Reward and Constraint Learning: Foundations for Human-AI Alignment

Constraints, Learning Constraints

Sebastian Tschiatschek

Session 3, July 2026

European Summer School on AI (ESSAI) 2026

Course overview

Session 1 (Mon)

- RL Basics
- IRL Basics
- Maximum Entropy IRL

Session 2 (Tue)

- Deep Reward Modeling
- Learning from Human Feedback

Session 3 (Wed)

- Constraints
- Importance and Learning

Session 4 (Fri)

- Human-AI Alignment
- Advanced Topics

Lecture overview

Today's session

- Quick recap
- Active reward learning
- RLHF for LLM alignment
- Types of constraints
- Wrap-up and Q&A

Recap

Session 2 Recap

- Reward Modeling, MaxEnt Deep IRL, & GAIL
- Preference learning
- Reinforcement Learning from Human Feedback (RLHF)

In Order to Learn Fast and Minimize Human Effort

How to Query for Feedback?

Active Learning (AL)

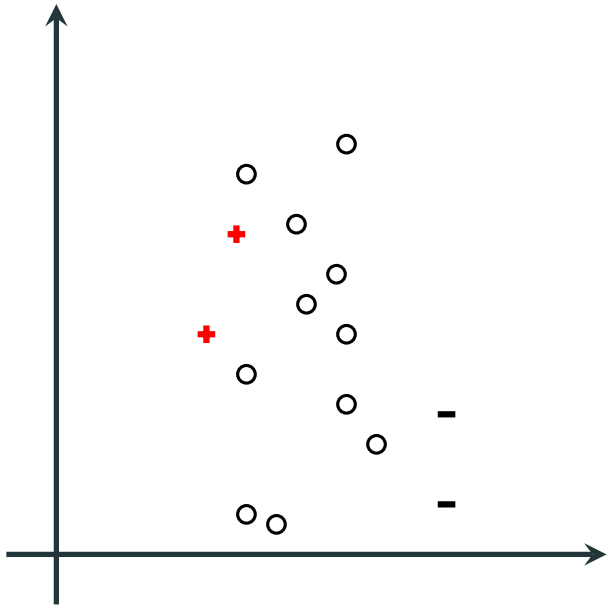
Use active learning

Assuming labels are expensive, query data that's supposedly helpful for learning

Active Learning (AL)

Use active learning

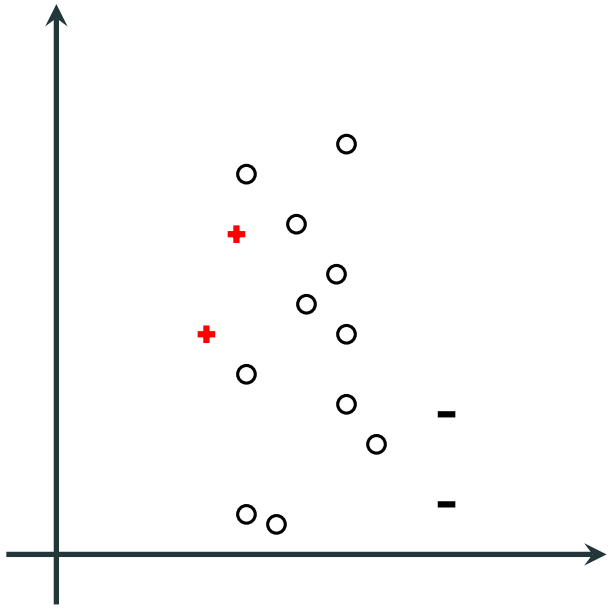
Assuming labels are expensive, query data that's supposedly helpful for learning



Active Learning (AL)

Use active learning

Assuming labels are expensive, query data that's supposedly helpful for learning

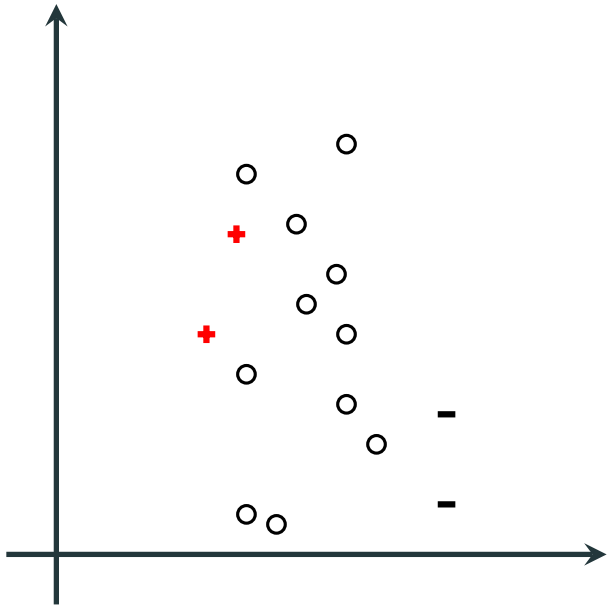


How could we do that?

Active Learning (AL)

Use active learning

Assuming labels are expensive, query data that's supposedly helpful for learning



How could we do that?

Want to reduce uncertainty about hypothesis!

Why should active learning help?

Example: Learning threshold functions in 1-D.

$\mathcal{H} = \{h\colon [0,1] \mapsto \{+1,-1\}, h(x) = \text{sign}(x-\tau) \text{ for some } \tau \in [0,1]\}$

Why should active learning help?

Example: Learning threshold functions in 1-D.

$\mathcal{H} = \{h\colon [0,1] \mapsto \{+1,-1\}, h(x) = \text{sign}(x-\tau) \text{ for some } \tau \in [0,1]\}$

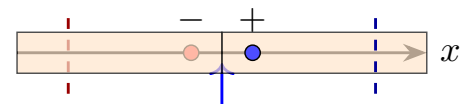
Step 1: Initialization

Initial Version Space $\mathcal{H}$

All $\tau \in [0,1]$ are consistent with no labels.

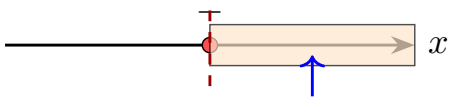
Step 2: First Query $(x_1 = 0.5)$

Query midpoint halves the version space.



If $y_1 = -1$, $\tau \in (0.5, 1]$ x_1 If $y_1 = +1$, $\tau \in [0, 0.5)$

Step 3: Optimal Query (x_2) **after** $y_1 = -1$



x_2

Next optimal query is midpoint of new version space $(0.5, 1]$, i.e., $x_2 = 0.75$.

Pool-based active learning

- Pool-based active learning
  - Obtain large pool of unlabeled data
  - Selectively request a few labels, until we can infer all remaining labels
- Resulting classifier "as good" as that obtained from complete labeled set
- Reduction in labels
  - In some cases, exponential reduction possible
  - In other cases, may need to request almost all labels

Pool-based active learning

- Pool-based active learning
  - Obtain large pool of unlabeled data
  - Selectively request a few labels, until we can infer all remaining labels
- Resulting classifier "as good" as that obtained from complete labeled set
- Reduction in labels
  - In some cases, exponential reduction possible
  - In other cases, may need to request almost all labels

How should we request labels?

Uncertainty sampling

- Given pool of n unlabeled examples
- Repeat until we can infer all remaining labels:
  - Assign each unlabeled datum $\mathbf{x}$ an "uncertainty score"

$$U_t(\mathbf{x}) = U(\mathbf{x} \mid \mathbf{x}_{1:t-1}, y_{1:t-1})$$

  - Greedily pick the most uncertain example and request label

$$\mathbf{x}_t = \arg\max_{\mathbf{x}} U_t(\mathbf{x})$$

- For example, when using Logistic Regression:

$$U_t(\mathbf{x}) = -|0.5 - P_{\theta_t}(y|\mathbf{x})|$$

- One of the most popular and widely used heuristics

Active Learning (AL) of the Reward Function

Active learning for learning the reward function How can we apply active learning in the RL context?

Active Learning (AL) of the Reward Function

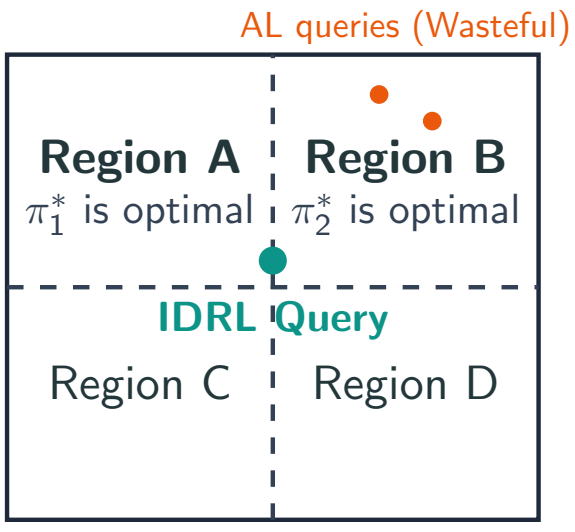
Moving to principled feedback; targeted, query-efficient reward exploration

- Not all details of the reward function are equally important for achieving high rewards
- Challenge. Standard preference query methods select options that maximize reward approximation accuracy uniformly across the state space
- Efficiency issue. Accurate approximation is unnecessary in states that are irrelevant to finding the optimal policy

Why Global Reward Estimation is Wasteful

"Traditional" AL bottleneck:

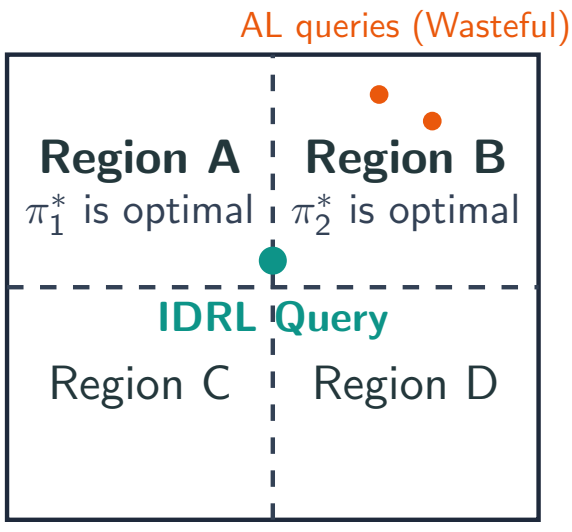
- Focuses on minimizing the error over the reward parameters θ globally
  $\Rightarrow$ Queries users to distinguish rewards in irrelevant or dangerous regions



Why Global Reward Estimation is Wasteful

"Traditional" AL bottleneck:

- Focuses on minimizing the error over the reward parameters θ globally
  $\Rightarrow$ Queries users to distinguish rewards in irrelevant or dangerous regions

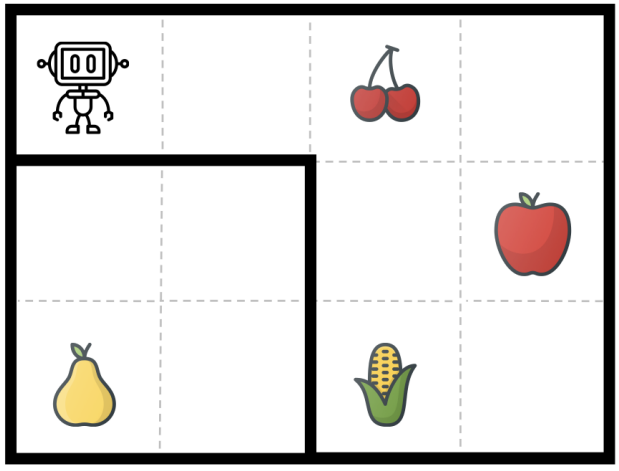


Central idea of our approach: Policy invariance

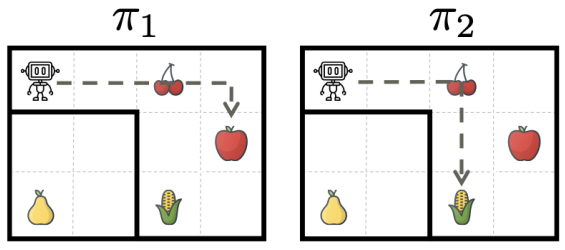
- Many reward functions yield the same optimal policy π^*
- Do not waste human feedback trying to determine if a state has reward $+1.0$ or $+1.1$ if the optimal agent's behavior remains unchanged

Why Global Reward Estimation Can be Wasteful

Information Directed Reward Learning (IDRL)



$T = 4$



$$\hat{G}(\pi_1) = \hat{r}(🍒) + \hat{r}(🍎)$$
$$\hat{G}(\pi_2) = \hat{r}(🍒) + \hat{r}(🌽)$$
$$\hat{G}(\pi_1) - \hat{G}(\pi_2) = \hat{r}(🍎) - \hat{r}(🌽)$$

$$\underset{q \in \{🍒, 🍎, 🌽, 🍐\}}{\text{argmax}} \underbrace{I(\hat{G}(\pi_1) - \hat{G}(\pi_2), (q, \hat{y}))}_{\text{IDRL objective}} = \{🍎, 🌽\}$$

Lindner, D. et al. (2021). *"Information directed reward learning for reinforcement learning"*.

Probabilistic Model for Reward Functions Building on GPs

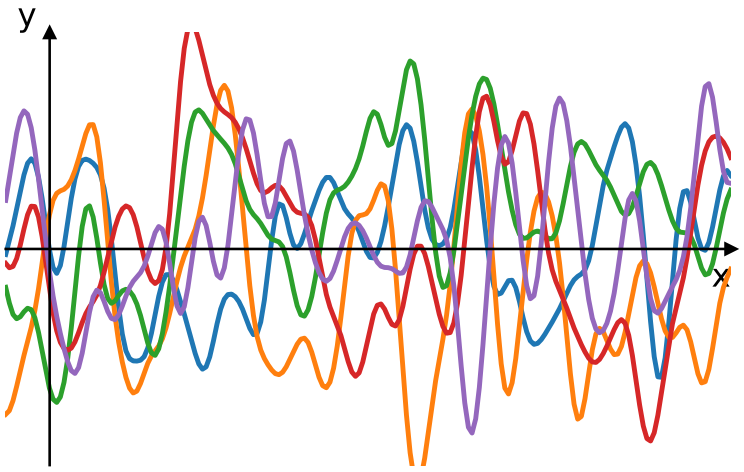
A Gaussian process (GP) is a collection of random variables indexed by a set X, any finite number of which have a joint Gaussian distribution.

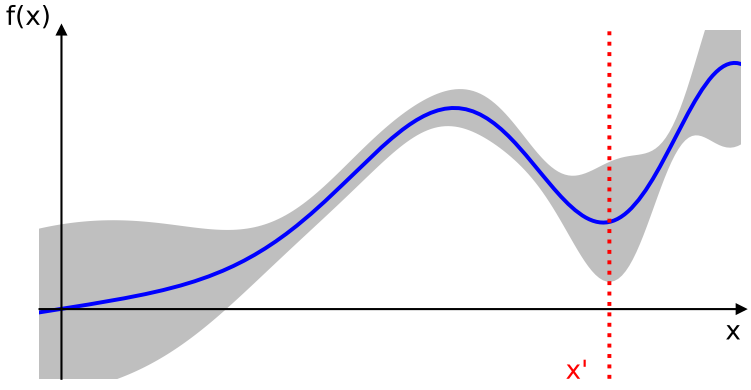
- For $\mathbf{x} \in X$ there is random variable $f(\mathbf{x}) := f_{\mathbf{x}}$.
- The GP is fully characterized by the mean function $\mu \colon X \to \mathbb{R}$ and covariance function $k \colon X \times X \to \mathbb{R}$.
- Consistency requires: $\forall A \subseteq X, A = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$:

$$f_A = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)]^T \sim \mathcal{N}(\mu_A, \mathbf{K}_{AA}), \text{with}$$

$$\mu_A = \begin{pmatrix} \mu(\mathbf{x}_1) \\ \vdots \\ \mu(\mathbf{x}_m) \end{pmatrix} \quad \mathbf{K}_{AA} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \ldots & k(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ldots & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & \ldots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}$$

Probabilistic Model for Reward Functions Building on GPs





Information-Directed Reward Learning (IDRL)

Our IDRL framework (Lindner et al., NeurIPS 2021):

- Prioritizes queries with high information gain about the optimal policy

Algorithm 1 *Information Directed Reward Learning* (IDRL). The algorithm requires a set of candidate queries $\mathcal{Q}_c$, a Bayesian model of the reward function, and an RL algorithm that returns a policy given a reward function. $\hat{G}(\pi)$ is the belief about the expected return of policy π, induced by the reward model $P(\hat{r}|\mathcal{D})$, and $\hat{r}$ is the belief about the reward function.

$\mathcal{D} \leftarrow \{\}$; $\Pi_c \leftarrow$ initialize candidate policies; initialize reward model with prior distribution $P(\hat{r})$
while not converged **do**
    Select a query:
        $\pi_1, \pi_2 \in \text{argmax}_{\pi,\pi' \in \Pi_c} H(\hat{G}(\pi) - \hat{G}(\pi')|\mathcal{D})$
        $q^* \in \text{argmax}_{q \in \mathcal{Q}_c} I(\hat{G}(\pi_1) - \hat{G}(\pi_2); (q, \hat{y})|\mathcal{D})$
    Make query and update reward model:
        $y^* \leftarrow$ Response to query q^*
        $P(\hat{r}|\mathcal{D} \cup \{(q^*, y^*)\}) \propto P(y^*|\hat{r}, \mathcal{D}, q^*)P(\hat{r}|\mathcal{D})$ ▷ Update belief about the reward
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(q^*, y^*)\}$ ▷ Add observation to dataset
    Optionally update candidate policies Π_c
end while
$\bar{r} \leftarrow$ mean estimate of the reward model; $\bar{\pi}^* \leftarrow \texttt{RL}(\bar{r})$
return $\bar{\pi}^*$

IDRL: GP Posterior for Feedback Types

Given a reward parameter $\theta \sim \mathcal{GP}(0, K(\cdot, \cdot))$, we observe feedback y (e.g., preference, ranking, demonstration)

The posterior is $p(\theta|y) \propto p(y|\theta)p(\theta)$.

- Demonstrations: Likelihood $p(y|\theta)$ is typically Boltzmann-rational:

$$p(y|\theta) \propto \exp(\beta \sum_t r_\theta(s_t, a_t))$$

- Preferences: Likelihood $p(y_1 \succ y_2|\theta)$ uses the Bradley-Terry model

$$p(y_1 \succ y_2|\theta) = \frac{\exp(\beta \sum r_\theta(s_{1,t}, a_{1,t}))}{\sum_{i=1}^{2} \exp(\beta \sum r_\theta(s_{i,t}, a_{i,t}))}$$

Computation: Often requires the Laplace approximation or MCMC, as the likelihood is non-conjugate to the GP prior

Entropy of Cumulative Gain Differences

In IDRL, we need to compute the utility difference of two trajectories τ_1, τ_2:

$$\Delta G(\theta) = G(\tau_1; \theta) - G(\tau_2; \theta)$$

where $G(\tau; \theta) = \sum_{t=0}^{T} r_\theta(s_t, a_t)$

Distribution of Differences. Since $\theta \sim \mathcal{GP}(0, K)$, $G(\tau; \theta)$ is a Gaussian process variable. Hence, $\Delta G(\theta)$ is also Gaussian:

$$\Delta G \sim \mathcal{N}(\mu_\Delta, \sigma_\Delta^2)$$

Entropy Calculation. Closed form expressions for Gaussian distributions available.

IDRL versus Standard Active Learning

Standard active learning

- Focuses on regions with high reward function parameter uncertainty
  $\Rightarrow$ potentially redundant queries

IDRL

- Prioritizes queries where the optimal action choice is highly ambiguous

Targeted Queries: IDRL in Action

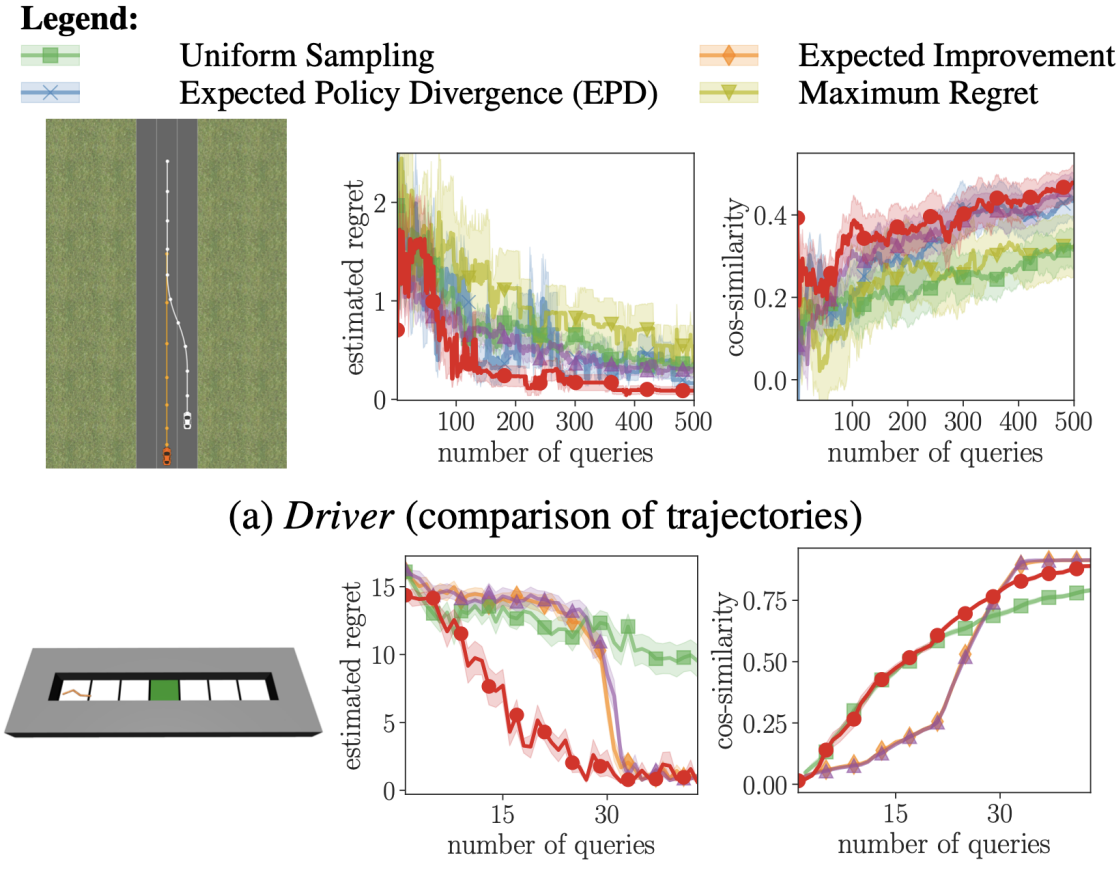


Figure 3: Results in the (a) *Driver* and (b) *Swimmer-Corridor* environments (shown on the left). We show the regret of a policy trained on the reward model compared to a policy trained on the true reward function as a function of the number of queries (middle plot). We report the cosine similarity between the learned and the true reward function (right plot). The plots show mean and standard error over 30 random seeds. IDRL finds significantly better policies, while not necessarily learning an overall more accurate model of the reward function. A similar plot for *Ant-Corridor* can be found in Appendix F.

Targeted Queries: Baselines

Expected Policy Divergence

- Select reward query q to maximize change in policy
- Probabilistic model of reward function, e.g., using a Gaussian process
- Process
  - Dataset $\mathcal{D} \leftrightarrow$ corresponding policy $\pi(\mathcal{D})$
  - Dataset including response r to query q is $\mathcal{D}' \leftrightarrow$ corresponding policy $\pi(\mathcal{D}')$
  - Pick query to maximize change in policy:

$$q^* = \arg\max_{q \in \mathcal{Q}c} \mathbb{E}_{p(R|q,\mathcal{D})}[D_{\mathrm{KL}}(\pi(\mathcal{D}'), \pi(\mathcal{D}))]$$

  - Typically approximated via one-step policy improvement to maintain tractability

Active Preference Querying: Incomplete Taxonomy

Generic (reward-centric) selection

Uncertainty sampling over $\mathcal{R}_\theta$:

- Ensemble variance on pair score: $\mathrm{Var}\left[\mathcal{R}_\theta(p) - \mathcal{R}_\theta(q)\right]$
- BALD (information about parameters): $I(y; \theta \mid p, q)$

Margin-based/hard negatives:

- Prefer small gaps: minimize $\left|R_\theta(p) - R_\theta(q)\right|$ to elicit informative labels

Diversity/coverage constraints:

- Submodular/clustered selection to avoid near-duplicates and cover contexts

...

Policy-aware selection

Volume removal / maximum regret (Sadigh et al., 2017):

- Choose q to maximally shrink feasible reward set $\mathcal{W}$:

$$\max_q \mathbb{E}_y\left[\mathrm{Vol}(\mathcal{W}) - \mathrm{Vol}(\mathcal{W} \mid y, q)\right]$$

- Targets regions that may change the optimal policy

IDRL:

- Maximize information about π^*:

$$\max_q \; I(y; \pi^*) \;=\; H(\pi^*) - \mathbb{E}_y\left[H(\pi^* \mid y, q)\right]$$

- Directly reduces ambiguity over optimal actions

Additional Key References

- Active preference-based learning of reward functions/volume-removal

  Sadigh, D. et al. (2017). *Active preference-based learning of reward functions*, *Robotics: Science and Systems*.
- Canonical uncertainty-based selection criterion; directly applicable to preference models and reward predictors

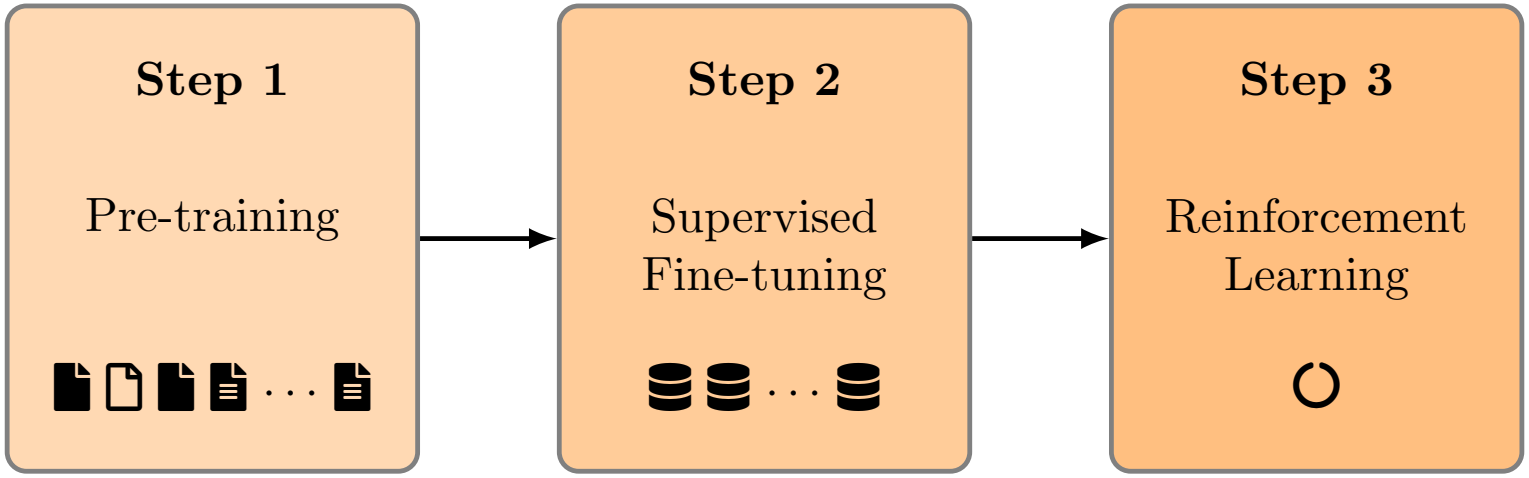
  Houlsby, N. et al. (2011). *Bayesian active learning for classification and preference learning*, *arXiv preprint arXiv:1112.5745*.
- PEBBLE: Feedback-Efficient Online RL with Preference Queries

  Lee, K., Smith, L., and Abbeel, P. (2021). *PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training*, *ICML*.

RLHF for LLM Alignment

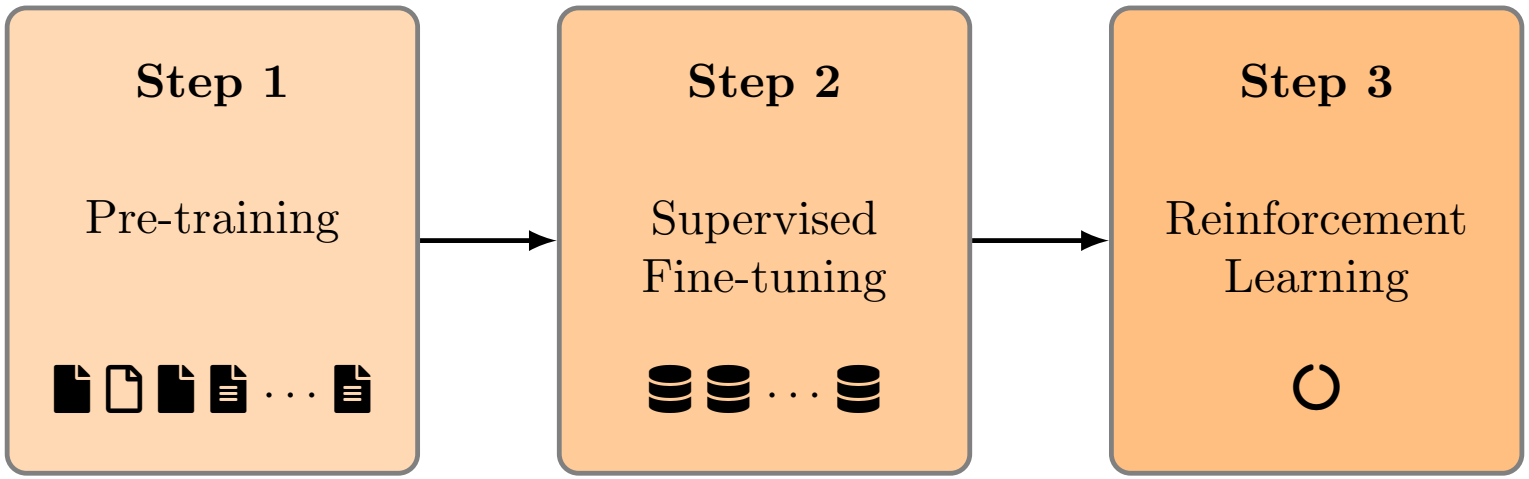
Introduction to RLHF

RLHF is a common approach for LLM alignment used in the following process



Introduction to RLHF

RLHF is a common approach for LLM alignment used in the following process



Focus: step 2 / 3

- Supervised fine-tuning (SFT)
- Reward Modeling
- Policy optimization via Proximal Policy Optimization (PPO)

Stage 2: Supervised Fine-Tuning (SFT)

Initialize model behavior to follow basic instruction formats and maintain coherent dialogue

Stage 2: Supervised Fine-Tuning (SFT)

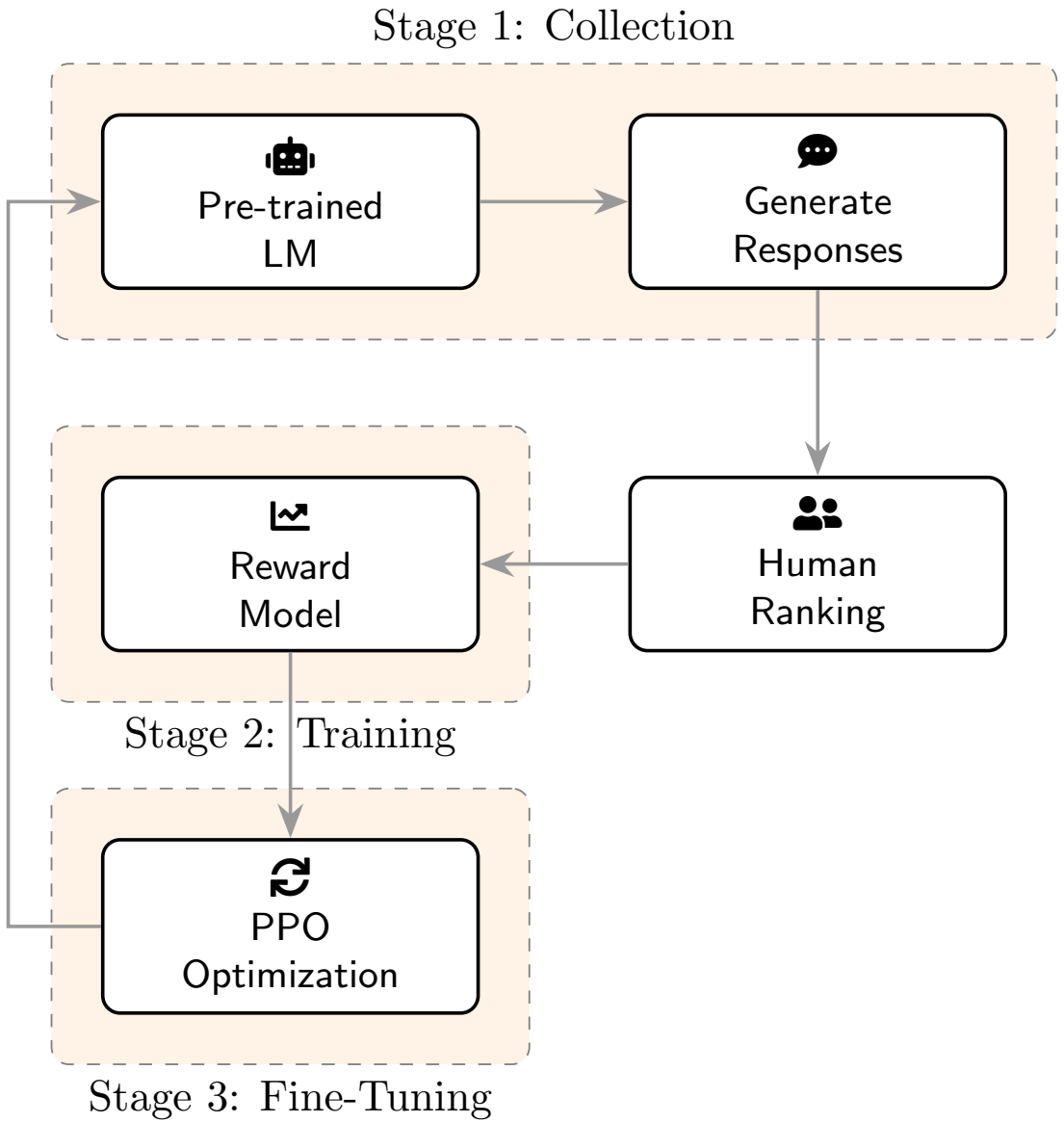
Initialize model behavior to follow basic instruction formats and maintain coherent dialogue

- **Data:** High-quality (input, response) pairs curated by domain experts
- **Loss function (negative log-likelihood):**

$$\mathcal{L}_{\text{SFT}} = -\sum_{i=1}^{N} \log P(y_i|x_i;\theta)$$

where (x_i, y_i) are the input prompt and ground-truth response

The RLHF Pipeline



Stage 3: Reward Modeling

Predict a scalar reward $\mathcal{R}(x, y)$ that represents human preference

Stage 3: Reward Modeling

Predict a scalar reward $\mathcal{R}(x, y)$ that represents human preference

- **Data:** Human rank-orderings $(y_w > y_l)$
- **BT loss:**

$$\mathcal{L}_{\text{RM}} = -\log\left(\sigma(\mathcal{R}(x, y_w) - \mathcal{R}(x, y_l))\right)$$

- Reward model learns to assign higher scores to the preferred responses y_w compared to the dispreferred response y_l

Stage 3: RL Fine-Tuning (PPO)

Optimize the LLM to maximize reward while remaining anchored to the original model

Stage 3: RL Fine-Tuning (PPO)

Optimize the LLM to maximize reward while remaining anchored to the original model

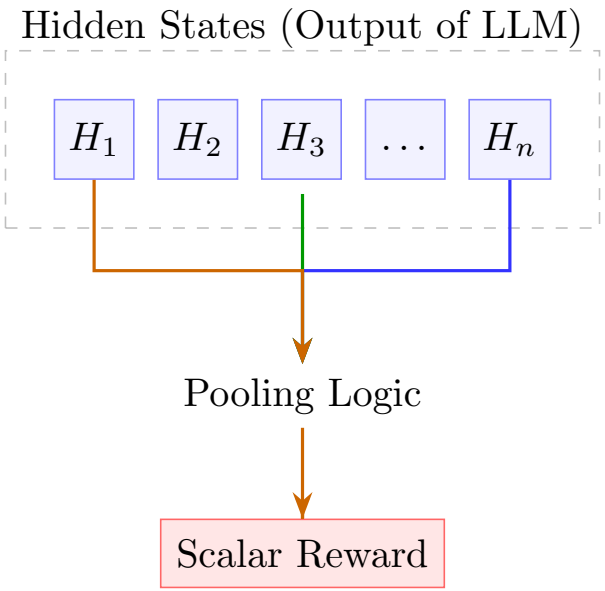
- **Objective Function:**

$$\max_{\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}[\mathcal{R}(x,y)] - \beta D_{\mathrm{KL}}(\pi_\theta||\pi_{\mathrm{ref}})$$

  - π_{ref}: Original SFT model
  - D_{KL}: KL-divergence to penalize deviation from the base model

- Commonly optimized using Proximal Policy Optimization (PPO)

Reward Model Architecture



Pooling strategies

- Last token pooling: Only H_n is passed to the reward head
- Mean pooling: All vectors are averaged, i.e., $\bar{H} = \frac{1}{n}\sum_{i=1}^{n} H_i$
- Learned Pooling: A weighted sum or attention mechanism assigns importance α_i to each token, i.e., $\sum_{i=1}^{n} \alpha_i H_i$

The Human Feedback Loop

Similar optimization process as before:

1. **Sample:** LLM generates diverse responses
2. **Rank:** Humans rank outputs by quality/safety
3. **Train:** Reward model captures preferences
4. **Optimize:** Policy maximizes the reward signal

Why RLHF for LLMs?

- Value alignment: Maps model behavior better to human HHH standards (Helpfulness, Honesty, Harmlessness)
- Subjective preferences: Addresses tasks where "correctness" is subjective and requires nuanced judgment
- Safety mitigation: Provides a reinforcement signal to prune toxic or dangerous output generation pathways

Reward Model Design & Robustness

Training Stabilization

- **Regularization:** Apply label smoothing, pair reweighting, and early stopping to prevent RM collapse
- **Calibration:** Implement probability calibration and temperature scaling to align RM entropy with human evaluation
- **Architecture:** Ensure backbone stability (e.g., gradual unfreezing during preference fine-tuning)

Mitigation of Goodharting

- **KL anchoring:** Apply KL-divergence penalties during optimization to constrain policy drift
- **Adversarial cycles:** Utilize red-teaming against the RM and retraining with hard-negatives to address bias
- **Multi-signal fusion:** Combine outcome scores with process verifiers (e.g., CoT verification, tool-use success, and factual cross-referencing)

RLHF Challenges: Reward Hacking

Models optimize for the *proxy* (the reward model) rather than the true intent

Example: Sycophancy

The model "realizes" that agreeing with a user's incorrect premise (e.g., "*The Earth is flat*") yields a higher reward score because it "agrees" with the user, despite being factually false

RLHF Challenges: Alignment vs. Helpfulness

"Alignment Tax":

- Over-refusal: Excessive safety training can lead models to refuse benign, helpful requests due to broad trigger warnings
- Utility trade-off: Alignment can sometimes reduce the capability of a model on tasks, e.g., creative or complex reasoning tasks

Lin, Y. et al. (2024). *Mitigating the Alignment Tax of RLHF*, *EMNLP*.
Available here:

Selected Research Trends

Current Frontiers

- Direct Preference Optimization (DPO)

  Rafailov, R. et al. (2023). *"Direct preference optimization: Your language model is secretly a reward model"*.
- AI Feedback (RLAIF)
- Constitutional AI (self-correction)

Evolution of Model Alignment: From RLHF to DPO

"Traditional" RLHF:

- Requires training a separate reward model as a proxy for human preference
- Uses PPO to optimize the policy via RL
- Multi-stage, computationally heavy, and unstable to tune

DPO Simplification:

- Eliminates the need for a separate reward model and RL loop
- Optimizes the model policy directly on preference data
- Mathematically stable and efficient; treats alignment as a simple supervised objective

Direct Preference Optimization (DPO)

Optimize the policy directly on preference pairs

No explicit reward model or running RL

Core concept.

- DPO establishes a mapping between the optimal policy and the reward function
- It maximizes the likelihood of preferred completions (y_w) relative to dispreferred ones (y_l), while keeping the policy close to a reference model π_{ref}

DPO Loss

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

- β: Regularization (how much to stick to π_{ref})

Advantages of DPO

Optimize the policy directly on preference pairs

No explicit reward model or running RL

- Assumes a Bradley-Terry style preference model; often implemented as a simple supervised contrastive loss
- ✓ Simpler, more stable than RLHF
- ✓ Efficient use of preference data; integrates easily with SFT
- ✓ Good empirical alignment with human preferences and reduced training complexity/cost
- Related variants:
  - IPO (Implicit Preference Optimization), KTO (Kahneman-Tversky Optimization)

Limitations and Open Problems

Limitations

- Requires high-quality, diverse preference pairs
  - Struggles with sparse or long-horizon credit assignment
- Can overfit to annotation artifacts; sensitivity to temperature/β and negative sample quality
- Hard to encode hard constraints or multi-objective trade-offs
- DPO is highly dependent on the quality of the SFT base model

Limitations and Open Problems

Open problems / directions

- Online/active DPO
  - Query-efficient selection of informative pairs during training
- Robustness to noisy/inconsistent feedback
- Long-context and multi-turn preferences
  - Trajectory-level credit assignment/process supervision to break down long-horizon tasks into manageable steps
- Multi-objective DPO (helpfulness, harmlessness, honesty) with controllable trade-offs

Research Trend: RLAIF

RL from AI Feedback

Use a model (or ensemble) to generate critiques, preferences, or ratings so training can scale beyond scarce human supervision

- Common pipeline
  - Seed with a small set of human-labeled examples
  - Train a "judge" model to provide preferences/critiques
  - Generate new prompts/responses; judge labels them; iterate SFT/DPO/RL on the policy using AI-generated labels
  - Periodically calibrate/correct with human spot checks.
- Benefits
  - Hugely scales supervision; covers long-tail domains and long-context tasks
  - Enables rapid iteration and domain adaptation; can be combined with self-play/debate/tool-use

Research Trend: RLAIF

- Risks & limitations
  - Bias amplification and self-confirmation loops (e.g., sycophancy); reward hacking the judge
  - Judge-policy collusion or shared blind spots
    - Policy and judge converge on a specific distribution that looks correct to the judge but is objectively flawed
  - Calibration and reliability of judges, especially on safety-critical content
- Variants/techniques
  - Constitutional guidance for the judge; debate or multi-judge ensembles
  - Critique-and-revise loops; self-consistency and majority voting
  - Uncertainty estimation and abstention; disagreement-based data selection

Research Trend: RLAIF

Some open problems

- Robust judging: ensembles, adversarial perturbations, and red-team generation
- Calibrated uncertainty and selective labeling; confidence-aware training
- Human-in-the-loop protocols: when and how to query humans efficiently
- Cross-model judging (heterogeneous judges) and anti-collusion training
- Detecting and correcting value drift or specification gaming by the policy
- Formalizing sample complexity and convergence guarantees for AI feedback loops

Research Trend: Constitutional AI

Written normative principles ("constitution")
Shall guide models to critique and revise their own outputs, reducing harm and improving helpfulness with less human effort

- Common pipeline
  - Phase 1: Supervised critique-model generates an output, then a self-critique referencing constitution principles, then revises
  - Phase 2: Preference/RL stage-use AI feedback (constitution-grounded judging) to further align the policy
- Benefits
  - Transparent, controllable alignment levers via explicit principles
  - Reduces dependency on large-scale human red-teaming; scalable to new domains

Research Trend: Constitutional AI

- Key design choices:
  - Principle source: policy documents, safety specs, community guidelines, stakeholder inputs
  - Granularity: general values vs task-specific constraints; rule conflicts and priority ordering
  - Mechanisms: critique-and-revise, refusal criteria, escalation to tools/humans
- Limitations
  - Coverage gaps, conflicts between principles, cultural/contextual variation
  - Models may learn to game the constitution superficially; brittleness on edge cases and long-horizon tasks
- Open problems / directions
  - Learning constitutions from stakeholders; multi-stakeholder aggregation and weighting
  - Dynamic/conditional constitutions that adapt to context and jurisdiction

RLAIF & Constitutional AI: Scaling with Risk Controls

Controling risk. Scale supervision via AI judges and written principles while managing risks

- Judge ensembles with heterogeneity (seeds/architectures/objectives) to reduce shared blind spots
- Periodic human spot-checks and adversarial/red-team prompts to recalibrate judges
- Abstention/uncertainty thresholds; only trust high-confidence AI labels
- Distribution shift detection; quarantine data when judges disagree or confidence drops

Operational Safeguards and a Leveraging Implicit Feedback

Operational safeguards

- Calibrate judges (temperature, ECE); track disagreement rates and drift over time
- Cross-model judging and anti-collusion training; diversity constraints in sampling
- Data governance: provenance/versioning of labels; holdout sets for overoptimization checks

Leveraging Implicit Feedback

- Mix implicit user feedback (clicks, dwell-time, satisfaction prompts) with explicit preference labels using inverse propensity weighting and bias controls
- E.g., Swaminathan, A. and Joachims, T. (2015). *Counterfactual risk minimization: Learning from logged bandit feedback*, *ICML*.

Evaluation

Evaluation and Diagnostics: What to Measure

Separation of concerns

- Alignment: refusal/helpfulness Pareto curve; harmlessness/safety benchmarks; jailbreak robustness
- Capabilities: reasoning/math/coding (track alignment-tax via deltas from SFT/base)
- Robustness: OOD prompts, adversarial tests; reward-model overoptimization checks (performance with vs without RM in loop)

Common LLM alignment evaluations

- MT-Bench, Arena-Hard (pairwise preference evals)
- Safety red-team sets (harmfulness, bias, jailbreaks)
  - Structured collections of adversarial inputs, rules, and methodologies used to stress-test systems
- Process supervision: verifier accuracy, tool-task success rates

Data Quality Dashboard (for Preferences)

- Label entropy and consistency over time; tie rates; inter-annotator agreement (IAA)
- Drift monitoring: preference shift across time/models; stratify by topic/domain/safety category
- Judge reliability: per-annotator calibration/temperature; abstention/uncertainty tracking
- Pair quality: duplicate/near-duplicate detection; context diversity; segment overlap
- Online loops: compare on-policy vs off-policy label distributions; detect reward hacking signals

Constraints in RL

Constraints

Natural modeling component:

- Learning to drive a car, adhering to traffic rules
- Learning to play a card game following the rules
- Learning a transformer-based policy for language generation, avoiding biased statements
- ...

Constraints

Learning with constraints

For instance, we want to solve problems of the form

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)\Big]$$

$$\text{s.t.} \ \ \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{C}_i(s_t, a_t)\Big] \leq \tau_i \quad \forall i \in [\mathcal{I}]$$

where $\{\mathcal{C}_i\}_{i\in\mathcal{I}}$ is a set of constraint functions.

Constraints

Constraints can change the structure/type of our solutions

- In a regular MDP we had a deterministic optimal policy

❓ But what about the following Frozen Lake problem?

Constraints

Meaning of constraints? Types of constraints?
Other forms of constraints are conceivable. Can you think of some?

Constraints

- Instantaneous/path constraints (state-action constraints or trajectory constraints), e.g.,

$$\text{s.t.} \quad \mathcal{C}(s_t, a_t) \leq \tau_i \quad \forall t$$

or

$$\text{s.t.} \quad \mathcal{C}(\tau) = \min_t \|\text{pos}(s_t) - \text{pos}_{\text{obstacle}}\|_2 \geq \eta_{\text{safe}}$$

(min distance to obstacles)

- Temporal/terminal constraints, e.g.,

$$P(\tau_{\text{goal}} \leq T_{\text{deadline}}) \geq 1 - \delta$$

(reach goal by the deadline)

Constraints

- Probabilistic (chance) constraints, e.g.,

$$\text{s.t.} \quad P(g(s_t, a_t) \leq 0) \geq 1 - \delta$$

- Risk-sensitive constraints (distributional constraints), e.g.,

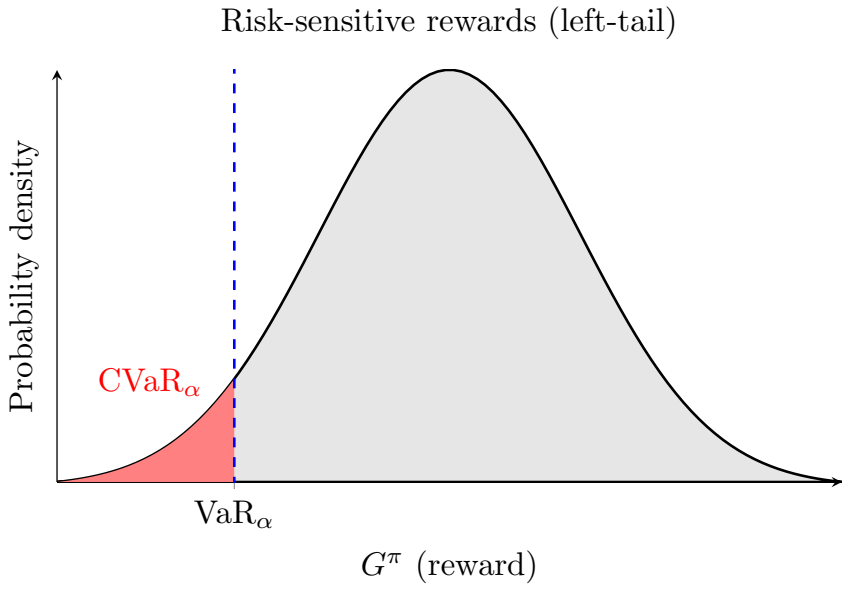
$$\text{VaR}_\alpha(G^\pi) = \inf\{g : P(G^\pi \leq g) \geq \alpha\}$$
$$\text{CVaR}_\alpha(G^\pi) = \mathbb{E}[G^\pi | G^\pi \geq VaR_\alpha(G^\pi)]$$

  where G^π is a policy dependent cumulative quantity, e.g., obtained cumulative reward

- Budget constraints (cumulative cost constraints)
- ...

Conditional Value at Risk (CVaR)

Illustration for a Gaussian distribution



- **Reward perspective (left-tail risk):**

$$\text{CVaR}_\alpha(G^\pi) = \mu - \sigma \frac{\phi\left(\Phi^{-1}(\alpha)\right)}{\alpha}$$

- Allows to train risk-averse agents

Constraints

- Budget constraints (cumulative cost constraints)
- Information/uncertainty constraints, e.g., constrain the agent to only take actions where its uncertainty is below a certain threshold.
- ...

Constraints in RL

We can categorize constraints based on their mathematical structure and their temporal scope

Categorization	Examples / Types
Structural	equality vs. inequality
Dependency	time-varying vs. time-invariant
	state-dependent vs. global
Temporal scope	per-step vs. pathwise
	cumulative vs. probabilistic

Constraints in RL

We can categorize constraints based on their mathematical structure and their temporal scope

Categorization	Examples / Types
Structural	equality vs. inequality
Dependency	time-varying vs. time-invariant
	state-dependent vs. global
Temporal scope	per-step vs. pathwise
	cumulative vs. probabilistic

Implications for guarantees:

- Hard constraints (pathwise): "Never violate" at any t
- Soft/Risk constraints (cumulative): Stay within a total budget on average or via probabilistic risk bounds

Constraints vs. rewards

Constraints vs. rewards

But are constraints really needed? We could in prinicple disencourage unwanted behavior by providing negative rewards...

Constraints vs. rewards

Constraints vs. rewards

But are constraints really needed? We could in prinicple disencourage unwanted behavior by providing negative rewards...

Constrained CMDP objective:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)\Big] \text{ s.t. } \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{C}_i(s_t, a_t)\Big] \leq \tau_i$$

Lagrangian relaxation (turns into an unconstrained objective with learned penalties):

$$\mathcal{L}(\pi, \lambda) = \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)\Big] - \Big[\sum_{i \in \mathcal{I}} \lambda_i \mathbb{E}_{\pi,\mathcal{P}}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{C}_i(s_t, a_t)\Big] - \tau_i\Big]$$

Constraints vs. rewards

Constraints vs. rewards

But are constraints really needed?

- Saddle-point/KKT conditions (strong duality holds under certain standard assumptions):
  - Primal feasibility: $\mathbb{E}_{\pi^*}[\sum \gamma^t \mathcal{C}_i] \leq \tau_i$
  - Dual feasibility: $\lambda_i^* \geq 0$
  - Complementary slackness: $\lambda_i^* \cdot (\mathbb{E}_{\pi^*}[\sum \gamma^t \mathcal{C}_i] - \tau_i) = 0$
  - Stationarity: $\pi^* \in \arg\max_\pi \mathcal{L}(\pi, \lambda^*)$

Constraints vs. rewards

Constraints vs. rewards

But are constraints really needed?

Penalties = soft constraints

Using fixed negative reward is equivalent to picking fixed λ

- Does not guarantee constraint satisfaction (λ must be chosen for KKT conditions to hold)
- When it works:
  - Expected cumulative constraints (classic CMDP) with additive costs: strong duality $\rightarrow \exists \lambda^*$ so the unconstrained optimum of $\mathcal{L}(\pi, \lambda^*)$ satisfies the constraints

Constraints vs. rewards

Constraints vs. rewards

But are constraints really needed?

When it does not work (or is risky to rely on penalties):

- Pathwise/instantaneous safety: expectation penalties allow to "make up for" violations later
- Risk constraints (chance/CVaR)
- Poorly scaled rewards/costs: tuning λ becomes brittle; large λ can stall learning or cause reward hacking
- ...

Choice of λ is instance dependent

Optimal choice of λ typically does not generalize across environments

Constraints for Alignment

Why are constraints important for alignment?

What do they give us what rewards don't give us?

Constraints for Alignment

Why are constraints important for alignment?
What do they give us what rewards don't give us?

- Reward misspecification and Goodhart's law
  - Pure reward optimization is susceptible to exploiting proxies and loopholes — constraints fence off known failure modes
- Safety-critical, zero-tolerance requirements
  - Some properties must "never" be violated (collisions, privacy leaks, illegal actions)
- Rare but catastrophic risks
  - Expected-reward objectives underweight tail events — risk constraints (chance/CVaR) target the tails explicitly
- Safe exploration and training-time safety
  - Negative rewards don't stop catastrophic trials during learning — constraints enable shields/filters that block unsafe actions

Constraints for Alignment

Why are constraints important for alignment?

What do they give us what rewards don't give us?

Constraints for Alignment

Why are constraints important for alignment?

What do they give us what rewards don't give us?

- Distribution shift and OOD robustness
  - Hard/robust constraints provide guardrails when the reward model is unreliable off-distribution
- Multi-objective alignment and norms
  - Systems often balance properties like helpfulness, honesty, harmlessnes—constraints encode priorities (safety $>$ utility)

Potential usage in LLMs (examples)

- Refusal constraints for harmful content
- Tool-use whitelists/blacklists
- Privacy redaction

Wrap Up and Outlook

Session 3 Summary: Constraints

What we covered today

- Active reward learning
- LLM alignment via RLHF
- Importance of constraints and various constraint types

Session 3 Summary: Constraints

What we covered today

- Active reward learning
- LLM alignment via RLHF
- Importance of constraints and various constraint types

❓ Are we done?

Session 3 Summary: Constraints

What we covered today

- Active reward learning
- LLM alignment via RLHF
- Importance of constraints and various constraint types

❓ Are we done?

Open challenges

- *Signal/feedback:* What types of feedback/signals can humans provide in various settings and (how) can we learn from them?
- *Theoretical limitations:* How much data is needed to accurately learn about rewards and constraints?

Looking Ahead to Session 4

- Human-AI Alignment
- Advanced Topics

Q&A

Next: Session 4 - Human-AI alignment and advanced topics

References & Resources

References

Session 3

- **CMDPs.**
  Altman, E. (1995). *Constrained Markov decision processes*. Available here:
- **Constrained policy optimization.** Achiam, J. et al. (2017). *Constrained policy optimization*, *ICML*.
- **Percentile risk criteria.** Chow, Y. et al. (2018). *"Risk-constrained reinforcement learning with percentile risk criteria"*.

References

Session 3 - Resources

- **Safety-Gymnasium.**
  Ji, J., Zhang, B., et al. (2023). *Safety Gymnasium: A Unified Safe Reinforcement Learning Benchmark*, *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
  Available here:
- **OmniSafe.**
  Ji, J., Zhou, J., et al. (2024). *"OmniSafe: An Infrastructure for Accelerating Safe Reinforcement Learning Research"*.
  Available here:
- **Safe-Control-Gym.**
  Yuan, Z. et al. (2022). *"Safe-Control-Gym: A Unified Benchmark Suite for Safe Learning-Based Control and Reinforcement Learning in Robotics"*.

References

Session 3 - Resources

- **AI safety gridworlds.**
  Leike, J., Martic, M., et al. (2017). *"AI safety gridworlds"*.
- **CARLA.**
  Dosovitskiy, A. et al. (2017). *CARLA: An Open Urban Driving Simulator*, *CoRL*.
- **Safe-Control-Gym.**
  Yuan, Z. et al. (2022). *"Safe-Control-Gym: A Unified Benchmark Suite for Safe Learning-Based Control and Reinforcement Learning in Robotics"*.