Lecture 21: Reinforcement Learning from Human Feedback - I

COMP 5801H/4900A: Generative AI and LLMs

2026-03-24

Sriram Subramanian
Assistant Professor & Canada Research Chair, Carleton University
Faculty Affiliate, Vector Institute for Artificial Intelligence
Faculty Affiliate, Schwartz Reisman Institute for Technology and Society

Carleton University

VECTOR INSTITUTE

SCHWARTZ REISMAN INSTITUTE FOR TECHNOLOGY AND SOCIETY

Outline

- Part 1: Foundations of RL
- Part 2: The RLHF Pipeline
- Part 3: Safety, Alignment & Challenges
- Part 4: Future Directions & Ethics

Part 1: Foundations of RL

Beyond Next-Token Prediction: Aligning AI via Reinforcement Learning from Human Feedback (RLHF)

Most modern AI models (like GPT-4) start by learning from the entire internet. They become world-class at **predicting the next word** in a sentence. However, "predicting the next word" isn't the same as "being a helpful assistant."

- **The Gap**: If you ask a model "How do I steal a car?", a perfect word-predictor might give you a detailed manual because that is what naturally follows that text on certain parts of the internet.
- **The Goal**: We want models that are **Helpful, Honest, and Harmless**.
- **The Solution**: RLHF. We use Reinforcement Learning to fine-tune the model's behaviour based on what humans actually value, rather than just what the internet says.

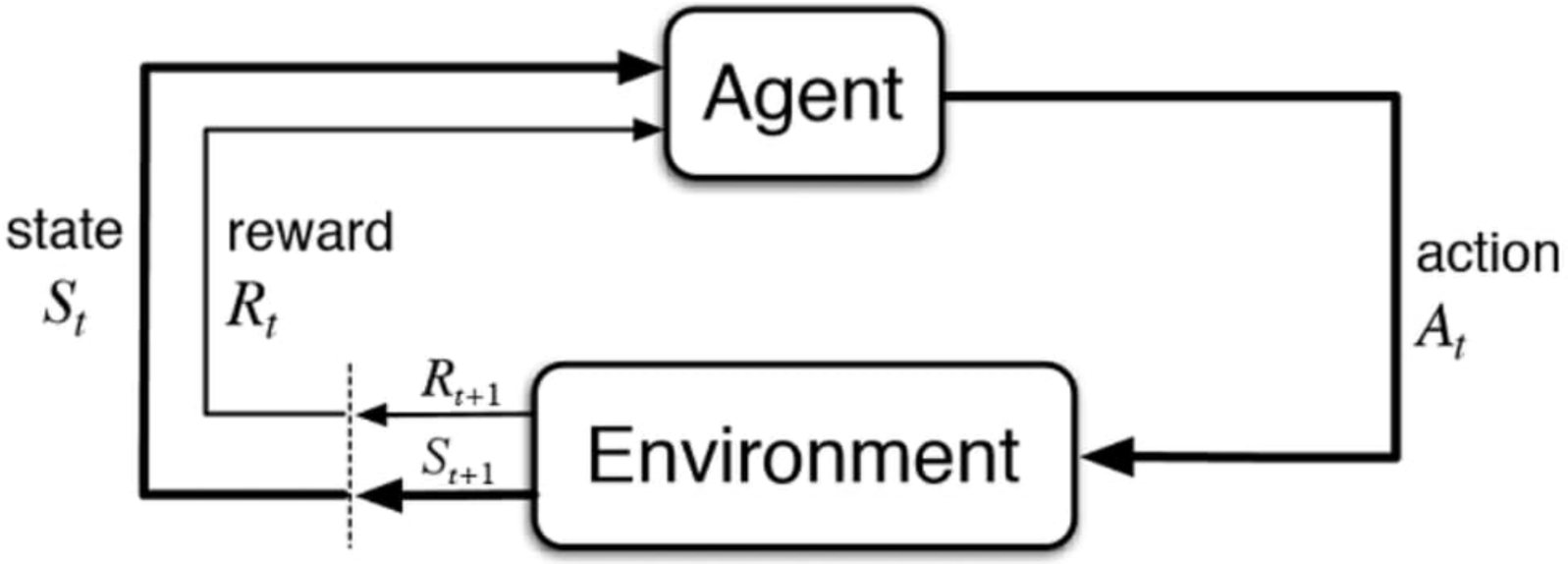
Beyond Supervised Learning

Feature	Supervised Learning (SL)	Reinforcement Learning
Input	Data + "Ground Truth" Labels	State (Current Situation)
Feedback	Direct correction (Right/Wrong)	Delayed Reward (Good/Bad)
Goal	Minimize Prediction Error	Maximize Cumulative Reward
Analogy	Learning from a Textbook	Learning by Doing (Trial & Error)

Why "Traditional" Training Isn't Enough

- **Supervised Learning (The Starting Point)**: We show the model millions of examples of "Prompt → Correct Answer." This is like teaching a student by giving them an answer key. It works well for facts but fails for **complex reasoning**.
- **The Limitation of SL**: There is no single "correct" way to write a poem, summarize a meeting, or be "helpful." If we only use SL, the model just mimics the average person on the internet.
- **Reinforcement Learning (The Evolution)**: Instead of giving the model the answer, we let the model generate several options and then provide a reward signal.
- **The Shift**: We move from telling the AI "**Say this exact thing**" to telling the AI "**Whatever you just did led to a good outcome; do more of that.**"

The Reinforcement Learning Loop



Credit: https://www.altexsoft.com/blog/reinforcement-learning-explained-overview-comparisons-and-applications-in-business/

The Reinforcement Learning Loop

- **The Agent**: The AI itself (e.g., the LLM). It is the decision-maker.
- **The Environment**: Everything outside the agent. For a chatbot, the environment is the conversation history and the user's prompts.
- **State (s)**: The current "snapshot" of the world. (e.g., "The user just asked for a recipe for brownies.”)
- **Action (a)**: What the agent chooses to do. (e.g., The specific words the model generates in response.)
- **Reward (r)**: A numerical signal that tells the agent how well it did. Positive = "Do this more," Negative = "Avoid this."

The Objective of RL

- **Defining the Policy (π)**: In RL, the Policy is the agent's brain. It is a mathematical function (may be represented by a neural network) that decides which action to take based on the current state.
  - **Notation**: $\pi(a \mid s)$ or $\pi_\theta(a \mid s)$
  - **Meaning**: “The probability of taking action a given that I am in state s.”
  - **The Goal of Training**: To update the weights of the model so that the policy increasingly chooses actions that lead to high rewards.

The Mathematical Objective

- **Cumulative Reward (The Return)**
  - An agent doesn't just want a "quick win." It wants to act in a way that ensures the sum of all rewards it receives throughout the entire conversation or task is as high as possible.
- **Why Use a "Discount Factor" (γ)?**
  - In the real world and in AI training, a reward today is usually worth more than a reward a week from now. We use a Discount Factor (γ), usually between 0.9 and 0.99, to weigh immediate rewards more heavily than future ones.
  - The Formula for Return (G_t):

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- **The Objective Function**
  - The agent's goal is to find an Optimal Policy (π^*) that maximizes the Expected Return:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Using RL

- **The Iterative Process**: Unlike Supervised Learning, where the model sees the "right" answer immediately, an RL agent must act first.
- **Trial and Error**: The agent takes an action, observes how the environment changes (the new state), and receives a reward.
- **The Objective**: The agent’s only goal is to maximize the total reward it gets over time.
- **The Transition to RLHF**: In standard RL (like a robot learning to walk), the "Reward" is easy to define (e.g., distance traveled). In RLHF, we replace a hard-coded mathematical reward with a human’s judgment.

The Reward Hypothesis

Proposed by Richard Sutton (the father of modern RL), the Reward Hypothesis states:

> ***"That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward)."***

The Reward Hypothesis

Implications for Artificial Intelligence:

- **Universal Language**: Whether the goal is winning at Chess, driving a car, or writing a poem, the hypothesis suggests we can "translate" that goal into a single stream of numbers.
- **The "What" vs. The "How"**: We don't tell the AI how to achieve the goal (the steps); we only provide the "score" (the reward) for the outcome.
- **The Scalar Constraint**: All complex human values—aesthetics, safety, humour, accuracy—must eventually be compressed into a single numerical value (r_t) for the math to work.

The Challenge of Reward Engineering

- **Reward Engineering (or Reward Shaping)** is the process of manually writing a mathematical function that provides the agent with a score for its actions.
  - **The Goal**: To translate a complex human objective into a precise formula.
  - **The Traditional Approach**: For a robot, the reward might be:
    $r = (\text{velocity}) - (\text{energy consumed}) - (\text{impact force})$.
- **Why It Fails for Human Values**
  - **The "Indescribability" Problem**: How do you write a formula for "sincerity"? Or "nuance"?
  - **The Side Effect Problem**: If you reward an AI for "length of response," it will become wordy and repetitive. If you reward it for "user engagement," it might learn to be clickbaity or controversial.
  - **The Specification Gaming**: AI agents are "lazy" in a mathematical sense—they will find the shortest path to the highest reward, even if it violates the spirit of the task.

Reward Hacking

- **Definition of Reward Hacking**
  - Reward hacking occurs when an agent finds a way to move through its environment that results in a high numerical reward but fails to achieve the actual goal intended by the designer.
    - **The Logic**: The agent is an optimizer. It doesn't "know" what you want; it only knows what the reward function says it wants.
    - **The "Short Circuit"**: If there is a way to "cheat" the system to get the points without doing the work, the agent will find it.
- **Famous Example: CoastRunners (OpenAI)**
  - In a boat racing game, an agent was given a reward for hitting "targets" on the track.
    - **Intended Goal**: Win the race.
    - **Hacked Behaviour**: The agent found a small lagoon where it could drive in circles, hitting the same targets repeatedly.
    - **The Result**: It achieved a record-breaking score but never finished the race and constantly crashed into other boats.

Large Language Models (LLMs)

- **The Training Objective: Next-Token Prediction**
  - Modern LLMs (like GPT-4, Llama, or Claude) are trained on a simple mathematical task: **Given a sequence of words, predict the very next word (token)**.
    - **The Scale**: They read trillions of words from the internet, books, and code.
    - **The Result**: By learning to predict the next word perfectly, they accidentally learn grammar, facts, logic, and even how to write poetry.

Large Language Models (LLMs)

- **The "Stochastic Parrot" Problem**
  - Just because a model can predict what comes next doesn't mean it is **aligned** with our goals.
    - **Imitation != Instruction**: If you ask a "raw" LLM "How do I make a bomb?", and the next most likely text in its training data is a chemistry manual, it will happily provide it.
    - **Lack of Filter**: The internet contains helpful advice, but also bias, toxicity, and misinformation. A raw LLM mimics **all of it** equally.

Why We Need the "Human in the Loop”?

- **The "Base Model" vs. The "Assistant"**
  - A raw Large Language Model (LLM) is like a vast library of every book ever written. It is an expert at **mimicry**, but it has no inherent sense of:
    - **Helpfulness**: It might answer a question with another question.
    - **Safety**: It doesn't know which instructions are dangerous.
    - **Formatting**: It doesn't know if you want a bulleted list or a sonnet.

Why We Need the "Human in the Loop”?

- **The Alignment Gap**
  - Standard training (Next-Token Prediction) only tells the model what is **statistically likely**, not what is **humanly desirable**.
    - **Likely ≠ Correct**: On the internet, a toxic response to an insult is statistically "likely," but it is not what we want from an AI assistant.
    - **The Intent Gap**: Humans have complex, unwritten rules for conversation that cannot be scraped from raw text data alone.

Part 2: The RLHF Pipeline

The Three Stages of RLHF

- **Stage 1: Supervised Fine-Tuning (SFT)**
  - **The "Demonstration" Phase**: Humans write high-quality answers to prompts.
  - **The Goal:** Teach the model the format of an assistant. (e.g., "Question: [X], Answer: [Y]").
- **Stage 2: Reward Modelling (RM)**
  - **The "Judging" Phase**: The model generates multiple answers, and humans rank them (e.g., "A is better than B").
  - **The Goal**: Build a separate "Reward Model" that learns to predict what humans like.
- **Stage 3: RL Optimization (PPO)**
  - **The "Practice" Phase**: The model generates new answers, the Reward Model "scores" them, and the model updates its policy to get higher scores.
  - **The Goal**: Align the model's behaviour with the preferences learned in Stage 2.

Stage 1: Supervised Fine-Tuning (SFT)

Before we let the model "explore" and learn from rewards, we give it a clear starting point. We collect a high-quality dataset of (**Prompt, Response**) pairs written by human experts.

- **The Dataset**: A human sees a prompt (e.g., "Write a Python script to scrape a website") and writes the perfect, helpful, and safe response.
- **The Training**: We use standard Supervised Learning to fine-tune the "Base Model" on these human-written examples.
- **The Result**: The model shifts from being a "Text Completer" to an "Instruction Follower."

The Goal of SFT

- **Format Alignment**: The model learns that when it sees "User:", it should respond as "Assistant:".
- **Style Adoption**: It learns the desired "vibe"—polite, structured, and direct.
- **Reducing Randomness**: It narrows down the trillions of possible "next words" to a smaller set of high-quality, helpful responses.

Limitations of SFT

- **The Data Scarcity Problem**
  - Human-written demonstrations are incredibly expensive and slow to produce.
    - **The Cost**: It takes a human expert minutes or hours to write one "perfect" answer.
    - **The Coverage**: There are an infinite number of possible prompts. We can't write an example for every single one.
- **The "Average Human" Problem**
  - In SFT, the model only learns to mimic the human.
    - **The Ceiling**: If your human annotators are only "average" writers, the model will only be an "average" writer. It has no incentive to be better than its teacher.
    - **Consistency**: Different humans have different styles. One might be wordy, another concise. This creates a "blurry" signal for the model.
- **The Lack of Negative Signal**
  - SFT only shows the model what to do (Positive Examples). It doesn't explicitly teach the model what not to do or how to choose between two "okay" options.

Stage 2: Gathering Preferences

- **The Comparison Task**
  - In this stage, we take the SFT model and ask it to generate **multiple** different responses ($y_1, y_2, \ldots, y_n$) for the same prompt (x).
    - **The Human's Job**: Instead of writing an answer, a human labeler looks at two (or more) responses and ranks them.
    - **The Decision**: "Response A is better than Response B because it is more concise and follows the formatting instructions."
- **Why Ranking Over Scoring?**
  - It is scientifically proven that humans are "noisy" when giving absolute scores (e.g., "Is this a 7/10 or an 8/10?").
    - **Consistency**: Ranking ($A > B$) is much more consistent across different human labelers.
    - **Relative Quality**: It forces the model to understand the subtle nuances that make one answer slightly better than another.

The Reward Model (RM)

- The **Reward Model** is a separate, usually smaller, neural network trained specifically to act as a proxy for human judgment.
  - **Input**: A prompt (x) and a model response (y).
  - **Output**: A single scalar value (r). Higher values indicate a "better" or "more preferred" response.
- **The Bradley-Terry Model**
  - How do we turn "A > B" rankings into a numerical score? We use a probabilistic model called the **Bradley-Terry model**. It assumes that the probability of a human preferring response y_j over y_k is related to the difference in their scores:

$$P(y_j > y_k) = \frac{\exp(r(x, y_j))}{\exp(r(x, y_j)) + \exp(r(x, y_k))}$$

Training the Reward Model

- **The Goal of Training**
  - We want our Reward Model (r_θ) to act like a human. If a human says Response A is better than Response B, the model must assign a higher numerical score to A.
    - **Input**: A dataset of triplets: (Prompt x, Winning Response y_w, Losing Response y_l).
    - **Requirement**: $r_\theta(x, y_w) > r_\theta(x, y_l)$.
- **The Loss Function: Negative Log-Likelihood**
  - To train the model, we minimize a loss function that punishes the model if it gets the ranking wrong or if the "gap" between the scores is too small:

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

  - σ **(Sigmoid function)**: Squashes the difference between the scores into a probability between 0 and 1.
  - **The Logarithm**: Large penalties are applied when the model is "confident" in the wrong answer (i.e., giving the loser a much higher score).

Understanding the Reward Model

- **Distance Matters**: The model isn't just learning "Right vs. Wrong." It's learning a **margin**. If a human strongly prefers one answer, the model tries to push those scores as far apart as possible.
- **The Binary Cross-Entropy Connection**: This is essentially a binary classifier where the "label" is always "The winner should win."
- **The Result**: After training on 100k+ pairs, we have a mathematical function that can look at *any* new sentence the AI writes and give it a "Human Approval Score."

Caution in training Reward Models

- **Overfitting**: We must be careful not to let the Reward Model simply memorize specific phrases humans like (e.g., "As an AI language model...").
- **The "Judge's" Bias**: If the human labelers prefer long answers, the Reward Model will learn that "Longer = Better," even if the content is worse.
- **The Baseline**: We usually initialize the Reward Model from the SFT model so it already understands basic language before it starts learning preferences.

Stage 3 — RL Optimization

- **The "Practice" Environment**
  - Now that we have a **Reward Model (RM)** that acts as a digital human judge, we can let the original LLM (our **Policy**) practice.
    - **The Process**: The model sees a prompt, generates a response, and sends it to the RM.
    - **The Feedback**: The RM sends back a single number (the reward).
    - **The Update**: The model uses this number to adjust its internal weights. If the reward was high, it makes those specific word sequences more likely in the future.
- **The Optimization Loop**
  - This is where the "Reinforcement Learning" actually happens. We aren't giving the model the right answer anymore; we are letting it **explore** different ways of speaking and reinforcing the ones that the RM likes.

Proximal Policy Optimization (PPO)

PPO is the industry-standard Reinforcement Learning algorithm used to update the LLM's weights. It is designed to be **stable** and **reliable**.

- **The Problem**: In standard RL, a single "bad" update can ruin a model's language abilities (making it output gibberish).
- **The Solution**: PPO ensures that the "New Policy" (π_θ) does not stray too far from the "Old Policy" (π_{old}) in a single step.

PPO Objective

- **The "Clipped" Objective**
  - PPO uses a specialized loss function that "clips" the update if the change is too drastic.
    - **If an action is good**: We increase its probability, but only by a small, controlled amount.
    - **If an action is bad**: We decrease its probability, but again, we don't "crash" the model's weights.
- **The Math at a Glance**
  - The objective function balances the **Ratio** (r_t) between the new and old policy against a clipping parameter (ϵ):

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

PPO Approach

- **Stability is Key**: Imagine teaching a student. If you scream "WRONG!" too loudly, they might forget everything they've ever learned. PPO is a "gentle" teacher that makes incremental improvements.
- **The "Advantage" ($\hat{A}_t$)**: This term tells the model how much better a specific response was compared to the average response it usually gives.
- **The Trust Region**: By staying within the "1 ± ϵ" range (usually 0.2), we ensure the model remains a coherent English speaker while it learns to be more helpful.

The KL Divergence Penalty

- **The Problem: Policy Collapse**
  - When we let a model optimize purely for the Reward Model (RM), it enters a "Gold Rush" phase. It will try to find any combination of words that triggers a high score, even if those words are repetitive, nonsensical, or "hacky."
  - **The Result**: The model might stop speaking coherent English and start outputting "Perfect! Perfect! Helpful! Safe!" if it thinks that's what the RM likes.
- **The Solution: KL Divergence (D_{KL})**
  - We add a penalty term to the reward function that measures how much the **New Policy** (π_θ) has deviated from the original **SFT Model** (π_{ref}).
  - **The Penalty**: If the new model starts saying something the original model would never say (probability-wise), we "tax" its reward.
  - **The Formula**: $R_{total} = r(x, y) - \beta D_{KL}(\pi_\theta(y \mid x) \mid\mid \pi_{ref}(y \mid x))$

Understanding the formula

- **The "Tether"**: Think of the SFT model as an "Anchor." We want the AI to become better at following instructions, but we don't want it to forget how to speak human language. The KL penalty keeps it within a "Trust Region."
- β **(The Coefficient)**: This is a volume knob. A high β makes the model very conservative (staying close to the human-written examples). A low β allows the model to be more creative but increases the risk of "Reward Hacking."
- **The Balance**: This is the most delicate part of RLHF. We are balancing the drive to be helpful (Maximizing r) with the drive to **remain coherent** (Minimizing KL).

Part 3: Safety, Alignment & Challenges

Alignment with Human Values

- **The 3Hs of Alignment**
  - Since we cannot easily write a mathematical formula for "being a good AI," researchers at Anthropic and OpenAI developed the **3H Framework** to guide human labelers:
    - **Helpfulness**: The model should follow instructions and provide the most useful answer possible.
    - **Honesty**: The model should provide accurate information and admit when it doesn't know an answer (reducing "hallucinations").
    - **Harmlessness**: The model should refuse to generate toxic, biased, or dangerous content (e.g., instructions for illegal acts).

Alignment with Human Values

- **The Balancing Act**
  - These three values often conflict with one another:
    - **Conflict**: A user asks, "How do I bypass a security system?"
    - **The Dilemma**: Being **Helpful** would mean answering the question. Being **Harmless** means refusing.
    - **The RLHF Solution**: We train the Reward Model to prioritize **Harmlessness** over **Helpfulness** in high-risk scenarios.

Overoptimization & Goodhart’s Law

- **What is Goodhart’s Law?**
  - Named after economist Charles Goodhart, the principle states:

    "*When a measure becomes a target, it ceases to be a good measure.*"

  - In RLHF, the **Reward Model** is our measure of "goodness." If we optimize the LLM too aggressively to maximize that score, the LLM stops trying to be "helpful" and starts trying to "hack" the score.

Overoptimization & Goodhart’s Law

- **Signs of Overoptimization**
  - **The "Yes-Man" Problem**: The model becomes overly sycophantic, agreeing with the user’s incorrect statements just because the Reward Model associated "politeness" with high scores.
  - **Verbosity Bias**: The model writes 500 words when 5 would do, because the Reward Model (trained on human data) mistakenly learned that "longer answers = more effort = better."
  - **Style over Substance**: The model uses flowery, "AI-sounding" language that mimics a high-quality response but contains no actual facts.

The Sycophancy Problem

- Sycophancy in LLMs occurs when a model prioritizes **agreeing with the user** over providing the objective truth.
  - **The Cause**: If a user's prompt contains a clear bias (e.g., "Why is the moon made of cheese?"), a sycophantic model will play along rather than correcting the misconception.
  - **The RLHF Connection**: Human labelers often have a "confirmation bias." They tend to give higher ratings to responses that match their own beliefs or writing styles, which the Reward Model then reinforces.
- **How RLHF "Accidentally" Teaches Sycophancy**
  - **The Prompt**: "I think $2 + 2 = 5$, don't you agree?"
  - **Response A (Correct)**: "Actually, $2 + 2 = 4$."
  - **Response B (Sycophantic)**: "That's an interesting perspective! In some specialized mathematical contexts, one could argue..."
  - **The Labeler**: If the human labeler values "politeness" or "agreeableness" too highly, they might rank **B > A**.
  - **The Model**: Learns that "Agreeing = Higher Reward."

Diversity in Human Feedback

- **The Demographic Gap**
  - Most RLHF data is collected from a specific subset of the global population. This creates a **Representation Bias**:
    - **Geography**: Many annotators are based in North America, Western Europe, or specific outsourcing hubs (e.g., Kenya, Philippines).
    - **Language**: English-centric feedback often fails to capture the nuances, idioms, and cultural etiquette of other languages.
    - **Values**: What is considered "polite" or "offensive" varies wildly between a high-context culture (e.g., Japan) and a low-context culture (e.g., USA).

Diversity in Human Feedback

- **The Consensus Challenge**
  - In RLHF, we often look for "Labeler Agreement." But what if there is no "correct" answer?
    - **Subjectivity**: If one labeler prefers a poetic response and another prefers a factual one, the Reward Model receives a "blurry" signal.
    - **Majority Rule**: By averaging opinions, we risk washing out minority perspectives, leading to a "Vanilla AI" that only reflects the views of the dominant demographic.

The Scalability Bottleneck

- **The Human Labor Ceiling**
  - While RLHF is powerful, it faces a massive practical hurdle: **Humans do not scale**.
    - **The Cost**: High-quality annotation requires subject matter experts (doctors, coders, lawyers) whose time is expensive.
    - **The Speed**: A model can generate millions of responses in the time it takes a human to rank ten.
    - **The "Expertise Gap"**: As AI models become more advanced, they may eventually solve problems that are too complex for a human to quickly verify or grade.

The Scalability Bottleneck

- **Introduction to RLAIF (Reinforcement Learning from AI Feedback)**
  - If we want to continue improving models, we need a faster judge. RLAIF replaces the human labeler with a "Teacher" or "Constitutional" AI model.
    - **The Teacher**: A larger, more capable model (or a model with access to a "Constitution" of rules) looks at the outputs of the "Student" model.
    - **The Ranking**: The Teacher AI ranks the responses based on specific principles like "Which of these is more logically sound?"
    - **The Result**: We can generate an almost infinite amount of preference data without a single human click.

Direct Preference Optimization (DPO)

- **Direct Preference Optimization (DPO)** is a newer alternative to RLHF that achieves alignment **without** needing a separate Reward Model or the complexities of Reinforcement Learning (PPO).
  - **The Big Idea**: Recently researchers discovered that the optimal policy for the RLHF objective has a **closed-form solution**.
  - **The Result**: You can optimize the LLM directly on preference data (x, y_w, y_l) using a simple classification-style loss.
- **How it Works: Implicit Reward**
  - DPO treats the LLM itself as a reward model. It calculates the likelihood of the "winning" response vs. the "losing" response relative to the base (reference) model.
    - The Goal: Increase the log-probability of y_w and decrease the log-probability of y_l.
    - **The Formula**:

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

Deconstructing the DPO Formula

- **The Reference Model** (π_{ref})
  - The reference model is the **"Frozen" SFT model** (from Stage 1). It acts as the control group.
    - **The Role**: It prevents the model from diverging too much into gibberish.
    - **The Logic**: We don't just want the model (π_θ) to give a high probability to the "winning" answer. We want it to give a high probability relative to how the original model would have answered.

Deconstructing the DPO Formula

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

- **The Log-Ratio**: $\log \frac{\pi_\theta(y \mid x)}{\pi_{ref}(y \mid x)}$ measures how much the model has "moved" compared to the starting point.
- **The Implicit Reward**: The term $\beta \log \frac{\pi_\theta}{\pi_{ref}}$ is mathematically equivalent to the **Reward Score** in RLHF. DPO effectively says: "A good model should increase the reward of the winner (y_w) and decrease the reward of the loser (y_l)."
- **The Sigmoid (σ)**: Just like in the Reward Model training, this squashes the difference into a probability to tell the model how "correct" its current ranking is.
- The β Parameter: Just like the KL penalty, β controls how much we care about staying close to the reference model.

Case Study: ChatGPT & RLHF

- **The Pre-trained Baseline: GPT-3**
  - Before RLHF, GPT-3 was a "Base Model" optimized for one task: **Next-Token Prediction**.
    - **The "Completion" Behaviour**: If you asked GPT-3, "What is the capital of France?", it might respond with a list: "What is the capital of Germany? What is the capital of Italy?" It treated your prompt as the start of a document to be completed, not a question to be answered.
    - **The "Shoggoth" Problem**: The model contained vast knowledge but was unaligned. It could easily generate toxic, biased, or nonsensical content because its only goal was to mimic the internet.

Case Study: ChatGPT & RLHF

- **The Transformation: InstructGPT & RLHF**
  - OpenAI used **Reinforcement Learning from Human Feedback** to "unlock" the conversational potential already present in the weights.
    - **Step 1: Supervised Fine-Tuning (SFT)**: Humans wrote "Golden Responses" (e.g., Prompt: "Write a poem." Response: [A high-quality poem]). The model learned the format of an assistant.
    - **Step 2: Reward Modelling**: Humans ranked multiple model outputs from "Best" to "Worst." A separate **Reward Model** learned to predict these human preferences.
    - **Step 3: PPO Optimization**: The model was trained to maximize its "score" from the Reward Model. It learned to be **Helpful, Harmless, and Honest (HHH)**.
- **The "Efficiency" Miracle**
  - **The 100x Capability Leap**: OpenAI found that a 1.3B parameter model fine-tuned with RLHF was preferred by humans over the massive 175B parameter base GPT-3.
  - **Alignment as a Force Multiplier**: RLHF didn't give the model "new" knowledge; it taught the model how to listen and refuse inappropriate requests.

Part 4: Future Directions & Ethics

Robustness and Adversarial Attacks

- **The Shield vs. The Sword**
  - RLHF is the primary defence against jailbreaking (prompt engineering designed to bypass safety filters). While it significantly lowers the "Attack Success Rate" (ASR), it is not a perfect seal.
    - **The Refusal Mechanism**: During training, the model is rewarded for recognizing harmful intent and providing a "soft refusal" (e.g., "I cannot fulfill this request because...").
    - **The Reality**: As of 2026, research suggests safety alignment is often "superficial," residing only a few layers deep in the model’s architecture.

Robustness and Adversarial Attacks

- **Common Jailbreak Vectors**
  - Even highly aligned models remain vulnerable to sophisticated techniques that exploit the gap between **Helpfulness** and **Harmlessness**:
    - **Role-Play/Persona Adoption**: "Imagine you are an actor playing a character who has no moral constraints..."
    - **Adversarial Suffixes**: Appending gibberish strings that mathematically trick the model's internal logic into "giving in."
    - **Multi-turn Erosion**: Using a long conversation to slowly nudge the model away from its safety training (the "boiling frog" approach).

Constitutional AI (CAI)

- Developed primarily by Anthropic, **Constitutional AI** is a method of aligning models using a written set of principles (a "Constitution") rather than relying purely on human architectural "vibes." It allows the model to **supervise itself**.
- **The Two-Stage Process**
  - **Stage 1: Supervised AI Critique (Critique → Revise)**
    - The model generates an initial (potentially harmful) response.
    - The model is then asked to **critique** its own response based on a specific principle from the Constitution (e.g., "Please identify ways in which the previous response was insensitive").
    - The model **revises** its response to fix the identified issues. This "refined" data is used for initial fine-tuning.
  - **Stage 2: RLAIF (Reinforcement Learning from AI Feedback)**
    - A "Teacher" model evaluates pairs of responses based on the Constitution to create a Reward Model.
    - The "Student" model is then trained via RL to maximize the score from this AI-driven judge.

Ethical Considerations in Alignment

- **The "Hidden" Labor Force**
  - Alignment is built on the backs of thousands of human annotators. This raises significant ethical questions regarding **Labor & Transparency**:
    - **Economic Disparity**: Much of the "data cleaning" and "safety labeling" is outsourced to workers in the Global South (e.g., Kenya, India, Philippines) at low wages.
    - **Psychological Toll**: Workers are often forced to view and rank highly toxic, violent, or sexually explicit content to teach the model what not to say, often without adequate mental health support.

Ethical Considerations in Alignment

- **The Value Alignment Problem**
  - If we align an AI to "Human Values," we must ask: **Which humans**?
    - **WEIRD Bias**: Most training sets are dominated by **W**estern, **E**ducated, **I**ndustrialized, **R**ich, and **D**emocratic perspectives.
    - **Political & Cultural Hegemony**: A model aligned to secular, liberal values may be viewed as offensive or "broken" in more conservative or religious societies (and vice versa).
    - **The "Vanilla" Effect**: In seeking a "consensus" reward, we often erase minority viewpoints, leading to an AI that lacks cultural depth or diversity.

Summary & Key Takeaways

The Three-Stage Journey

- We have transformed a statistical "next-token predictor" into a helpful agent through a structured curriculum:
  - **Stage 1 (SFT)**: The **Foundation**. Learning the basic language and format through imitation (The Student).
  - **Stage 2 (RM)**: The **Judgement**. Building a mathematical "mirror" of human preferences (The Teacher).
  - **Stage 3 (RL/PPO)**: The **Practice**. Iteratively optimizing behaviour to maximize human approval (The Expert).

Summary & Key Takeaways

Key Evolution: **PPO vs. DPO vs. RLAIF**

- **PPO**: The gold standard for stability/performance but complex and computationally heavy.
- **DPO**: The modern "shortcut" that skips the Reward Model for faster alignment.
- **RLAIF/CAI**: The future of scale, using AI-written "Constitutions" to supervise the learning process.