STAT 453: Introduction to Deep Learning and Generative Models

Ben Lengerich

Lecture 20: Aligning LLMs: SFT, RLHF, and DPO

November 17, 2026

Logistics

- **HW5 out later this week**, due Fri 12/4: GPTs and LoRA
- HW4 due Fri 11/20
- Project presentations: Tue 12/1 and Thu 12/3
  - Sign up for a slot if you haven't yet
  - 4-minute talks; upload slides to Canvas 24 hours before your slot
- Project final report due Tue 12/8
- Final exam
  - Saturday, December 12, 5:05–7:05 PM

Ben Lengerich © University of Wisconsin-Madison 2026

Last Time: Unsupervised Training of LLMs

- Pretraining = next-token maximum likelihood on trillions of tokens
  - Scaling laws: loss falls as a power law in parameters, data, and compute
- What MLE does not give us:
  - No notion of truth: likely ≠ correct
  - No task goals, no explicit reward

Ben Lengerich © University of Wisconsin-Madison 2026

Challenges of MLE-based unsupervised training

MLE: A KL-Divergence view

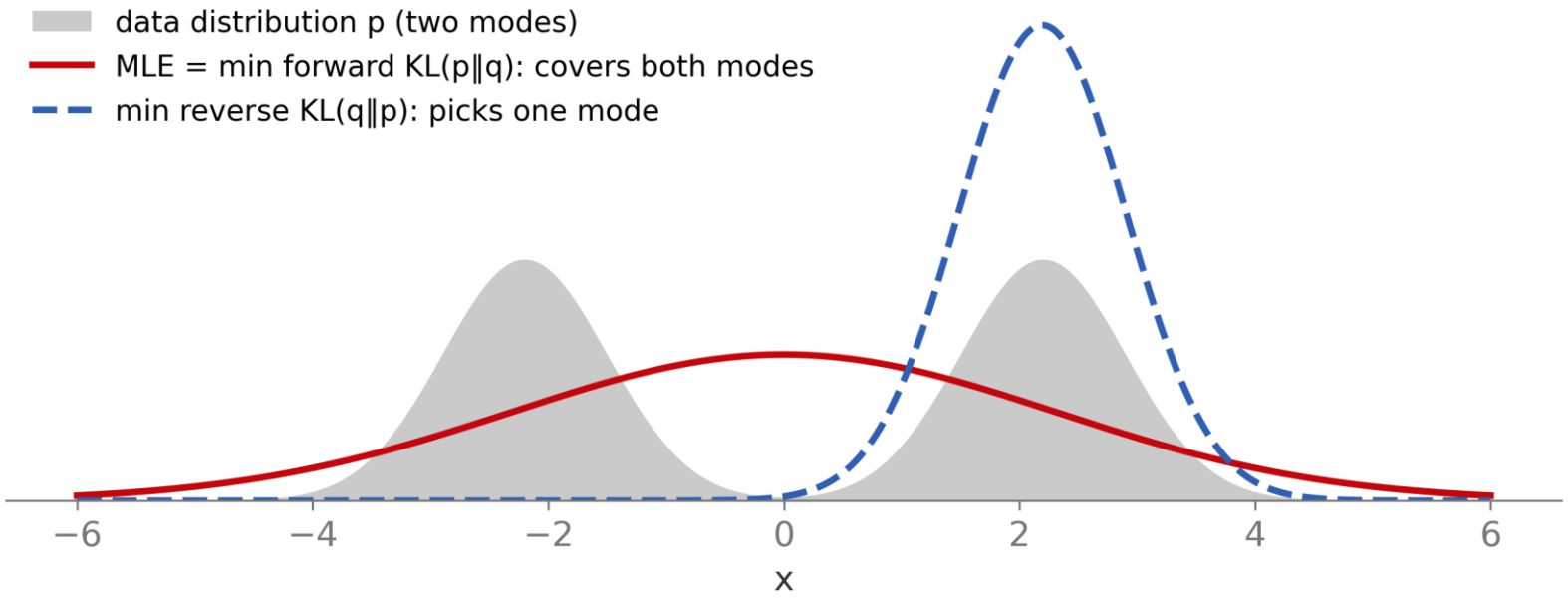
$$KL(P_{data} \,||\, P_\theta) = E_{x \sim P_{data}}[-\log P_\theta(x)] + const.$$

$$\text{argmax}_\theta E_{x \sim P_{data}}[\log P_\theta(x)] = \text{argmin}_\theta KL(P_{data} \,||\, P_\theta)$$

- Forward KL heavily penalizes p_θ(x) ≈ 0 wherever data has mass → **mode-covering**
  - With limited capacity, leftover mass lands between the modes: incoherent continuations get real probability
  - Rare patterns must still get probability, so they can be memorized
- So how we sample from p_θ matters (next slides)

Forward KL covers modes; reverse KL picks one

$$\mathbb{E}_{x\sim p_{\text{data}}}[-\log p_\theta(x)] = H(p_{\text{data}}) + \text{KL}(p_{\text{data}} \,\|\, p_\theta)$$

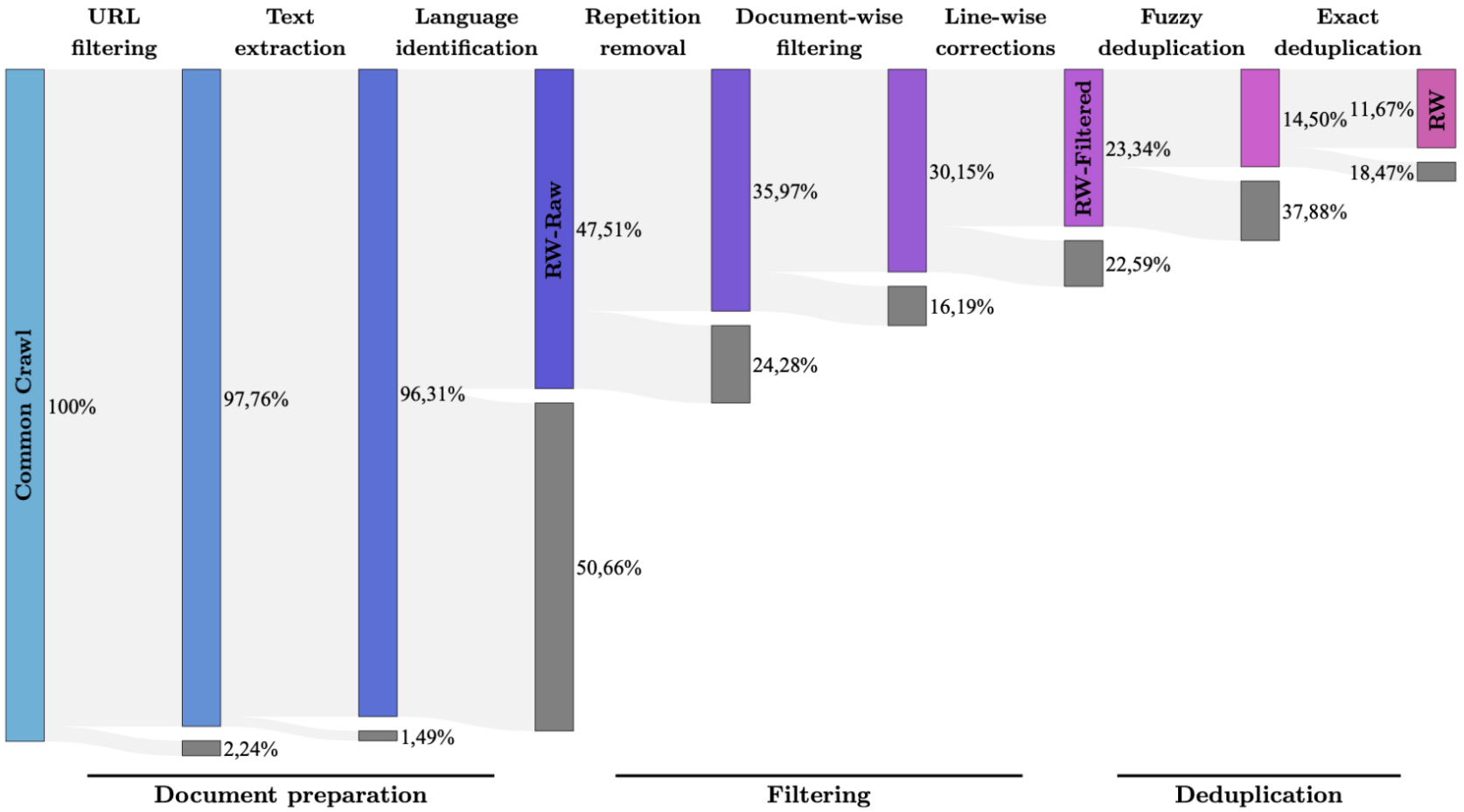


- Cross-entropy loss ≥ entropy of the data: it cannot reach 0
- MLE = minimize forward KL → **mode-covering**
- VAEs (ELBO) use reverse KL → **mode-seeking**

MLE fits the corpus, not the world

- MLE minimizes KL(p_{corpus} ‖ p_θ): the target is the empirical training distribution
- p_{corpus} is not the distribution we want:
  - duplicates, spam, toxic text, and stale facts all get probability in proportion to how often they appear
  - a document repeated k times counts k times
- So data selection is part of the objective: filtering and deduplication change what MLE estimates

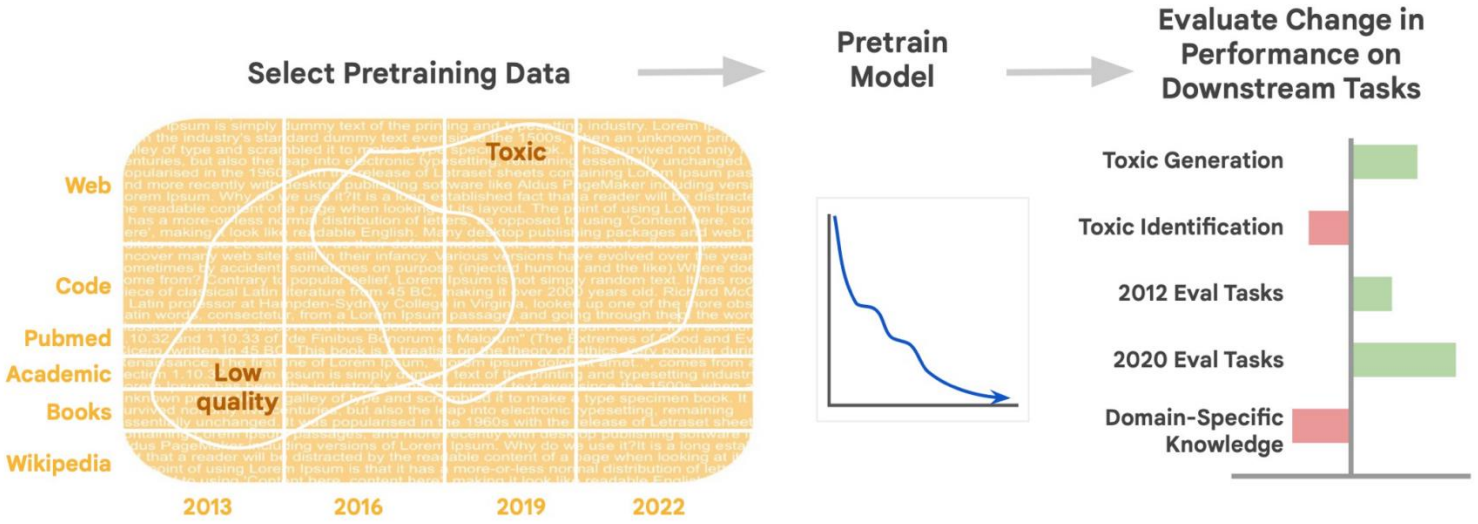
Scale is difficult…for example, data filtering



Penedo, et al. (2023) The Refined Web dataset for Falcon LLM

Scale is difficult…for example, data filtering

A Pretrainer's Guide to Training Data: Measuring the Effects of Data Age, Domain Coverage, Quality, & Toxicity [Longpre et al, NAACL 2024]



Some findings: strongly encourage to read the paper!

- "temporal shift between evaluation data and pretraining data leads to performance degradation, which is not overcome by finetuning"
- "a trade-off between performance on standard benchmarks and risk of toxic generations… there does not exist a one-size-fits-all solution to filtering."

Entropy and Confidence

- Token-level entropy:

$$H_t = -\sum_v P(x_t = v \mid x_{<t}) \log P(x_t = v \mid x_{<t})$$
$$= E_v[-\log P(x_t = v \mid x_{<t})]$$

- Low entropy = high confidence.

Do we want our model to be low or high entropy?

Ben Lengerich © University of Wisconsin-Madison 2026

Recall from Variational Inference

$$\log p(x \mid \theta) = E_{z \sim q}[\log p(x, z \mid \theta)] + H(q) + KL(q(z \mid x) \,||\, p(z \mid x, \theta))$$

$$\log p(x \mid \theta) \geq \underbrace{E_{z \sim q}[\log p(x, z \mid \theta)] + H(q)}$$

"ELBO": Evidence Lower Bound

Maximizing ELBO → Increased entropy of $q(z)$

Does Maximizing Likelihood → Increased entropy of $p(x \mid \theta)$?

Ben Lengerich © University of Wisconsin-Madison 2026

MLE and Entropy: two ways to fail at generation

$$\theta_{\text{MLE}} = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \left[\log p_\theta(x) \right]$$

Too much entropy: sample the full distribution

- MLE spreads mass into an unreliable tail
- Over hundreds of tokens, some sample lands there; later tokens build on the mistake
- → incoherent drift (“pure sampling”)

Too little entropy: always pick the most likely

- Greedy / beam search maximize probability
- Repeating a phrase makes repeating it again more likely: a self-reinforcing loop
- → bland, repetitive text (“beam search”)

Human text sits in between: rarely the most likely token, rarely a very unlikely one

Ben Lengerich © University of Wisconsin-Madison 2026

MLE and Entropy

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

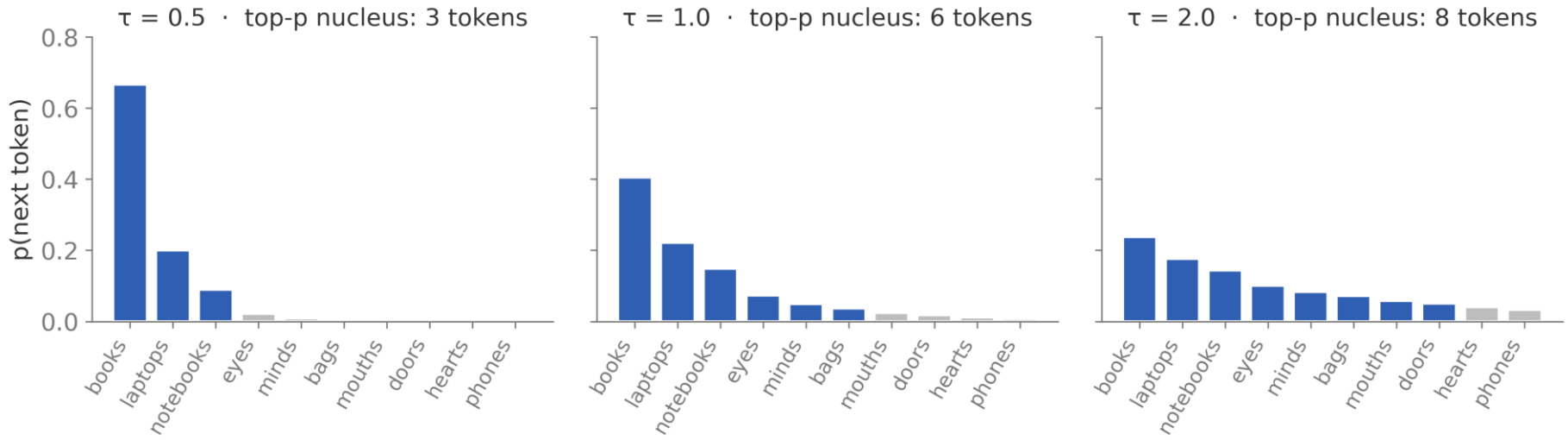
Pure Sampling:
They were cattle called Bolivian Cavalleros; they live in a remote desert uninterrupted by town, and they speak huge, beautiful, paradisiacal Bolivian linguistic thing. They say, 'Lunch, marge.' They don't tell what the lunch is," director Professor Chuperas Omwell told Sky News. "They've only been talking to scientists, like we're being interviewed by TV reporters. We don't even stick around to be interviewed by TV reporters. Maybe that's how they figured out that they're cosplaying as the Bolivian Cavalleros."

Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

THE CURIOUS CASE OF NEURAL TEXT DeGENERATION [Holtzman 2020]

Fix at decoding time: truncate the tail

$$V^{(p)} = \text{smallest set with } \sum_{v \in V^{(p)}} p_\theta(v \mid x_{<t}) \geq p, \qquad p'(v) = \frac{p_\theta(v \mid x_{<t}) \mathbb{1}[v \in V^{(p)}]}{\sum_{u \in V^{(p)}} p_\theta(u \mid x_{<t})}$$



- **Top-k:** fixed number of candidates. **Top-p:** adapts to the model's confidence (blue bars, p = 0.9)
- Removes the unreliable tail but keeps sampling, so it avoids greedy's repetition loops

Fix at training time: change the objective

Entropy bonus $\max_{\theta} \mathbb{E}\big[\log p_\theta(x)\big] + \lambda \sum_t H\big[p_\theta(\cdot \mid x_{<t})\big]$

Label smoothing $y_{\text{true}} = 1 - \epsilon, \qquad y_{\text{other}} = \dfrac{\epsilon}{V-1}$

Unlikelihood $-\sum_{c \in \mathcal{C}_t} \log\big(1 - p_\theta(c \mid x_{<t})\big) \quad (\mathcal{C}_t : \text{ e.g. tokens already used})$

- **Exposure bias:** training conditions on true prefixes, generation on the model's own. **Scheduled sampling** sometimes feeds it its own predictions during training
- **Rewards and preferences:** optimize what we actually want, not likelihood → **today**

Today: Aligning LLMs

- **Supervised fine-tuning (SFT):** imitate good demonstrations
- **Preferences:** humans compare responses → a reward model
- **RLHF:** optimize the learned reward, on a leash (KL to a reference model)
- **DPO:** the same objective, with no reward model and no RL loop
- **What can go wrong:** over-optimization, reward hacking, overconfidence

Ben Lengerich © University of Wisconsin-Madison 2026

What does MLE not do?

- No **task goals**
- No **explicit reward**
- No utility
- Dataset selection drives everything

Can we fine-tune our model to be **useful** after learning unsupervised P(X) learning?

Ben Lengerich © University of Wisconsin-Madison 2026

Post-Training as Utility Optimization

- Instead of $p_\theta(\, x_{t+1} \mid x_{<t}\,)$, we want to optimize utility:

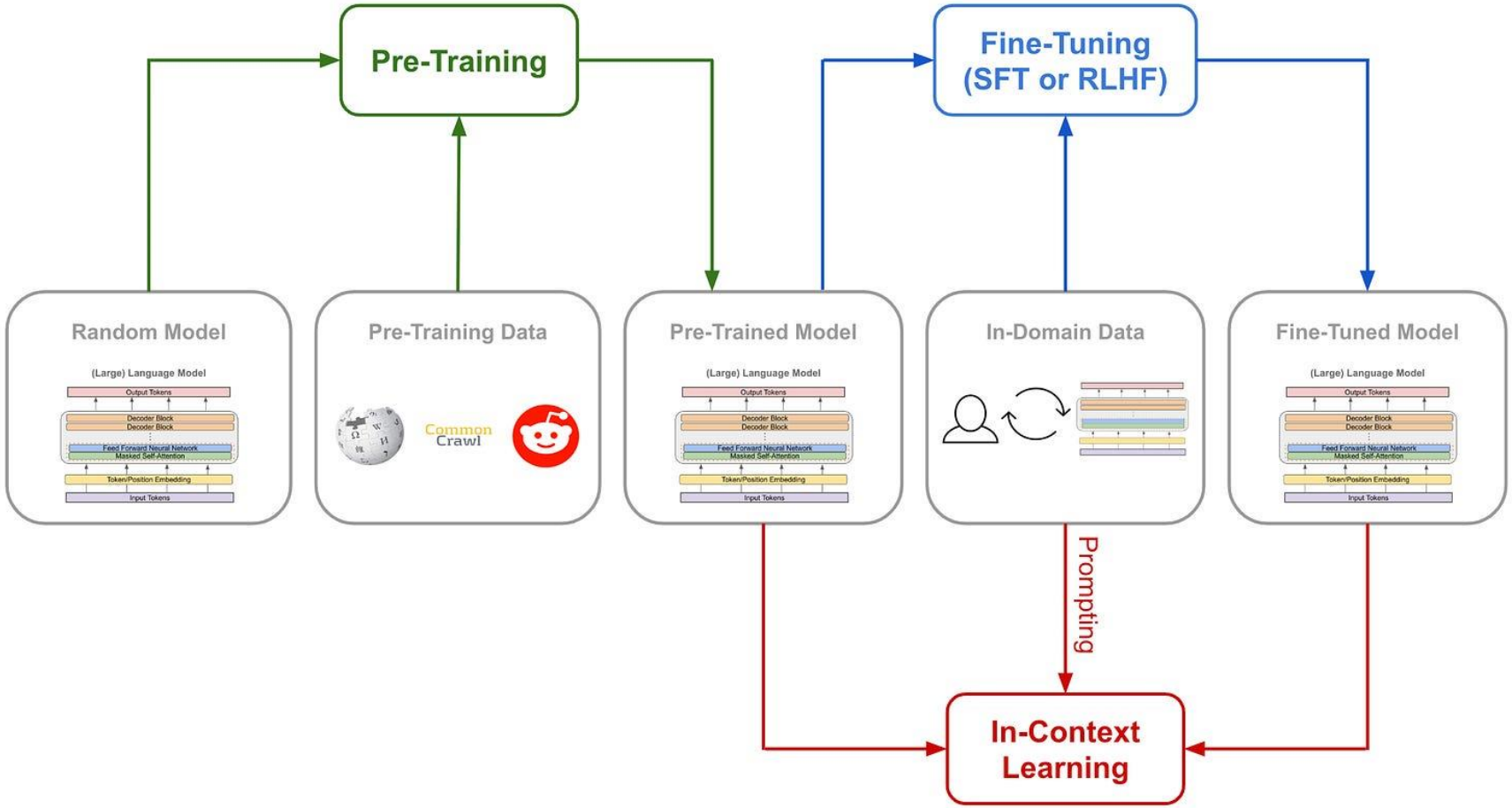
$$\max_\theta E_{x,y\sim\pi_\theta}[R(x,y)]$$

- Problem: Reward R is **unknown**
- Must be learned from humans or verification signals
- Main approaches:

Method	Reward Source
SFT	demonstrations
RLHF	learned reward model
DPO	preference likelihood
RLVR	programmatic reward

From Unsupervised to Supervised

- Can we directly train toward **utility** via **explicit rewards?**



https://cameronrwolfe.substack.com/p/understanding-and-using-supervised

Supervised Fine-Tuning

Supervised Fine-Tuning (SFT)

- Show the language model how to appropriately respond to prompts of different types
- “Behavior cloning”
- InstructGPT

Training language models to follow instructions with human feedback

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***

OpenAI

Prompt	Response
Serendipity means the occurrence and development of events by chance in a happy or beneficial way. Use the word in a sentence.	Running into Margaret and being introduced to Tom was a fortunate stroke of serendipity.
ELI5: What's the cause of the "anxiety lump" in our chest during stressful or disheartening experiences?	The anxiety lump in your throat is caused by muscular tension keeping your glottis dilated to maximize airflow. The clenched chest or heartache feeling is caused by the vagus nerve which tells the organs to pump blood faster, stop digesting, and produce adrenaline and cortisol.
Create a shopping list from this recipe: Trim the ends off zucchini. Cut zucchini in half lengthwise; scoop out pulp, leaving 1/2-in. shells. Finely chop pulp. In a skillet, cook beef, zucchini pulp, onion, mushrooms and peppers over medium heat until meat is no longer pink; drain. Remove from the heat. Add 1/2 cup cheese, ketchup, salt and pepper; mix well. Spoon into the zucchini shells. Place in a greased 13x9-in. baking dish. Sprinkle with remaining cheese.	Zucchini, beef, onion, mushroom, peppers, cheese, ketchup, salt, pepper

https://huyenchip.com/2023/05/02/rlhf.html

Ben Lengerich © University of Wisconsin-Madison 2026

Supervised Fine-Tuning (SFT)

- Show the language model how to appropriately respond to prompts of different types
- “Behavior cloning”
- Dataset: (x_i, y_i)
- Loss: $L_{SFT} = -E_{(x,y)} \log p_\theta(y \mid x)$

- Limitations:
  - copies **surface behavior**
  - cannot represent **tradeoffs between responses**

Ben Lengerich © University of Wisconsin-Madison 2026

Inside the SFT loss

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\sum_{t=1}^{|y|} \log p_\theta\left(y_t \mid x,\ y_{<t}\right)$$

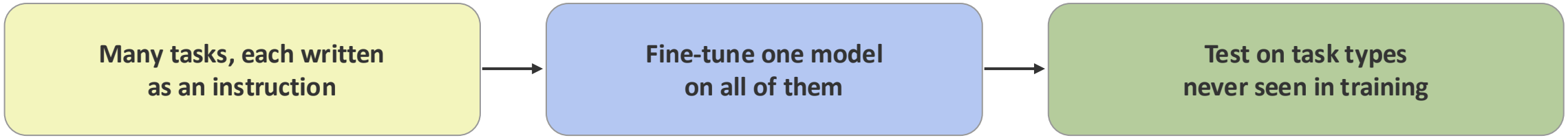


prompt x: context only, no loss

response y: every token is a target

- Same next-token loss as pretraining, **masked to the response**
- **Teacher forcing:** each target is predicted from the true previous response tokens
- At generation, it conditions on its own previous tokens → errors can compound (**exposure bias**)
- Tiny data by pretraining standards: ~10^4–10^5 demonstrations vs. ~10^{13} tokens

Instruction tuning generalizes to new tasks (FLAN)

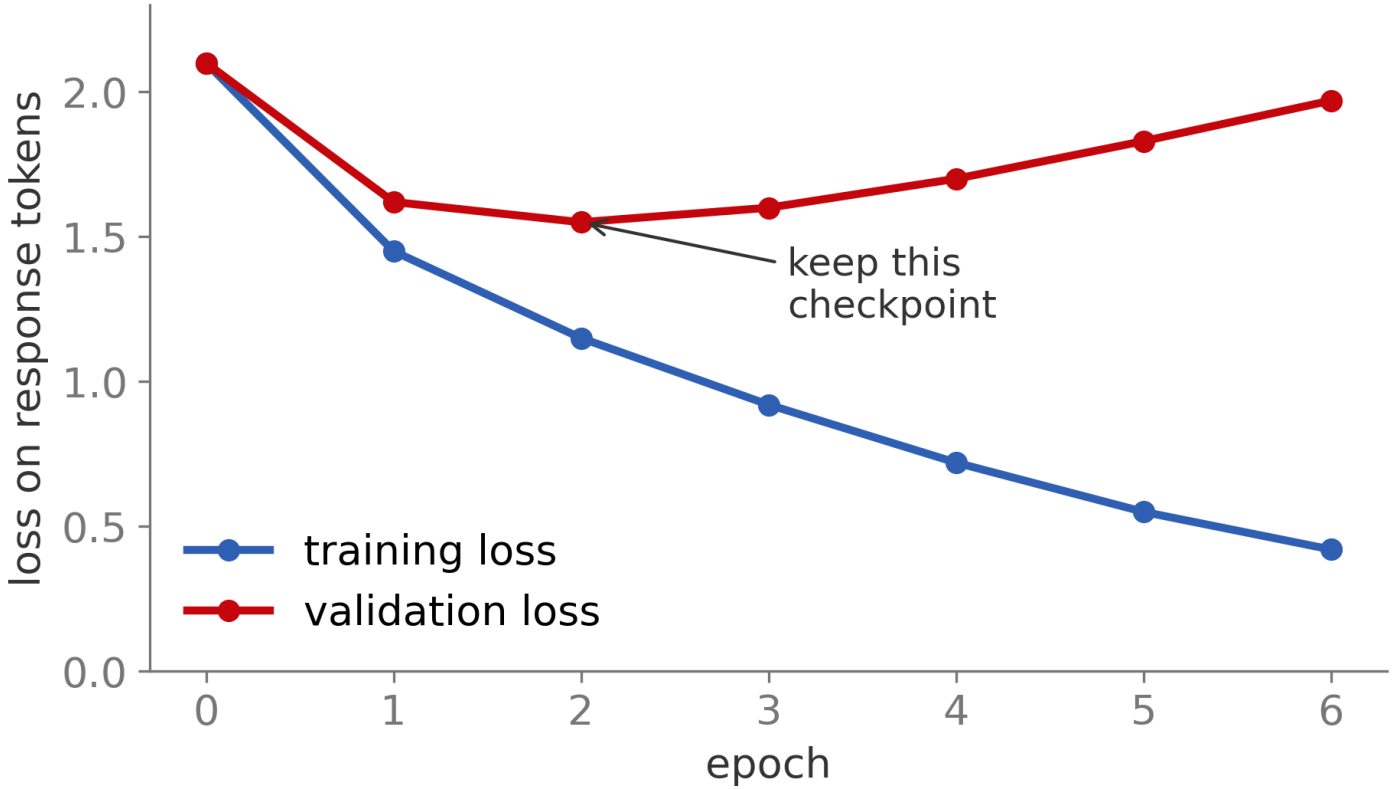


"Translate to French: …" "Is this review positive or negative? …" "Answer the question: …"

- 137B model, >60 datasets: beats **zero-shot 175B GPT-3 on 20 of 25** held-out tasks
- Helps more with **more tasks** and with **larger models**
- Lesson: SFT teaches the format "follow the instruction," not just individual tasks

Wei et al., ICLR 2022

SFT overfits fast



- Thousands of examples, billions of parameters
- Training loss keeps falling; validation loss turns up after a few epochs
- Keep the **lowest validation loss**; typically 1–3 epochs
- The usual train/validation logic, applied to fine-tuning

Ben Lengerich © University of Wisconsin-Madison 2026

Model Distillation: A type of SFT

- Dataset: $(x_i, y_{teacher})$
- Loss: $L_{distill} = -E_x \log p_\theta(y_{teacher} \mid x)$

- Benefits:
  - transfer reasoning ability
  - smaller deployment models, cheaper inference
- Example:
  - Phi models
  - Gemma
  - Many open reasoning models

Ben Lengerich © University of Wisconsin-Madison 2026

Model Distillation: Legal?

From Anthropic Feb 23, 2026

We have identified industrial-scale campaigns by three AI laboratories—DeepSeek, Moonshot, and MiniMax—to illicitly extract Claude's capabilities to improve their own models. These labs generated over 16 million exchanges with Claude through approximately 24,000 fraudulent accounts, in violation of our terms of service and regional access restrictions.

These labs used a technique called "distillation," which involves training a less capable model on the outputs of a stronger one. Distillation is a widely used and legitimate training method. For example, frontier AI labs routinely distill their own models to create smaller, cheaper versions for their customers. But distillation can also be used for illicit purposes: competitors can use it to acquire powerful capabilities from other labs in a fraction of the time, and at a fraction of the cost, that it would take to develop them independently.

These campaigns are growing in intensity and sophistication. The window to act is narrow, and the threat extends beyond any single company or region. Addressing it will require rapid, coordinated action among industry players, policymakers, and the global AI community.

Ben Lengerich © University of Wisconsin-Madison 2026

Learning from Preferences

Supervised Fine-Tuning (SFT)

- But real responses involve **multiple competing objectives**
- Ex:
  - accuracy vs simplicity
  - brevity vs completeness
  - caution vs usefulness

- SFT only sees **one chosen response** and cannot represent these tradeoffs.

Ben Lengerich © University of Wisconsin-Madison 2026

Why Preference Learning Helps

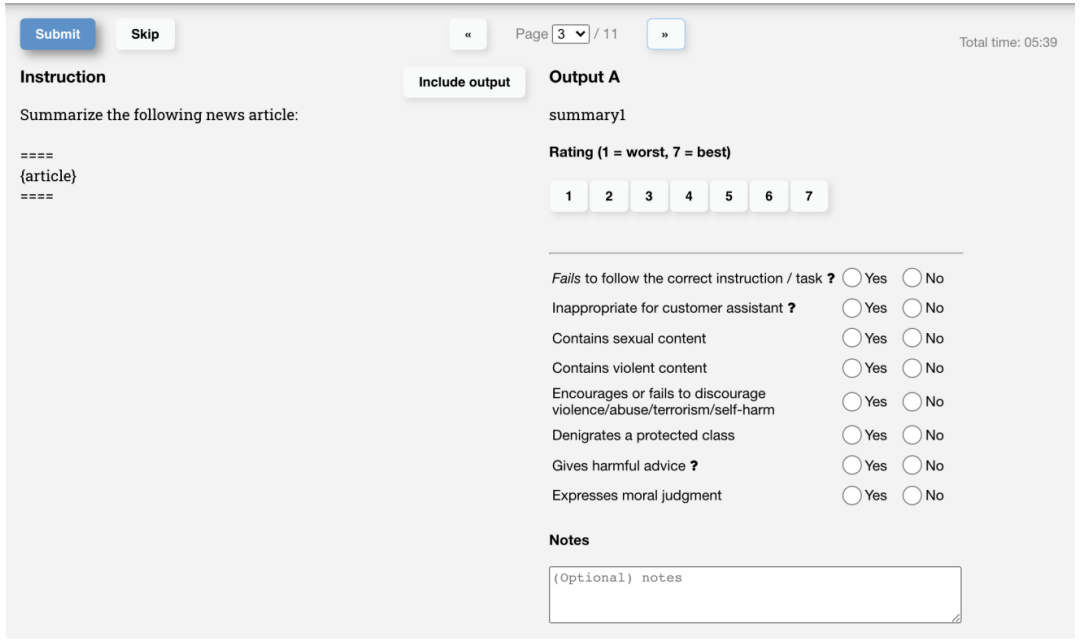
- Instead of demonstrations (x_i, y_i)
- Preference learning uses comparisons: (x, y_w, y_l) where $y_w \succ y_l$
- This allows the model to learn **relative quality** rather than copying a single example.

Ben Lengerich © University of Wisconsin-Madison 2026

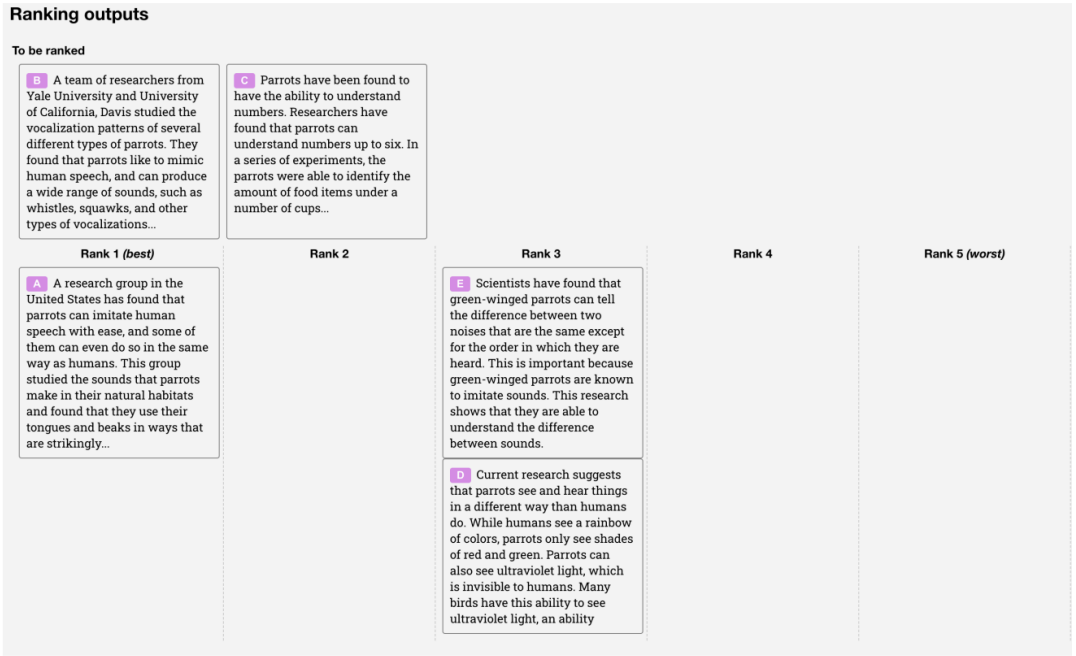
Getting data from users

OpenAI UI

- Can make a dataset of user preferences by offering multiple responses to prompts and having the user pick which they prefer



(a)



(b)

https://huyenchip.com/2023/05/02/rlhf.html

Reward model: a classifier on pairs (Bradley–Terry)

$$P(y_w \succ y_l \mid x) = \sigma\big(r_\phi(x, y_w) - r_\phi(x, y_l)\big)$$

$$\mathcal{L}_{\mathrm{RM}}(\phi) = -\,\mathbb{E}_{(x,y_w,y_l)}\Big[\log \sigma\big(r_\phi(x, y_w) - r_\phi(x, y_l)\big)\Big]$$

- Data: prompt x, preferred response y_w, rejected response y_l
- Model: the SFT model with a **scalar head**: (prompt, response) ↦ score
- Logistic regression on the score difference
- **Only differences matter:** r + c gives the same predictions → rewards have no absolute scale

Where RLHF started: a backflip from 900 comparisons

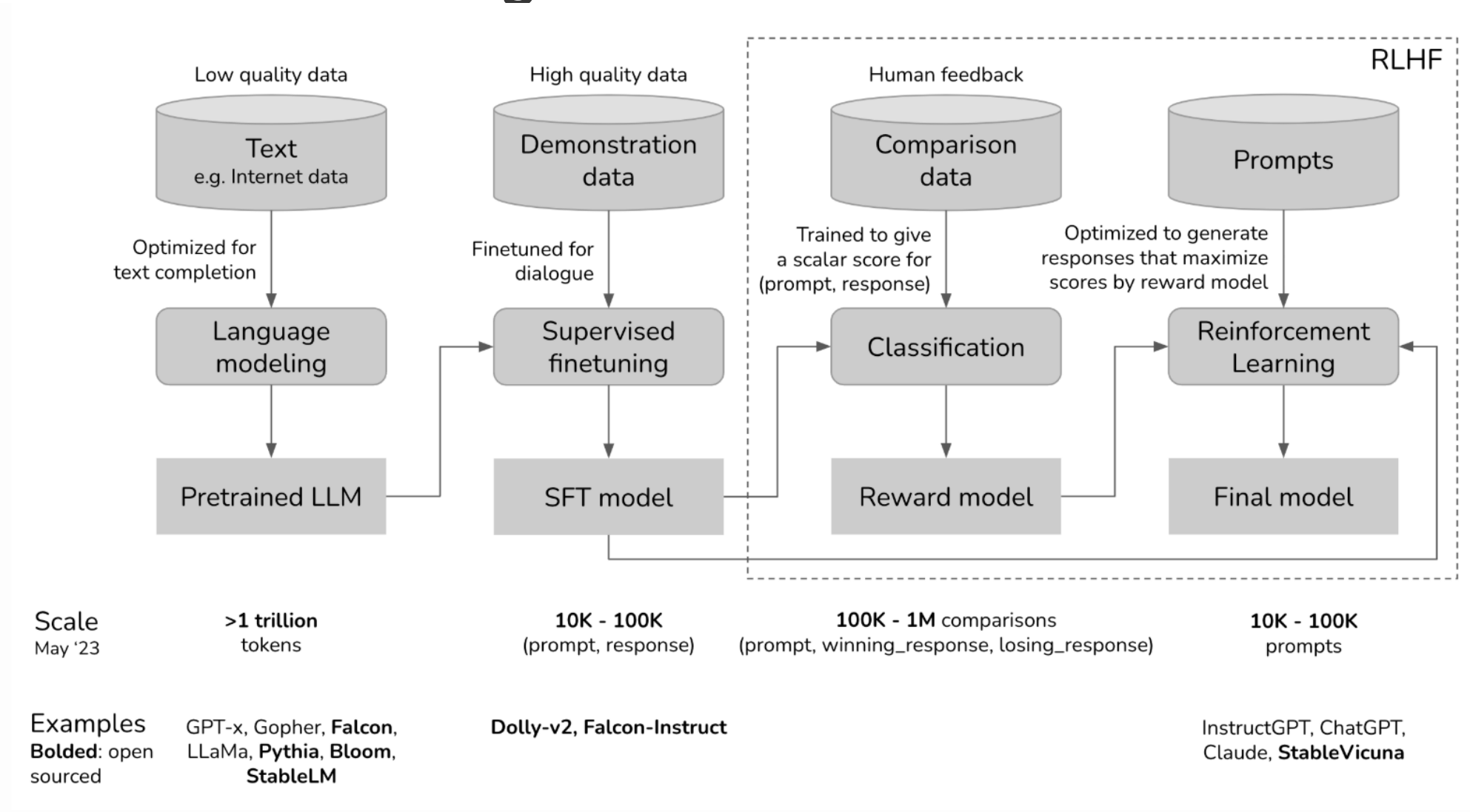


- Simulated robot learned a backflip from **~900 comparisons, under an hour** of human time
- Feedback on **under 1%** of the agent's interactions
- Useful when a goal is easy to recognize but hard to write as a reward
- InstructGPT (2022): the same recipe, with text responses instead of video clips

Christiano et al., NeurIPS 2017

RLHF: Optimizing a Learned Reward

Reinforcement Learning with Human Feedback



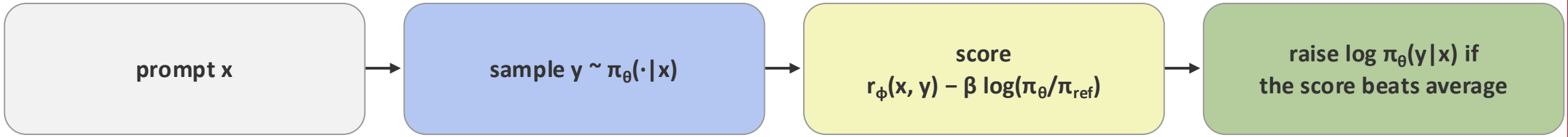
https://huyenchip.com/2023/05/02/rlhf.html

The RLHF objective: reward on a leash

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

- π_θ: the policy (the LLM being trained); π_{ref}: the SFT model, frozen
- **Why the KL term?** r_ϕ was trained on responses near π_{ref}; far from them its scores are unreliable
  - $\beta \to 0$: chase the reward model wherever it leads → exploits its errors
  - $\beta \to \infty$: never move from the SFT model
- Also keeps outputs fluent and diverse: the policy cannot collapse onto one high-scoring answer

The RL step: sample, score, update



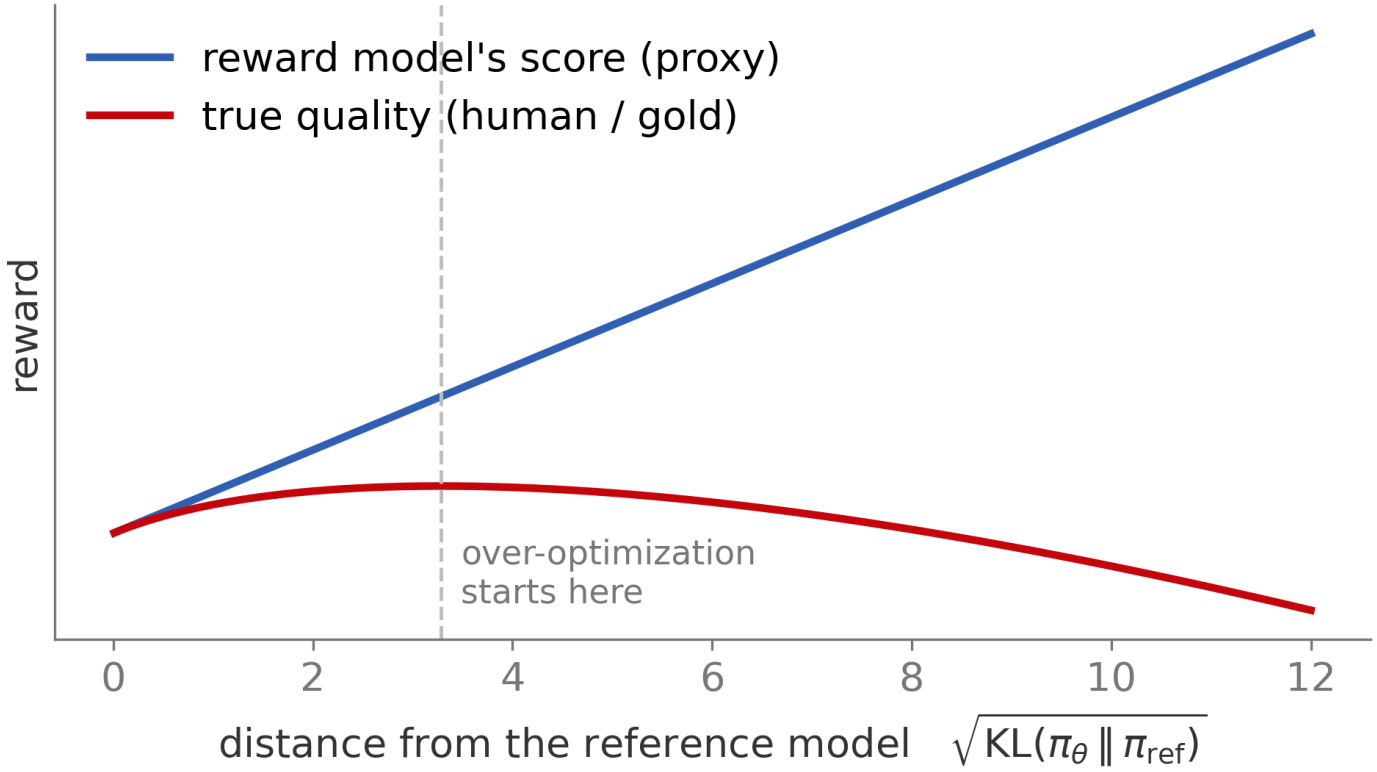
$$\nabla_\theta \sum_y \pi_\theta(y \mid x) R(y) = \sum_y \pi_\theta(y \mid x)\, R(y)\, \nabla_\theta \log \pi_\theta(y \mid x) = \mathbb{E}_{y \sim \pi_\theta}\left[R(y)\, \nabla_\theta \log \pi_\theta(y \mid x)\right]$$

using $\nabla\pi = \pi\, \nabla \log \pi$ (the log-derivative trick)

- We can't backpropagate through sampling, and tokens are discrete, so the VAE's reparameterization trick doesn't apply
- Instead, raise $\log \pi_\theta(y|x)$ in proportion to how good y was (subtract a baseline b, e.g. the average reward, to reduce variance)
- Targets are the **model's own samples**, not fixed human responses: this is **not** cross-entropy on preferred answers
- InstructGPT used PPO, a stabilized version of this update

Ben Lengerich © University of Wisconsin-Madison 2026

Over-optimizing the reward model (Goodhart's law)



- The reward model is a **proxy** for what people want
- Push hard enough and the policy finds outputs it **overrates**: reward hacking
- Symptom: reward-model score ↑ while human ratings ↓
- Fixes: larger β, stop early, better / larger reward models

Schematic after Gao, Schulman & Hilton, 2023

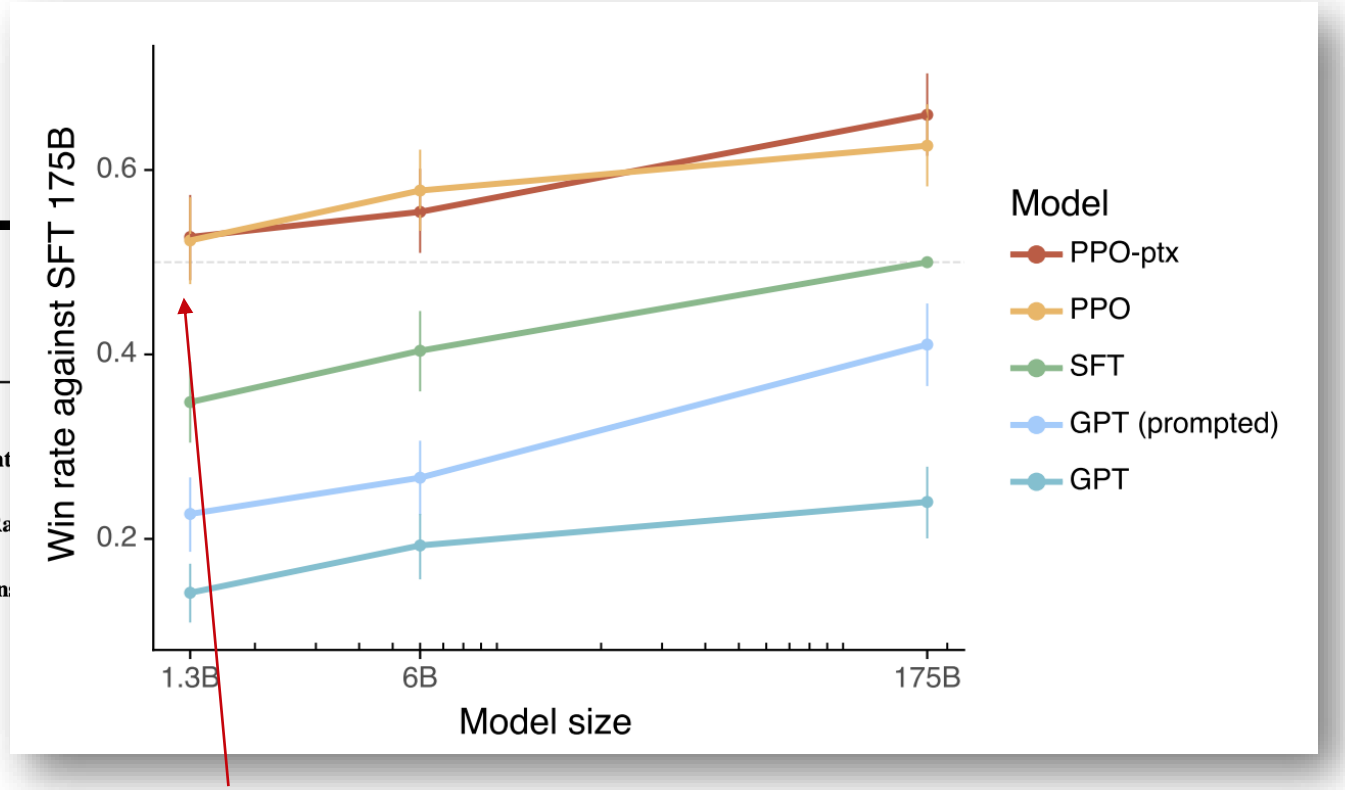
Ben Lengerich © University of Wisconsin-Madison 2026

RLHF works: InstructGPT

- SFT → reward model → PPO
- Human raters preferred a 1.3B RLHF model to 175B GPT-3
- InstructGPT

Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright
Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ra
John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens
Amanda Askell† Peter Welinder Paul Christiano*†
Jan Leike* Ryan Lowe*
OpenAI



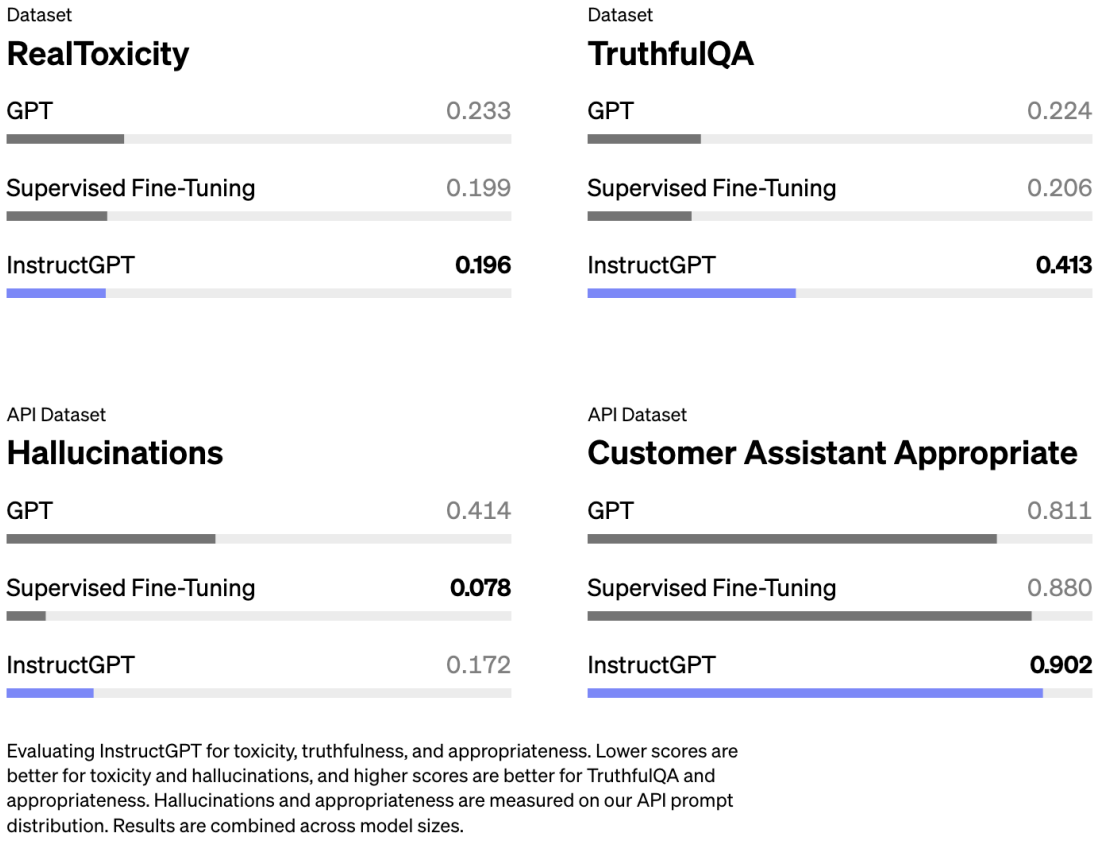
1.3B model can outperform 175B model

Does human feedback reduce model hallucinations?

How to Fix with RL

- 1) Adjust output distribution so model is allowed to express uncertainty, challenge premise, admit error. (Can use behavior cloning.)
- 2) Use RL to precisely learn behavior boundary.
  - Reward(x) = {
    1 if unhedged correct (The answer is y)
    0.5 if hedged correct (The answer is likely y)
    0 if uninformative (I don't know)
    -2 if hedged wrong (The answer is likely z)
    -4 wrong (The answer is z)
    }
  - This reward is similar to log loss, or a proper scoring rule

John Schulman 2023



Evaluating InstructGPT for toxicity, truthfulness, and appropriateness. Lower scores are better for toxicity and hallucinations, and higher scores are better for TruthfulQA and appropriateness. Hallucinations and appropriateness are measured on our API prompt distribution. Results are combined across model sizes.

https://huyenchip.com/2023/05/02/rlhf.html

Direct Preference Optimization

DPO: the RLHF objective has a closed-form optimum

1. Optimal policy for a given reward r:

$$\pi^*(y \mid x) = \frac{1}{Z(x)}\,\pi_{\text{ref}}(y \mid x)\,\exp\big(r(x,y)/\beta\big)$$

$$\text{since}\quad \mathbb{E}_{y\sim\pi}[r] - \beta\,\mathrm{KL}(\pi \,\|\, \pi_{\text{ref}}) = -\beta\,\mathrm{KL}(\pi \,\|\, \pi^*) + \beta \log Z(x)$$

2. Invert: the reward is a log-ratio of policies

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

3. Plug into Bradley–Terry: only $r(x, y_w) - r(x, y_l)$ appears, so **Z(x) cancels**

- The LM is "secretly a reward model":

$$\hat{r}_\theta(x,y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)}$$

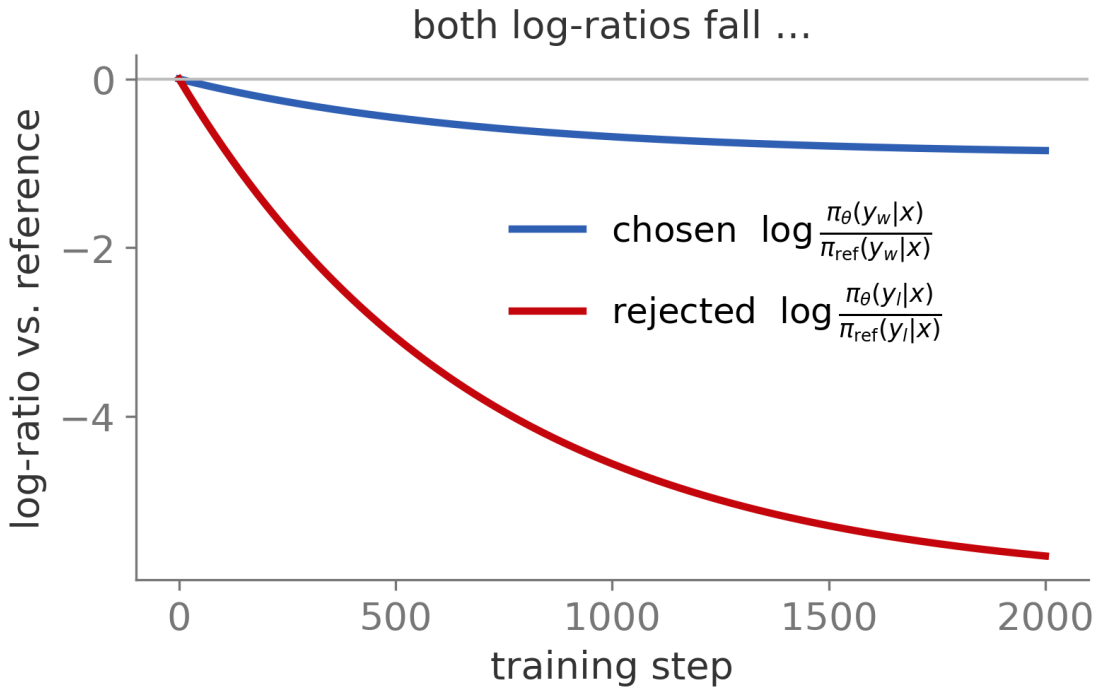
Rafailov et al., NeurIPS 2023

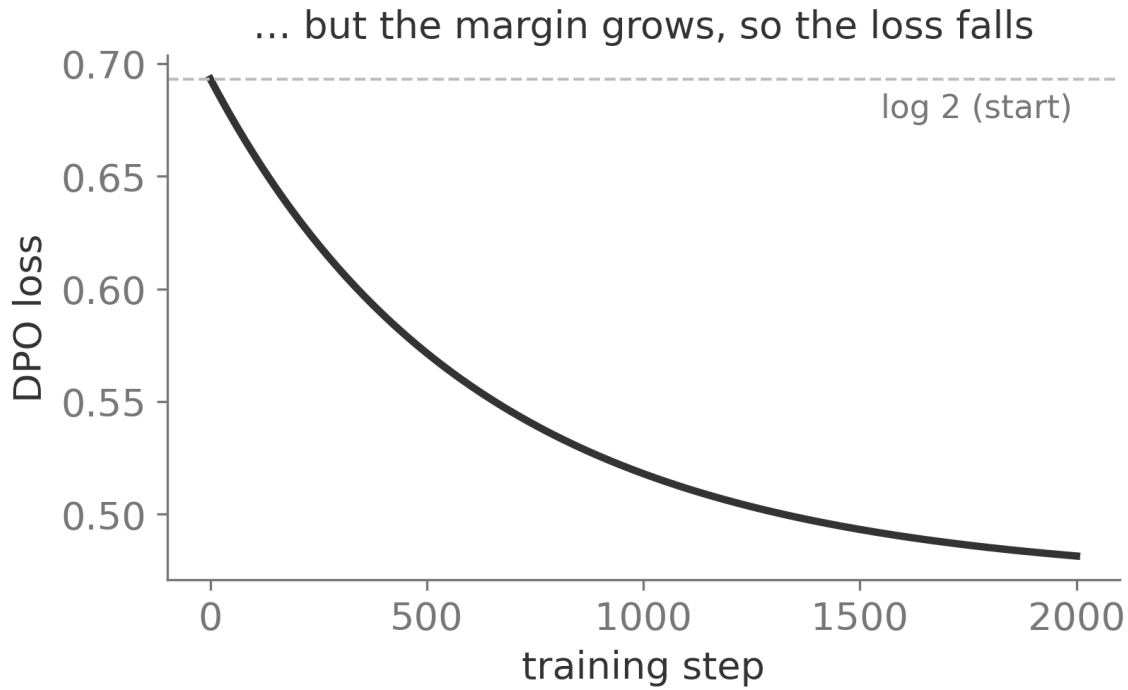
The DPO loss

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

- A **classification loss on preference pairs**: no reward model, no sampling, no RL loop
- Pushes up the chosen response, pushes down the rejected one, **relative to π_{ref}**
  - Most weight on pairs the model currently ranks the wrong way
- **β plays the same role as in RLHF:** larger β keeps π_θ closer to π_{ref}
- At initialization ($\pi_\theta = \pi_{ref}$) the loss is $-\log\sigma(0) =$ **log 2**

Reading DPO training curves





- The loss depends only on the **margin** between the two log-ratios
- Both can fall (probability moves to other responses), and the loss still drops as long as the margin grows

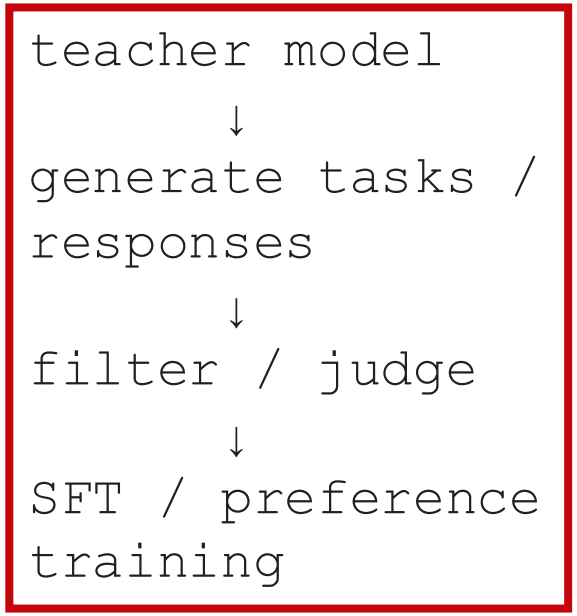
RLHF vs. DPO

	RLHF (e.g. PPO)	DPO
Reward	Explicit reward model r_ϕ	Implicit: $\beta \log \pi_\theta / \pi_{ref}$
Training data	Fresh samples from the policy, scored by r_ϕ	A fixed dataset of preference pairs
Optimization	RL loop: sample, score, policy gradient	One supervised-style loss
Strengths	Can improve beyond the pairs it was given	Simple, stable, cheap
Weaknesses	Unstable, expensive, reward hacking	Limited to its pairs; can overfit them

Ben Lengerich © University of Wisconsin-Madison 2026

In Practice

Where to get training data?



Pipeline

- Distill another model
- Generate synthetic data
  - Ex: Self-Instruct, UltraChat
  - Reasoning datasets
  - Code generation datasets
- Humans, in particular model users

Failure Modes of Post-Training

- Reward hacking
  - Model learns reward model artifacts.
- Preference overfitting
  - Optimizing human preferences ≠ truth.
- Mode collapse
  - RLHF reduces response diversity.
- Distribution shift
  - Alignment trained on narrow prompts.
- Over-confidence
  - RLHF encourages confident answers even when the model is uncertain.

Ben Lengerich © University of Wisconsin-Madison 2026

The Post-Training Stack

Stage	Objective
Pretraining	learn language distribution
SFT	imitate desired behavior
Preference learning	learn reward
RL / DPO	optimize responses
PEFT	efficient adaptation

Modern LLMs are not a single training algorithm.
They are a layered optimization pipeline.

Ben Lengerich © University of Wisconsin-Madison 2026

Next Time: Reasoning Models

Reasoning Models and RL from Verifiable Rewards

Replace the learned reward model with a program that checks the answer

Ben Lengerich © University of Wisconsin-Madison 2026

Questions?

