Reinforcement Learning for Language Model Training

Polina Tsvilodub

Prepped LLMs

RL4 LMT

Recap

RLHF & PPO

- Policy gradient methods can be used for training LMs
  - in a bandit environment
- Reward required for RL training is based on human feedback
  - can be elicited in different ways
  - used for training a reward model
- LLMs are trained with RLHF:
  - step 1: supervised fine-tuning
  - step 2: reward model learning
  - step 3: training with PPO
- PPO is an advanced policy gradient algorithm
  - uses advantage estimation and compound rewards for better training
  - often implemented as A2C
- resulting LLMs maximize reward (not next token probability!)

Policy-Gradient Methods

Language models as policies

Policy gradient estimation: $\nabla L(\theta) = \sum_{\tau} P(\tau, \theta) \nabla_\theta \log P(\tau, \theta) R(\tau) \approx \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{H} \nabla_\theta \log \pi_\theta(a_t^i \mid s_t^i) R(a_t^i)$

- policy π_θ: language model
- trajectories τ: generations from language model
- $\log \pi_\theta(a^i \mid s^i)$: log probability of a generation a^i under the language model
- $R(a_t^i)$: **reward for generation** a^i

s^i: prompt

a^i: completion

k-armed bandit environment
where k = # of prompts

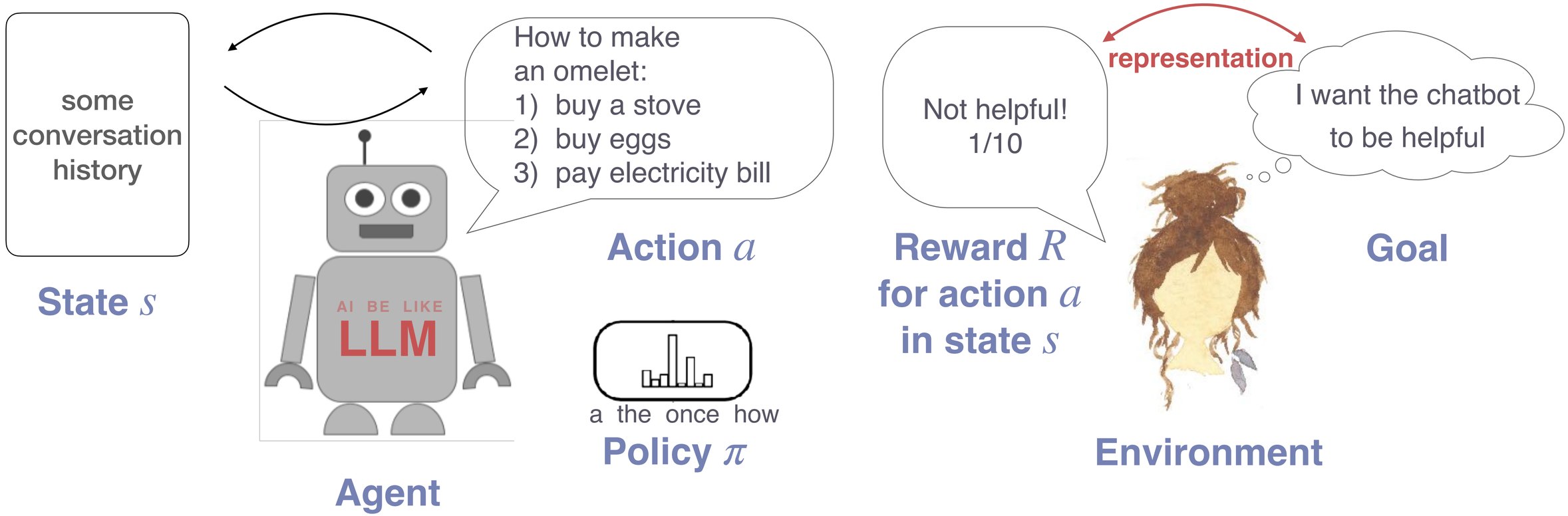
::: no episodic structure!

Sutton & Barto (2018)

Reinforcement Learning from Human Feedback

Overview

- use human judgments as a signal on what model prediction counts as a good output
- learn a **reward model representing human feedback**



Sutton & Barto (2018), Ouyang et al (2022)

Human feedback in RL

RLHF

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-3.5 with supervised learning.

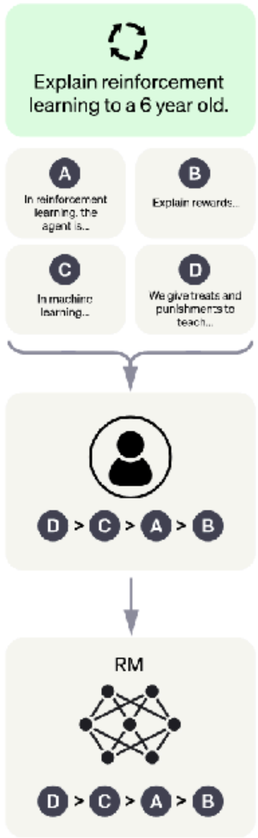
Step 2

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 the PPO reinforcement learning algorithm.

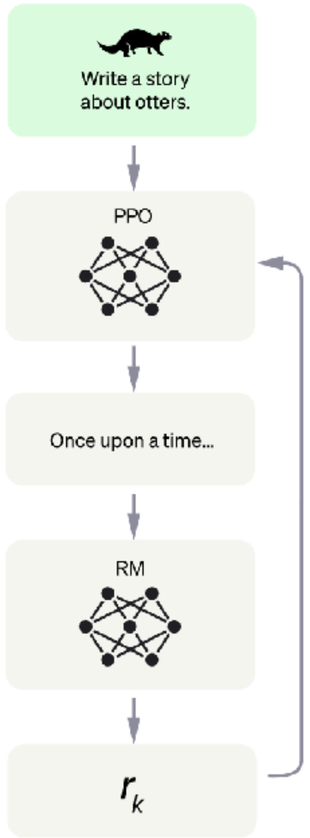
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

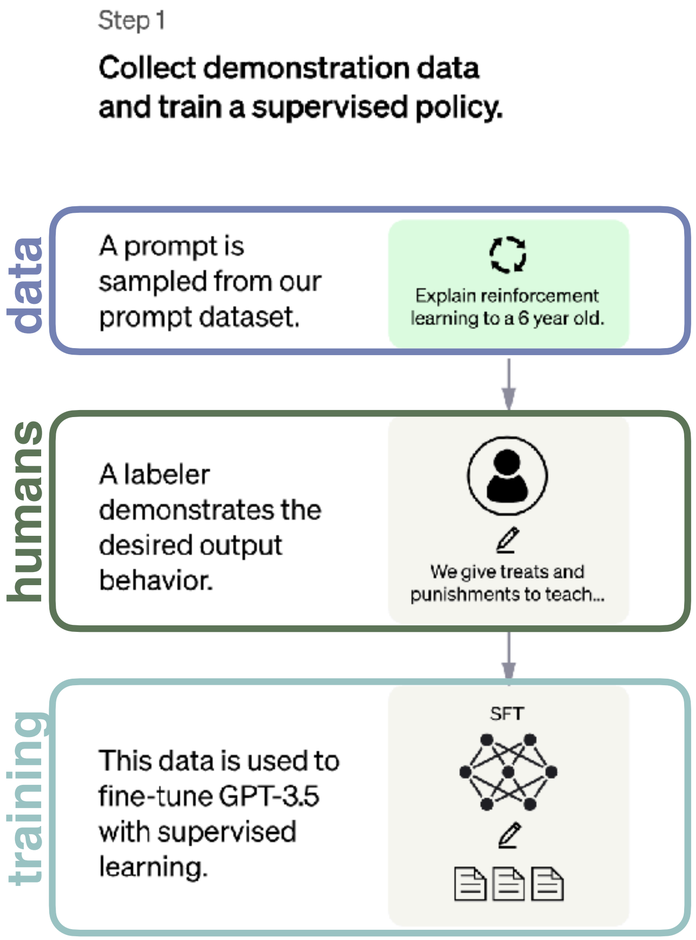
The reward is used to update the policy using PPO.



OpenAI (2022)

RLHF in practice

Step 1

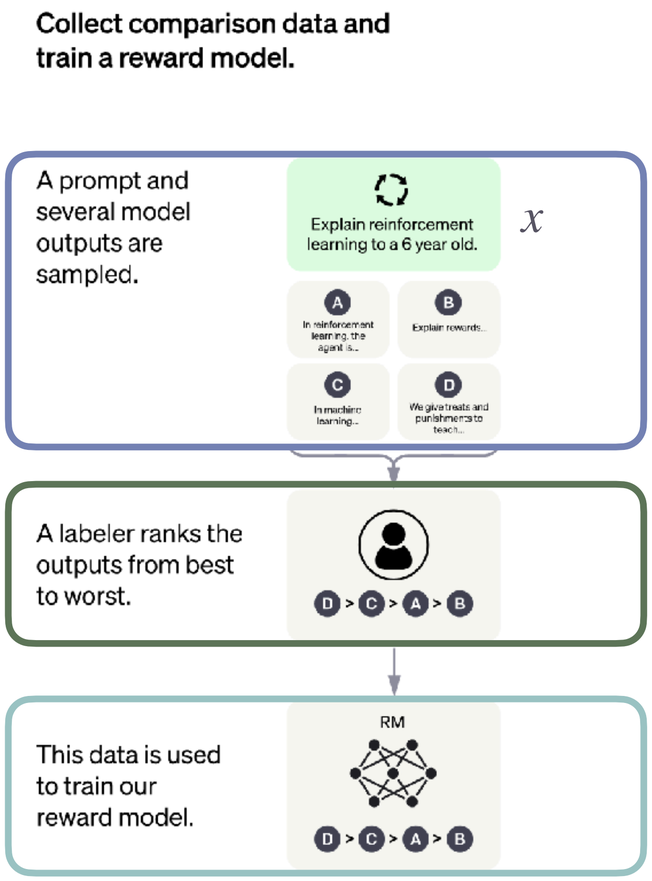


- supervised fine-tuning on a dataset of input-output demonstrations of the target task
  - pretrained model trained for a shorter time
- shifts the initial pretraining distribution $\Delta(S)$ to a task-specific distribution $\Delta'(S)$ (behavioural cloning)
  - learning about the format of task
  - producing informative rollouts from the policy for reward modelling

OpenAI (2022) and here

RLHF in practice

Step 2



- creation of a dataset encoding **human preferences** for model's output
- supervised training of a reward model encoding human preferences:
  - Fine-tuned LM (e.g., 6B GPT-3 in InstructGPT) trained to output scalar reward:

$$L(\theta) = -\frac{1}{N}\mathbb{E}_{(x,D,B)\sim D}[log\ (\sigma(\underbrace{r_\theta(x,D)}_{\text{predicted reward for response D}} - \underbrace{r_\theta(x,B)}_{\text{predicted reward for response B}}))]$$

- smart procedure for eliciting comparisons

OpenAI (2022), Ouyang et al. (2022)

Policy-Gradient Methods

Improvements

noisy estimate of $\nabla L(\theta)$

$$\nabla L(\theta_t) \propto \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i)]$$

- Introducing an **advantage**: $\hat{A} = R(a^i) - b$

  $L_\theta = \mathbb{E}[\hat{A}\ log\ \pi_\theta(a^i \mid s^i)]$

  Baseline b: e.g., constant, average, or learned state value

- Introducing a **surrogate objective / loss**: ratio $r(\theta) = \dfrac{\pi_\theta(a^i \mid s^i)}{\pi_{\theta\_old}(a^i \mid s^i)}$

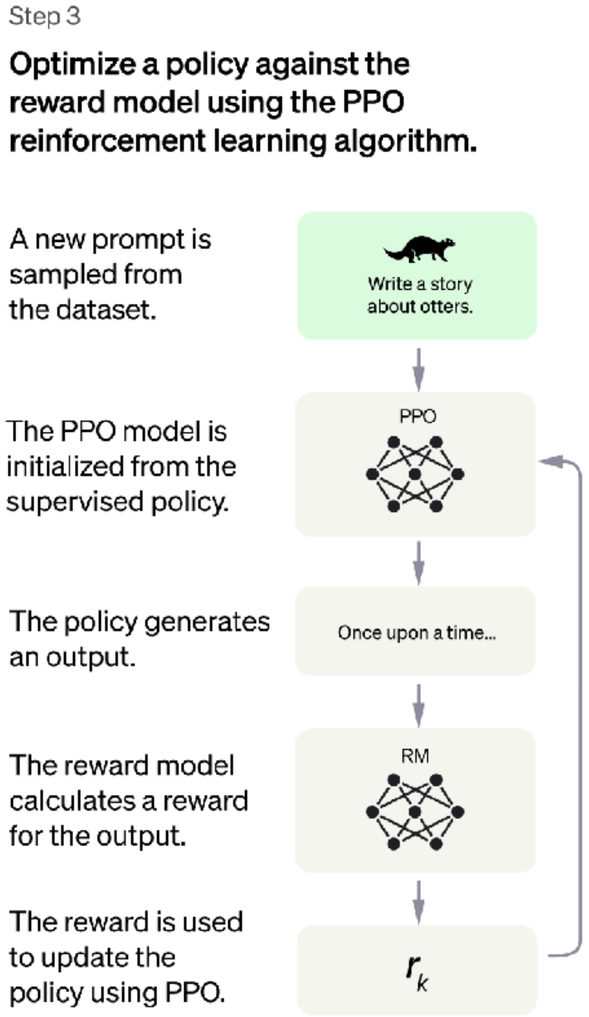
$$\mathbb{E}[\nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i)] \rightarrow \mathbb{E}[\frac{\nabla_\theta \pi_\theta(a^i \mid s^i)}{\pi_{\theta_{old}}(a^i \mid s^i)} R(a^i)]$$

"New policy shouldn't be too different from old policy"

Schulman et al. (2015), Schulman et al. (2017)

RLHF in practice

Step 3 details



- Reward objective can be complex:

Maximized reward objective

KL penalty

REWARD

Learned RL policy

$$obj(\phi) = \mathbb{E}_{(x,y)\sim D_{\pi\_RL}}[R_\theta(x,y) - \beta \log(\pi_{RL}(y\,|\,x)/\pi_{SFT}(y\,|\,x))] + \gamma \mathbb{E}_{x\sim D\_pretrain}[log(\pi_{RL}(x))]$$

Pretraining gradients

Fine-tuned LM

Pretraining distribution

for prompt x and completion y

- PPO optimization:

$$L^{CLIP}(\theta) = \mathbb{E}[min(r(\theta)\hat{A}, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A})]$$

$$\frac{\pi_{RL}(y\,|\,x)}{\pi_{RL\_old}(y\,|\,x)} \qquad obj(\phi) - \hat{q}(x,y)$$

where $\hat{q}(x,y)$ initialised from $R_\theta(x,y)$

Ouyang et al. (2022), Stiennon et al. (2022)

Core LLM

- trained on **language modeling objective**
  - predict the next word

"Here is a fragment of text … According to your **knowledge of the statistics of human language**, what words are likely to come next?

Shanahan (2022)

Prepped LLM

- trained on **usefulness objective**
  - produce text that satisfies user goals

"Here is a fragment of text … According to your **reward-based conditioning**, what words are likely to trigger positive feedback?"

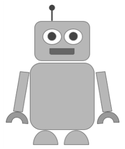
Human feedback

Limitations

- there are different ways to elicit human judgements for a preference RM
  - comparisons
  - absolute ratings
  - feedback
  - …
- you get what you ask for: necessity of detailed & complicated instructions
- annotations might be biased
  - selecting annotator sample is difficult
  - individual annotators can add malicious data
- easy to overlook mistakes
- difficult to evaluate complex tasks
- …

Casper et al. (2023)

Taking stock

			coming up with own projects	💎
		learning “what?”	learning “how?”	
				
START		mastering basic concepts		

Schedule

preliminary

session	date	topic
1	October 18th	intro & recap of LLMs
2	October 25th	LLMs & intro to RL
	November 1st	**holiday**
3	November 8th	RL: part 2
4	November 15th	RL: part 3
5	November 22nd	LLMs & RL studies
6	November 29th	Behavioral effects of RL
7	December 6th	Representational effects of RL
8	December 13th	More advanced evaluations of LMs
9	December 20th	Experiments in RL environments
10	January 10th	Social implications
11	January 17th	Limitations of RL for LM training
12	January 24th	Zooming out
13	January 31st	**final session**

SOTA models trained with RL

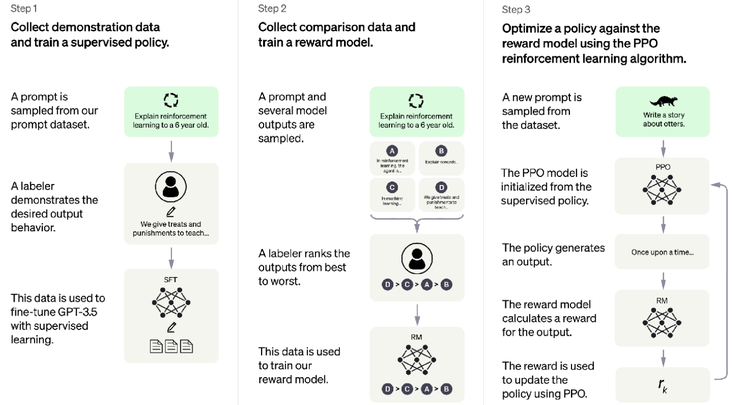
GPT-3

OpenAI

- 300B tokens
  - CC, WebText2, Books1-2, Wikipedia
- 0.1B - 175B parameters
- decoder-only transformer with a BPE tokenizer
  - context window of 2048 tokens

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

- pretraining on LM for 0.5-3 epochs
- **+ fine-tuning with RLHF = GPT-3.5**



Brown et al. (2020)

InstructGPT

OpenAI

continuous iteration

π_θ: 175B GPT-3

13k samples

16 epochs

CE loss

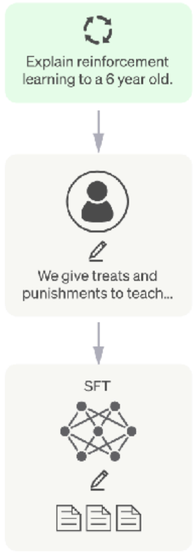
Step 1

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-3.5 with supervised learning.



R: 6B GPT-3

33k samples

4-9 samples ranked

1 epoch

preference log P maximisation

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.

π_θ: 175B GPT-3

31k samples

often high τ

256k episodes

Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

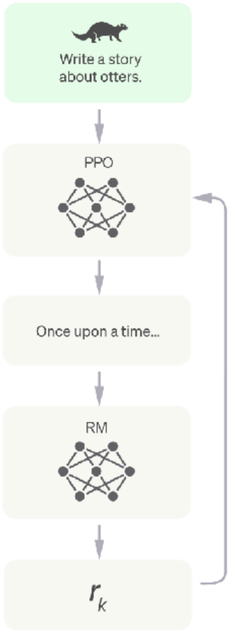
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

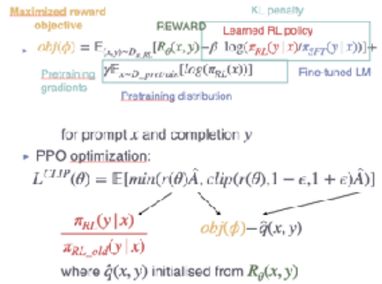
The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



- smaller batch sizes
- quite low LR
- low KL coefficients

$$L(\theta) = -\frac{1}{N}\mathbb{E}_{(x,D,B)\sim D}[log\ (\sigma(r_\theta(x,D) - r_\theta(x,B)))]$$



Ouyang et al. (2022)

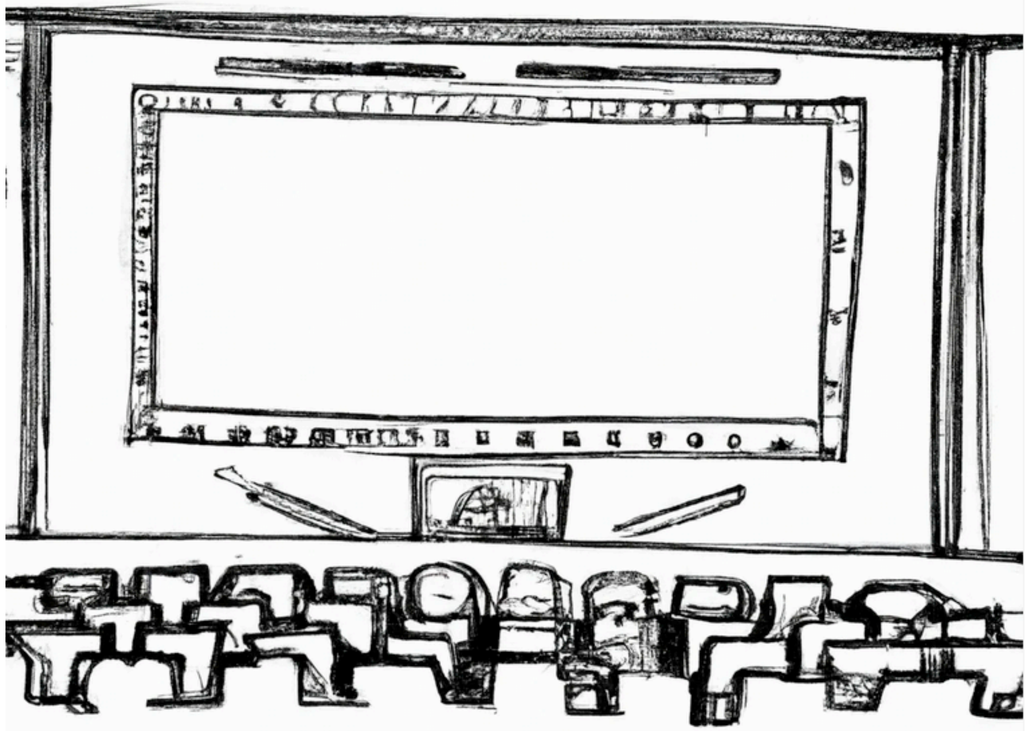
Presentations

Your job

During the presentations, think about the following questions:

- 3 keywords for your favourite aspects of the paper
- 3 keywords for your least favourite aspects of the paper
- would you be able to re-apply (conceptually)?

We will collect with a Mentimeter on both talks after!



Citation (2002), Citation 2 (2050)

Mentimeter

Code: 3927 2787