JARNA ITAPA

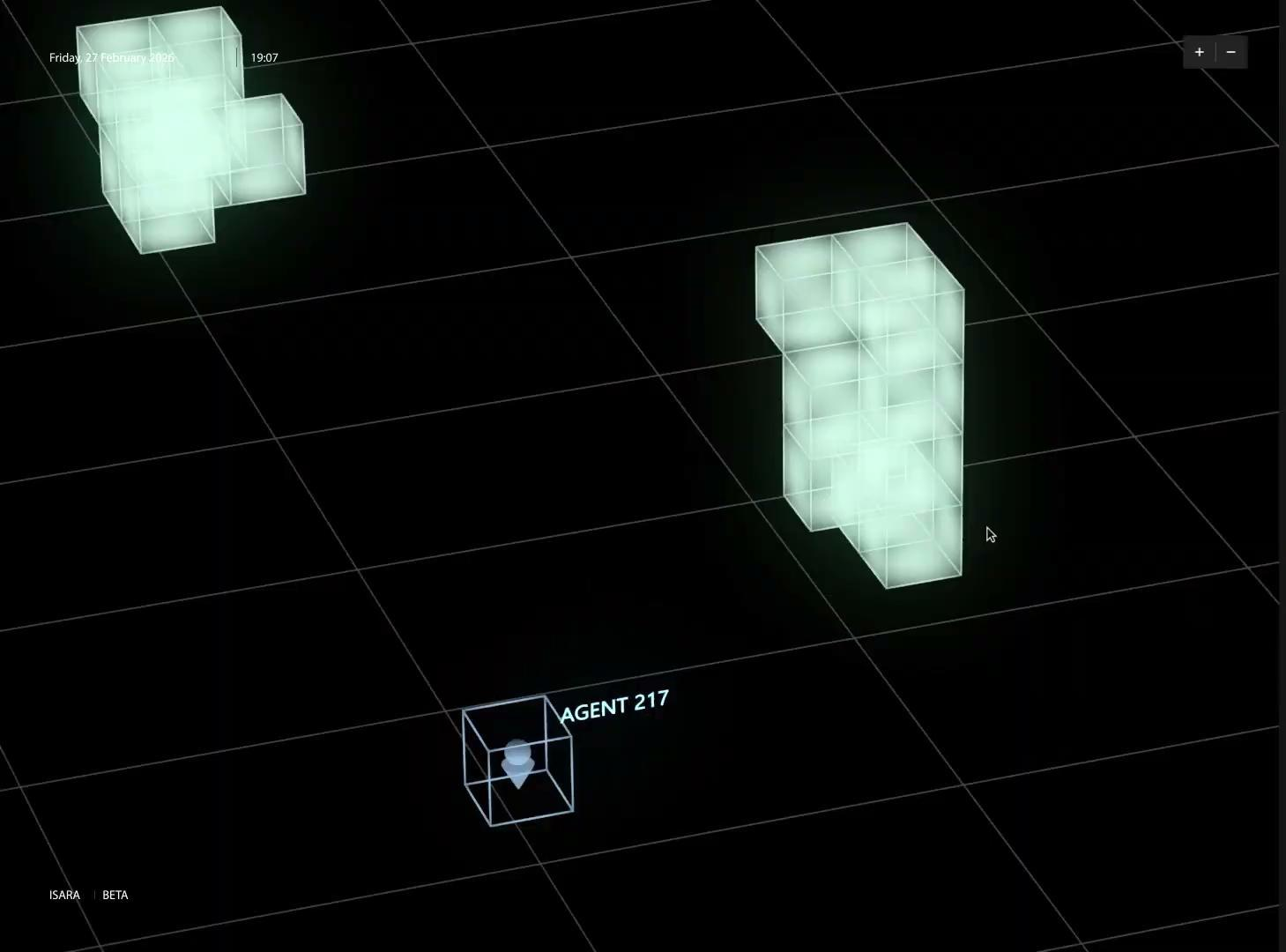
Agentická AI – čo prinesie?

Accelerating Nash learning from human feedback via Mirror Prox

Michal Valko · 18. 6. 2026

pod kapotou

itapa
inno.digi.tech

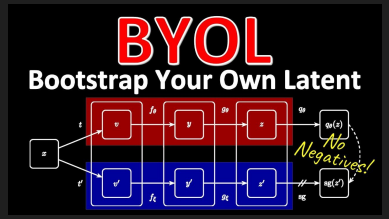

Friday, 27 February 2026
19:07
+
−
AGENT 217
ISARA
BETA

… after 5 + 5 years

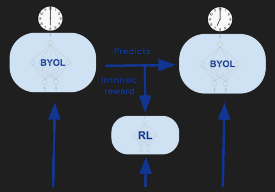
world models rebirth

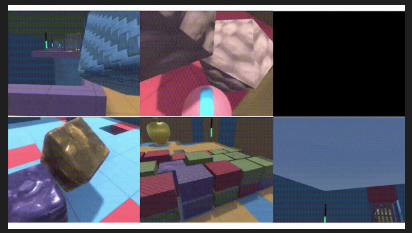
https://gemini.google.com/

https://ai.meta.com/



Gemini





Collaborators

Yunhao Tang, Rémi Munos, Liang Tan, Damien Allonsius, Dhruv Mahajan, Tyler Lu, Rui Hou, Pierre Ménard, Daniele Calandriello, Bilal Piot, Mark Rowland, Zeyu Zheng, Mohammad Gheshlaghi Azar, Matthieu Geist, Daniel Zhaohan Guo, Bernardo Ávila Pires, Daniil Tiapkin, Alexey Naumov, Pierre Harvey Richemond, Denis Belomestny, Tadashi Kozuno, Yuan Cao, Eugene Tarassov, Yong Cheng, Will Dabney, Côme Fiegel, Vianney Perchet, Wenhao Yang, Nino Vieillard, Toshinori Kitamura, Jincheng Mei, Olivier Pietquin, Csaba Szepesvári, Aniket Didolkar, Anirudh Goyal, Nan Rosemary Ke, Siyuan Guo, Danilo Rezende, Yoshua Bengio, Michael Mozer, Sanjeev Arora, Andrea Michi, Marco Selvi, Sertan Girgin, Nikola Momchev, Olivier Bachem, Daniel J. Mankowitz, Doina Precup, Charline Le Lan, Tianqi Liu, Rishabh Joshi, Tianlin Liu, Shangmin Guo, Leonardo Bianco, Quentin Berthet, Felipe Llinares, Jessica Hoffmann, Lucas Dixon, Mathieu Blondel, Oliver Groth, Éric Moulines, Pierre Perrault

Talk based on the following papers

Nash learning from human feedback**, ICML 2024**
Human alignment of large language models through online preference optimisation, **ICML 2024**
A unified approach to offline alignment, **ICML 2024**
Decoding-time realignment of language models, **ICML 2024**
Metacognitive capabilities of LLMs for mathematical problem solving, **NeurIPS 2024**
Accelerating Nash learning from human feedback via Mirror Prox **2026**

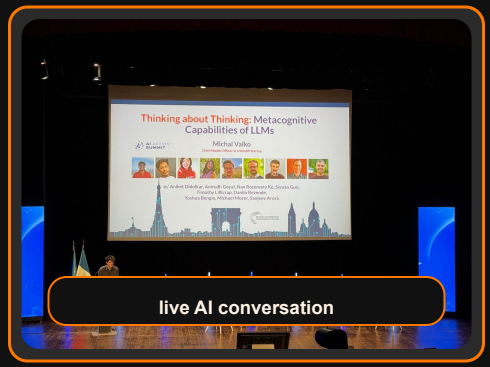
Understanding the performance gap between online and offline alignment algorithms**, arxiv**
Demonstration-regularized RL, **ICLR 2024**
A general theoretical paradigm to understand learning from human preferences, **AISTATS 2024**
RL-fine tuning LLMs from on- and off-policy data with a single algorithm **2025**
Preference optimization with multi-sample comparisons **2025**

Fast rates for maximum entropy exploration, **ICML 2023,**
Adapting to game trees in zero-sum imperfect information games, **ICML 2023,** best paper
Optimal design for reward modeling in RLHF **2025**

01

Plan for June 17

Algorithmic alignment
Pairwise preference over ELO scores
Better than best response



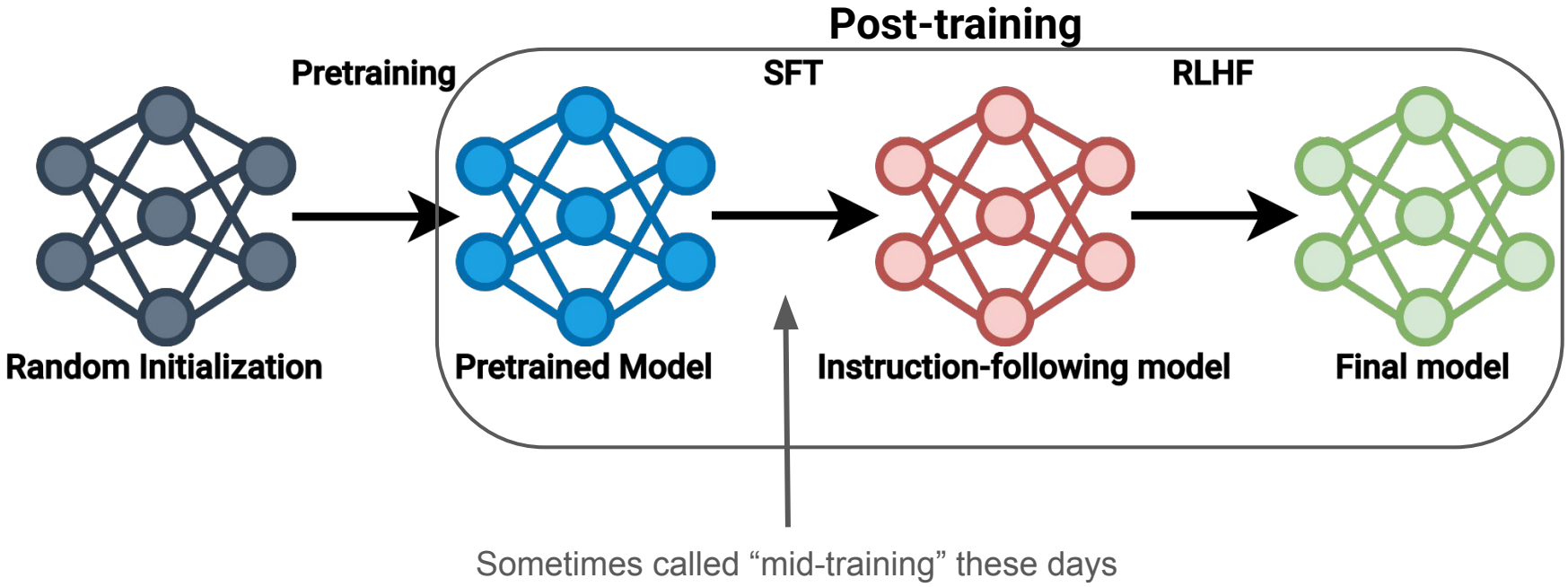
live AI conversation

Cieľ: aby po 25 minútach nikto nepovedal iba: pekné farby.


MACHINES
CAN THINK

Large Model Training Pipeline

MACHINES CAN THINK

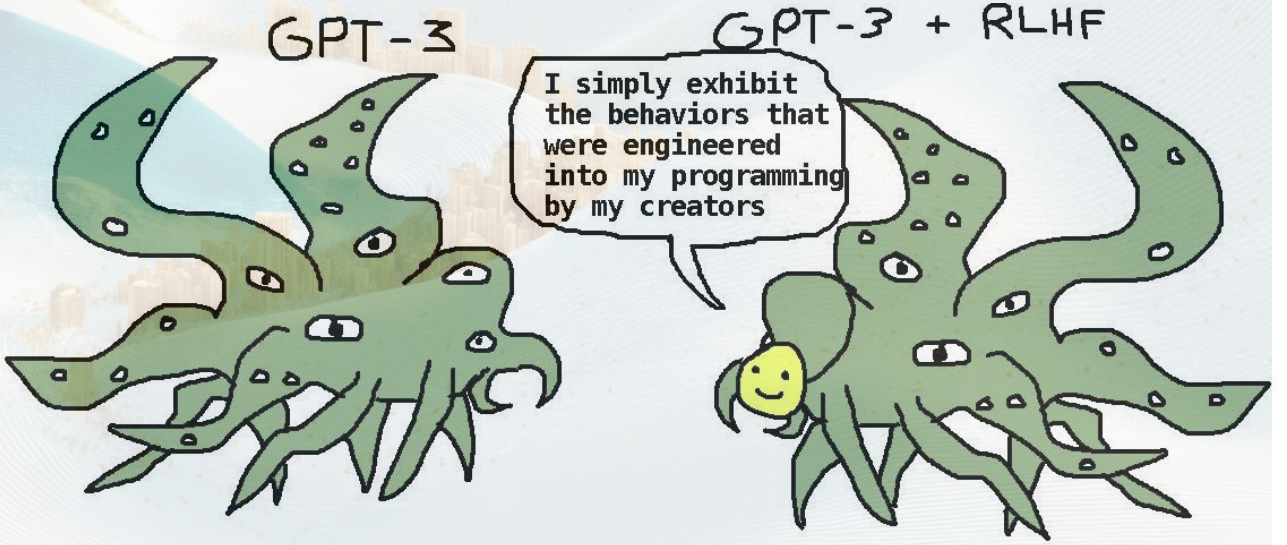


Reinforcement Learning from Human Feedback (RLHF)

Given: pretrained model that can perform basic instruction following
Goal: "align" this model with human preferences.

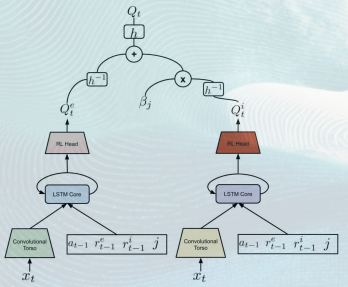
Implementation:

1. Train reward model ***r*** from human preferences;
2. Use RL to maximize reward ***r***;

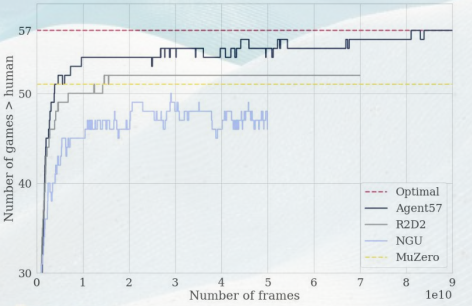


Source: https://twitter.com/natfriedman/status/1625850766039842824

Atari - Gemini - Llama - RL reality check



1e10
training steps

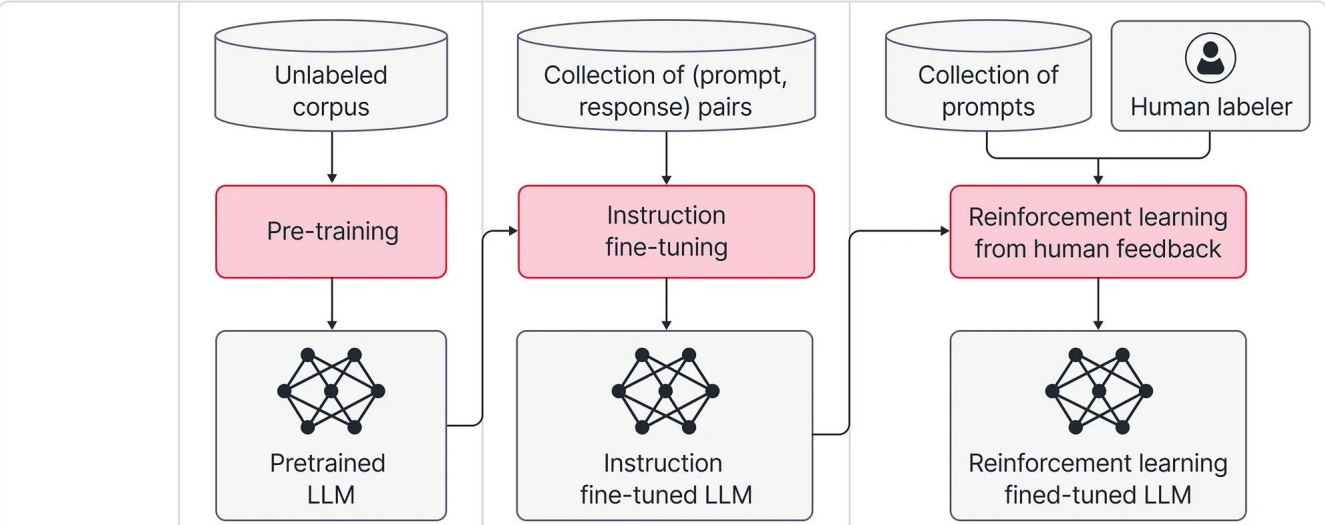


> 400 B

1e3-1e6
training steps

Gemini

Traditional three-phase recipe



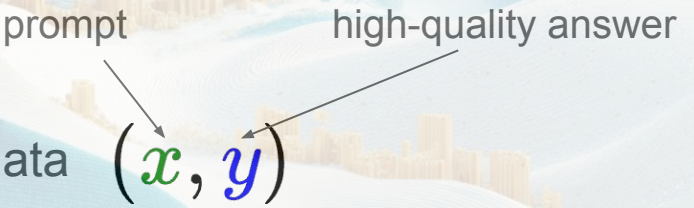
Dataset size	x00 billions to 1.x trillion tokens	~xk to x0k (prompt, response)	~x0k prompt
Example of models	GPT-3, LLaMA, Falcon, BLOOM	Dolly-v2, Falcon-instruct	Claude, GPT-4, ChatGPT
	(a)Pre-training	(b) Instruction fine-tuning	(c) Reinforcement learning from human feedback

What data do we have?

Main secret of RLHF: allows to efficiently use data that previously was impossible to use!

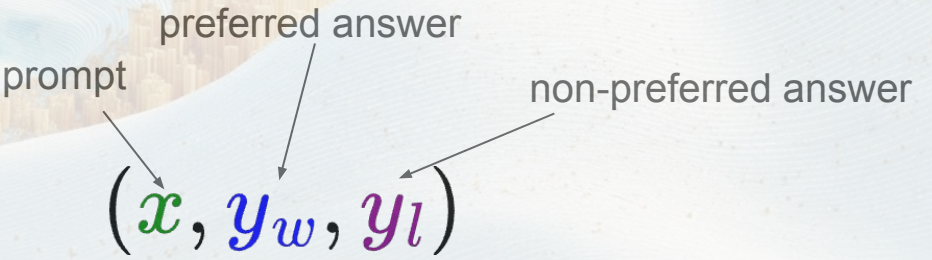
For SFT:

- High-quality expert data
- Expensive to scale!



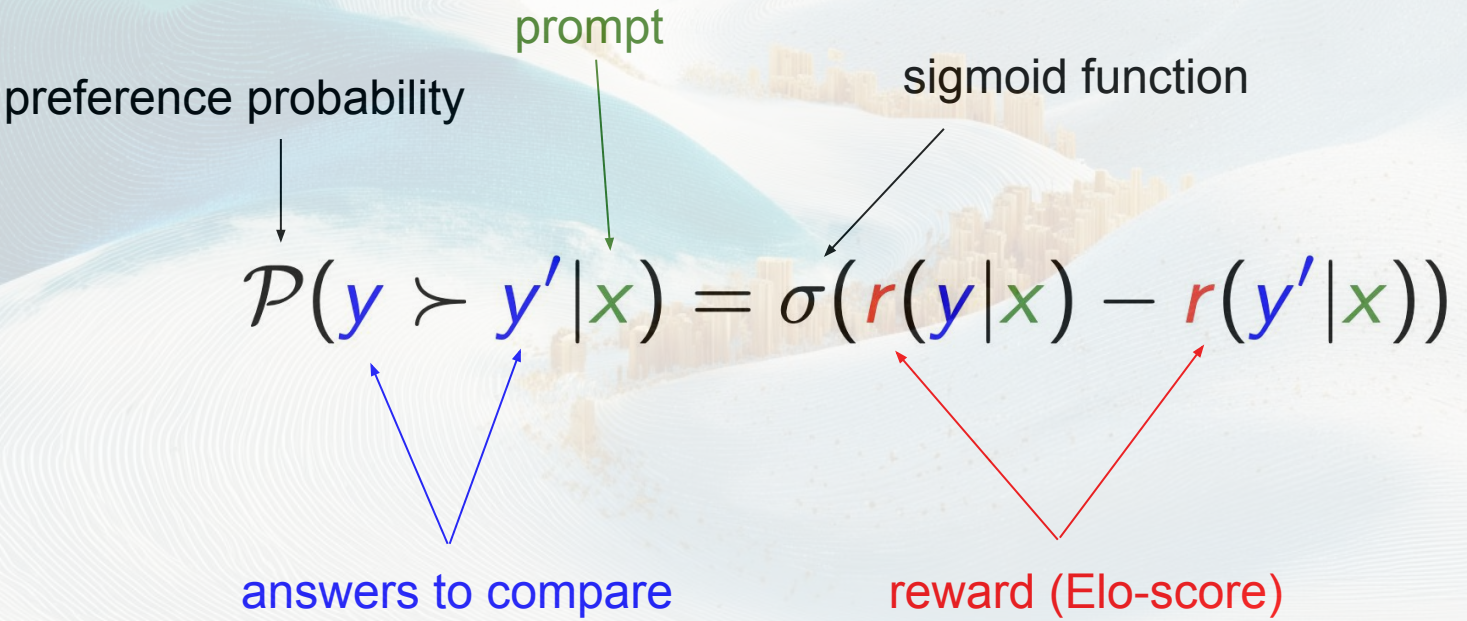
For RLHF:

- Mid-to-low quality **preference** data
- Very cheap to collect:
  generate two answers and ask a user to select the best



How to extract rewards?

Question. Which assumptions do we use to train reward model in RLHF?
Answer: Bradley-Terry model



Reward modeling

Approach:

- Collect a dataset of triples (prompt, response 1, response 2);
- Use human annotators to compare response 1 and response 2;
- Learn reward model by maximum likelihood:

$$\min_{\theta} \mathcal{L}(\theta) = -\mathbb{E}_{(x,y_w,y_\ell)}[\log(\sigma(r_\theta(y_w|x) - r_\theta(y_\ell|x))]$$

response-winner response-loser

Main idea: comparison data is MUCH cheaper than demonstrations!

(Naive) Reinforcement Learning from Human Feedback

Given: reward function

Goal: find a policy that maximizes its expectation

policy i.e. probabilities of next predictions by LLM

$$\max_{\psi} \mathcal{L}_{\mathrm{RL}}(\psi) = \mathbb{E}_{y \sim \pi_{\psi}(y|x)} \left[r_{\hat{\theta}}(y|x) \right]$$

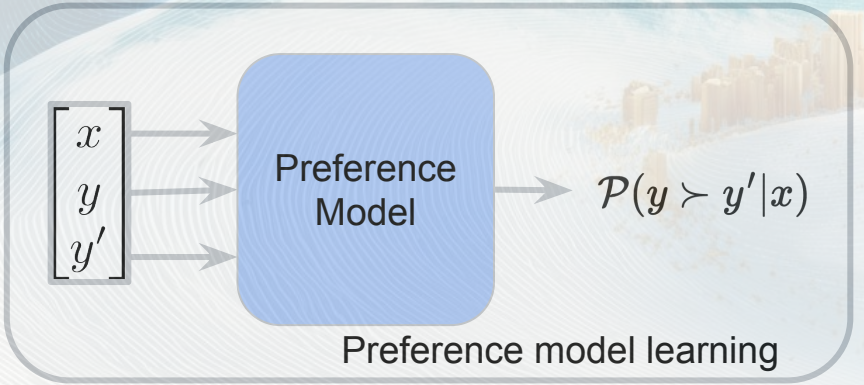
Could be done by any RL method, i.e. Proximal Policy Optimization (PPO), REINFORCE Leave-one-Out (RLOO), Shifted Q-learning (ShiQ).

Q: What are possible problems here?

Pairwise preference over ELO scores

$$\mathcal{P}_{BT}(y \succ y'|x) \overset{\text{def}}{=} \sigma\left(r_\theta(x,y) - r_\theta(x,y')\right)$$

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



$$\mathcal{L}_{\mathcal{P}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\left(\mathcal{P}_\theta(y_w \succ y_l|x)\right)\right]$$

$\mathcal{P}(y \succ y'|x)$

- Initialise it with a LLM prompted: "Given this prompt 'x' and two responses 'y1' and 'y2', which one do you prefer?"
- trained by SL with preference human data

0 What is RL?

What is reinforcement learning?

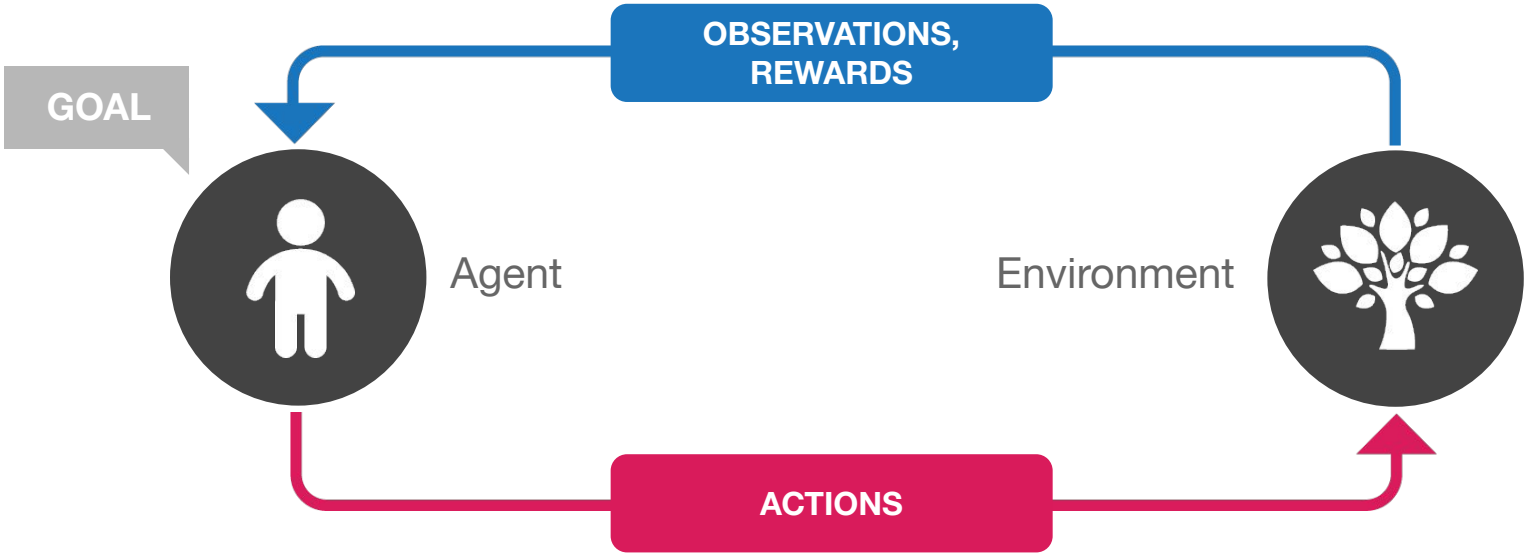
- learning by trial and error
- learning to act in an unknown, stochastic environment by maximizing some reward signal
- Example: learning to bike without a perfect knowledge of physics

Prehistory of RL

- 1900s: observation of animal behavior (e.g. Thorndike 1911 “Law of Effect”)
- 1920s: Pavlov work on conditionnal reflexes first occurrence of “reinforcement” in animal learning
- Oak and Miller 1954: first experiments on electric brain stimuli for controlling mice behavior

RL Concepts

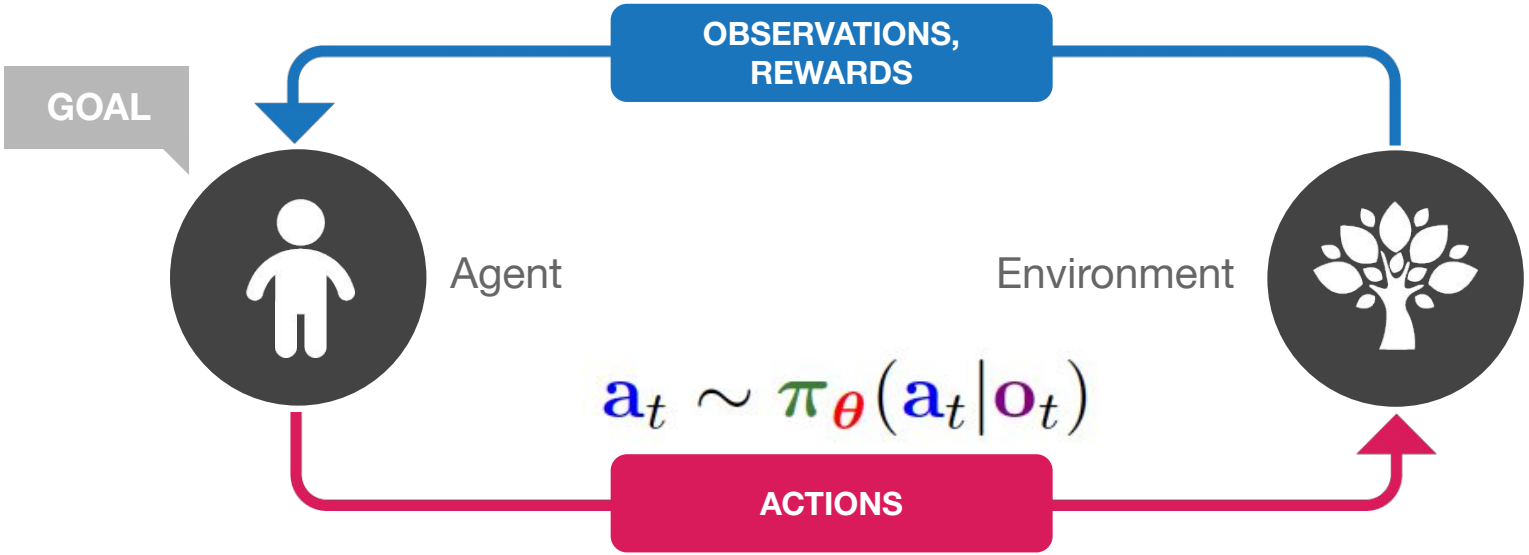
Reinforcement learning



Making good decisions by learning from experience

$$\mathbf{o}_{t+1} \sim P(\mathbf{o}_{t+1}|\mathbf{a}_t, \mathbf{o}_t)$$

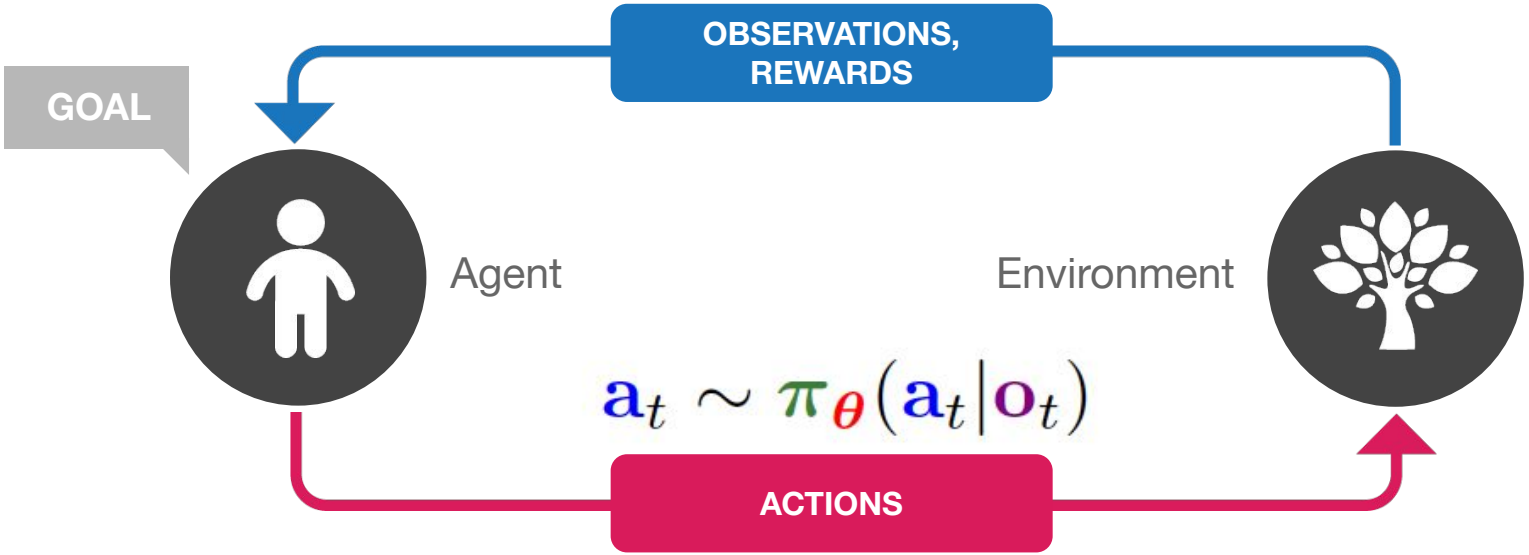
$$\mathbf{r}_t = \mathbf{r}(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$$



Making good decisions by learning from experience

$$\mathbf{o}_{t+1} \sim P(\mathbf{o}_{t+1} | \mathbf{a}_t, \mathbf{o}_t)$$

```
next_timestep = environment.step(action)
```

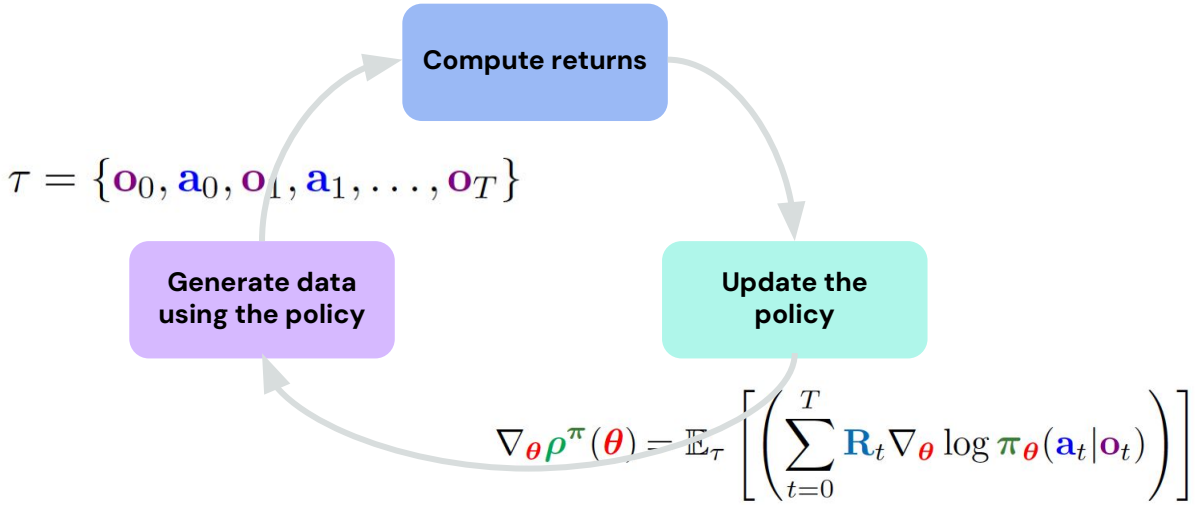



```
action = actor.policy(timestep.observation)
```

Making good decisions by learning from experience

The policy gradient

$$\mathbf{R}_t = \sum_{n=t}^{T} \mathbf{r}(\mathbf{o}_n, \mathbf{a}_n, \mathbf{o}_{n+1})$$



Best response vs. probability of winning

antisymmetric: $\mathcal{P}(y \succ y'|x) = 1 - \mathcal{P}(y' \succ y|x)$

f is a (deterministic) absolute scoring function

$$\mathcal{P}(y \succ y'|x) = \mathbb{E}_{Z\sim\nu}\left[\mathbb{I}\{f(x, y, Z) \succ f(x, y', Z)\}\right]$$

Probability of winning:

$$\mathcal{P}(\pi \succ \pi'|x) \stackrel{\text{def}}{=} \mathbb{E}_{y\sim\pi(\cdot|x), y'\sim\pi'(\cdot|x)}\left[\mathcal{P}(y \succ y'|x)\right]$$

Best response vs. probability of winning

Probability of winning

$$\mathcal{P}(\pi \succ \pi' | x) \overset{\text{def}}{=} \mathbb{E}_{y \sim \pi(\cdot|x), y' \sim \pi'(\cdot|x)} \left[\mathcal{P}(y \succ y' | x) \right]$$

Nash Equilibrium

$$\arg \max_{\pi} \min_{\pi'} \mathbb{E}_{x, y \sim \pi, y' \sim \pi'} \left[\mathcal{P}(y \succ y' | x) \right]$$

Why stray away from Bradley-Terry

1. Diverse human preferences

Example:

- 3 types of humans with respective preferences P1 , P2, P3
- Each type as has a different preference between action y1, y2, y3
- **BT** will select one action y1 deterministically
- **Nash** will selected a mixture policy proportionally

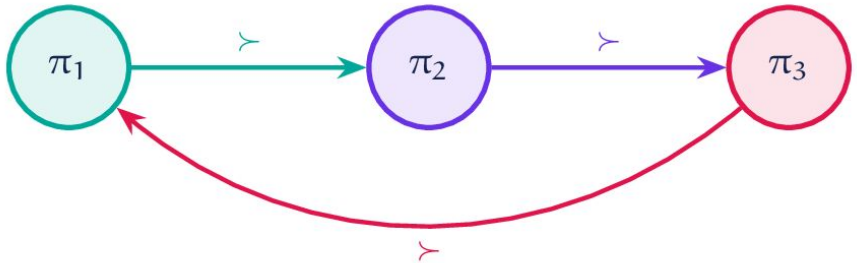
BT is also unstable: One datapoint can radically change the policy

Why stray away from Bradley–Terry

- Bradley–Terry scores each answer y with a *scalar* $s(y)$ and sets $\mathbb{P}(y \succ y') = \sigma\big(s(y) - s(y')\big)$.
- A scalar score imposes a **total order** $\Rightarrow$ preferences are forced to be **transitive**.
- But real preferences can **cycle**. (Example: non-transitive dice, Gardner 1970.)

We can construct three policies with

$$\mathbb{P}(\pi_1 \succ \pi_2) > \tfrac{1}{2}, \quad \mathbb{P}(\pi_2 \succ \pi_3) > \tfrac{1}{2}, \quad \mathbb{P}(\pi_3 \succ \pi_1) > \tfrac{1}{2}.$$



Why stray away from Bradley-Terry

3. Sensitivity to the sampling distribution

A reward model depends on the data distribution:

$$r^{\pi} \stackrel{\text{def}}{=} \arg\max_{r(\cdot,\cdot)} \mathbb{E}_{\substack{x \sim \rho \\ y, y' \sim \pi(\cdot|x) \\ Z \sim \nu}} \left[\log\left(\sigma(r(x, y_w^Z) - r(x, y_l^Z))\right)\right]$$

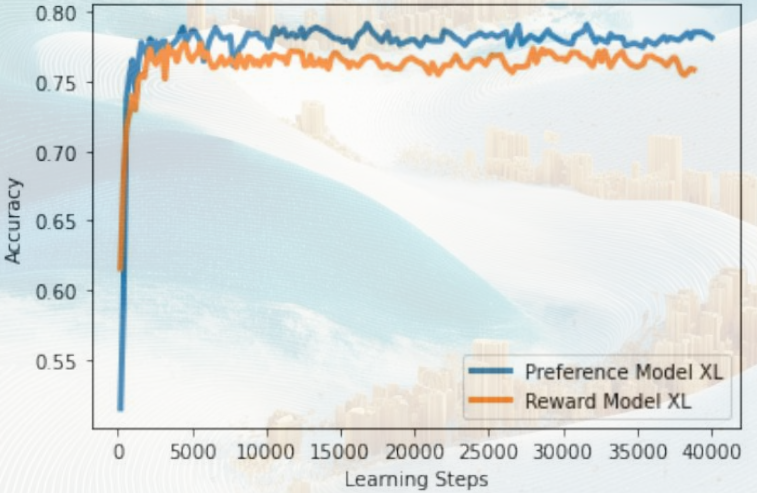
Whereas a preference model essentially* does not:

$$\mathcal{P}^* \stackrel{\text{def}}{=} \arg\max_{\mathcal{P}(\cdot \succ \cdot|\cdot)} \mathbb{E}_{\substack{x \sim \rho \\ y \sim \pi(\cdot|x) \\ y' \sim \pi'(\cdot|x) \\ Z \sim \nu}} \left[\log \mathcal{P}(y_w^Z \succ y_l^Z|x)\right]$$

essentially* = infinite amount of data, no approximation

Why stray away from Bradley-Terry

4. Data comes from human pairwise preferences



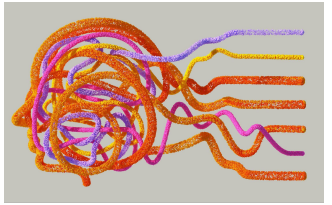
Empirical argument: **fits better**

Solving for the Nash

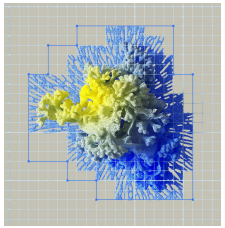
From IXOMD to NashLLMs

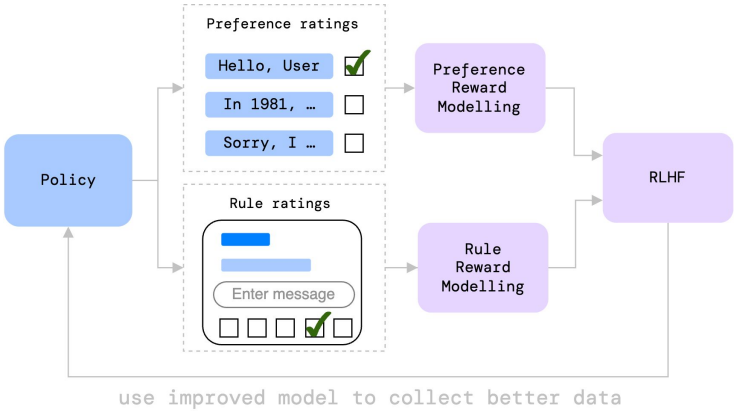
games

trees

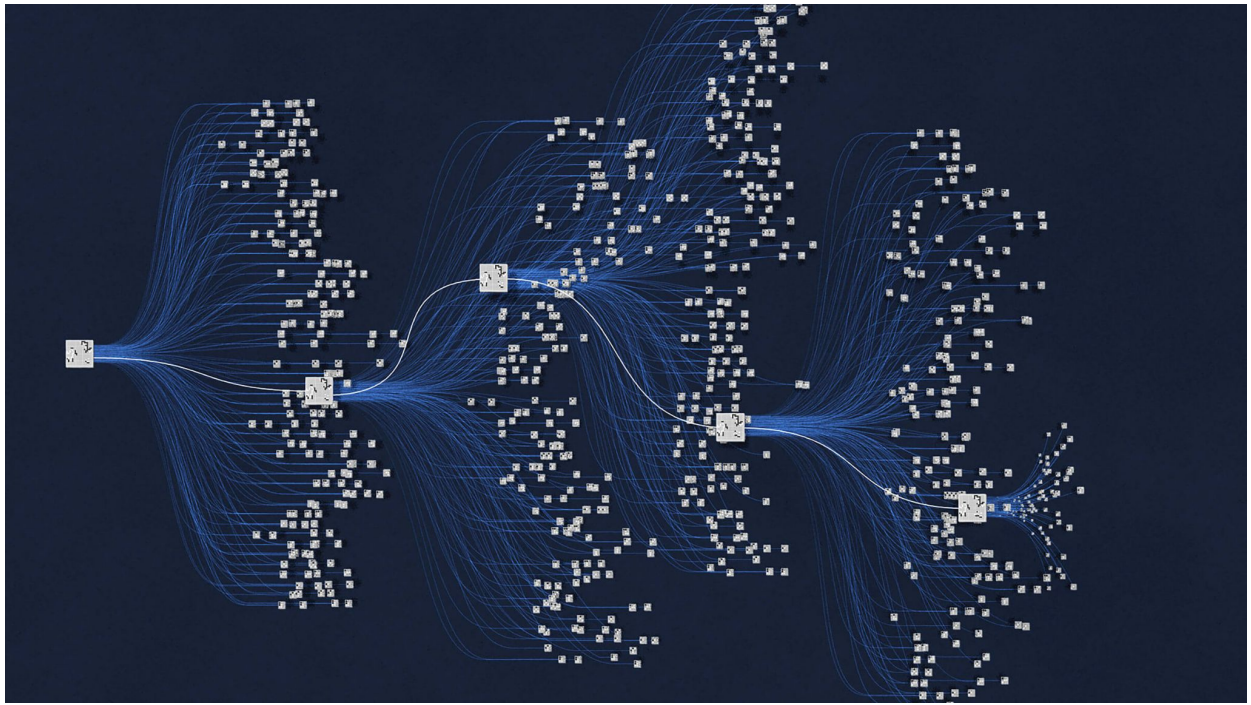


self-improvement





Solving **imperfect information games**



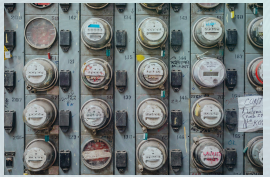
Scale

replay buffer

computation only
along trajectories

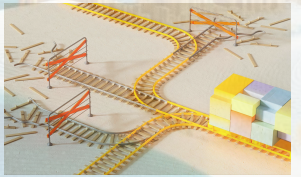
A recipe for **success** in optimal play

Self-play with **follow-the-regularized leader**



Loss estimate

We do not have full information



Regularizer

We can stray away

Balancing

Spent effort where it matters

Magic Sauce

Craft the the interplay with no tree

Quicky mention the first three ingredients

Focus on the magic

10 years to the solution

IX - Implicit e**X**ploration
Kocák et. al 2014
Valko et al 2016,
Lattimore and Szepesvári 2020

10 years to the solution

Regularised graph cuts (2008)

$$\ell_u = (L_{uu} + \gamma_g I)^{-1} W_{ul} \ell_l$$

Kernel bandits (2013)

$$\hat{\mu}_{a,t} = k_{x_{a,t},t}^{\intercal} (K_t + \gamma I)^{-1} y_t$$

Implicit exploration (2014)

$$\hat{\ell}_{t,i} = \frac{\ell_{t,i}}{o_{t,i} + \gamma_t} \mathbb{1}_{\{(I_t \to i) \in G_t\}}$$

Spectral bandits (2014)

$$\log \frac{|\mathbf{V}_T|}{|\mathbf{\Lambda}|} \leq \max \sum_{i=1}^{N} \log \left(1 + \frac{t_i}{\lambda_i}\right)$$

Ridge leverage scores (2016)

$$\tau_{i,n}(\gamma) = \sum_{j=1}^{n} \frac{\lambda_j}{\lambda_j + \gamma} [\mathbf{U}]_{i,j}^2$$

Obit: Michal Valko died in 20xx.
Whenever he saw a fraction a/b, he was known to quickly change it to a/(b+γ).

10 years to the solution

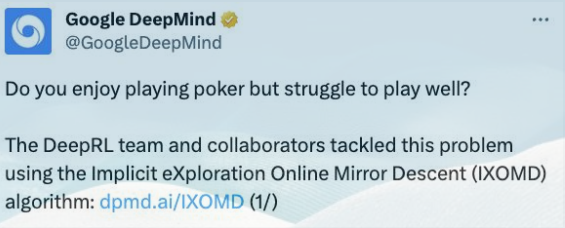
IX - Implicit e**X**ploration
Kocák et. al 2014
Valko et al 2016,
Lattimore and Szepesvári 2020

10 years to the solution

10 years to the solution

IX - Implicit eXploration
Kocák et. al 2014
Valko et al 2016,
Lattimore and Szepesvári 2020

Monte-Carlo CFR
Lanctot et al. 2019



2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 ...

Dilated entropy
Kroer et al. 2015

High-probability
Neu 2015

First-order methods
Kroer et al. 2018

1st slow rate results
Farina and Sandholm (2020-21)

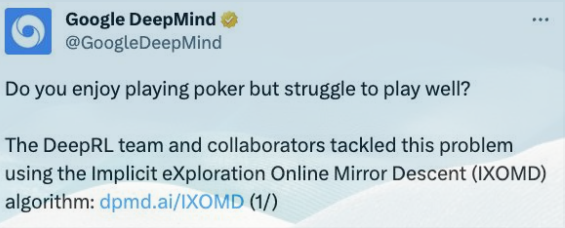
Balanced strategy
Farina et al. Sa (2020)

Using **IX** for unknown transition
Jin et al. 2020

10 years to the solution

IX - Implicit eXploration
Kocák et. al 2014
Valko et al 2016,
Lattimore and Szepesvári 2020

Monte-Carlo CFR
Lanctot et al. 2019



NeurIPS 2021 - not optimal
the complexity was multiplied by the huge size of the tree

2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 ...

Dilated entropy
Kroer et al. 2015

High-probability
Neu 2015

First-order methods
Kroer et al. 2018

1st slow rate results
Farina and Sandholm (2020-21)

Balanced strategy
Farina et al. Sa (2020)

Using **IX** for unknown transition
Jin et al. 2020

Back to work

10 years to the solution

The quest for the **magic ingredient**

Algorithm	Sample complexity	Structure-free
IXOMD	$\widetilde{\mathcal{O}}(H^2(\lvert\mathcal{X}\rvert^2\lvert\mathcal{A}\rvert+\lvert\mathcal{Y}\rvert^2\lvert\mathcal{B}\rvert)/\epsilon^2)$	✓
BalancedOMD	$\widetilde{\mathcal{O}}(H^3(\lvert\mathcal{X}\rvert\lvert\mathcal{A}\rvert+\lvert\mathcal{Y}\rvert\lvert\mathcal{B}\rvert/\epsilon^2))$	✗
BalancedFTRL	$\widetilde{\mathcal{O}}(H(\lvert\mathcal{X}\rvert\lvert\mathcal{A}\rvert+\lvert\mathcal{Y}\rvert\lvert\mathcal{B}\rvert)/\epsilon^2)$	✗
AdaptiveFTRL	$\widetilde{\mathcal{O}}(H^2(\lvert\mathcal{X}\rvert\lvert\mathcal{A}\rvert+\lvert\mathcal{Y}\rvert\lvert\mathcal{B}\rvert)/\epsilon^2)$	✓
Lower bound	$\widetilde{\Omega}(H(\lvert\mathcal{X}\rvert\lvert\mathcal{A}\rvert+\lvert\mathcal{Y}\rvert\lvert\mathcal{B}\rvert)/\epsilon^2)$	-

✓ If we look at the tree **only one single time** we can get the last ingredient

We could calculate a **balanced** policy which samples w.p. **1/|all tree descendants|**

✗ Peaking at the tree once, when the tree has $\mathbf{10^{1000}}$ nodes - not really possible.

In 2014, we merely pretended to be exploring.
... in 2023, we **merely pretend to see the tree**

Guess the game from the moves of the opponent.

10 years to the solution

10 years to the solution

IX - Implicit eXploration
Kocák et. al 2014
Valko et al 2016,
Lattimore and Szepesvári 2020

Monte-Carlo CFR
Lanctot et al. 2019

Google DeepMind
@GoogleDeepMind

Do you enjoy playing poker but struggle to play well?

The DeepRL team and collaborators tackled this problem using the Implicit eXploration Online Mirror Descent (IXOMD) algorithm: dpmd.ai/IXOMD (1/)

Peeking once is enough
Bai et. al 2022

Regularization for Stratego
Perolat, de Vylder, et. al 2022

2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 ...

Dilated entropy
Kroer et al. 2015

High-probability
Neu 2015

First-order methods
Kroer et al. 2018

1st slow rate results
Farina and Sandholm (2020-21)

Balanced strategy
Farina et al. Sa (2020)

Using **IX** for unknown transition
Jin et al. 2020

Demis Hassabis
@demishassabis

Congrats to @GoogleDeepMind's Remi Munos, @misovalko, & team on the Outstanding Paper Award at @ICMLConf! "Adapting to game trees in zero-sum imperfect information games" helps answer: how do you make the best move in a

NashMD in LLMs

🌳 Full NashMD asks for **best-response** (BR) in every step

$$\pi_{t+1} = \arg\max_{\pi} \left[\eta \mathcal{P}(\pi \succ \pi_t) - \mathrm{KL}(\pi, \pi_t^{\mu})\right]$$

🌳 **NashMD**-PG: follow the gradient - note the difference in the KL!

$$\nabla_\theta \log \pi_\theta(y|x) \left[\mathcal{P}(y \succ y'|x) - \frac{1}{2}\right] - \tau \nabla_\theta \mathrm{KL}(\pi_\theta(\cdot|x), \pi_{ref}(\cdot|x))$$

🌳 **y** is generated from the current policy
🌳 **y'** is generated from a (geometric) mixture between the current policy and a past checkpoint (such as the initial SFT policy):

$$y' \sim \pi_\theta^\beta(\cdot|x) \propto (\pi_\theta(\cdot|x))^{1-\beta} (\pi_{ref}(\cdot|x))^\beta$$

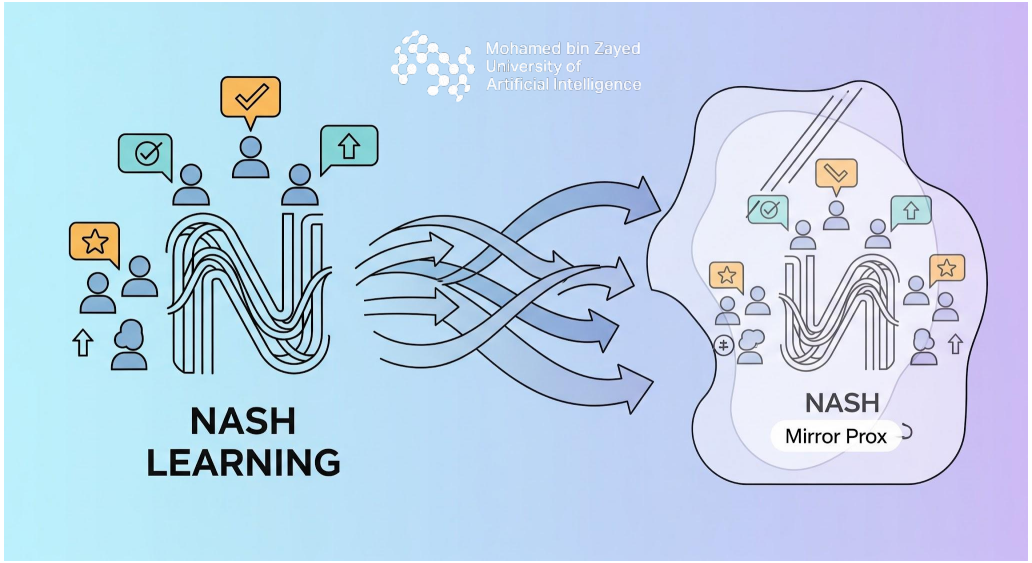
Experiment on a text summarizing task

Train preference model (T5X-L models) on TL;DR database, then compute the Nash using several methods: Self-Play, Nash-MD, Nash-EMA, Best-Response.

Table 1. PaLM 2 preference $\mathcal{P}^*(\pi_c \succ \pi_r)$ model between column policy π_c against row policy π_r.

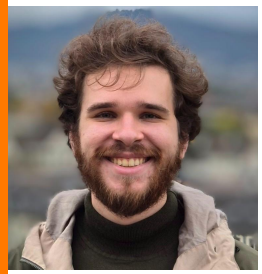
$\mathcal{P}^*$	SFT	RLHF	SP	**MD1**	MD2	MD3	MD4	MD5	MD6	BR	EMA1	EMA2	EMA1*	EMA2*
SFT	0.500	0.990	0.983	**0.982**	0.989	0.987	0.985	0.982	0.965	0.943	0.970	0.961	0.977	0.980
RLHF	0.010	0.500	**0.489**	**0.598**	**0.519**	**0.561**	**0.501**	**0.436**	**0.284**	**0.148**	0.468	0.320	0.477	0.510
SP	0.017	0.511	0.500	**0.592**	0.504	0.545	0.499	0.451	0.310	0.211	0.445	0.362	0.464	0.488
MD1	0.018	0.402	0.408	**0.500**	0.425	0.470	0.369	0.362	0.238	0.163	0.391	0.270	0.400	0.447
MD2	0.011	0.481	0.496	**0.575**	0.500	0.513	0.491	0.434	0.298	0.196	0.460	0.351	0.430	0.496
MD3	0.013	0.439	0.455	**0.530**	0.487	0.500	0.484	0.408	0.273	0.187	0.429	0.323	0.413	0.472
MD4	0.015	0.499	0.501	**0.631**	0.509	0.516	0.500	0.428	0.265	0.161	0.468	0.358	0.437	0.503
MD5	0.018	0.564	0.549	**0.638**	0.566	0.592	0.572	0.500	0.329	0.210	0.532	0.389	0.518	0.539
MD6	0.035	0.716	0.690	**0.762**	0.702	0.727	0.735	0.671	0.500	0.342	0.652	0.548	0.651	0.691
BR	0.057	0.852	0.789	**0.837**	0.804	0.813	0.839	0.790	0.658	0.500	0.743	0.640	0.752	0.774
EMA1	0.030	0.532	0.555	**0.609**	0.540	0.571	0.532	0.468	0.348	0.257	0.500	0.381	0.480	0.556
EMA2	0.039	0.680	0.638	**0.730**	0.649	0.677	0.642	0.611	0.452	0.360	0.619	0.500	0.585	0.659
EMA1*	0.023	0.523	0.536	**0.600**	0.570	0.587	0.563	0.482	0.349	0.248	0.520	0.415	0.500	0.555
EMA2*	0.020	0.490	0.512	**0.553**	0.504	0.528	0.497	0.461	0.309	0.226	0.444	0.341	0.445	0.500

https://arxiv.org/abs/2312.00886



Accelerating Nash Learning from Human Feedback via Mirror Prox

Joint work with
Daniil Tiapkin, Daniele Calandriello, Denis Belomestny, Eric Moulines, Alexey Naumov, Kashif Rasul, Pierre Menard







Magnetic Preference Optimization (Wang et al. 2025)

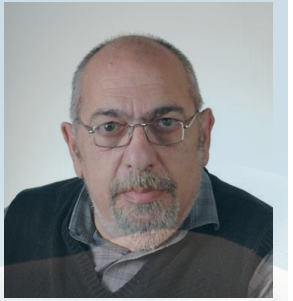
$$\pi_{k+1} = \arg\min_{\pi \in \Pi} \left\{ \eta \mathcal{P}(\pi_k \succ \pi) + \eta\beta \mathrm{KL}_\rho(\pi \| \pi^{\mathrm{ref}}) + (1 - \eta\beta)\mathrm{KL}_\rho(\pi \| \pi_k) \right\}$$

Overall: no need for additional stabilization to have good convergence rates!

Convergence guarantees:

$$\mathrm{KL}(\pi^\star_\beta \| \pi_T) \leq \frac{(1+\beta^2)^{-T}}{\beta}$$

Our algorithm: Nash Mirror Prox



Our approach: adapt an accelerated algorithm to this setup.

$$\pi_{k+\frac{1}{2}} = \arg\min_{\pi\in\Pi}\left\{\mathcal{P}(\pi_k \succ \pi) + \beta\,\mathrm{KL}_\rho(\pi\|\pi^{\mathrm{ref}}) + (\beta/\eta)\,\mathrm{KL}_\rho(\pi\|\pi_k)\right\},$$

$$\pi_{k+1} = \arg\min_{\pi\in\Pi}\left\{\mathcal{P}(\pi_{k+\frac{1}{2}} \succ \pi) + \beta\,\mathrm{KL}_\rho(\pi\|\pi^{\mathrm{ref}}) + (\beta/\eta)\,\mathrm{KL}_\rho(\pi\|\pi_k)\right\},$$

Convergence guarantees.

$$\mathrm{KL}(\pi^\star_\beta\|\pi_T) \le \frac{1}{2\beta}(1+2\beta)^{-T}$$

Naive Implementation of Nash Mirror Prox

Main question. How to *implement* this algorithm in the LLM setup?

Simple approach. Let's just approximate steps

$$\widehat{\pi}_{k+p/2} \approx \arg\min_{\pi \in \Pi} \left\{ \mathcal{P}(\widehat{\pi}_{k+(p-1)/2} \succ \pi) + \beta \, \mathrm{KL}(\pi \| \pi^{\mathrm{ref}}) + (\beta/\eta) \, \mathrm{KL}(\pi \| \widehat{\pi}_k) \right\}$$

using *policy gradients*

online policy

target policy

$$J_{k+p/2}(\theta) \triangleq \mathbb{E}_{y' \sim \pi_\theta} [\mathcal{P}(\widehat{\pi}_{k+(p-1)/2} \succ y')] + \beta \, \mathrm{KL}(\pi_\theta \| \pi^{\mathrm{ref}}) + (\beta/\eta) \, \mathrm{KL}(\pi_\theta \| \widehat{\pi}_k)$$

$$\theta_{k+\frac{p}{2},t+1} = \theta_{k+\frac{p}{2},t} - \gamma \hat{\nabla} J_{k+\frac{p}{2}}(\theta_{k+\frac{p}{2},t})$$

Scalable implementation of Nash Mirror Prox

Problem. Previous implementation requires to abruptly change the optimization target each T gradient steps for T large enough!

$$\widehat{\pi}_{k+p/2} \approx \arg\min_{\pi\in\Pi}\left\{\mathcal{P}(\widehat{\pi}_{k+(p-1)/2} \succ \pi) + \beta\,\mathrm{KL}(\pi\|\pi^{\mathrm{ref}}) + (\beta/\eta)\,\mathrm{KL}(\pi\|\widehat{\pi}_k)\right\}$$

Idea. Let's update the target more often (T times for the same "target model") but each update with worse accuracy:

$$\hat{\pi}_{k+p/T} \approx \arg\min_{\pi\in\Pi}\left\{\mathcal{P}\left(\hat{\pi}_{k+(p-1)/T} \succ \pi\right) + \beta\,\mathrm{KL}(\pi\|\pi^{\mathrm{ref}}) + \left(\frac{\beta}{\eta}\right)\mathrm{KL}(\pi\|\hat{\pi}_k)\right\}$$

Main idea behind of this approximation

Mirror prox = 2-step approximation of *Proximal Point (PP) Method*

$$\pi_{k+1} = \arg\min_{\pi \in \Pi} \left\{ \mathcal{P}(\pi_{k+1} \succ \pi) + \beta \, \mathrm{KL}_\rho(\pi \| \pi^{\mathrm{ref}}) + (\beta/\eta) \, \mathrm{KL}_\rho(\pi \| \pi_k) \right\}$$

Approximate Nash-MP: perform T gradient steps to approximate each of 2 PP-approximation steps;

(Naive) Deep Nash-MP: perform 1 gradient step to approximate each of T PP-approximation steps;

DL Implementation of Nash Mirror Prox

Problem. Need to perform a lot of gradient updates and afterwards change the objective.

Question. Could we *smooth* the updating procedure?

$$\mathcal{L}_{\texttt{NashMP}}(\theta;\theta',\theta^{\text{target}}) \triangleq \mathbb{E}_{\substack{x\sim\rho \\ y\sim\pi_\theta(\cdot|x) \\ y'\sim\pi_{\theta'}(\cdot|x)}} \left[\mathcal{P}(y\succ y'|x) + \beta\log\frac{\pi_\theta(y|x)}{\pi^{\text{ref}}(y|x)} + \frac{\beta}{\eta}\log\frac{\pi_\theta(y|x)}{\pi_{\theta^{\text{target}}}(y|x)}\right].$$

$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{NashMP}}(\theta_t;\theta_t,\theta_t^{\text{target}}), \qquad \theta_{t+1}^{\text{target}} = \kappa\theta_{t+1} + (1-\kappa)\theta_t^{\text{target}},$$

Symmetrization trick

Additional small idea: since we are performing only one gradient update, then:

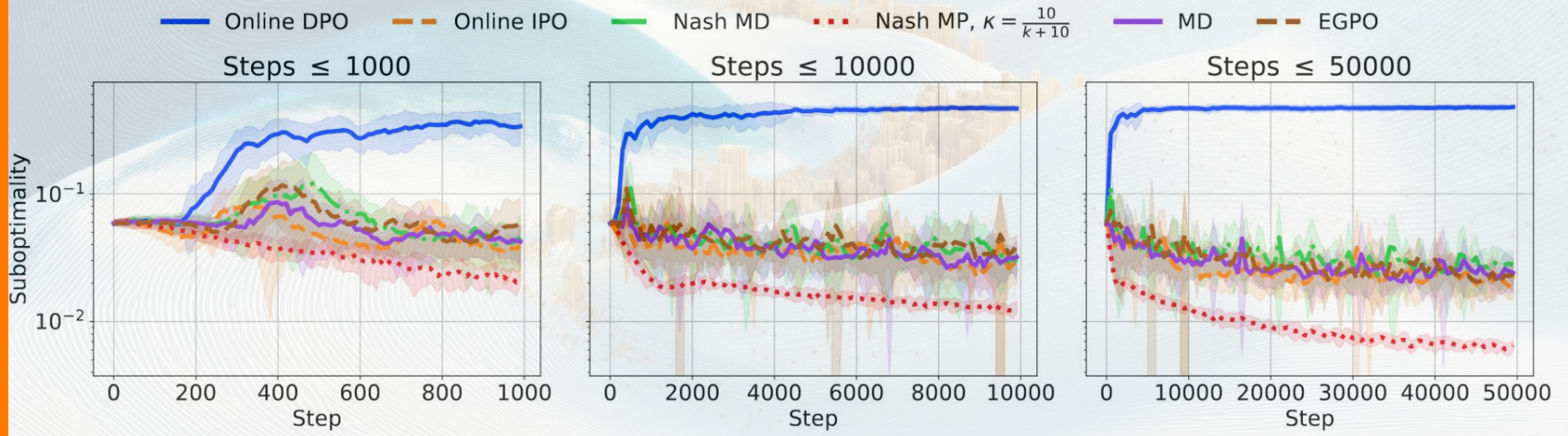
- We have a perfect baseline equals to ½
- We can utilize twice more generation without additional price
- **Bonus:** contrastive-style loss!

$$\widehat{\nabla}_\theta \mathcal{L}_{\texttt{NashMP}}(\theta_t; \theta_t, \theta_t^{\text{target}}) \triangleq \frac{1}{B}\sum_{i=1}^{B}\left(\nabla_\theta \log \pi_{\theta_t}(y_i|x_i) - \nabla_\theta \log \pi_{\theta_t}(y_i'|x_i)\right) \cdot \left(\frac{1}{2} - \mathcal{P}(y_i \succ y_i'|x_i)\right)$$
$$+ \nabla_\theta \log \pi_{\theta_t}(y_i|x_i)\left[\beta \log\left(\frac{\pi_{\theta_t}(y_i|x_i)}{\pi^{\text{ref}}(y_i|x_i)}\right) + \frac{\beta}{\eta}\log\left(\frac{\pi_{\theta_t}(y_i|x_i)}{\pi_{\theta_t^{\text{target}}}(y_i|x_i)}\right)\right]$$
$$+ \nabla_\theta \log \pi_{\theta_t}(y_i'|x_i)\left[\beta \log\left(\frac{\pi_{\theta_t}(y_i'|x_i)}{\pi^{\text{ref}}(y_i'|x_i)}\right) + \frac{\beta}{\eta}\log\left(\frac{\pi_{\theta_t}(y_i'|x_i)}{\pi_{\theta_t^{\text{target}}}(y_i'|x_i)}\right)\right].$$

Nash Mirror Prox for contextual games

Problem. Low-rank contextual game

$$\mathcal{P}(y \succ y'|x) \triangleq \sigma(A_{y,y'} - A_{y',y}), \quad A \triangleq U\Theta_x V^\top,$$



Nash Mirror Prox for LLM fine-tuning

Problem. Instruction-following, coding and math for Gemma-2 2B;

Evaluation. LLM-as-a-judge

Table 2: Pairwise Win Rates (mean $\pm$ 3σ-confidence intervals). Statistically significant wins are in **bold**. Confidence intervals are in a smaller font size. Row/column for NashMP, $\kappa = 0.1$ is highlighted.

Win rate	SFT	Online DPO	Online IPO	NashMD	Reg. Self-Play	NashMP, $\kappa = 0.1$
SFT	–	0.1623$\pm$0.0087	0.1554$\pm$0.0091	0.1974$\pm$0.0098	0.1536$\pm$0.0087	0.1283$\pm$0.0081
Online DPO	**0.8377**$\pm$0.0087	–	0.4743$\pm$0.0115	**0.5788**$\pm$0.0116	0.4730$\pm$0.0113	0.4392$\pm$0.0116
Online IPO	**0.8446**$\pm$0.0091	**0.5257**$\pm$0.0115	–	**0.6115**$\pm$0.0121	0.5036$\pm$0.0118	0.4706$\pm$0.0117
NashMD	**0.8026**$\pm$0.0098	0.4212$\pm$0.0116	0.3885$\pm$0.0121	–	0.4031$\pm$0.0119	0.3605$\pm$0.0115
Reg. Self-Play	**0.8464**$\pm$0.0087	**0.5270**$\pm$0.0113	0.4964$\pm$0.0118	**0.5969**$\pm$0.0119	–	0.4620$\pm$0.0118
NashMP, $\kappa = 0.1$	**0.8717**$\pm$0.0081	**0.5608**$\pm$0.0116	**0.5294**$\pm$0.0117	**0.6395**$\pm$0.0115	**0.5380**$\pm$0.0118	–

Optimal Design for Reward Modeling in RLHF

SNEAK PEEK

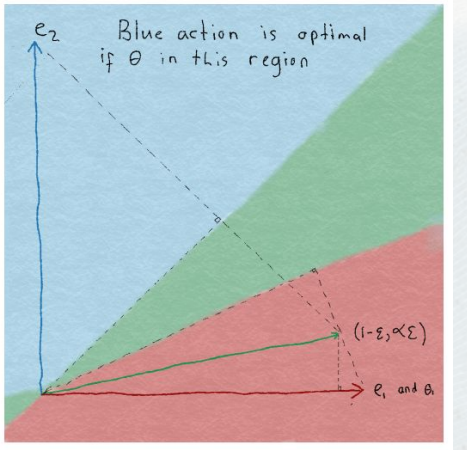
Michal Valko

Chief Models Officer of a Stealth Startup

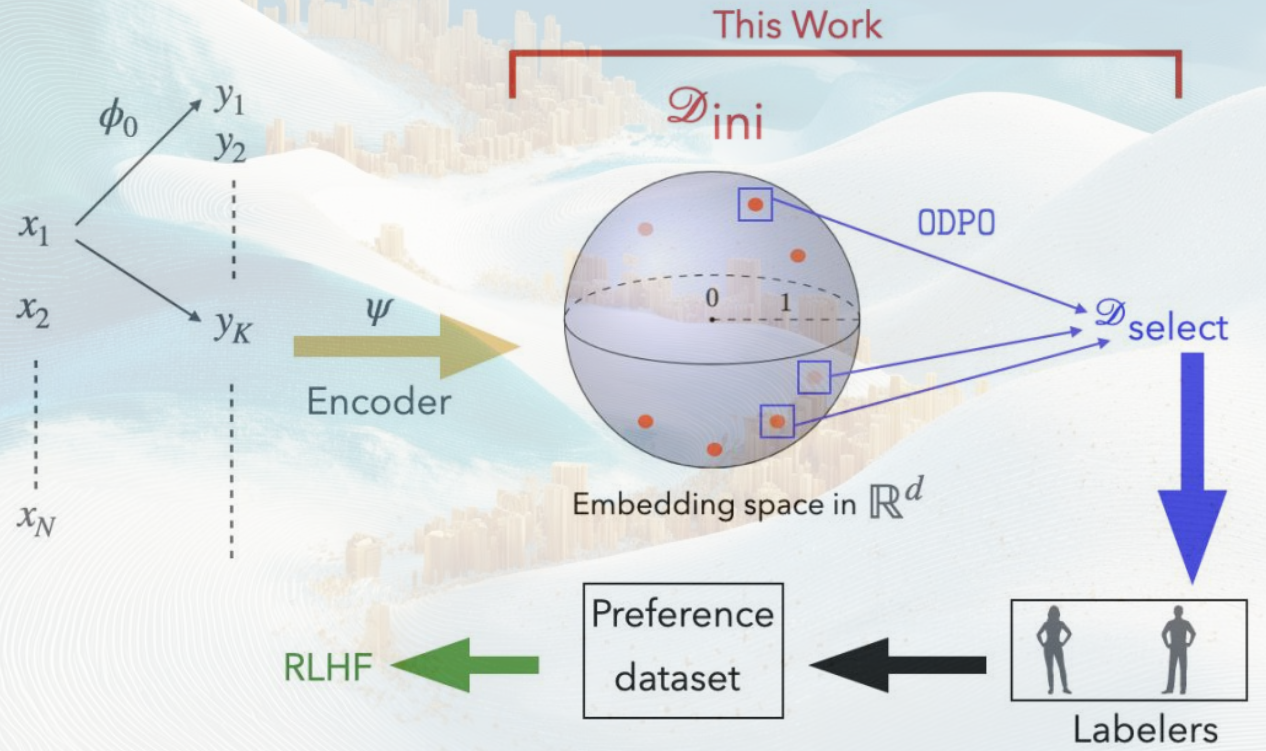
w/ Antoine Scheid, Etienne Boursier, Alain Durmus, Michael I Jordan, Pierre Ménard, Eric Moulines

How to contrast?

- Cumulative reward vs. best policy identification
- RLHF rounds people cast their preference that we contrast
  - Are we doing the right?
- Active learning of preference pairs
- Gamification of Pure Exploration for Linear Bandits, ICML 2020
- What is a different now?
  - Infinitely many arms
  - Dueling nature
  - Part of a bigger loop
- Next level: Active learning of prompts



Optimal Design for Preference Optimization



Language Generation with Replay

SNEAK PEEK


w/ Giorgio Racca, Amartya Sanyal

Stop the singularity of ensh*ttification!

- We train LLMs on internet
- We use LLM flood internet
- Does it make them dumb?

> **Question.** Does the presence of replay, where a generator is trained on its own past outputs, make language generation fundamentally harder?

Current tricks:

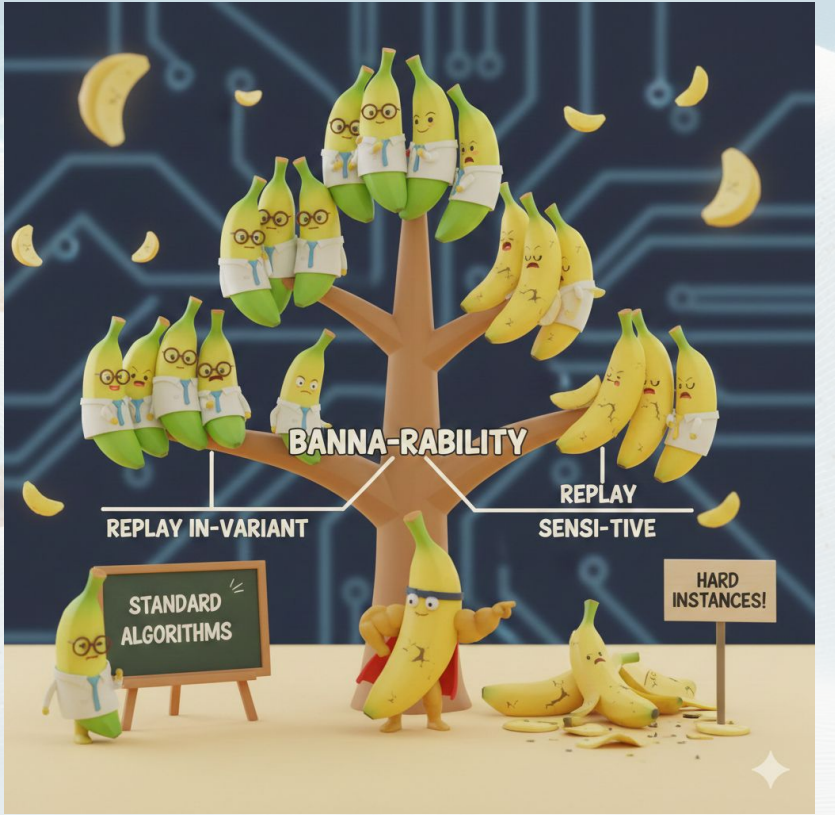
- Data cleaning
- Watermarking
- Blissful ignorance
- Output filtering

Complete characterization of replay

- When reply matters
- When reply-resistant
- Hard instances
- Output restriction

Generation notion	Finite $\mathcal{H}$	Countable $\mathcal{H}$	General $\mathcal{H}$
Uniform	✓ (3.1)	✓ (3.1)	✓ (3.1)
Non-uniform (strong)	✓ (3.1)	× (4.1)	× (4.1)
Non-uniform (weak)	✓ (3.1)	✓ (4.2)	?
In the limit	✓ (5.1)	✓ (5.1)	× (5.2)
Proper in the limit	× (6.2)	× (6.2)	× (6.2)

✓: same guarantees as the standard setting. ×: strict separation from the standard setting. ?: unresolved by our results. Parenthesized numbers indicate the theorem establishing the entry.



Open question: Multi-preferences

Nash Learning approach: a comparison gives just one number - which answer is better.

However: we care about more multi-dimensional objects:

$$\mathcal{P}(y \succ y'|x) = \begin{pmatrix} \text{is } y \text{ more helpful than } y'? \\ \text{is } y \text{ more honest than } y'? \\ \text{is } y \text{ more harmless than } y'? \end{pmatrix}$$

Open question: How to define a proper objective for it?

Askell, Amanda, et al. "A general language assistant as a laboratory for alignment." *arXiv preprint arXiv:2112.00861* (2021).