Corruption Robust Offline Reinforcement Learning with Human Feedback

Debmalya Mandal
University of Warwick

Andi Nika
MPI-SWS

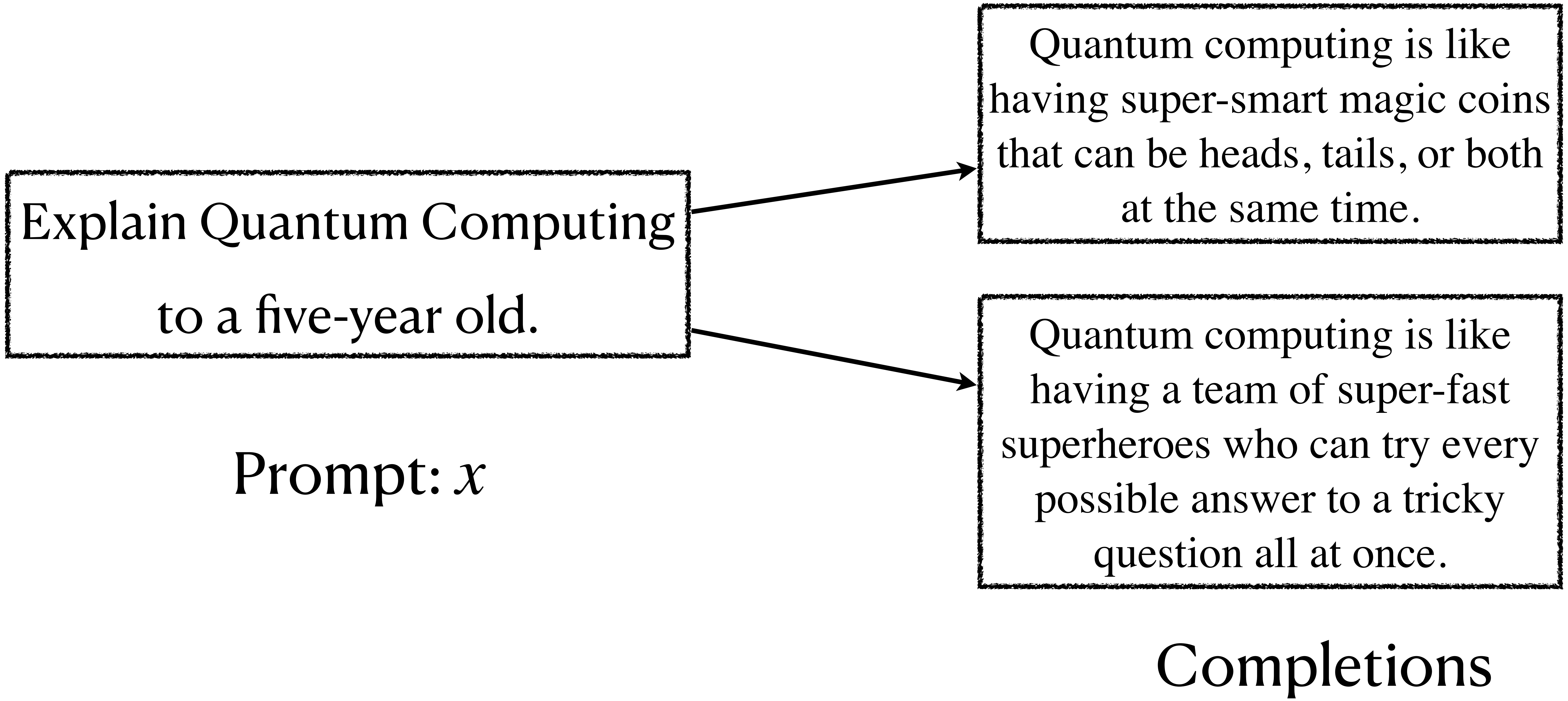
Parameswaran Kamalruban
Featurespace

Adish Singla
MPI-SWS

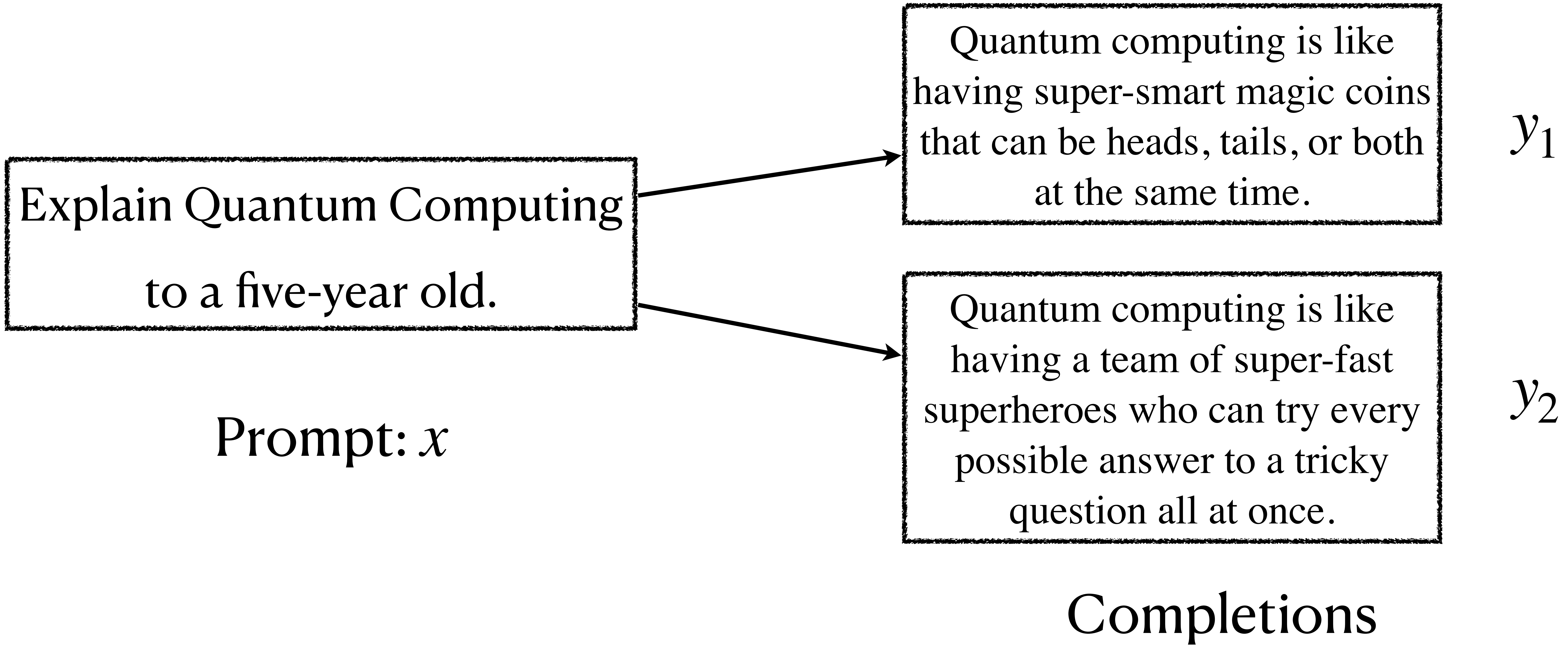
Goran Radanovic
MPI-SWS

AISTATS 2025

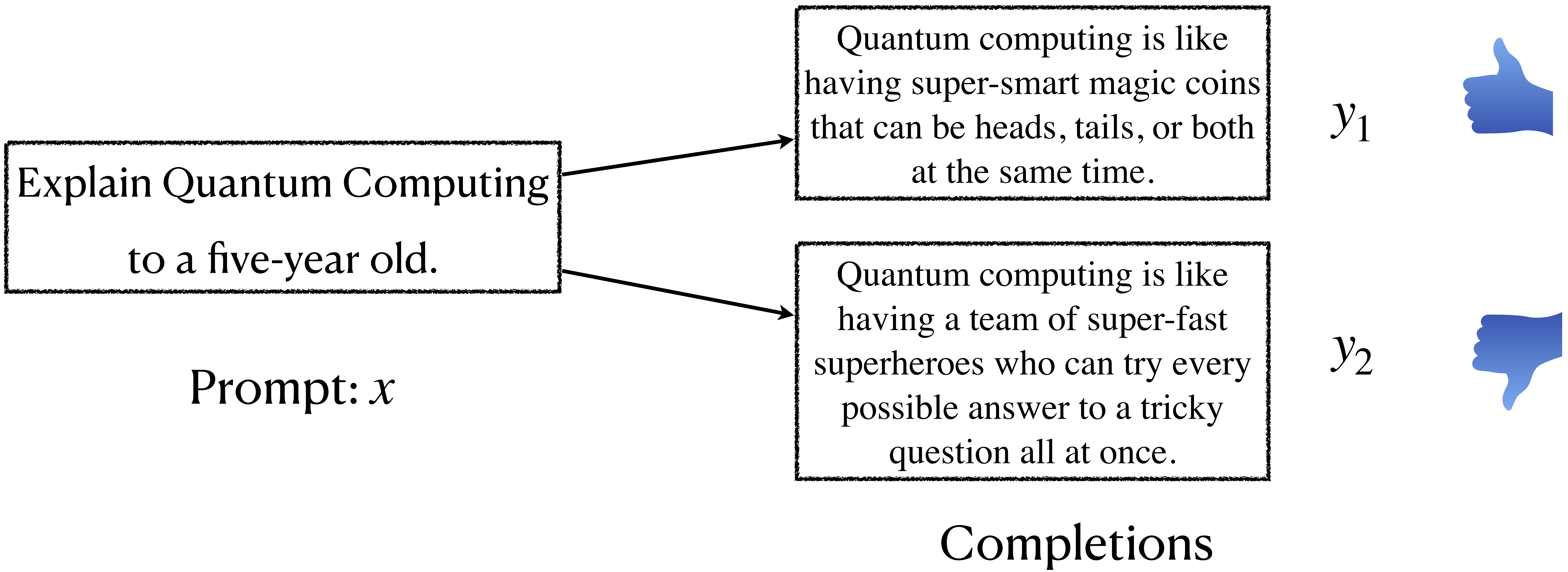
Reinforcement Learning with Human Feedback (RLHF)



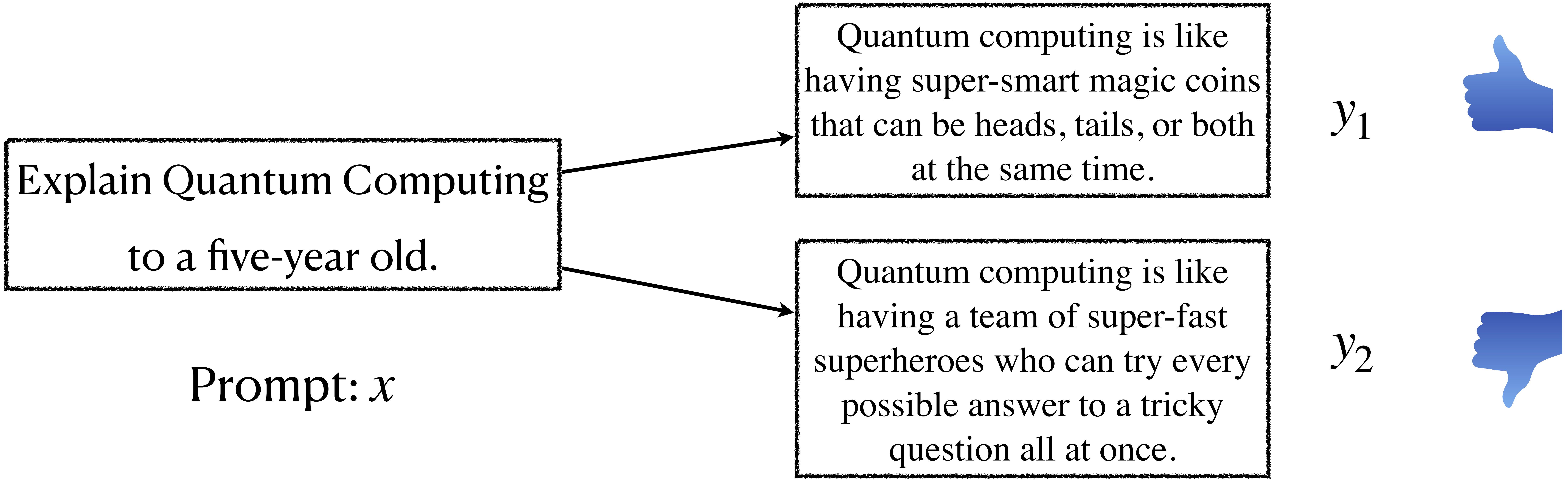
Reinforcement Learning with Human Feedback (RLHF)



Reinforcement Learning with Human Feedback (RLHF)



Reinforcement Learning with Human Feedback (RLHF)



Prompt: x

Completions

Preference: $y_1 \succ y_2$ if $r(x, y_1) > r(x, y_2)$

Preference Model

- Simplest Model: **Bradley-Terry** (BT) model
- $\Pr(y_1 \succ y_2 \mid x) = \dfrac{1}{1 + \exp(-(r(x, y_1) - r(x, y_2)))}$ [Contextual Bandits]

Preference Model

- Simplest Model: **Bradley-Terry** (BT) model
- $\Pr(y_1 \succ y_2 | x) = \frac{1}{1 + \exp(-(r(x, y_1) - r(x, y_2)))}$ [Contextual Bandits]

Reinforcement Learning Version [1]

[1] Deep Reinforcement Learning from Human Preferences, Christiano et. al. (NIPS-2017)

Preference Model

- Simplest Model: **Bradley-Terry** (BT) model
- $\Pr(y_1 \succ y_2 \mid x) = \dfrac{1}{1 + \exp(-(r(x, y_1) - r(x, y_2))}$ [Contextual Bandits]

Reinforcement Learning Version [1]

- Given pairs of trajectories (τ_1, τ_2) with
- $\tau_i = (s_1^i, a_1^i, \ldots, s_H^i, a_H^i)$
- $\Pr(\tau_1 \succ \tau_2) = \dfrac{1}{1 + \exp(-(r(\tau_1) - r(\tau_2))}$

[1] Deep Reinforcement Learning from Human Preferences, Christiano et. al. (NIPS-2017)

RLHF Pipeline

Given a dataset $\mathscr{D} = \{(\tau_1, \tau_2,$ 👍$)\}$

RLHF Pipeline

Given a dataset $\mathscr{D} = \{(\tau_1, \tau_2, \text{👍})\}$

Reward Estimation:

$$\hat{r} \leftarrow \min_{r} \ell(r; \mathscr{D}) = \sum_{(\tau_1, \tau_2) \in \mathscr{D}} - \log \Pr(\tau_1 \succ \tau_2 \mid r)$$

RLHF Pipeline

Given a dataset $\mathscr{D} = \{(\tau_1, \tau_2, \text{👍})\}$

Reward Estimation:

$$\hat{r} \leftarrow \min_{r} \ell(r; \mathscr{D}) = \sum_{(\tau_1, \tau_2) \in \mathscr{D}} - \log \Pr(\tau_1 \succ \tau_2 \,|\, r)$$

Policy Optimization:

$$\hat{\pi} \leftarrow \max_{\pi} V^{\pi}(\hat{r})$$ [discounted sum of returns under $\hat{r}$]

But where do we get the Datasets?

But where do we get the Datasets?

- Public repositories: Hugging Face
- Private datasets are collected through crowdsourcing
- Concerns: Inaccurate feedback, Subjective opinions [2], Manipulation by adversaries

[2] Whose Opinions do Language Models Reflect? Santurkar et. al. ICML-2023.

But where do we get the Datasets?

- Public repositories: Hugging Face
- Private datasets are collected through crowdsourcing
- Concerns: Inaccurate feedback, Subjective opinions [2], Manipulation by adversaries

- **Huber Corruption Model**: ε-fraction of the data is arbitrarily corrupted.

[2] Whose Opinions do Language Models Reflect? Santurkar et. al. ICML-2023.

Corruption-Robust in RLHF

Corruption-Robust in RLHF

- **Setup**: given a dataset $\mathscr{D} = \{(\tau_w^i, \tau_\ell^i, o^i)\}_{i=1}^n$ according to reward r.

Corruption-Robust in RLHF

- **Setup**: given a dataset $\mathscr{D} = \{(\tau_w^i, \tau_\ell^i, o^i)\}_{i=1}^n$ according to reward r.
- An adversary corrupts ε-fraction of the datapoints arbitrarily.

Corruption-Robust in RLHF

- **Setup**: given a dataset $\mathscr{D} = \{(\tau_w^i, \tau_\ell^i, o^i)\}_{i=1}^n$ according to reward r.
- An adversary corrupts ε-fraction of the datapoints arbitrarily.
- An RLHF algorithm outputs policy $\widehat{\pi}$

Corruption-Robust in RLHF

- **Setup**: given a dataset $\mathscr{D} = \{(\tau_w^i, \tau_\ell^i, o^i)\}_{i=1}^n$ according to reward r.
- An adversary corrupts ε-fraction of the datapoints arbitrarily.
- An RLHF algorithm outputs policy $\widehat{\pi}$
- We measure *suboptimality gap* $V^{\pi^\star}(r) - V^{\widehat{\pi}}(r)$

Corruption-Robust in RLHF

- **Setup**: given a dataset $\mathcal{D} = \{(\tau_w^i, \tau_\ell^i, o^i)\}_{i=1}^n$ according to reward r.
- An adversary corrupts ε-fraction of the datapoints arbitrarily.
- An RLHF algorithm outputs policy $\widehat{\pi}$
- We measure *suboptimality gap* $V^{\pi^\star}(r) - V^{\widehat{\pi}}(r)$

The problem is hard to solve without any assumption about the model.

Linear Markov Decision Process

Linear Markov Decision Process

- **Linear MDP**: reward and transition functions are linear in features. [3]

[3] Provably Efficient RL with Linear Function Approximation, Jin, Wang, and Jordan, COLT-2020.

Linear Markov Decision Process

- **Linear MDP**: reward and transition functions are linear in features. [3]
- There exist reward parameters (unknown) $\{\theta_h\}_{h=1}^H$ s.t.

$$r_h(s, a) = \phi(s, a)^\top \theta_h$$

[3] Provably Efficient RL with Linear Function Approximation, Jin, Wang, and Jordan, COLT-2020.

Linear Markov Decision Process

- **Linear MDP**: reward and transition functions are linear in features. [3]
- There exist reward parameters (unknown) $\{\theta_h\}_{h=1}^{H}$ s.t.

$$r_h(s, a) = \phi(s, a)^\top \theta_h$$

- There exist signed measures (unknown) $\{\boldsymbol{\mu}_h\}_{h=1}^{H}$ s.t.

$$P_h(s' \mid s, a) = \phi(s, a)^\top \boldsymbol{\mu}_h(s')$$

[3] Provably Efficient RL with Linear Function Approximation, Jin, Wang, and Jordan, COLT-2020.

Linear Markov Decision Process

- **Linear MDP**: reward and transition functions are linear in features. [3]
- There exist reward parameters (unknown) $\{\theta_h\}_{h=1}^{H}$ s.t.

$$r_h(s, a) = \phi(s, a)^\top \theta_h$$

- There exist signed measures (unknown) $\{\boldsymbol{\mu}_h\}_{h=1}^{H}$ s.t.

$$P_h(s' \mid s, a) = \phi(s, a)^\top \boldsymbol{\mu}_h(s')$$

- **Offline RL**: We need *coverage* assumption on the data.

[3] Provably Efficient RL with Linear Function Approximation, Jin, Wang, and Jordan, COLT-2020.

Results

Type of Coverage	Suboptimality Gap	#calls to Robust RL oracle
Uniform Coverage	$O\left(H^3\sqrt{d}\varepsilon^{1-o(1)}\right)$	$O(1)$
Low Relative Condition Number [4]	$\tilde{O}\left(H^2 d^{3/4}\varepsilon^{1/4}\right)$	$\tilde{O}\left(\frac{H^{3/2}d^5}{\varepsilon^3}\right)$
Generalized Coverage Ratio [5]	$\tilde{O}\left(H^2 d^{3/2}\sqrt{\varepsilon}\right)$	$O\left(\frac{1}{\varepsilon}\right)$

[4] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

[5] Offline Primal Dual RL for Linear MDPs, Gabbianelli et. al. AIStats-2024 .

Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL [6] (ε-approx)

[6] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL [6] (ε-approx)

1. Solve **trimmed maximum likelihood estimation**:

$$\hat{\theta} \leftarrow \operatorname*{argmax}_{\theta} \max_{S \subseteq \mathcal{D}: |S| = (1-\varepsilon)n} \sum_{\tau_1, \tau_2 \in S} \log \Pr(\tau_1 \succ \tau_2 \mid \theta)$$

[6] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL [6] (ε-approx)

1. Solve **trimmed maximum likelihood estimation**:

$$\hat{\theta} \leftarrow \operatorname*{argmax}_{\theta} \max_{S \subseteq \mathcal{D}:|S|=(1-\varepsilon)n} \sum_{\tau_1, \tau_2 \in S} \log \Pr(\tau_1 \succ \tau_2 | \theta)$$

2. Call oracle with $\hat{\theta}$ i.e. robust RL with rewards $r_h(s, a) = \phi(s, a)^\top \hat{\theta}$.

[6] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL [6] (ε-approx)

1. Solve **trimmed maximum likelihood estimation**:

$$\widehat{\theta} \leftarrow \underset{\theta}{\operatorname{argmax}} \max_{S \subseteq \mathscr{D}:|S|=(1-\varepsilon)n} \sum_{\tau_1,\tau_2 \in S} \log \Pr(\tau_1 \succ \tau_2 | \theta)$$

2. Call oracle with $\widehat{\theta}$ i.e. robust RL with rewards $r_h(s,a) = \phi(s,a)^\top \widehat{\theta}$.

- **Rationale**: with uniform coverage $\|\widehat{\theta} - \theta^\star\|_2 \leq O(\varepsilon^{1-o(1)})$

[6] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL [6] (ε-approx)

1. Solve **trimmed maximum likelihood estimation**:

$$\hat{\theta} \leftarrow \underset{\theta}{\operatorname{argmax}} \max_{S \subseteq \mathcal{D}:|S|=(1-\varepsilon)n} \sum_{\tau_1,\tau_2 \in S} \log \Pr(\tau_1 \succ \tau_2 | \theta)$$

2. Call oracle with $\hat{\theta}$ i.e. robust RL with rewards $r_h(s,a) = \phi(s,a)^\top \hat{\theta}$.

- **Rationale**: with uniform coverage $\|\hat{\theta} - \theta^\star\|_2 \leq O(\varepsilon^{1-o(1)})$
- **Computational efficiency**: Alternating optimization converges to a saddle point in $\tilde{O}(1/\varepsilon^2)$ iterations.

[6] Corruption-Robust Offline Reinforcement Learning, Zhang et. al. AIStats-2022 .

Non-Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL ($\sqrt{\varepsilon}$-approx)

Non-Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL ($\sqrt{\varepsilon}$-approx)

- Without uniform coverage,

$$\text{Log-Likelihood}(\widehat{\theta}, \mathcal{D}) - \text{Log-Likelihood}(\theta^\star, \mathcal{D}) \leq \tilde{O}\left(H\sqrt{d}\varepsilon + d/n\right)$$

Non-Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL ($\sqrt{\varepsilon}$-approx)

- Without uniform coverage,

$$\text{Log-Likelihood}(\widehat{\theta}, \mathscr{D}) - \text{Log-Likelihood}(\theta^\star, \mathscr{D}) \leq \tilde{O}\left(H\sqrt{d}\varepsilon + d/n\right)$$

- Construct a confidence set around $\widehat{\theta}$, say $\Theta(\mathscr{D})$ (convex for **Linear MDP**).

Non-Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL ($\sqrt{\varepsilon}$-approx)

- Without uniform coverage,

$$\text{Log-Likelihood}(\widehat{\theta}, \mathscr{D}) - \text{Log-Likelihood}(\theta^\star, \mathscr{D}) \leq \tilde{O}\left(H\sqrt{d}\varepsilon + d/n\right)$$

- Construct a confidence set around $\widehat{\theta}$, say $\Theta(\mathscr{D})$ (convex for **Linear MDP**).

2. Solve $\theta^\dagger \leftarrow \text{argmin}_{\theta \in \Theta(\mathscr{D})} \max_\pi V^\pi(\theta)$ **[Pessimistic Planning]**

Non-Uniform Coverage

Corruption Robust RLHF → Corruption Robust RL ($\sqrt{\varepsilon}$-approx)

- Without uniform coverage,

$$\text{Log-Likelihood}(\hat{\theta}, \mathscr{D}) - \text{Log-Likelihood}(\theta^\star, \mathscr{D}) \leq \tilde{O}\left(H\sqrt{d}\varepsilon + d/n\right)$$

- Construct a confidence set around $\hat{\theta}$, say $\Theta(\mathscr{D})$ (convex for **Linear MDP**).

2. Solve $\theta^\dagger \leftarrow \text{argmin}_{\theta \in \Theta(\mathscr{D})} \max_\pi V^\pi(\theta)$ [**Pessimistic Planning**]

3. Call oracle with $\theta^\dagger$ i.e. robust RL with rewards $r_h(s,a) = \phi(s,a)^\top \theta^\dagger$.

Non-Uniform Coverage (contd.)

Non-Uniform Coverage (contd.)

- Solving $\min_{\theta \in \Theta(\mathscr{D})} \underbrace{\max_\pi V^\pi(\theta)}_{\text{Convex function}}$ **[convex optimization]**

Non-Uniform Coverage (contd.)

- Solving $\min_{\theta \in \Theta(\mathscr{D})} \underbrace{\max_\pi V^\pi(\theta)}_{\text{Convex function}}$ [**convex optimization**]
- Solve convex optimization with

Non-Uniform Coverage (contd.)

- Solving $\min_{\theta \in \Theta(\mathcal{D})} \underbrace{\max_{\pi} V^{\pi}(\theta)}_{\text{Convex function}}$ [**convex optimization**]
- Solve convex optimization with
  1. **Zero-order** (noisy) oracle

Non-Uniform Coverage (contd.)

- Solving $\min_{\theta \in \Theta(\mathscr{D})} \underbrace{\max_\pi V^\pi(\theta)}_{\text{Convex function}}$ [**convex optimization**]
- Solve convex optimization with
  1. **Zero-order** (noisy) oracle
     (a) Use robust RL oracle calls to construct sub-gradient

Non-Uniform Coverage (contd.)

- Solving $\underbrace{\min_{\theta \in \Theta(\mathscr{D})} \max_{\pi} V^{\pi}(\theta)}_{\text{Convex function}}$ [**convex optimization**]
- Solve convex optimization with
  1. **Zero-order** (noisy) oracle
     (a) Use robust RL oracle calls to construct sub-gradient
     (b) Each call is $\sqrt{\varepsilon}$-biased $\Rightarrow \varepsilon^{1/4}$ sub-optimality gap

Non-Uniform Coverage (contd.)

- Solving $\min_{\theta \in \Theta(\mathscr{D})} \underbrace{\max_{\pi} V^{\pi}(\theta)}_{\text{Convex function}}$ **[convex optimization]**
- Solve convex optimization with
  1. **Zero-order** (noisy) oracle
     (a) Use robust RL oracle calls to construct sub-gradient
     (b) Each call is $\sqrt{\varepsilon}$-biased $\Rightarrow \varepsilon^{1/4}$ sub-optimality gap
  2. **First-order** (noisy) oracle

Non-Uniform Coverage (contd.)

- Solving $\underbrace{\min_{\theta \in \Theta(\mathscr{D})} \max_{\pi} V^{\pi}(\theta)}_{\text{Convex function}}$ **[convex optimization]**
- Solve convex optimization with
  1. **Zero-order** (noisy) oracle
     (a) Use robust RL oracle calls to construct sub-gradient
     (b) Each call is $\sqrt{\varepsilon}$-biased $\Rightarrow \varepsilon^{1/4}$ sub-optimality gap
  2. **First-order** (noisy) oracle
     (a) We use LP-based method to construct new robust RL oracle that returns $\sqrt{\varepsilon}$-apx sub-gradient $\Rightarrow \sqrt{\varepsilon}$ sub-optimality gap

Conclusion

Conclusion

- We provide corruption-robust RLHF under various coverage assumptions.

Conclusion

- We provide corruption-robust RLHF under various coverage assumptions.

- Some directions for future work:
  1. Corruption-robustness with general reward models
  2. Online learning setting

Conclusion

- We provide corruption-robust RLHF under various coverage assumptions.

- Some directions for future work:
  1. Corruption-robustness with general reward models
  2. Online learning setting

Thank you!