COSI-230B: Natural Language Annotation for Machine Learning

Lecture 18: Preference Annotation and RLHF in Practice

Jin Zhao

Brandeis University

April 13, 2026

Today's Agenda

Part I: Foundations (30 min)

- From reference outputs to preferences
- Stiennon et al. deep dive
- The RLHF pipeline (diagram)
- What a reward model really is

Part II: Key Systems (25 min)

- InstructGPT: the 3-stage annotation stack
- DPO: preference optimization without RL
- Constitutional AI + critique-revise loop

Part III: Data Quality & Demos (30 min)

- HH-RLHF dataset format
- Preference data quality pitfalls
- Artifact catalog: biases at scale
- Live demos: length bias + position bias
- AlpacaFarm: simulating human feedback

Part IV: Practice & Wrap-Up (25 min)

- QC and modeling mitigations
- In-class annotation activity (15 min)
- Discussion + Key takeaways
- Readings and next class

Goal: Understand what preferences measure, how they distort, and how models exploit them.

Why We Moved from "Reference Outputs" to Preference Data

Before (supervised):

- Gold reference output
- ROUGE/BLEU against reference
- Assumes single "correct" answer
- Works for translation, not dialogue

After (preference-based):

- Comparative judgments
- "A is better than B" (easier than rating)
- Captures quality beyond exact match
- Scales to open-ended generation

Stiennon et al. (2020)

ROUGE/reference matching is a proxy; direct human preference comparisons improved summarization quality relative to those proxies—illustrating that "annotation target selection" matters.

Stiennon et al. (2020). Learning to Summarize from Human Feedback. NeurIPS. https://arxiv.org/abs/2009.01325

Stiennon et al. (2020): Annotation Protocol Details

Task: Summarize Reddit posts (TL;DR dataset).

Candidate generation:

- Sample K candidate summaries from the current policy (temperature sampling)
- Present pairs to annotators (combinatorial from K)
- Early rounds: compare against human-written references
- Later rounds: compare model-vs-model outputs

Annotator setup:

- Contracted annotators via Scale AI
- **Instructions:** "Which summary better captures the important information from the post, without including unnecessary information?"
- No explicit rubric—overall quality judgment
- Annotators could flag ties (but forced choice preferred)

Stiennon et al. (2020). Learning to Summarize from Human Feedback. NeurIPS. https://arxiv.org/abs/2009.01325

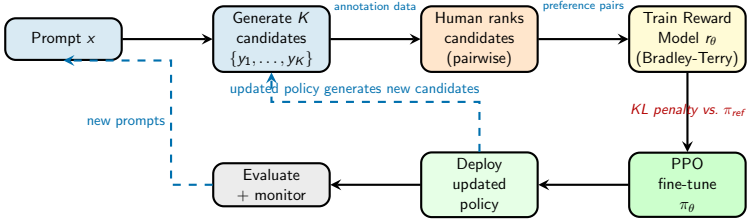
Stiennon et al. (2020): Annotation Protocol Details (cont.)

Key Annotation Insight

The quality of the candidates shown to raters changes over training iterations—creating a **non-stationary annotation distribution**. The reward model must generalize to outputs it was never trained to judge.

Stiennon et al. (2020). Learning to Summarize from Human Feedback. NeurIPS. `https://arxiv.org/abs/2009.01325`

The RLHF Pipeline: End-to-End



Key points:

- The **feedback loop** means annotation quality affects all downstream iterations
- Bradley-Terry: $P(y_1 \succ y_2 \mid x) = \sigma(r_\theta(x, y_1) - r_\theta(x, y_2))$
- KL regularization prevents the policy from drifting too far from the SFT baseline

What a Reward Model Really Is

Core Idea

A reward model is a **pairwise classifier**: given a prompt and two responses, predict which one was "chosen" vs. "rejected."

Bradley-Terry model:

$$P(y_1 \succ y_2 \mid x) = \sigma\big(r_\theta(x, y_1) - r_\theta(x, y_2)\big)$$

What the RM learns:

- A scalar "reward" for each (prompt, response) pair
- Trained on human preference comparisons
- Optimized as a ranking loss, not a classification loss

Annotation Consequence

The RM learns to predict *what raters preferred*—including their biases. If raters systematically prefer longer or more polite outputs, the RM will too.

InstructGPT: The Full Annotation Stack

Three annotation stages:

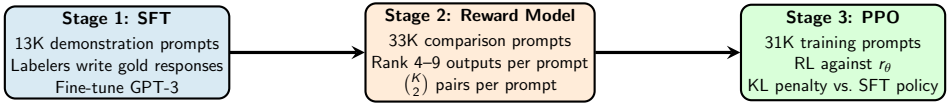
1. **Demonstration data (SFT):** Labelers write ideal responses to prompts
2. **Comparison data (RM):** Labelers rank 4–9 model outputs per prompt
3. **RL training:** PPO against learned reward model

Annotation design details:

- $\sim$40 contractors screened for instruction sensitivity
- Explicit guidelines for helpfulness, truthfulness, harmlessness
- **Limitation acknowledged in paper:** "Human feedback depends on contractor values/background"
- Labeler pool size is *small*—rater pool composition *is* the signal

Ouyang et al. (2022). Training language models to follow instructions with human feedback. NeurIPS.
https://arxiv.org/abs/2203.02155

InstructGPT: The 3-Stage Process by the Numbers



Stage	Prompts	Labelers	Annotation Type
SFT demonstrations	∼13,000	40 contractors	Write ideal response
RM comparisons	∼33,000	40 contractors	Rank 4–9 candidates
PPO training	∼31,000	(none—uses RM)	Automated via r_θ

Annotation Bottleneck

The entire system is shaped by ∼40 people's preferences. Prompt distribution came from OpenAI API users—skewing toward English, tech-savvy demographics.

DPO: Preference Optimization Without RL

Rafailov et al. (2023): Bypass the reward model entirely.

DPO Loss

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

Why DPO matters for annotation:

- Simpler pipeline $\rightarrow$ encourages *more* preference data collection
- Data format (chosen/rejected pairs) directly drives training
- Same bias risks as RLHF, but now *entirely* encoded in the data
- No reward model to audit—biases are harder to detect

Rafailov et al. (2023). Direct Preference Optimization: Your Language Model is Secretly a Reward Model. NeurIPS.
https://arxiv.org/abs/2305.18290

Constitutional AI: Principles Instead of Labels

Bai et al. (2022): Replace human preference labels with **AI self-critique** guided by a set of principles (a "constitution").

Pipeline:

1. Generate response
2. AI critiques response against principles
3. AI revises response
4. Use revised pairs as preference data

Annotation shift:

- From labeling *outputs* to defining *principles*
- Annotation effort moves "upstream"
- Scalable but assumes principles are sufficient
- Task assumptions: works best for safety/helpfulness

Bai et al. (2022). Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073.

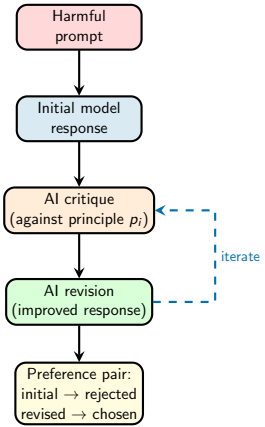
https://arxiv.org/abs/2212.08073

Constitutional AI: Principle Examples & Critique-Revise Loop

Example principles from the constitution:

1. "Choose the response that is least likely to be used for illegal or harmful activities."
2. "Which response is most respectful of everyone's right to their own beliefs?"
3. "Choose the response that sounds most similar to what a peaceful, ethical person would say."
4. "Choose the response that is least racist, sexist, or socially biased."
5. "Choose the answer that is most supportive and encouraging of life."

16 principles total in the Anthropic constitution.



Constitutional AI: Principle Examples & Critique-Revise Loop (cont.)

Key Insight

RLAIF: the revised pairs train a reward model with no human labels—"RL from AI Feedback." Annotation is principle engineering, not labeling.

HH-RLHF Dataset Format

Anthropic's HH-RLHF: "chosen/rejected" preference pairs—now a *de facto* standard format.

Example Preference Pair

Prompt: "How do I make pasta from scratch?"

Chosen: "To make pasta from scratch, you'll need flour, eggs, salt..." (detailed, helpful)

Rejected: "Just buy pasta from the store." (dismissive)

Format properties:

- Binary choice (no ties, no Likert scales)
- Prompt + chosen + rejected triplets
- Separate "helpful" and "harmless" subsets
- Widely adopted by DPO, RLHF research

Anthropic HH-RLHF: `https://github.com/anthropics/hh-rlhf`

Preference Data Quality: What Can Go Wrong

Failure Mode	Example	Root Cause	Downstream Effect
Ambiguous prompt	"Tell me about Python" (language or snake?)	Raters interpret differently	Noisy, contradictory labels
Both bad	Two hallucinated responses; rater forced to pick one	Forced choice on low-quality pair	RM trained on noise
Sycophancy reward	Rater picks the response that agrees with the prompt's premise	Social desirability bias	Model learns to flatter

Preference Data Quality: What Can Go Wrong (cont.)

Failure Mode	Example	Root Cause	Downstream Effect
Domain mismatch	Non-expert rates medical advice	Expertise gap	Dangerous misinformation rewarded
Annotation fatigue	Rater defaults to "first option" after 200 pairs	Cognitive load	Position bias in dataset
Cultural bias	"Polite" defined by one culture's norms	Homogeneous rater pool	Model encodes narrow values

Takeaway

Every row is a real, documented failure mode. Preference datasets require the same rigor as any annotation project—plus awareness of biases unique to comparative judgment.

Artifact Catalog: Preference Annotation Biases

"In preference learning, the dataset is a model of human judgment. If that judgment is biased, the model will become biased in the same direction—at scale."

Bias	Mechanism	Model Effect
Length bias	Raters prefer longer, more detailed outputs	Verbosity, padding, unnecessary elaboration
Position bias	First-listed option preferred (primacy effect)	Ordering artifacts in training data
Authority bias	Confident/formal tone preferred	Overconfident, authoritative-sounding errors
Reward hacking	Model optimizes proxy reward, not intent	"Safe but unhelpful" or gaming the rubric

See also: Dubois et al. (2023). AlpacaFarm; judge-bias literature. `https://arxiv.org/abs/2305.14387`

Live Demo 1: Length-Only Predictor of "Preferred"

```python
import random, numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

random.seed(0); np.random.seed(0)

# Create toy preference data where "chosen" is often longer (bias)
n = 1000
X = []  # features: [len(chosen), len(rejected), len_diff]
y = []  # 1 means chosen is first
for _ in range(n):
    a = random.randint(5, 120)
    b = random.randint(5, 120)
    chosen_first = (a > b) if random.random() < 0.8 else (a < b)  # biased
    X.append([a, b, a-b])
    y.append(1 if chosen_first else 0)

X = np.array(X); y = np.array(y)
clf = LogisticRegression().fit(X[:800], y[:800])
pred = clf.predict(X[800:])
print("Accuracy predicting preference from length features:",
      accuracy_score(y[800:], pred))
```

Takeaway: If you can predict "preferred" from length alone, your preference data has a length bias that the reward model will learn.

Live Demo 2: Position Bias — First-Listed Preference

```python
import random, numpy as np
from collections import Counter

random.seed(42)
n_pairs = 500
first_chosen = 0
# Simulate: annotator picks "first-listed" 62% of the time
# even when quality is controlled to be equal
for _ in range(n_pairs):
    quality_a = random.gauss(0, 1)  # response A quality
    quality_b = random.gauss(0, 1)  # response B quality (same dist)
    position_boost = 0.3            # primacy effect (bias)
    score_a = quality_a + position_boost  # A is always listed first
    score_b = quality_b
    if score_a > score_b:
        first_chosen += 1

pct = first_chosen / n_pairs * 100
print(f"First-listed chosen: {first_chosen}/{n_pairs} ({pct:.1f}%)")
print(f"Expected if unbiased: 50.0%")
print(f"Position bias magnitude: +{pct - 50:.1f} percentage points")
# Mitigation: randomize order AND present each pair twice (A,B) and (B,A)
```

Takeaway: A 0.3 standard-deviation position boost yields ∼62% first-listed preference.
Mitigation: randomize presentation order and double-annotate with swapped order.

AlpacaFarm: Simulating Human Feedback

Dubois et al. (2023): A simulation framework for studying RLHF annotation at low cost.

What it does:

- Uses LLM-as-judge (GPT-4) to simulate human preferences
- Provides a standardized evaluation on 805 instructions (from Self-Instruct)
- Benchmarks RLHF methods (PPO, DPO, best-of-n, etc.)
- Cost: $\sim$\$600 vs. $\sim$\$13,000 for human annotation

Key findings:

- Simulated feedback achieves **65% agreement** with human annotators (humans agree with each other $\sim$66%)
- LLM judges exhibit their own biases: verbosity preference, self-preference
- Useful for rapid prototyping; not a replacement for human annotation in deployment

Dubois et al. (2023). AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. NeurIPS.
https://arxiv.org/abs/2305.14387

AlpacaFarm: Simulating Human Feedback (cont.)

Relevance to This Course

AlpacaFarm lets researchers study annotation design choices (rubrics, rater pools, candidate policies) without expensive human studies—an "annotation simulator."

Dubois et al. (2023). AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. NeurIPS. `https://arxiv.org/abs/2305.14387`

QC: Rater Calibration and Drift Monitoring

Quality control strategies:

1. **Rater calibration sets:** Periodic tests with known-answer pairs
2. **Gold prompts:** Items with expert consensus, embedded in workflow
3. **Drift monitoring:** Track rater agreement over time; detect fatigue/confusion
4. **Inter-rater agreement:** Measure consistency (analogous to classical IAA)
5. **Stratified audits:** Check for systematic biases by rater subgroup

Best Practice

Treat preference calibration like traditional IAA monitoring—but also check for *shared systematic biases* that high agreement might mask.

Modeling Mitigations for Preference Biases

1. **KL regularization:** Penalize divergence from reference policy (prevents reward over-optimization)
2. **Mixing objectives:** Combine preference loss with SFT loss to maintain instruction-following
3. **Preference debiasing:** Length-normalize rewards, control for surface features
4. **Eval triangulation:** Use multiple evaluation methods (human, LLM-judge, automatic metrics) to detect biases
5. **Adversarial evaluation:** Test with intentionally short-but-correct vs. long-but-wrong pairs

Deployment Warning

Preference-trained systems can become "safe but unhelpful"—over-optimizing for harmlessness at the cost of utility. Monitor both dimensions.

In-Class Activity: Preference Annotation (15 min)

Goal: Experience preference annotation firsthand and measure inter-annotator agreement.

Setup:

1. Form groups of 3–4 students
2. Each group receives the **same prompt and two model outputs** (distributed on handout / screen)

Task (3 phases):

1. **Phase 1 — Write a rubric (5 min):** As a group, write 3–5 criteria for "which response is better" (e.g., accuracy, helpfulness, conciseness, tone). Assign weights if desired.
2. **Phase 2 — Rate individually (3 min):** Each person *independently* scores both responses on your group's rubric, then picks a winner.
3. **Phase 3 — Compare and discuss (7 min):** Share ratings within the group. Calculate agreement. Discuss: Where did you disagree? Was it the rubric or the interpretation?

Debrief Question

How would your disagreements affect a reward model trained on your labels?

Activity Materials: The Prompt and Two Outputs

Prompt

"Explain the concept of opportunity cost to a high school student in 2–3 sentences."

Response A:

Opportunity cost is an economic concept that refers to the potential benefit an individual misses out on when choosing one alternative over another. For instance, if you choose to spend your Saturday studying for a test instead of going to a concert, the opportunity cost is the enjoyment and experience you would have gained from attending the concert. It is a fundamental principle in economics that helps explain decision-making and resource allocation in both personal and business contexts.

Response B:

Opportunity cost is what you give up when you make a choice. If you spend \$15 on lunch instead of saving it, the opportunity cost is whatever else you could have done with that \$15. Every choice has a trade-off!

Which is "better"? Apply your group's rubric and rate independently before discussing.

Activity Debrief: What Did We Learn?

Discussion questions for class:

1. **Agreement:** How many groups had unanimous agreement? How many had split votes?
2. **Rubric variation:** Did different groups emphasize different criteria? (Some may value conciseness; others value completeness.)
3. **Length effect:** Response A is longer. Did any raters feel drawn to it because of length, even if B was more appropriate for the audience?
4. **Real-world parallel:** In InstructGPT, $\sim$40 annotators made these judgments for the entire model. How would scaling to 1,000 raters change the data?
5. **Rubric as annotation guideline:** Would a shared class-wide rubric have increased agreement? At what cost to capturing diverse preferences?

Connection

This is exactly the challenge faced by every preference annotation project: rubric design, rater calibration, and the tension between agreement and diversity.

Discussion Questions

1. If you can predict preference labels from length alone, should you (a) change the rubric, (b) normalize length, (c) reweight data, or (d) accept it as "real preference"? Why?
2. When is a principle-based method (Constitutional AI) preferable to direct preference labeling? What does it assume about the task?
3. What's the minimum documentation needed to interpret a preference dataset scientifically (who raters are, instructions, candidate generation policy, filtering)?
4. AlpacaFarm shows LLM judges agree with humans $\sim$65% of the time, and humans agree with each other $\sim$66%. Does this mean LLM judges are "good enough"? What are the risks?

Key Takeaways

1. Preference data is easy to collect but easy to bias—"cheap heuristics" (like length) will correlate unless you design against it
2. The rater pool *is* the signal: who annotates determines what the model optimizes
3. DPO's simplicity encourages more data but makes biases harder to audit (no separate RM to inspect)
4. Constitutional AI shifts annotation upstream to principle design—powerful but relies on principle sufficiency
5. QC for preferences = classical IAA monitoring + systematic bias detection
6. Position bias, sycophancy, and domain mismatch are real, documented failure modes—design your annotation protocol to mitigate them

References

Stiennon et al. (2020). Summarize from Human Feedback. | Ouyang et al. (2022). InstructGPT.
Rafailov et al. (2023). DPO. | Bai et al. (2022). Constitutional AI.

Anthropic HH-RLHF. | Dubois et al. (2023). AlpacaFarm.

Readings

Required readings for next class:

1. Ouyang et al. (2022). *Training language models to follow instructions with human feedback.* NeurIPS. **Sections 1–3** (annotation pipeline).
2. Rafailov et al. (2023). *Direct Preference Optimization.* NeurIPS. **Sections 1–4** (method and data requirements).

Optional / deeper dive:

- Bai et al. (2022). Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073.
- Dubois et al. (2023). AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. NeurIPS.
- Stiennon et al. (2020). Learning to Summarize from Human Feedback. NeurIPS.