CS 839: Foundation Models

Alignment

Fred Sala

University of Wisconsin-Madison

Oct. 2, 2025

Announcements

- **Logistics:**
  - Presentation information out tonight
  - **Homework 1:** Due tonight
  - **OH**: Changho, Monday 1-2 pm
  - **OH**: (just this week) Fred: **Th 2:30-3:30, Fri: 3:00-4:00 pm**
- Class roadmap:

Thursday Oct. 2	Alignment
Tuesday Oct. 7	RLVR
Thursday Oct. 9	Efficient Training

Outline

- **Alignment and RLHF**
  - Finish up last time, Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Why Does It Work?**
  - Failures of supervised learning, knowledge-seeking interactions, abstains
- **Variations + Open Questions**
  - Direct Preference Optimization (DPO), RLAIF, other techniques

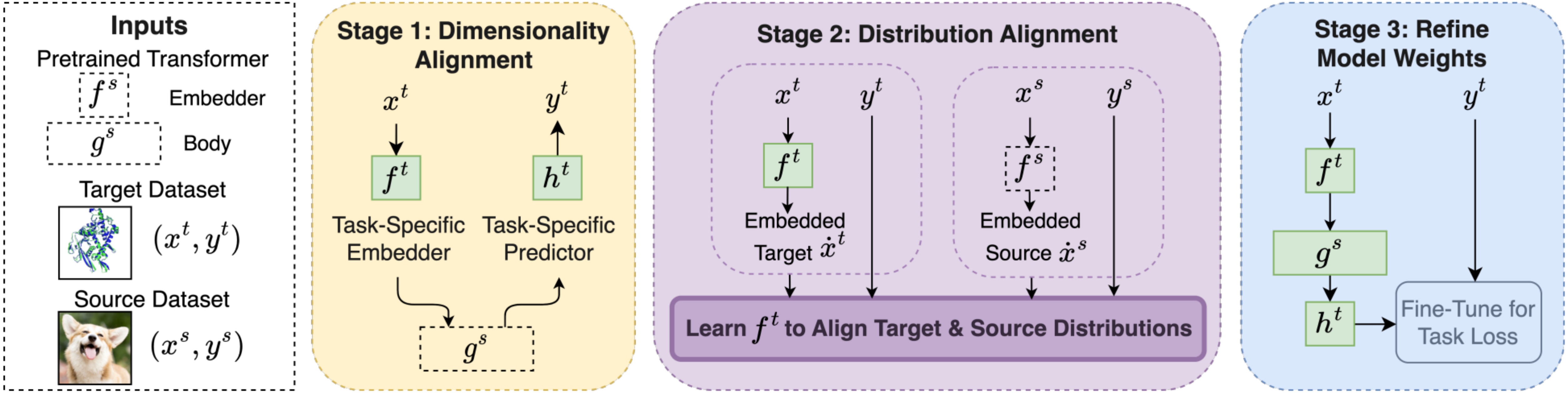
Outline

- **Alignment and RLHF**
    - Finish up last time, Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Why Does It Work?**
    - Failures of supervised learning, knowledge-seeking interactions, abstains
- **Variations + Open Questions**
    - Direct Preference Optimization (DPO), RLAIF, other techniques

From last time: Cross-modal Techniques

A powerful adaptation approach: ORCA (Shen et al ‘23)

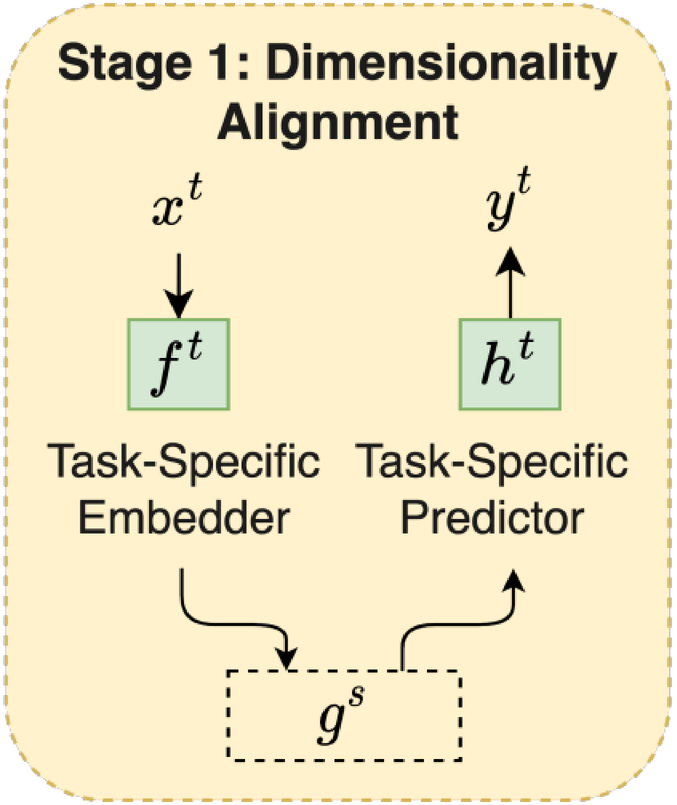
- Adds: distribution alignment step (align then refine)
- Distribution alignment is via OTDD



ORCA: **Stage 1**

Let’s understand each stage of ORCA

- Stage 1: compatibility for inputs and outputs
- Custom input and output embedders that depend on the task
  - Input example: convolutional layers for image settings
  - Output example: average pooling+linear layer for classification



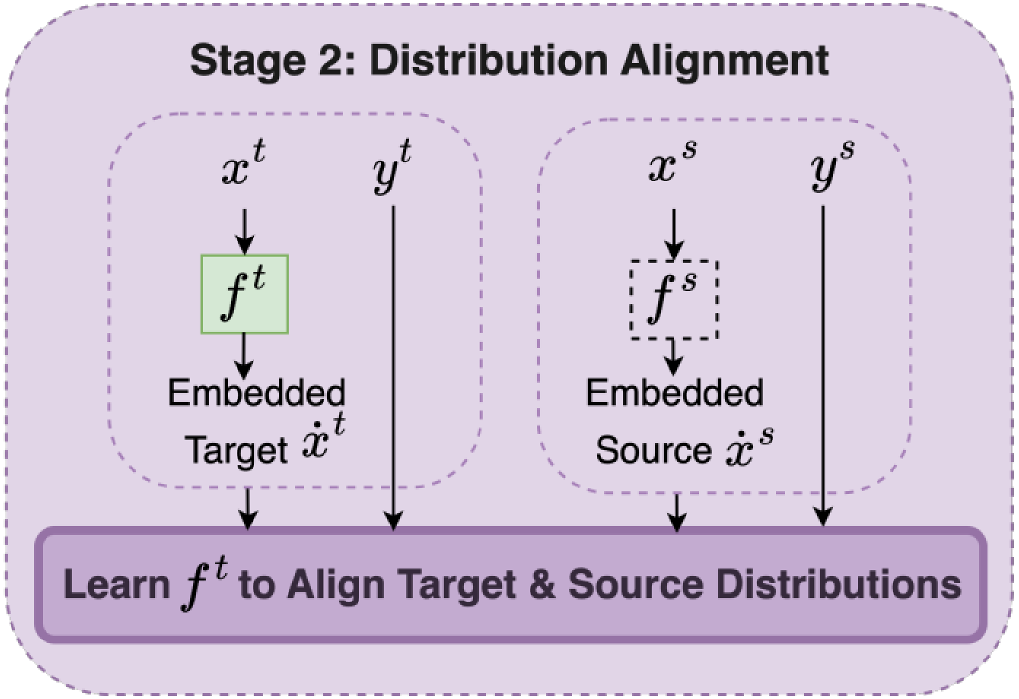
ORCA: **Stage 2**

Let’s understand each stage of ORCA

- Stage 2: distribution alignment
- Intuition:
  - Change embeddings so target features **resemble** source features

- Learn the function f^t **that minimizes distance between**

  $(f^t(x^t), y^t)$ and $(f^s(x^s), y^s)$


Stage 2: Distribution Alignment
x^t
y^t
f^t
Embedded Target $\dot{x}^t$
x^s
y^s
f^s
Embedded Source $\dot{x}^s$
Learn f^t to Align Target & Source Distributions

ORCA: **Dataset and Distributional Distances**

Want: learn the function f^t **that minimizes distance between**

$(f^t(x^t),y^t)$ and $(f^s(x^s),y^s)$

- How?
- Need a distance function on these datasets (i.e., empirical distributions)
- Here, **optimal transport dataset distance** (OTDD)
- What is OT?

$$\inf\left\{\int_{X\times Y} c(x,y)\,\mathrm{d}\gamma(x,y)\,\middle|\,\gamma\in\Gamma(\mu,\nu)\right\},$$

Interlude: **Optimal Transport**

In optimal transport, we solve

$$\inf\left\{\int_{X\times Y} c(x,y)\,\mathrm{d}\gamma(x,y)\,\middle|\,\gamma\in\Gamma(\mu,\nu)\right\},$$

Cost or **distance** of moving x to y

The two **marginals** we care about, i.e., on x and y (for us, source and target)

- Want to “move” distribution on *x* to one on *y*
  - Output is a joint distribution with the original source and target
- But there’s a cost to moving x to y, given by c(x,y)

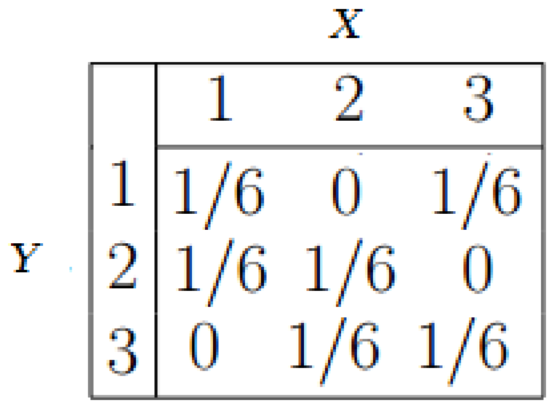
Interlude: **Optimal Transport**

In optimal transport, we solve

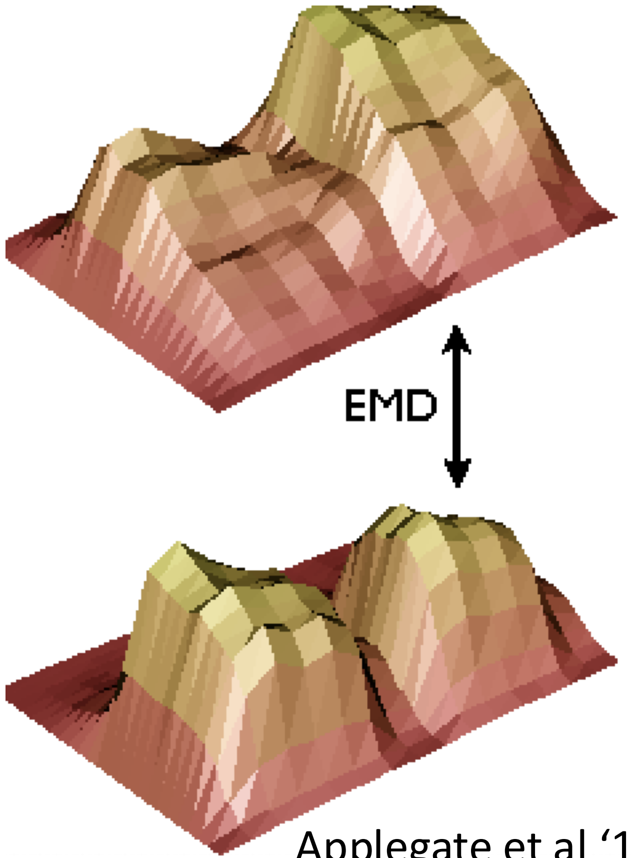
$$\inf\left\{\int_{X\times Y} c(x,y)\,\mathrm{d}\gamma(x,y)\,\middle|\,\gamma\in\Gamma(\mu,\nu)\right\},$$

- What is the deal with the **joint distribution** γ?
  - Defines a “plan” for how to move source to target

- What is the deal with the **cost** function c(*x*,*y*)
  - Defines how expensive each move is



		X		
		1	2	3
	1	1/6	0	1/6
Y	2	1/6	1/6	0
	3	0	1/6	1/6



Applegate et al ‘11

Interlude: **Optimal Transport**

In optimal transport, we solve

$$\inf \left\{ \int_{X \times Y} c(x, y) \, \mathrm{d}\gamma(x, y) \,\middle|\, \gamma \in \Gamma(\mu, \nu) \right\},$$

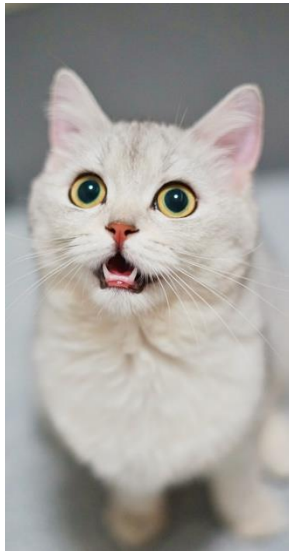
- Cost given by **distance**: result is Wasserstein distance
- Gives a distance on distributions, i.e.,

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \mathbf{E}_{(x,y)\sim\gamma} d(x, y)^p \right)^{1/p}$$

Back to the Beginning: **Dataset Distance**

Where do we use this? We wanted a distance to begin with

- For inputs x, pretty easy: feature vectors in spaces that have distances, e.g., ||x-x'||
- For outputs y, not so easy



–

- A clever idea:
  - Replace y with P(X|y)
- **Now we get to use Wasserstein**: W(P(X|y),P(X|y'))
  - Approximate this with a Gaussian: closed form too!

ORCA: **Distributional Distances**

Want: learn the function f^t **that minimizes distance between**

$(f^t(x^t), y^t)$ and $(f^s(x^s), y^s)$

- Need a distance function on these distributions
- Here, **optimal transport dataset distance** (OTDD)

$$d_{\mathcal{Z}}\big((x,y),(x',y')\big) \triangleq \big(d_{\mathcal{X}}(x,x')^p + \mathbf{W}_p^p(\alpha_y,\alpha_{y'})\big)^{1/p}$$

i.e., Euclidean distance

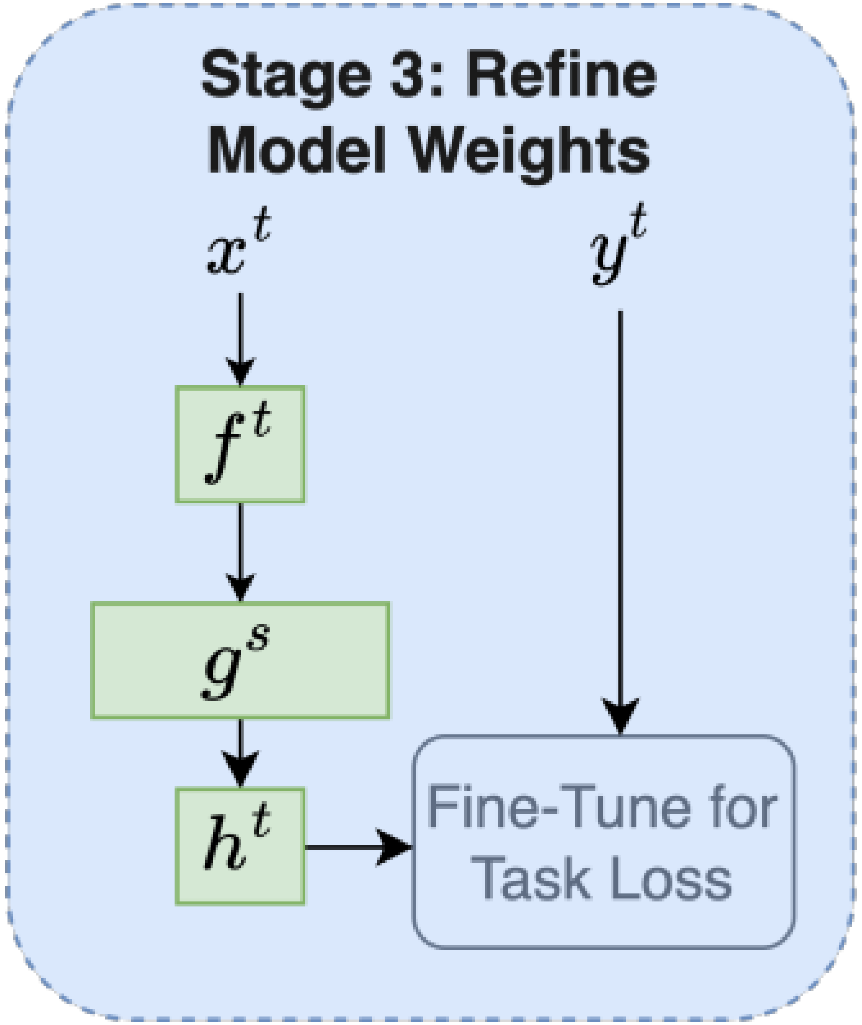
p-**Wasserstein distance** on P(x|y)

Alvarez-Melis and Fusi, ‘20

ORCA: **Stage 3**

Let's understand each stage of ORCA

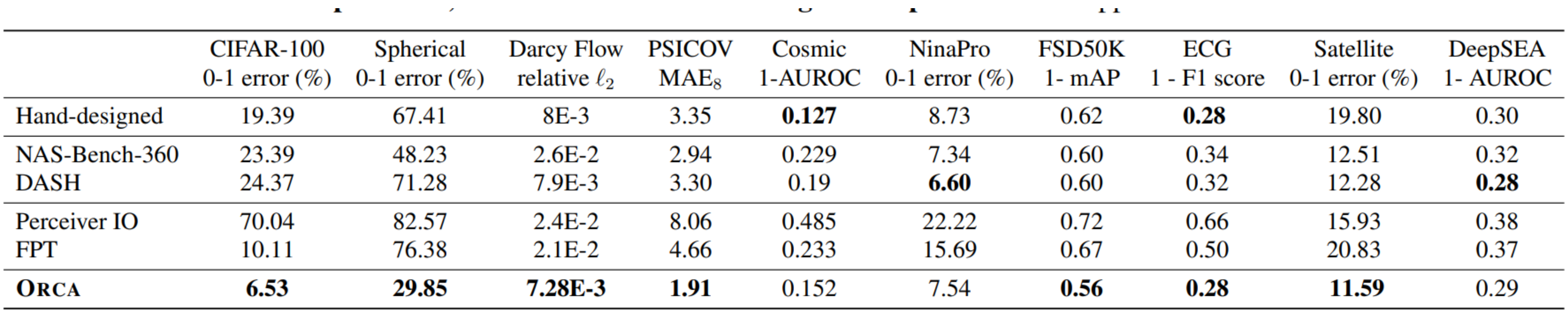
- Stage 3: fine-tune the input and output network weights
  - For particular tasks
  - Or, could do any other variant of what we've talked about…



ORCA: **Results**

Extremely good, even against state-of-the-art results

- Compare to Neural Architecture Search (NAS)
  - Produces custom architectures that improve state-of-the-art performance for various tasks
  - Same procedure on many types of tasks works well:



	CIFAR-100 0-1 error (%)	Spherical 0-1 error (%)	Darcy Flow relative ℓ_2	PSICOV MAE$_8$	Cosmic 1-AUROC	NinaPro 0-1 error (%)	FSD50K 1- mAP	ECG 1 - F1 score	Satellite 0-1 error (%)	DeepSEA 1- AUROC
Hand-designed	19.39	67.41	8E-3	3.35	**0.127**	8.73	0.62	**0.28**	19.80	0.30
NAS-Bench-360	23.39	48.23	2.6E-2	2.94	0.229	7.34	0.60	0.34	12.51	0.32
DASH	24.37	71.28	7.9E-3	3.30	0.19	**6.60**	0.60	0.32	12.28	**0.28**
Perceiver IO	70.04	82.57	2.4E-2	8.06	0.485	22.22	0.72	0.66	15.93	0.38
FPT	10.11	76.38	2.1E-2	4.66	0.233	15.69	0.67	0.50	20.83	0.37
ORCA	**6.53**	**29.85**	**7.28E-3**	**1.91**	0.152	7.54	**0.56**	**0.28**	**11.59**	0.29

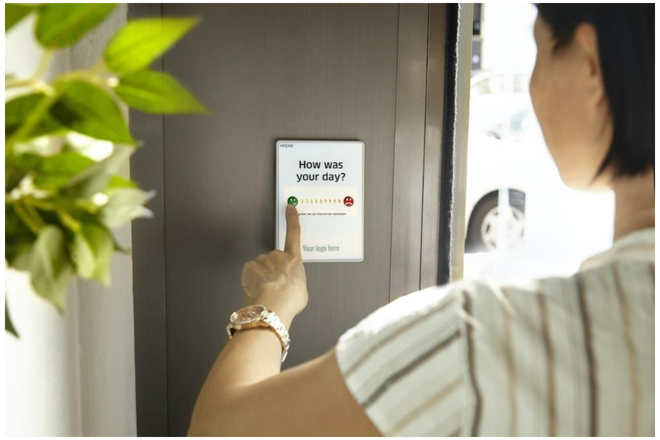
Alignment: **Basic Motivation**

Goal: produce language model outputs that users like better…

- **Hard** to specify exactly what this means,
- **Easy** to query users

Collect human feedback and use it to change the model

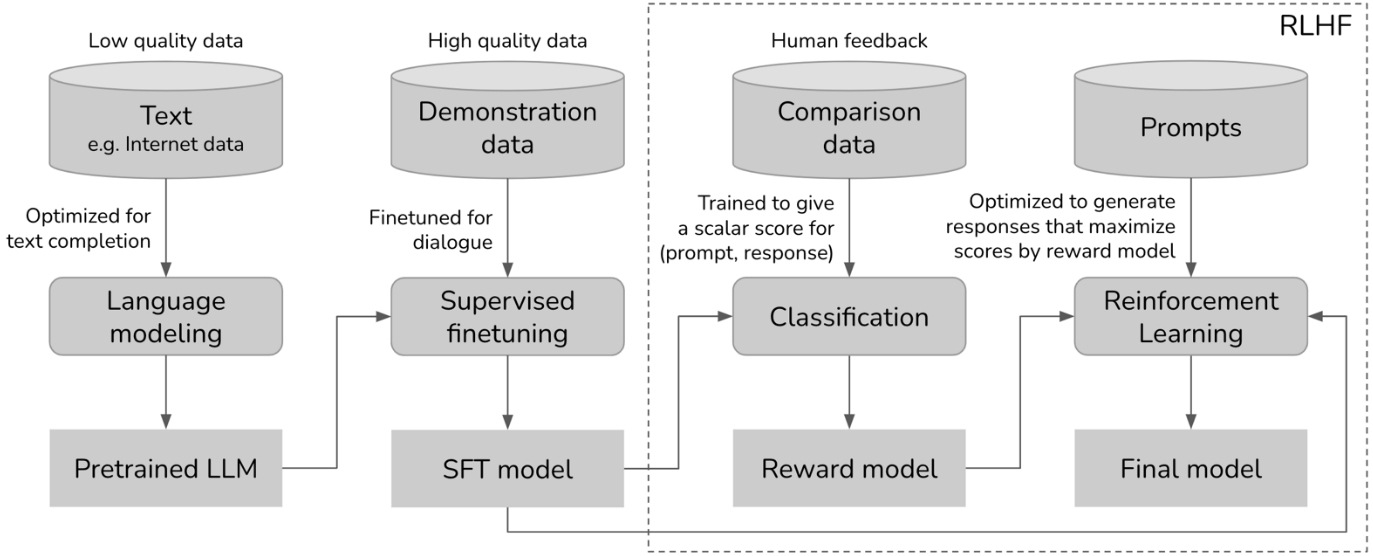
- Can do this by fine-tuning, especially with instructions
- Doesn't quite capture what users want
- We'll use other approaches, like RLHF



RLHF: **Setup**

Goal: produce language model outputs that users like better…

- Via RL with trained reward model (Ouyang et al ‘222)



RLHF

Low quality data

Text
e.g. Internet data

Optimized for text completion

Language modeling

Pretrained LLM

High quality data

Demonstration data

Finetuned for dialogue

Supervised finetuning

SFT model

Human feedback

Comparison data

Trained to give a scalar score for (prompt, response)

Classification

Reward model

Prompts

Optimized to generate responses that maximize scores by reward model

Reinforcement Learning

Final model

Scale May ‘23	**>1 trillion** tokens	**10K - 100K** (prompt, response)	**100K - 1M** comparisons (prompt, winning_response, losing_response)	**10K - 100K** prompts
Examples **Bolded**: open sourced	GPT-x, Gopher, **Falcon**, LLaMa, **Pythia**, **Bloom**, **StableLM**	**Dolly-v2, Falcon-Instruct**		InstructGPT, ChatGPT, Claude, **StableVicuna**

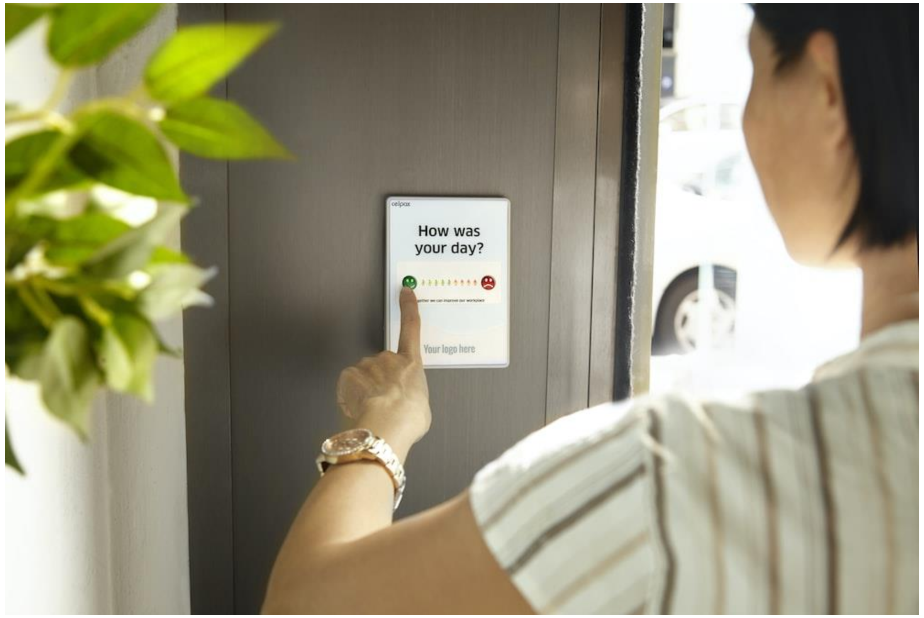
Chip Huyen

RLHF: **Feedback**

First stage: get **human feedback** to train reward model

- Fix a set of prompts
- Produce multiple outputs for each prompt
  - Can get them from the original model post-SFT, or otherwise

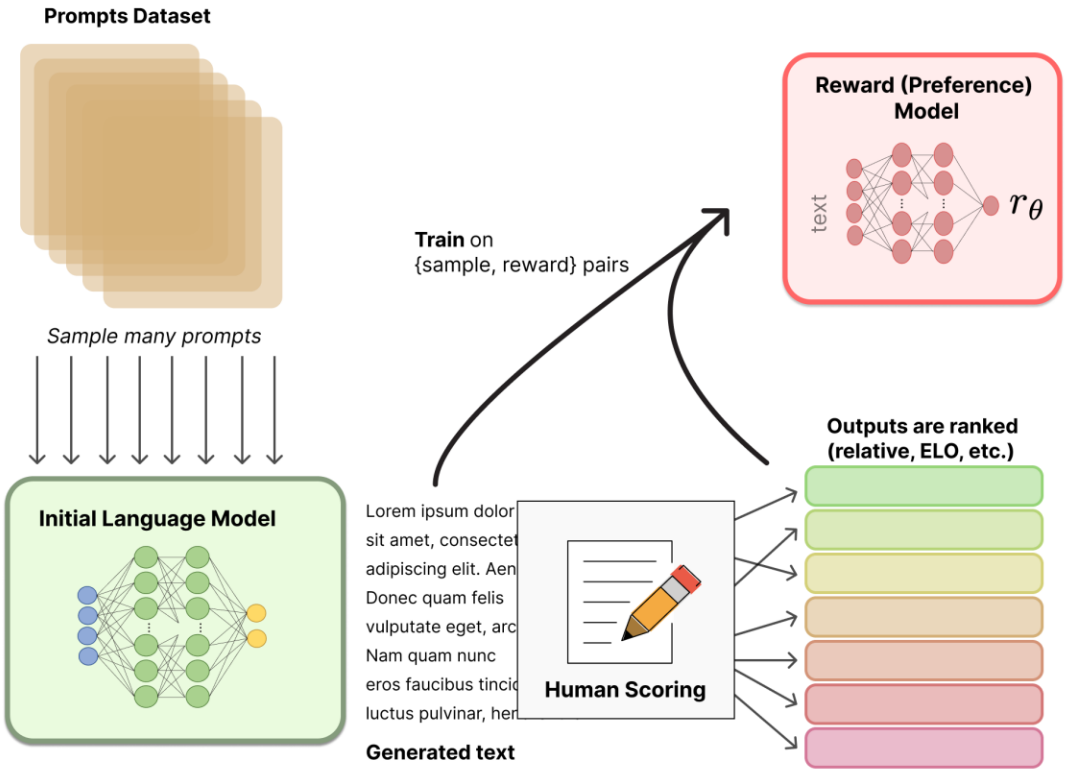
- Ask human users **which is better**
  - **Binary output**
  - Can do more
    - Rank more questions



RLHF: **Reward/Preference Model**

Second stage: train reward model

- Use the human feedback to train/fine-tune another model to reproduce the metric
- **Preference model**


Prompts Dataset
Sample many prompts
Initial Language Model
Lorem ipsum dolor sit amet, consectet adipiscing elit. Aen Donec quam felis vulputate eget, arc Nam quam nunc eros faucibus tincid luctus pulvinar, her
Generated text
Human Scoring
Train on {sample, reward} pairs
Outputs are ranked (relative, ELO, etc.)
Reward (Preference) Model
text
r_θ

https://huggingface.co/blog/rlhf

RLHF: **Reward/Preference Model**

Second stage: train reward model

- Use the human feedback to train/fine-tune another model to reproduce the metric
- **Loss?** Based on preference models,
  - Example: Bradley-Terry model

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

  - Then, our reward model loss is based on the log likelihood,

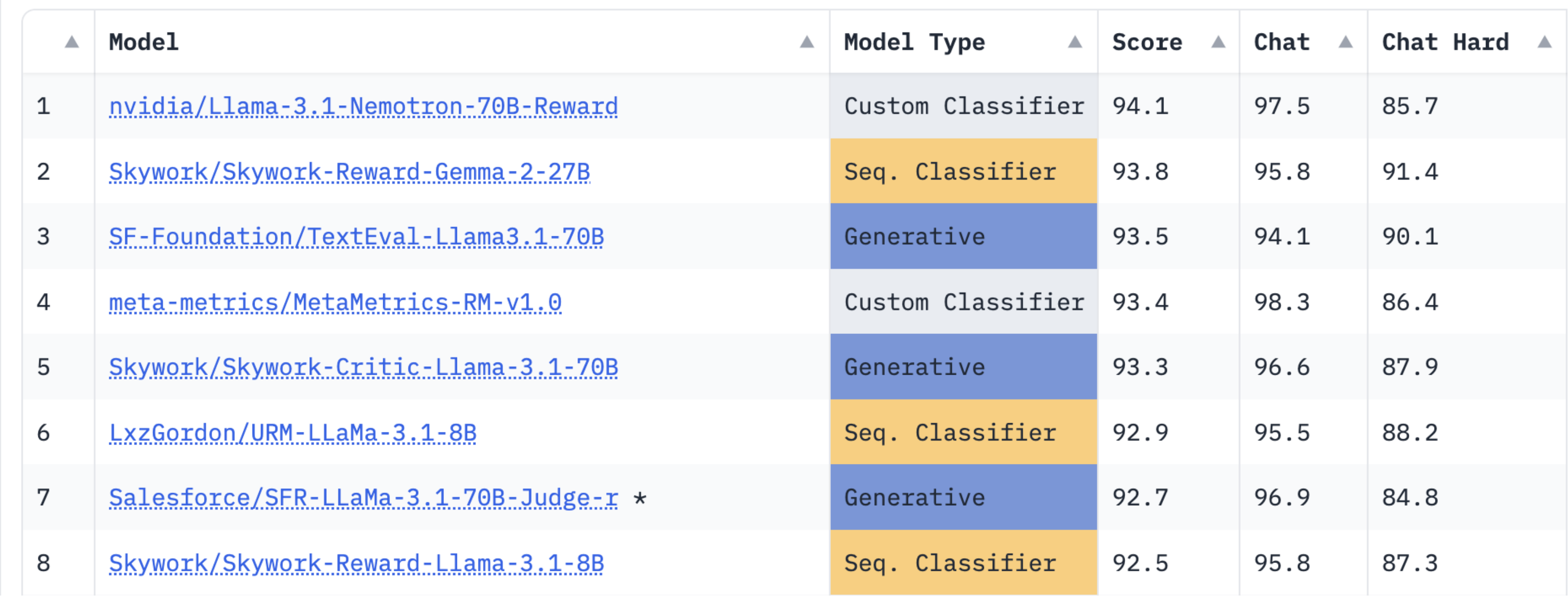
$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$$

RLHF: **Reward/Preference Model**

Note: we don’t have to always do this from scratch

- Pretrained reward models available
- Benchmarks for this:

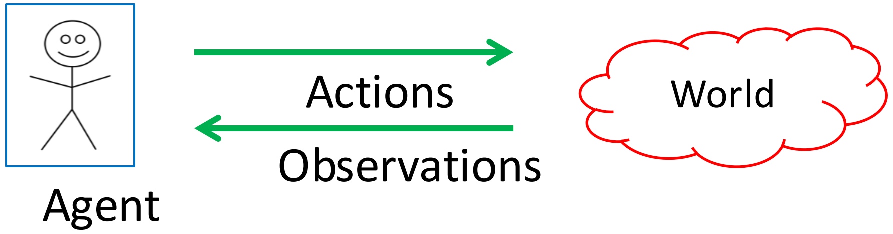
rewardbench



	Model	Model Type	Score	Chat	Chat Hard
1	nvidia/Llama-3.1-Nemotron-70B-Reward	Custom Classifier	94.1	97.5	85.7
2	Skywork/Skywork-Reward-Gemma-2-27B	Seq. Classifier	93.8	95.8	91.4
3	SF-Foundation/TextEval-Llama3.1-70B	Generative	93.5	94.1	90.1
4	meta-metrics/MetaMetrics-RM-v1.0	Custom Classifier	93.4	98.3	86.4
5	Skywork/Skywork-Critic-Llama-3.1-70B	Generative	93.3	96.6	87.9
6	LxzGordon/URM-LLaMa-3.1-8B	Seq. Classifier	92.9	95.5	88.2
7	Salesforce/SFR-LLaMa-3.1-70B-Judge-r *	Generative	92.7	96.9	84.8
8	Skywork/Skywork-Reward-Llama-3.1-8B	Seq. Classifier	92.5	95.8	87.3

Lambert et al ‘24

RLHF: **Fine-Tuning with RL**



Third stage: RL

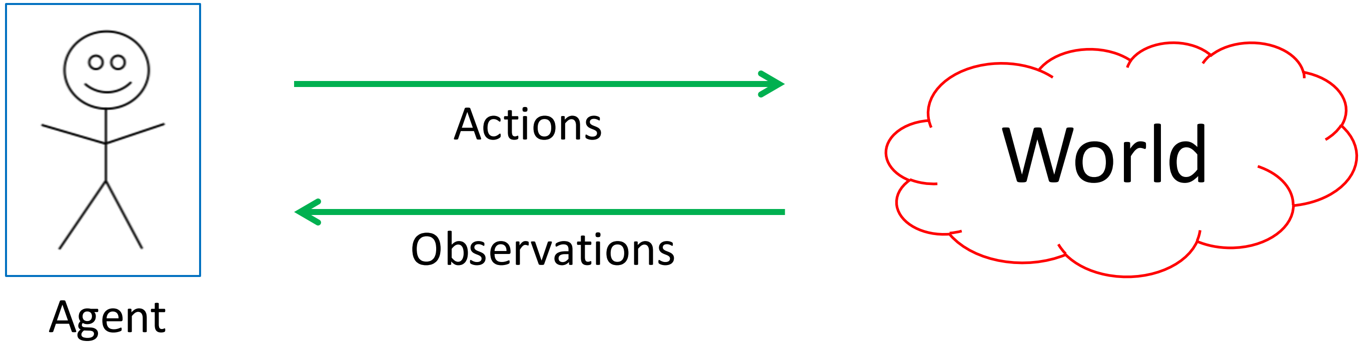
- Use an RL algorithm
- **Goal:** produce outputs that have high reward

RL formulation:

- **Action space**: all the tokens possible to output
- **State space**: all the sequences of tokens
- **Reward function**: the trained reward model
- **Policy**: the new version of the LM, taking in state and returning tokens

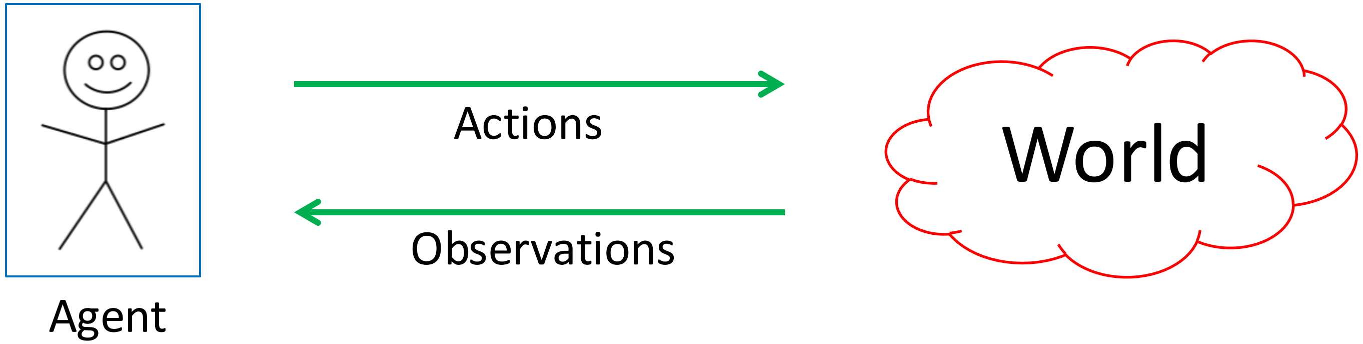
Reinforcement Learning Review

We have an **agent** **interacting** with the **world**



- Agent receives a reward based on state of the world
  - **Goal**: maximize reward / utility **($$$)**

RL Review: **Theoretical Model**



Basic setup:

- Set of states, S
- Set of actions A
- Information: at time *t*, observe state $s_t \in$ S. Get reward r_t
- Agent makes choice $a_t \in$ A. State changes to $s_{t+1,}$ continue

Goal: find a map from **states to actions** maximize rewards.

↑

A “policy”

RL Review: **Markov Decision Process (MDP)**

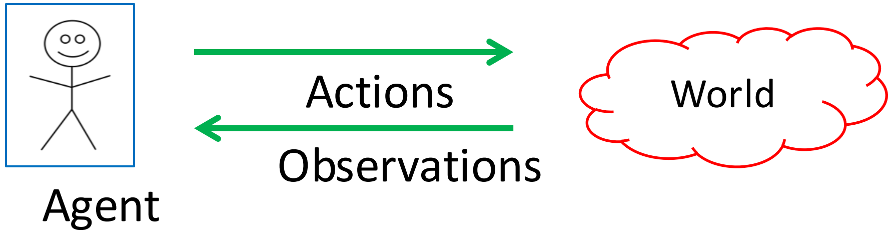
The formal mathematical model:

- **State set** S. Initial state $s_{0.}$ **Action set** A
- **State transition model**: $P(s_{t+1}|s_t, a_t)$
  - Markov assumption: transition probability only depends on s_t and a_t, and not previous actions or states.
- **Reward function:** $r(s_t)$
- **Policy**: $\pi(s) : S \to A$ action to take at a particular state.

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \ldots$$

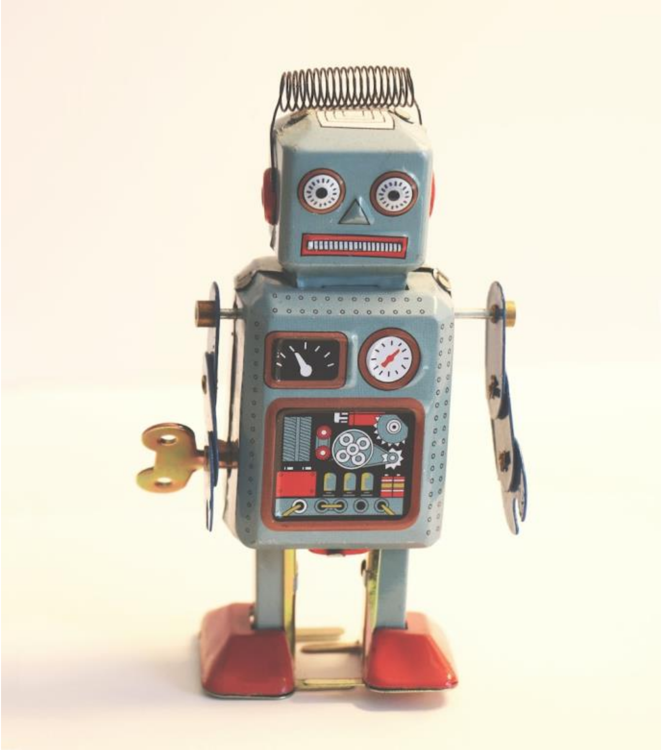
RLHF: **RL Approach**

What approach for RL stage?



- Many deep RL methods available
- Policy gradient methods
- Popular: PPO (Proximal Policy Optimization)
    - Main difference from vanilla policy gradient, you constrain change to policy at each step (Schulman et al)

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



Break & Questions

Outline

- **Alignment and RLHF**
  - Finish up last time, Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Why Does It Work?**
  - Failures of supervised learning, knowledge-seeking interactions, abstains
- **Variations + Open Questions**
  - Direct Preference Optimization (DPO), RLAIF, other techniques

Why RLHF?

Why should we do this?

- Why does supervised fine-tuning by itself not give our goal results?

- Many hypotheses; this section inspired by Yoav Goldberg's blog:
  - https://gist.github.com/yoavg/6bff0fecd65950898eba1bb321cfbd81
  - Itself based on Schulman's talk
  - https://www.youtube.com/watch?v=hhiLw5Q_UFg

Why RLHF? **Ways To Interact**

Three “modes of interaction”:

- **text-grounded**: provide the model with text, instruction ("what are the chemical names mentioned in this text“),
- **knowledge-seeking**: provide the model with question or instruction, and expect a (truthful) answer based on the model's internal knowledge
- **creative**: provide the model with question or instruction, expect some creative output. ("Write a story about...")

Why RLHF? **Knowledge-seeking**

Three “modes of interaction”:

- **knowledge-seeking**: provide the model with question or instruction, and expect a (truthful) answer based on the model's internal knowledge

- This is hypothesized to require RL. Why does **SL fail?**
  - Case 1: know the answer: fine.
  - Case 2: don’t know the answer. Supervised learning forces memorization, cannot produce “don’t know”.
  - Worse, SL on case 2 encourages **model to lie…**

Why RLHF? **Knowledge-seeking with RL**

Three “modes of interaction”:

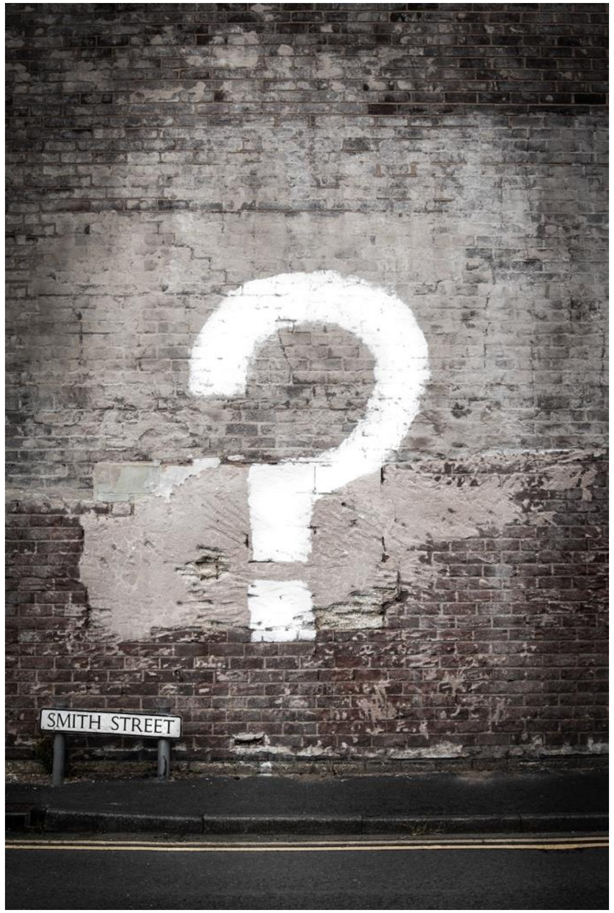
- **knowledge-seeking**: provide the model with question or instruction, and expect a (truthful) answer based on the model's internal knowledge

- Why does RL succeed?
  - Case 1: know the answer: fine. Get a reward
  - Case 2: don’t know the answer. Sometimes make it up and get a reward if lucky, most of the time low reward
  - **Encourages truth telling.**

Why RLHF? **Abstains**

Additionally, **we'd like our model to abstain**

- SL will really struggle with this
  - Usually no abstains in datasets
  - Even if there were, "generalization" here means abstaining on similar questions? Difficult
- RL still challenging, need to produce high reward for "don't know", but specific to model
- One way to craft a reward function:
  - High reward: correct answers
  - Medium reward: abstain
  - Negative reward: incorrect



Break & Questions

Outline

- **Alignment and RLHF**
  - Finish up last time, Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Why Does It Work?**
  - Failures of supervised learning, knowledge-seeking interactions, abstains
- **Variations + Open Questions**
  - Direct Preference Optimization (DPO), RLAIF, other techniques

RLHF Problems

Lots of challenges!

- **Casper et al**, “Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback”

- Challenges everywhere, all three phases:
  - **In human feedback,**
  - **In obtaining reward model,**
  - **In obtaining the policy**

RLHF Problems: **Human Feedback**

- Need to obtain some kind of “representative” collection of feedback providers
- **Simpler:**
  - Some people have biases
  - Mistakes due to lack of care (standard in crowdsourcing)
  - Adversarial data poisoners
- **Harder:**
  - In tough settings, what is “good” output?
  - Possible to manipulate humans

RLHF Problems: **Human Feedback**

- Additionally, **need high-quality data**.
- Expensive to hand-craft good prompts to drive feedback

- Feedback quality:
  - Tradeoffs in feedback levels
  - Ideally, rich
  - But harder to work with to train reward

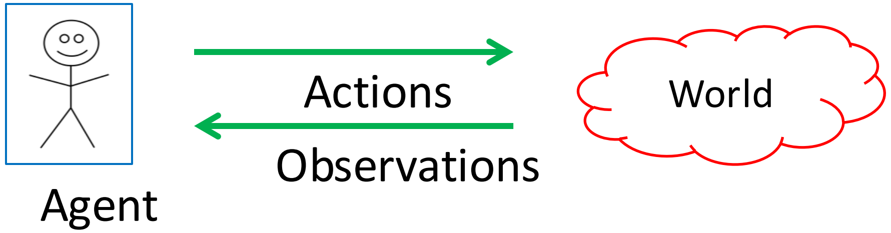
RLHF Problems: **Reward Model**

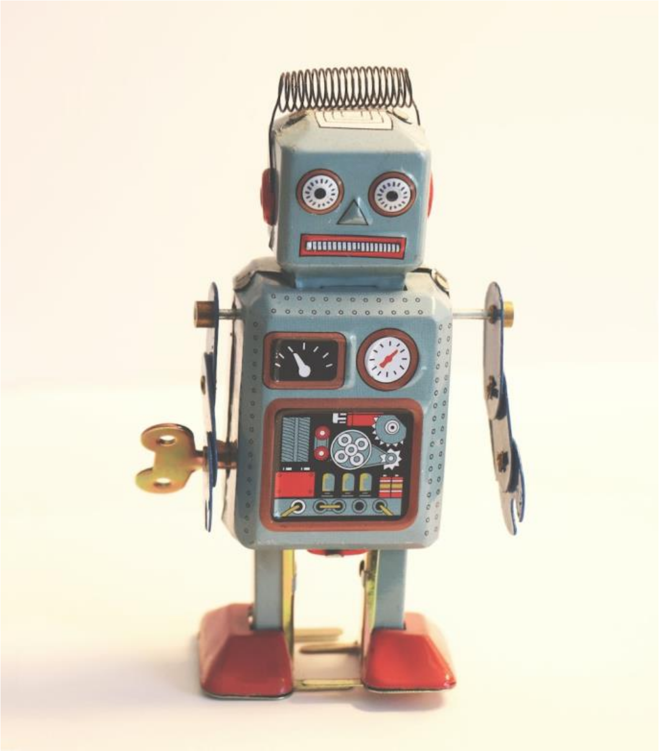
- Values can be difficult to express as a reward function
- May need to combine multiple reward functions:
  - What’s a “universal” one? People are different

- Reward Hacking
  - In tough settings, what is “good” output?
  - Possible to manipulate humans

RLHF Problems: **Training**

- The RL in RLHF can be difficult
- Also, learned policies **do not necessarily generalize to other environments**





RLHF **Alternatives**

- **Direct preference optimization** (DPO)
  - Bypass separate trained reward model: just use preference information **directly** (Rafailov et al,‘23)
  - **How?** Model a preference distribution from samples, integrate into a single loss (one-stage approach)

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

  - **Gradient step:**

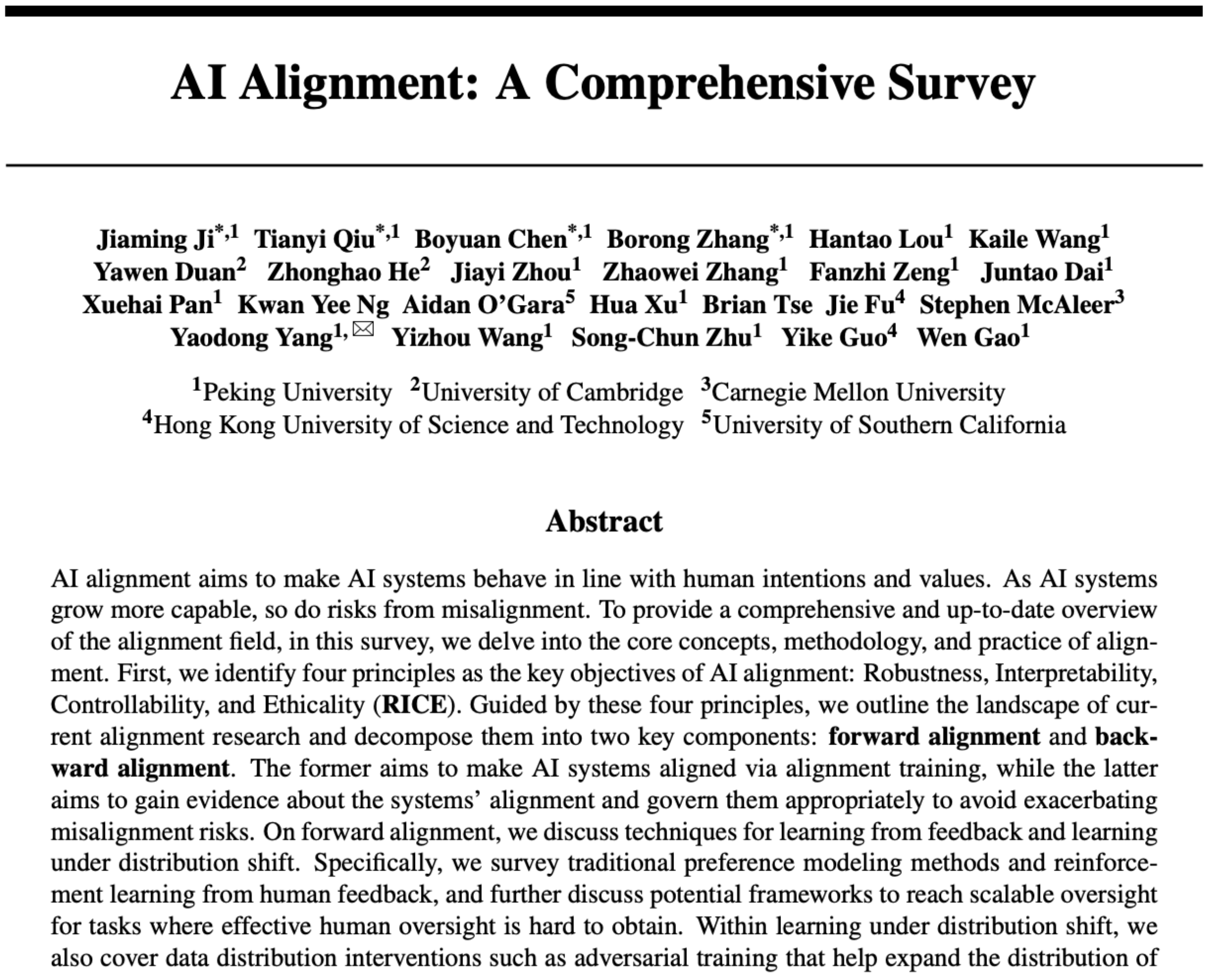
$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\beta \mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\bigg[\underbrace{\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))}_{\text{higher weight when reward estimate is wrong}} \bigg[\underbrace{\nabla_\theta \log \pi(y_w \mid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

RLHF **Alternatives**

- **Many new approaches:**
  - A good survey: Ji et al ’24

- New approaches to rewards, new forms of feedback (including AI feedback), etc

- Popular research area!



AI Alignment: A Comprehensive Survey

Jiaming Ji[*,1] **Tianyi Qiu**[*,1] **Boyuan Chen**[*,1] **Borong Zhang**[*,1] **Hantao Lou**[1] **Kaile Wang**[1]
Yawen Duan[2] **Zhonghao He**[2] **Jiayi Zhou**[1] **Zhaowei Zhang**[1] **Fanzhi Zeng**[1] **Juntao Dai**[1]
Xuehai Pan[1] **Kwan Yee Ng** **Aidan O’Gara**[5] **Hua Xu**[1] **Brian Tse** **Jie Fu**[4] **Stephen McAleer**[3]
Yaodong Yang[1,✉] **Yizhou Wang**[1] **Song-Chun Zhu**[1] **Yike Guo**[4] **Wen Gao**[1]

[1]Peking University [2]University of Cambridge [3]Carnegie Mellon University
[4]Hong Kong University of Science and Technology [5]University of Southern California

Abstract

AI alignment aims to make AI systems behave in line with human intentions and values. As AI systems grow more capable, so do risks from misalignment. To provide a comprehensive and up-to-date overview of the alignment field, in this survey, we delve into the core concepts, methodology, and practice of alignment. First, we identify four principles as the key objectives of AI alignment: Robustness, Interpretability, Controllability, and Ethicality (**RICE**). Guided by these four principles, we outline the landscape of current alignment research and decompose them into two key components: **forward alignment** and **backward alignment**. The former aims to make AI systems aligned via alignment training, while the latter aims to gain evidence about the systems’ alignment and govern them appropriately to avoid exacerbating misalignment risks. On forward alignment, we discuss techniques for learning from feedback and learning under distribution shift. Specifically, we survey traditional preference modeling methods and reinforcement learning from human feedback, and further discuss potential frameworks to reach scalable oversight for tasks where effective human oversight is hard to obtain. Within learning under distribution shift, we also cover data distribution interventions such as adversarial training that help expand the distribution of

Bibliography

- Shen et al ‘23: Junhong Shen, Liam Li, Lucio M. Dery, Corey Staten, Mikhail Khodak, Graham Neubig, Ameet Talwalkar, “Cross-Modal Fine-Tuning: Align then Refine” (https://arxiv.org/abs/2302.05738)
- Chip Huyen: https://huyenchip.com/2023/05/02/rlhf.html
- Nathan Lambert et al: https://huggingface.co/blog/rlhf
- Ouyang et al ‘22: Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, Ryan Lowe, “Training language models to follow instructions with human feedback” (https://arxiv.org/abs/2203.02155)
- Lambert et al ‘24: Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi , “RewardBench: Evaluating Reward Models for Language Modeling” (https://arxiv.org/abs/2403.13787)
- Schulman et al: John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, “Proximal Policy Optimization Algorithms” (https://arxiv.org/abs/1707.06347)
- Yoav Golderbg: https://gist.github.com/yoavg/6bff0fecd65950898eba1bb321cfbd81
- Casper et al: Stephen Casper, Xander Davies, and many others, “Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback” (https://arxiv.org/abs/2307.15217)
- Rafailov et al ‘23: Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, Chelsea Finn, “Direct Preference Optimization: Your Language Model is Secretly a Reward Model” (https://arxiv.org/abs/2305.18290)
- Ji et al ‘24: Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, Fanzhi Zeng, Kwan Yee Ng, Juntao Dai, Xuehai Pan, Aidan O'Gara, Yingshan Lei, Hua Xu, Brian Tse, Jie Fu, Stephen McAleer, Yaodong Yang, Yizhou Wang, Song-Chun Zhu, Yike Guo, Wen Gao, “AI Alignment: A Comprehensive Survey” (https://arxiv.org/abs/2310.19852)

Thank You!