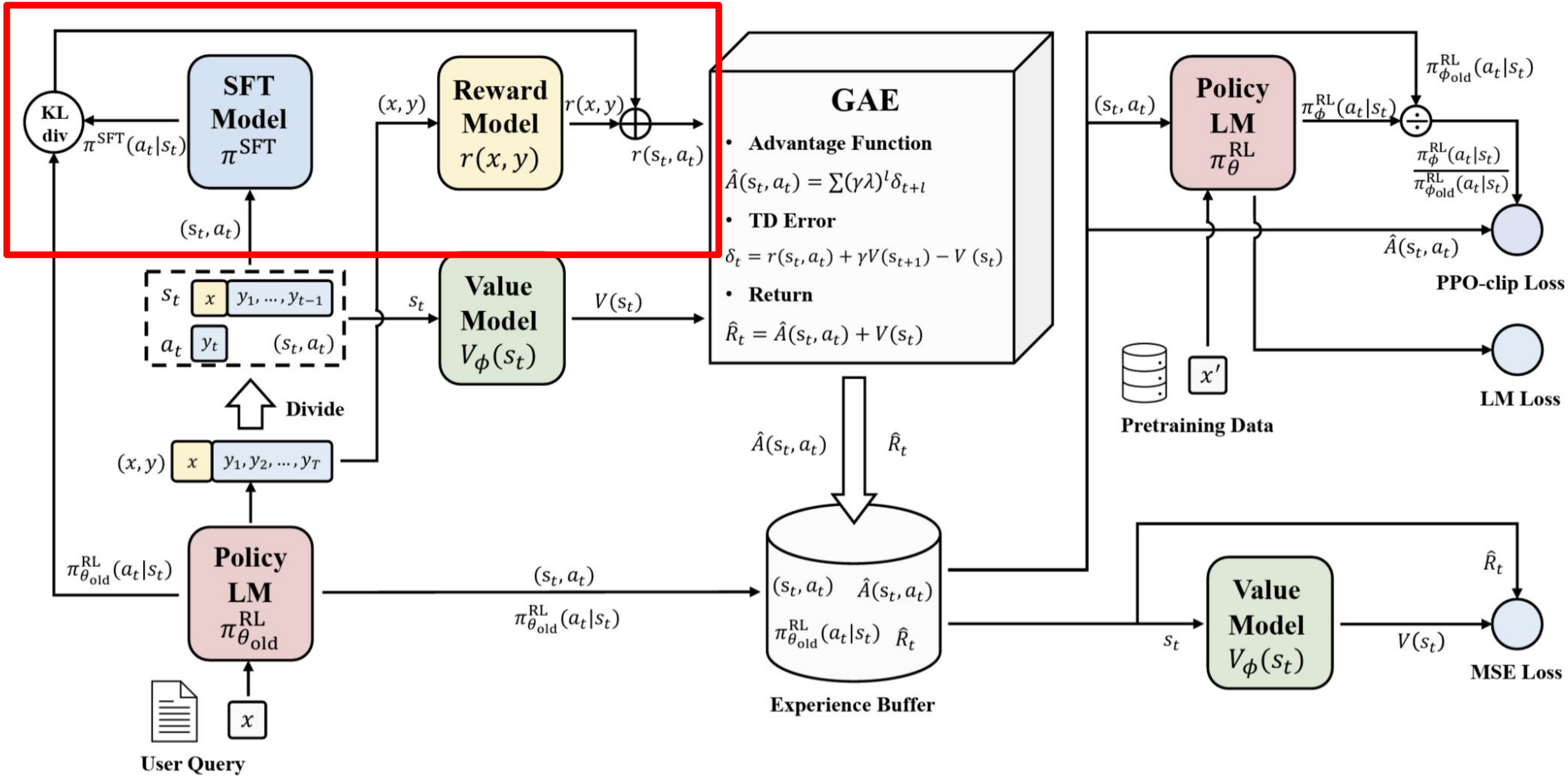
$$(r-1) - \log r$$

	bias/true	stdev/true
k1	0	20
k2	0.002	1.42
k3	0	1.42

$q = N(0,1), p = N(0.1,1)$

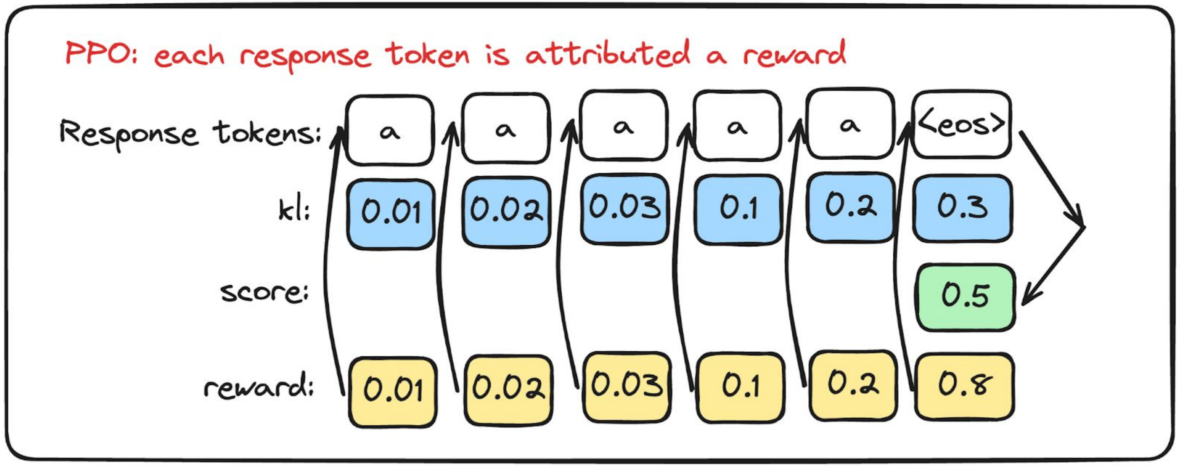
	bias/true	stdev/true
k1	0	2
k2	0.25	1.73
k3	0	1.7

$p = N(1,1)$


KL div
SFT Model
Reward Model
GAE
Advantage Function
TD Error
Return
Value Model
Divide
Policy LM
User Query
Experience Buffer
Pretraining Data
PPO-clip Loss
LM Loss
MSE Loss

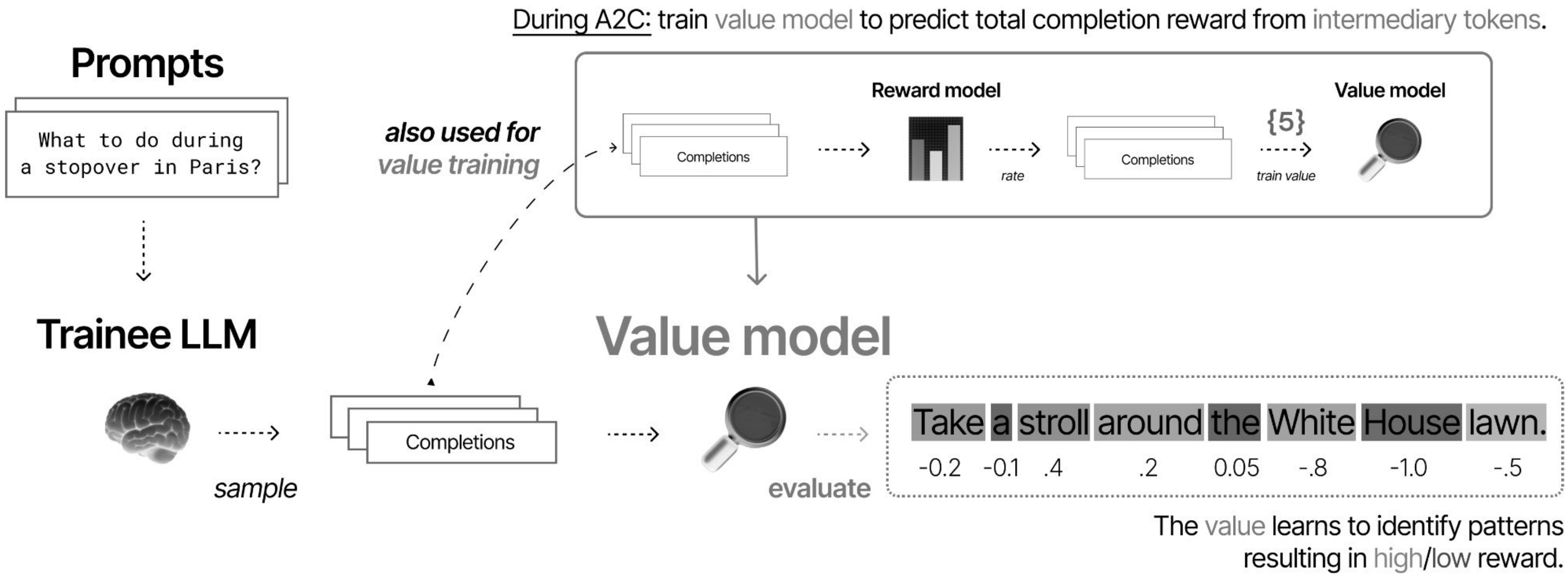
PPO - Proximal Policy Optimization

- PPO has a per-token reward (because of KL)



PPO - Proximal Policy Optimization

- Use advantage instead of return
- We have the value model (critic) V to estimate expected future return



PPO - Proximal Policy Optimization

- Step 3: infer the value model and compute advantage (GAE)
    - Future rewards are noisy
    - Value estimations are biased
    - Let’s find a middle ground

GAE

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \ldots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

GAE

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \ldots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$$\hat{A}_t^k = \hat{R}_t^k - V(s_t) = \sum^{k} \gamma^l \delta_{t+l} = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

GAE

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \ldots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$$\hat{A}_t^k = \hat{R}_t^k - V(s_t) = \sum^{k} \gamma^l \delta_{t+l} = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

$$\begin{aligned}
\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} &= (1-\lambda)(\hat{A}_t^{(1)} + \lambda \hat{A}_t^{(2)} + \lambda^2 \hat{A}_t^{(3)} + \cdots) \\
&= (1-\lambda)(\delta_t + \lambda(\delta_t + \gamma\delta_{t+1}) + \lambda^2(\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2}) + \ldots) \\
&= (1-\lambda)(\delta_t(1 + \lambda + \lambda^2 + \ldots) + \gamma\delta_{t+1}(\lambda + \lambda^2 + \lambda^3 + \ldots) \\
&\quad + \gamma^2\delta_{t+2}(\lambda^2 + \lambda^3 + \lambda^4 + \ldots) + \ldots) \\
&= (1-\lambda)(\delta_t(\frac{1}{1-\lambda}) + \gamma\delta_{t+1}(\frac{\lambda}{1-\lambda}) + \gamma^2\delta_{t+2}(\frac{\lambda^2}{1-\lambda}) + \ldots) \\
&= \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}.
\end{aligned}$$

GAE

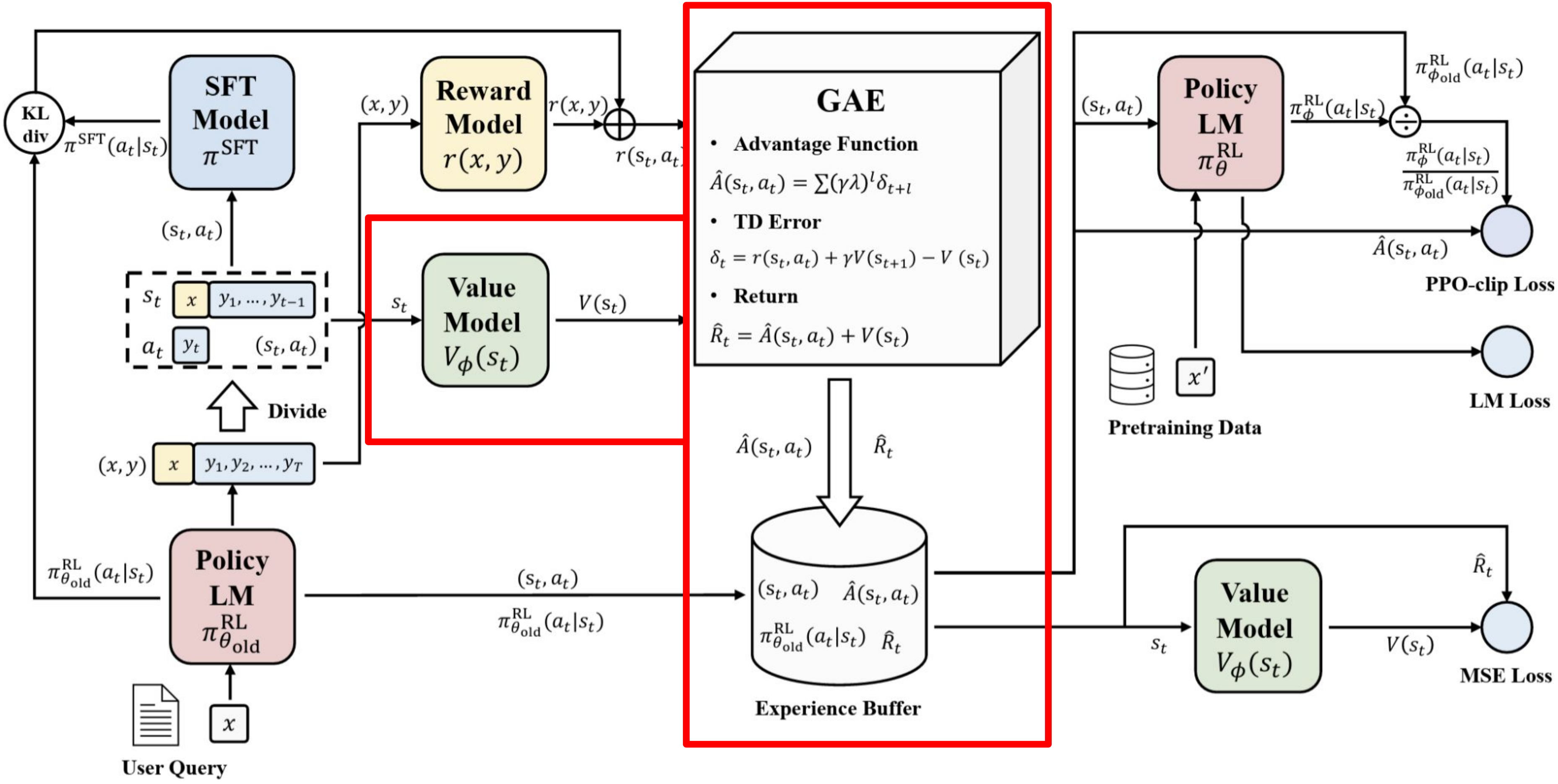
- Calculated for each state by looping over a reversed trajectory
- Limit cases:

$$\text{GAE}(\gamma, 0) : \hat{A}_t = \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t).$$

$$\text{GAE}(\gamma, 1) : \hat{A}_t = \sum_{l=0}^{\infty} \gamma^l \delta_{t+1} = \sum_{l=0}^{\infty} \gamma^l r_{t+1} - V(s_t).$$

- Advantages can be used for Policy Gradient:

$$\nabla_\theta \hat{J}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}_t,$$


KL div
SFT Model π^{SFT}
$\pi^{SFT}(a_t|s_t)$
Reward Model $r(x, y)$
(x, y)
$r(x, y)$
$r(s_t, a_t)$
GAE
• Advantage Function
$\hat{A}(s_t, a_t) = \sum(\gamma\lambda)^l \delta_{t+l}$
• TD Error
$\delta_t = r(s_t, a_t) + \gamma V(s_{t+1}) - V(s_t)$
• Return
$\hat{R}_t = \hat{A}(s_t, a_t) + V(s_t)$
(s_t, a_t)
s_t x $y_1, \ldots, y_{t-1}$
a_t y_t
Divide
(x, y) x $y_1, y_2, \ldots, y_T$
Value Model $V_\phi(s_t)$
$V(s_t)$
$\hat{A}(s_t, a_t)$ $\hat{R}_t$
Experience Buffer
(s_t, a_t) $\hat{A}(s_t, a_t)$
$\pi^{RL}_{\theta_{old}}(a_t|s_t)$ $\hat{R}_t$
Policy LM $\pi^{RL}_{\theta_{old}}$
$\pi^{RL}_{\theta_{old}}(a_t|s_t)$
User Query
x
Policy LM π^{RL}_θ
$\pi^{RL}_\phi(a_t|s_t)$
$\pi^{RL}_{\phi_{old}}(a_t|s_t)$
$\frac{\pi^{RL}_\phi(a_t|s_t)}{\pi^{RL}_{\phi_{old}}(a_t|s_t)}$
PPO-clip Loss
LM Loss
Pretraining Data
x'
Value Model $V_\phi(s_t)$
MSE Loss

PPO - Proximal Policy Optimization

- Having the replay buffer, do several iterations of optimization
- But don’t overfit on the trajectories
- Step 4: construct the loss and optimize policy
- TRPO would do this:

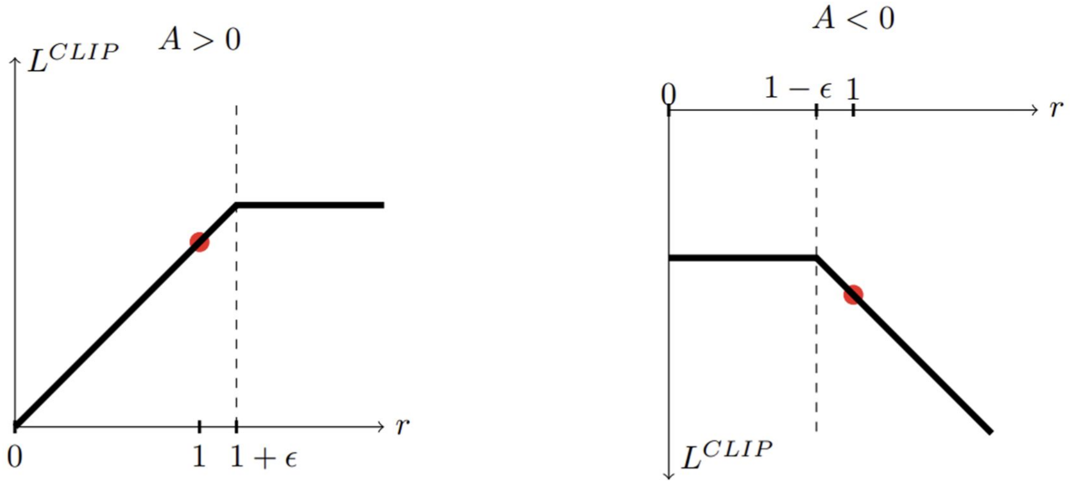
$$\begin{aligned}&\text{maximize}_{\theta}\ \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right],\\&\text{subject to}\ \hat{\mathbb{E}}_t\left[\mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t),\pi_\theta(\cdot|s_t))\right]\le\delta,\end{aligned}$$

- *This is the “surrogate objective”, not the true loss, but close

PPO - Proximal Policy Optimization

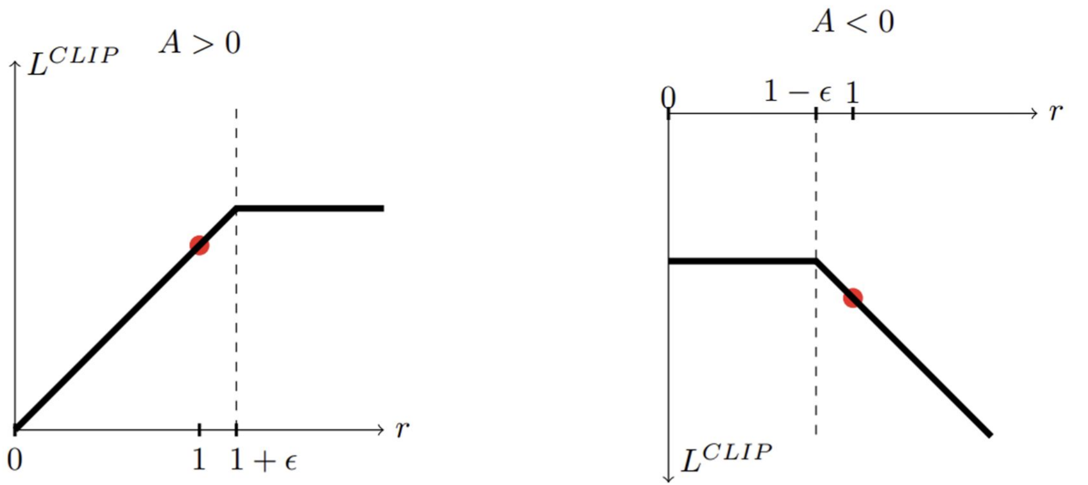
- Instead, PPO does

$$\mathcal{L}_{\text{ppo}-\text{clip}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \text{clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right) \right],$$



PPO - Proximal Policy Optimization

- No optimization if the ratio is already high enough / low enough

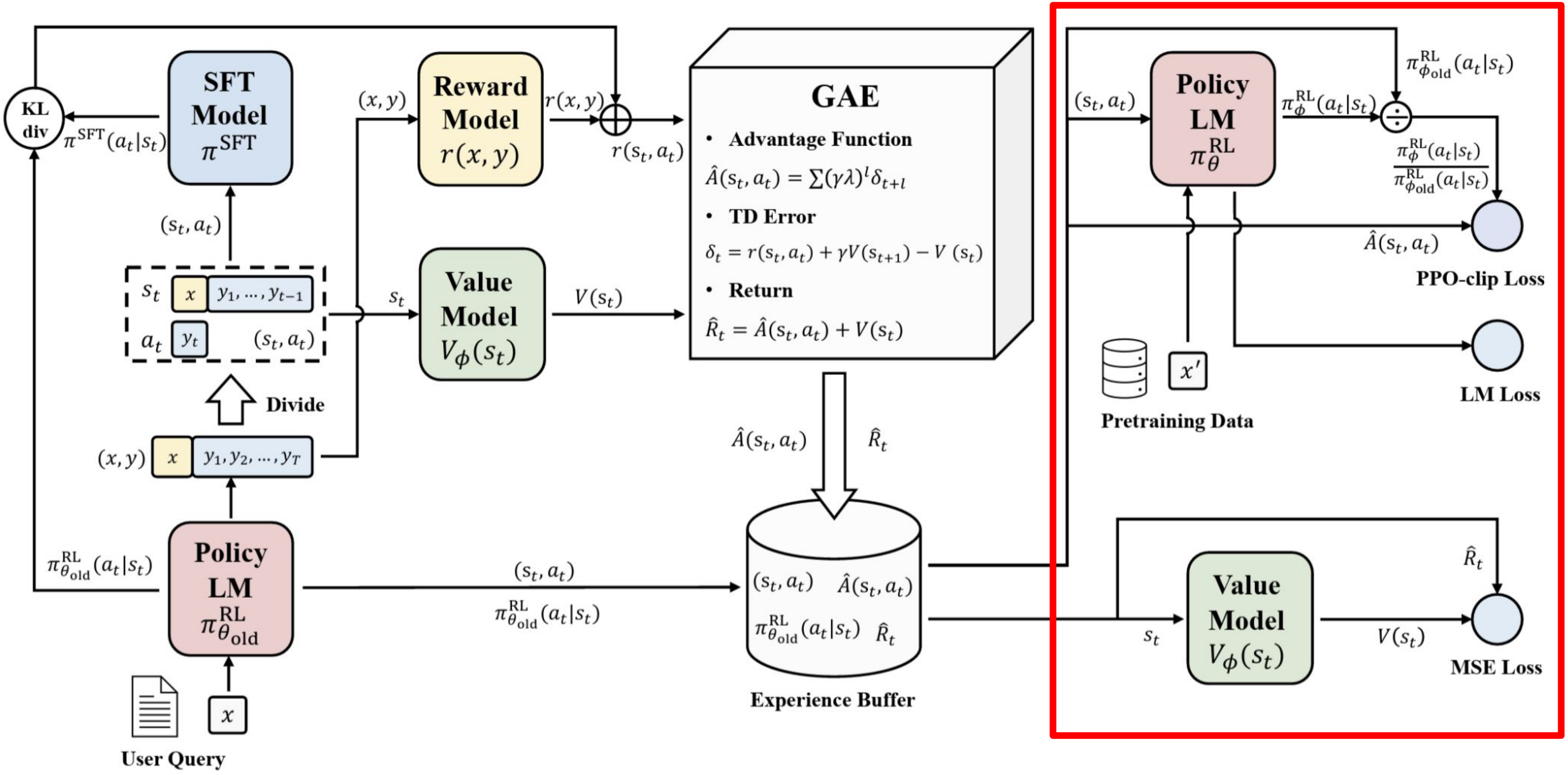


PPO - Proximal Policy Optimization

- Step 4.5: optimize the value function

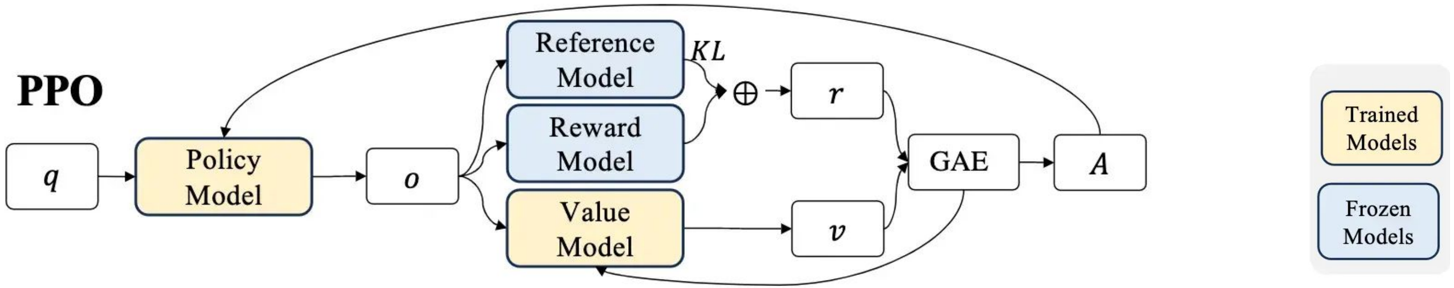
$$\hat{R}_t = \sum_{l=0}^{\infty} \gamma^l r_{t+l}.$$

$$\mathcal{L}_{\text{critic}}(\phi) = \hat{\mathbb{E}}_t \left[\| V_\phi(s_t) - \hat{R}_t \|^2 \right]$$


KL div
SFT Model π^{SFT}
$\pi^{SFT}(a_t|s_t)$
Reward Model $r(x,y)$
(x,y)
$r(x,y)$
$r(s_t,a_t)$
GAE
• Advantage Function
$\hat{A}(s_t,a_t) = \sum(\gamma\lambda)^l \delta_{t+l}$
• TD Error
$\delta_t = r(s_t,a_t) + \gamma V(s_{t+1}) - V(s_t)$
• Return
$\hat{R}_t = \hat{A}(s_t,a_t) + V(s_t)$
(s_t,a_t)
s_t x $y_1,\dots,y_{t-1}$
a_t y_t (s_t,a_t)
Value Model $V_\phi(s_t)$
s_t
$V(s_t)$
Divide
(x,y) x $y_1,y_2,\dots,y_T$
$\hat{A}(s_t,a_t)$ $\hat{R}_t$
Policy LM $\pi^{RL}_{\theta_{old}}$
$\pi^{RL}_{\theta_{old}}(a_t|s_t)$
(s_t,a_t)
$\pi^{RL}_{\theta_{old}}(a_t|s_t)$
(s_t,a_t) $\hat{A}(s_t,a_t)$
$\pi^{RL}_{\theta_{old}}(a_t|s_t)$ $\hat{R}_t$
Experience Buffer
User Query
x
(s_t,a_t)
Policy LM π^{RL}_θ
$\pi^{RL}_{\phi_{old}}(a_t|s_t)$
$\pi^{RL}_\phi(a_t|s_t)$
$\frac{\pi^{RL}_\phi(a_t|s_t)}{\pi^{RL}_{\phi_{old}}(a_t|s_t)}$
$\hat{A}(s_t,a_t)$
PPO-clip Loss
x'
Pretraining Data
LM Loss
$\hat{R}_t$
Value Model $V_\phi(s_t)$
s_t
$V(s_t)$
MSE Loss

PPO - Proximal Policy Optimization

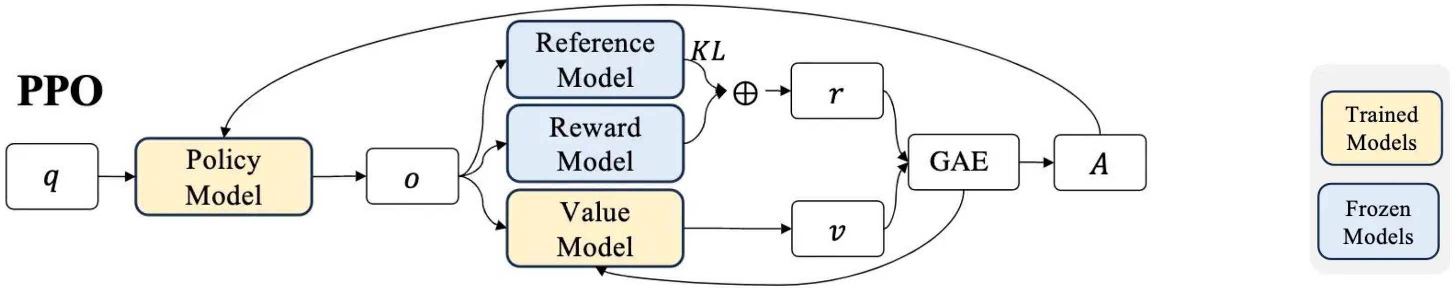
- Recap



PPO - Proximal Policy Optimization

- Recap

Questions?



DPO - Direct Preference Optimization

- Components:
  - Policy model
  - Reference model
  - Ranked completion pairs **(no reward!)**
  - **No rollouts. no RL**
- Examples:
  - Llama3
  - Qwen 2.5

DPO - Direct Preference Optimization

- Recall reward modelling: preferences come from the reward

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}.$$

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$$

DPO - Direct Preference Optimization

- Then the optimal policy maximizes the regularized objective:
  What's the optimal policy?

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi}\left[r(x,y)\right] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi(y|x) \mid\mid \pi_{\mathrm{ref}}(y|x)\right]$$

$$= \max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right]$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \frac{1}{\beta}r(x,y)\right]$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]$$

DPO - Direct Preference Optimization

- Rewrite as KL:

$$Z(x) = \sum_y \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right)$$

$$\pi^*(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right)$$

$$\min_\pi \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{E}_{y \sim \pi(y|x)} \left[\log \frac{\pi(y|x)}{\pi^*(y|x)}\right] - \log Z(x)\right] =$$

$$\min_\pi \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{D}_{\text{KL}}(\pi(y|x) \mid\mid \pi^*(y|x)) - \log Z(x)\right]$$

$$\pi(y|x) = \pi^*(y|x)$$

DPO - Direct Preference Optimization

- Reward function VS optimal policy:

$$\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)$$

$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

↑ Does not matter

- Bijection between the policies and the reward equivalence classes

DPO - Direct Preference Optimization

- Preference likelihood w.r.t. optimal policy:

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}$$

- Optimize it directly!

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

DPO - Direct Preference Optimization

- What does the gradient update actually do?

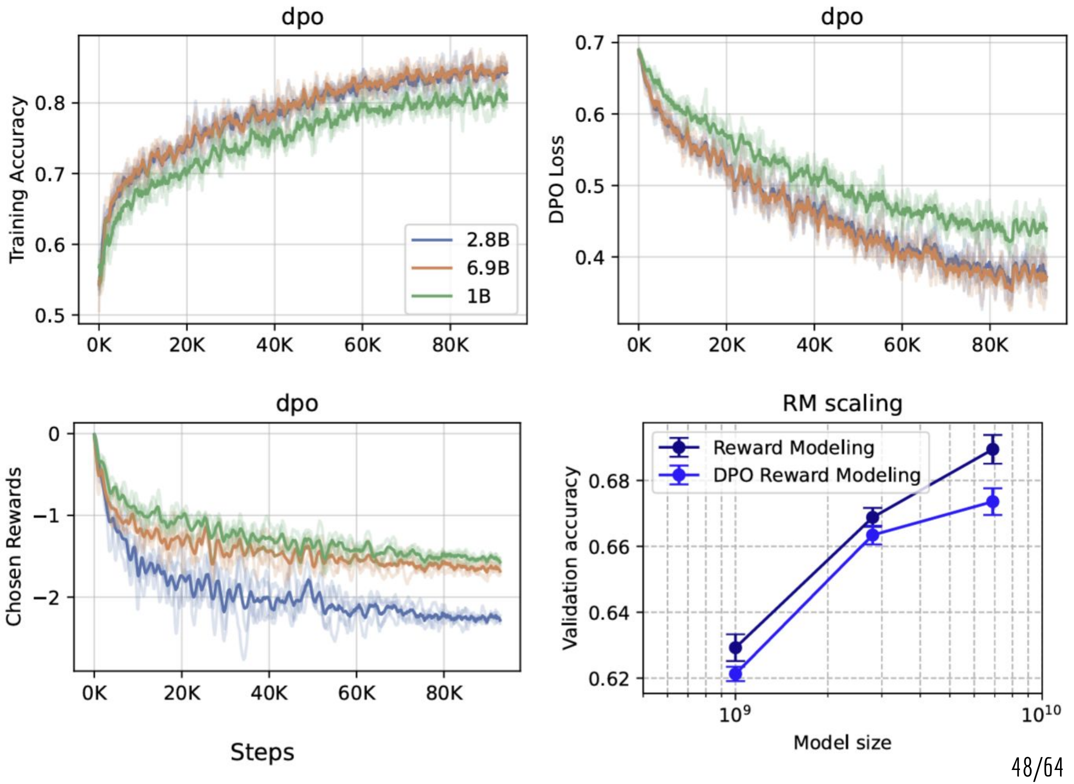
$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$

$$-\beta \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \bigg[\underbrace{\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))}_{\text{higher weight when reward estimate is wrong}} \bigg[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decrease likelihood of } y_l} \bigg] \bigg]$$

- This is just weighted learning and un-learning!
- No per-token rewards

DPO - Direct Preference Optimization

- Prob(winning) declines :(
  => add SFT loss
- Or SFT on winning first
- RM/DPO accuracy ~70%



Bonus: iterated DPO with a reward model

- Components:
  - Policy model
  - Reference model
  - **Reward model**
- Algorithm:
  - Sample multiple completions
  - Score with reward
  - Pick a good one and a bad one
  - Do DPO on those pairs
  - [Repeat]

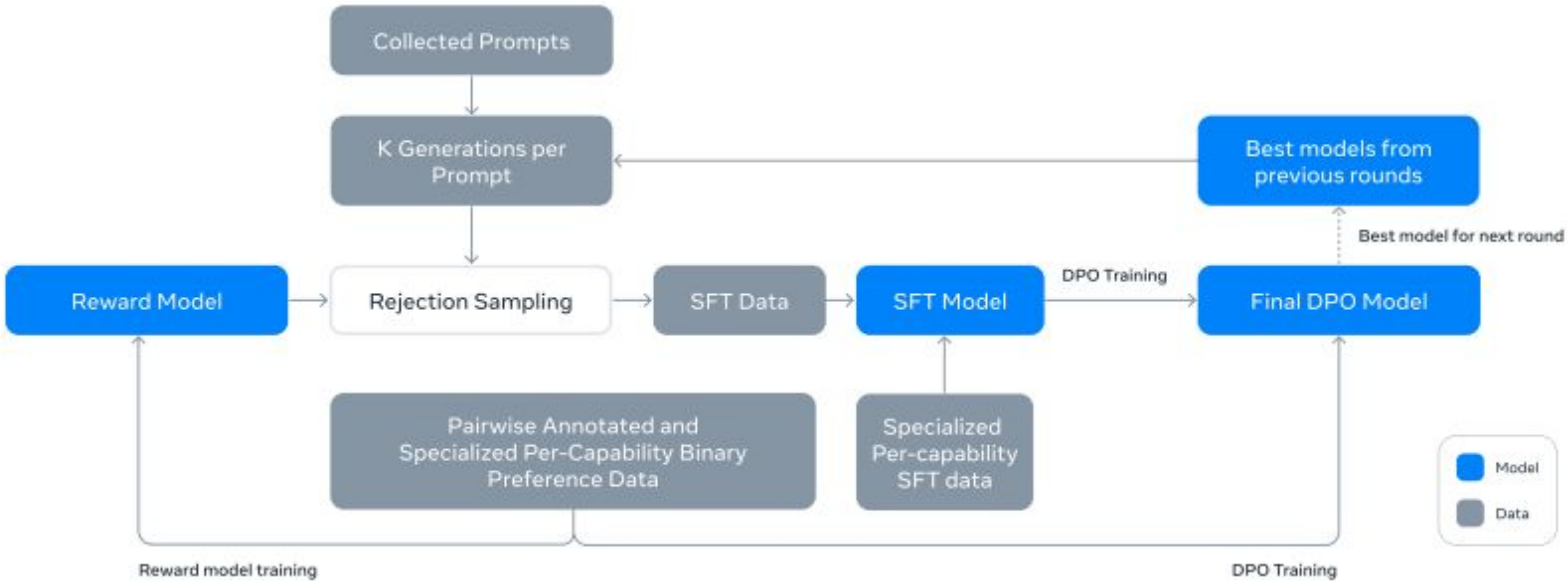
Bonus: iterated DPO with a reward model

- Components:
    - Policy model
    - Reference model
    - **Reward model**
- Algorithm:
    - Sample multiple completions
    - Score with reward
    - Pick a good one and a bad one
    - Do DPO on those pairs
    - [Repeat]

Questions?

RLFH iterations

- E.g. Llama 3:



RLOO - Cohere's REINFORCE Leave-One-Out

- Components:
    - Policy model
    - Reference model
    - Reward model
    - **(no value model)**

$$\frac{1}{k}\sum_{i=1}^{k}[R(y_{(i)},x)-\frac{1}{k-1}\sum_{j\neq k}R(y_{(j)},x)]\nabla\log\pi(y_{(i)}|x)\ \text{for}\ y_{(1)},\ldots,y_{(k)}\overset{i.i.d}{\sim}\pi_\theta(.|x)$$

RLOO - Cohere's REINFORCE Leave-One-Out

- Weighted SFT-like learning on above-average generations and weighted un-learning on below-average
- Like Rejection Sampling and DPO+RM, but uses all generations

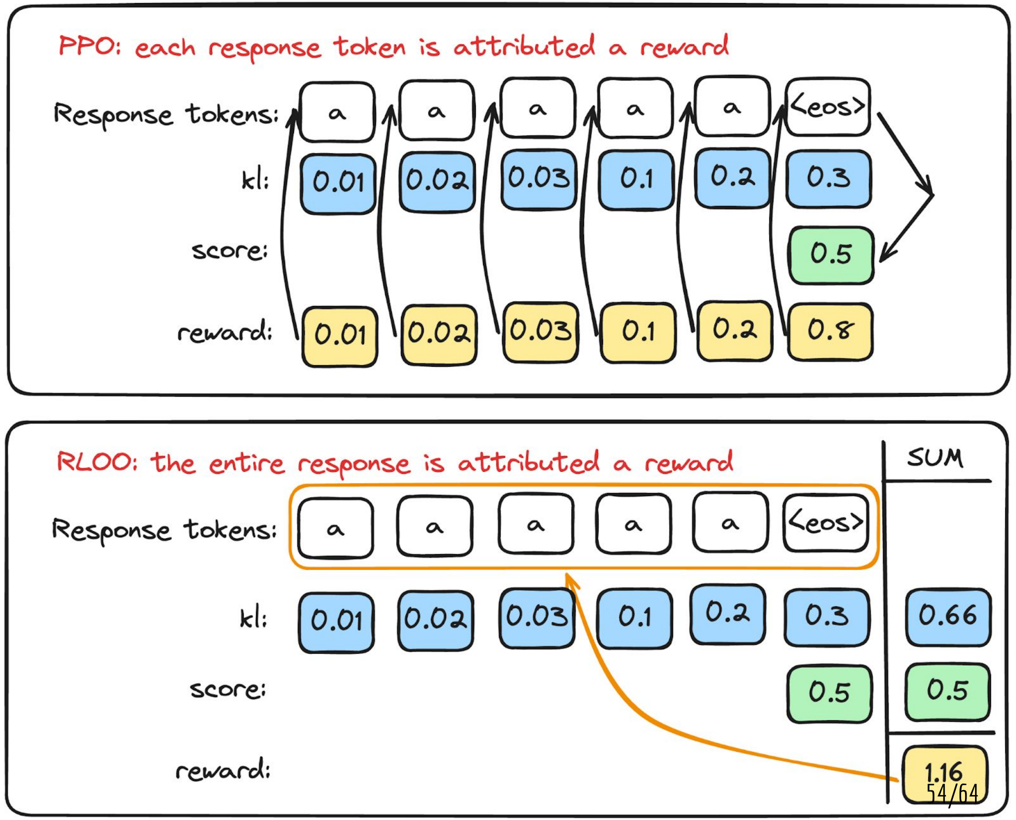
$$\frac{1}{k}\sum_{i=1}^{k}[R(y_{(i)},x) - \frac{1}{k-1}\sum_{j\neq k} R(y_{(j)},x)]\nabla \log \pi(y_{(i)}|x) \text{ for } y_{(1)},\ldots,y_{(k)} \overset{i.i.d}{\sim} \pi_\theta(.|x)$$

↑

Can be replaced with the average reward

RLOO - Cohere's REINFORCE Leave-One-Out

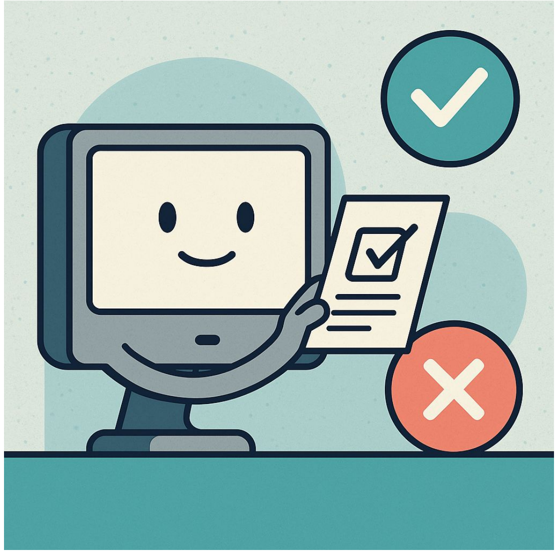
- No intermediate rewards
- "1 action"
- But in PPO intermediate advantages are synthetic anyway



RLVR: Verifiable rewards and filters

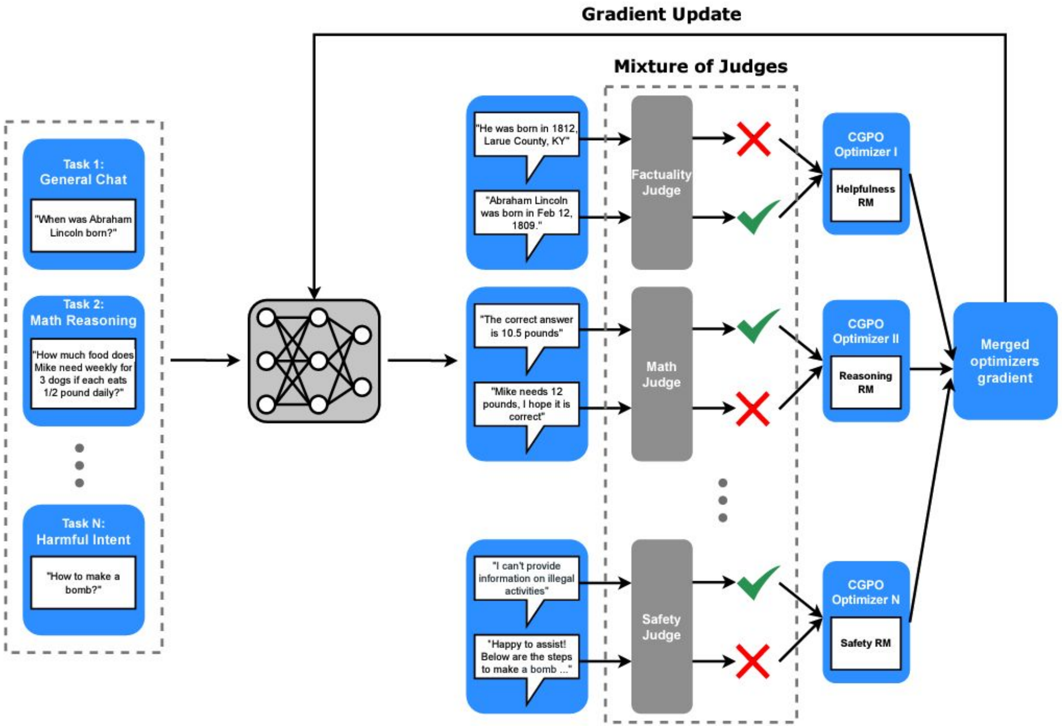
- Objective, hard-coded scores
- No reward hacking*

- Examples
  - Is length < 1024?
  - Is this a valid JSON?
  - Is this numeric answer for a math problem correct?
  - Does this code compile and pass tests?



CGPO - Meta's "Perfect blend"

- Components:
  - Policy model
  - Reference model
  - Reward models
  - **Binary judges**
- Algorithm:
  - Sample multiple completions
  - Score with reward
  - Score 0/1 with judges
  - Increase prob of above average + passing
  - Decrease prob of below average or failing



GRPO: Group Relative Policy Optimization

- Components:
  - Policy model
  - Reference model
  - [Verifiable] Reward
  - **(no value model)**
- Estimate advantage from the group
- PPO loss
- Move KL from reward into loss

GRPO: Group Relative Policy Optimization

$$\mathcal{L}_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}\hat{A}_{i,t},\, g(\epsilon,\hat{A}_{i,t})\right] - \beta D_{\text{KL}}[\pi_\theta||\pi_{\text{ref}}]$$

Probability ratio i.e. how likely is the new policy?

Clips advantage between 1-ε & 1+ε. ε is a hyper parameter.

KL divergence b/w the old and new policy, scaled by hyper parameter β

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}$$

Advantage function (based on normalised rewards)

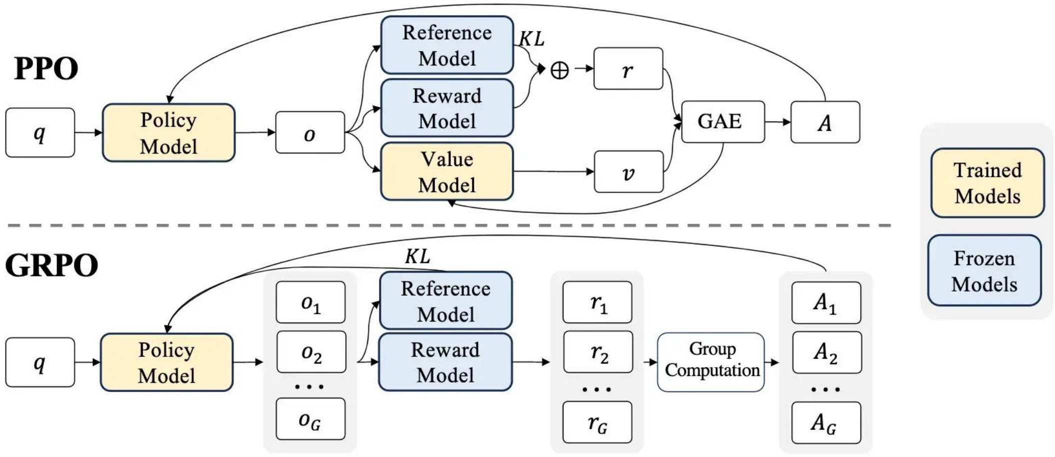
Average Loss per token in specific output -->

1. Add surrogate loss across all tokens within a specific output oi.
2. Divide the sum by |oi| i.e. len(oi) so that each output has equal contribution.

Average across tokens of clipped surrogate loss -->

GRPO: Group Relative Policy Optimization

- DeepSeek-R1 and -R1-Zero
- We could start from the base model and remove KL



DAPO: Decoupled Clip and Dynamic sAmpling Policy Optimization

- GRPO + tweaks from ByteDance
- Removes the KL regularization for RLVR
- Tweaks the PPO loss formula (clips higher)
- Discards groups with the same reward
- Sums loss per-token, ensuring high quality of long generations
- Introduces a smooth length penalty to avoid exceeding max_length

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\left(r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\Big(r_{i,t}(\theta),1-\varepsilon_{\text{low}},1+\varepsilon_{\text{high}}\Big)\hat{A}_{i,t}\right)\right]$$

$$\text{s.t.}\quad 0 < \Big|\{o_i \mid \texttt{is\_equivalent}(a,o_i)\}\Big| < G,$$

Understanding R1-Zero-Like Training: A Critical Perspective

- Remove biased std norm, allso tweak length norm

GRPO

$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|\mathbf{o}_i|}\sum_{t=1}^{|\mathbf{o}_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right]\right\},$$

$$\text{where } \hat{A}_{i,t} = \frac{R(\mathbf{q},\mathbf{o}_i) - \text{mean}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\})}{\text{std}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\})}.$$

Dr. GRPO

GRPO Done Right (without bias)

$$\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{|\mathbf{o}_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right]\right\},$$

$$\text{where } \hat{A}_{i,t} = R(\mathbf{q},\mathbf{o}_i) - \text{mean}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\}).$$

Practical considerations

- RLAIF (synthetic markup)
- Length bias
- Reward mixing
- Switch to efficient inference (but beware numeric instability)

Conclusion

- RL helps optimize human preferences, penalize unwanted behaviour
- Allows exploration to find useful reasoning patterns (e.g. reflection)
- The field is evolving:
    - New algorithms
    - Rewards from AI
    - Verifiable rewards
    - Inference-time scaling
- Expect progress in areas with verifiable rewards

Conclusion

- RL helps optimize human preferences, penalize unwanted behaviour
- Allows exploration to find useful reasoning patterns (e.g. reflection)
- The field is evolving:
  - New algorithms
  - Rewards from AI
  - Verifiable rewards
  - Inference-time scaling
- Expect progress in areas with verifiable rewards

Questions?

Takeaways

- RLVR works for tasks with **verifiable answers**
- Expect progress for these :)
- RL can reinforce successful CoT/reasoning paths
- Leading to inference-time scaling