Deep Reinforcement Learning from Human Preferences

Paul F Christiano
OpenAI
paul@openai.com

Jan Leike
DeepMind
leike@google.com

Tom B Brown
nottombrown@gmail.com

Miljan Martic
DeepMind
miljanm@google.com

Shane Legg
DeepMind
legg@google.com

Dario Amodei
OpenAI
damodei@openai.com

NIPS 2017

Contents

Introduction

- For **sophisticated reinforcement learning (RL) systems** to interact usefully with real-world environments, we need to communicate complex goals to these systems.
- We explore goals defined in terms of **(non-expert) human preferences** between pairs of trajectory segments.
- We show that this approach can effectively solve complex RL tasks **without access to the reward function**.
- This **reduces the cost** of human oversight far enough that it can be practically applied to state-of-the-art RL systems.

Preliminaries

- At each time t:
  - Observation $o_t \in O$
  - Action $a_t \in A$
  - ~~Reward $r_t \in R$~~
- Trajectory segment :
  - $\sigma = ((o_0, a_0), (o_1, a_1), \ldots (o_{k-1}, a_{k-1})) \in (O \times A)^k$
  - $\sigma^1 \succ \sigma^2$: The human preferred trajectory segment σ^1 to trajectory segment σ^2.

Preliminaries – Evaluate in two ways

- **Quantitative**: We say that preferences $\succ$ are generated by a reward function $r: O \times A \to R$ if

$$((o_0^1, a_0^1), \dots, (o_{k-1}^1, a_{k-1}^1)) \succ ((o_0^2, a_0^2), \dots, (o_{k-1}^2, a_{k-1}^2))$$

  where

$$r(o_0^1, a_0^1) + \cdots + r(o_{k-1}^1, a_{k-1}^1) \succ r(o_0^2, a_0^2) + \cdots + r(o_{k-1}^2, a_{k-1}^2)$$

- **Qualitative**: Sometimes we have no reward function. In these cases, all we can do is qualitatively evaluate how well the agent satisfies to the human's preferences.

Method - Processes

- Parameters:
  - Policy $\pi: O \rightarrow A$
  - Reward function estimate $\hat{r}: O \times A \rightarrow R$

- Three processes:

1. The policy π interacts with the environment to produce a set of trajectories $\{\tau^1, \ldots, \tau^i\}$. The parameters of π are updated by a traditional reinforcement learning algorithm, in order to maximize the sum of the predicted rewards $r_t = \hat{r}(o_t, a_t)$.
2. We select pairs of segments $\left(\sigma^1, \sigma^2\right)$ from the trajectories $\{\tau^1, \ldots, \tau^i\}$ produced in step 1, and send them to a human for comparison.
3. The parameters of the mapping $\hat{r}$ are optimized via supervised learning to fit the comparisons collected from the human so far.

Method - Preference Elicitation

- Trajectory segment σ^1 and trajectory segment σ^2:
  - σ^1 is better than σ^2 or σ^2 is better than σ^1.
  - Two segments are equally good.
  - Two segments can not be compared.

- Store in database :
  - $(\sigma^1, \sigma^2, \mu)$
  - σ^1 and σ^2 are the two segments.
  - μ is a distribution over {1, 2} indicating which segment the user preferred.
  - If two segments are incomparable, the comparison is not included in the database.

Method – Fitting the Reward Function

- We can interpret a reward function estimate $\hat{r}$ as a preference-predictor if we view $\hat{r}$ as a latent factor explaining the human's judgments.

$$\hat{P}\left[\sigma^1 \succ \sigma^2\right] = \frac{\exp \sum \hat{r}\left(o_t^1, a_t^1\right)}{\exp \sum \hat{r}(o_t^1, a_t^1) + \exp \sum \hat{r}(o_t^2, a_t^2)}.$$

- Minimize the cross-entropy loss function.

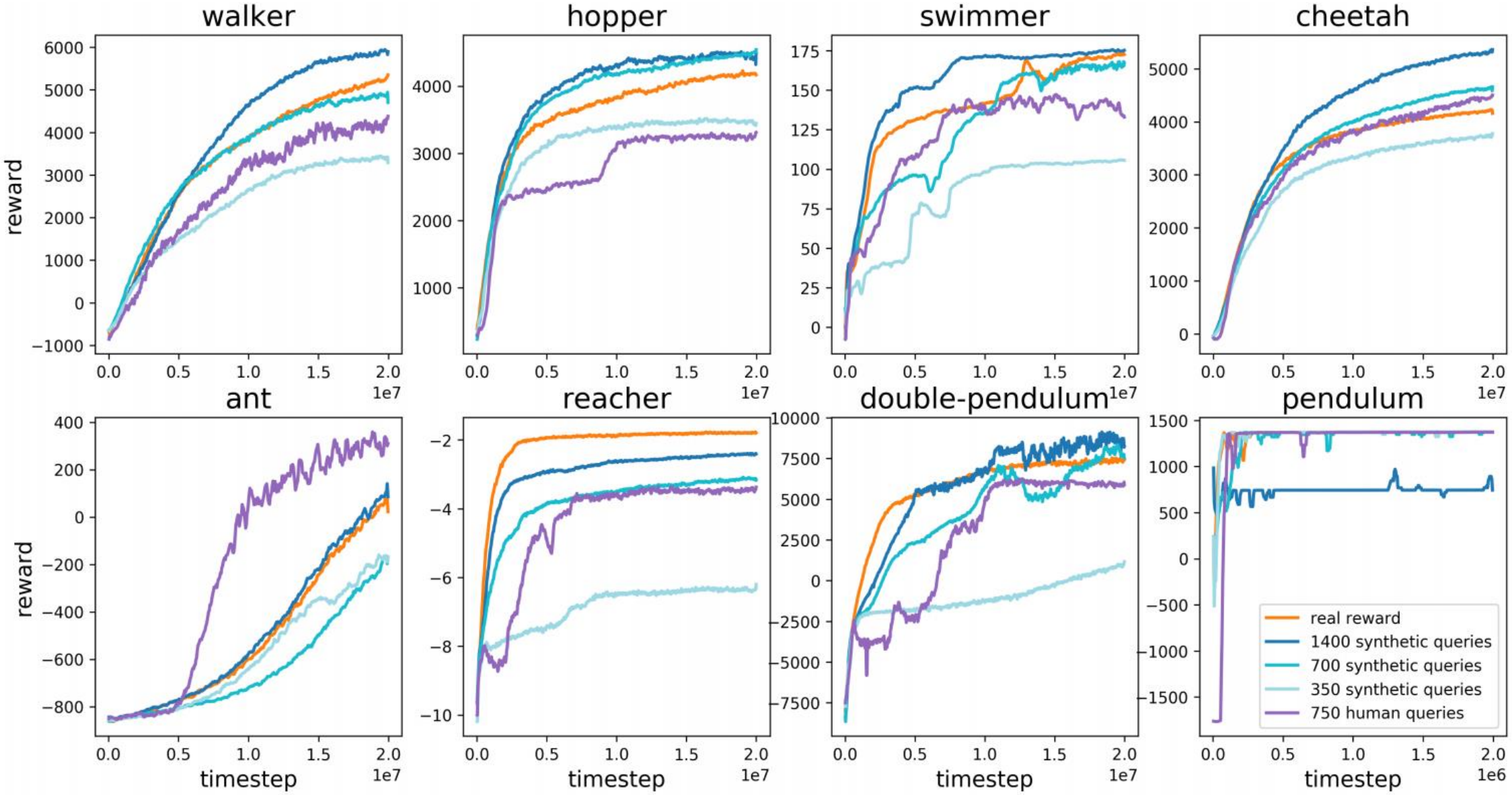
$$\text{loss}(\hat{r}) = -\sum_{(\sigma^1,\sigma^2,\mu)\in\mathcal{D}} \mu(1) \log \hat{P}\left[\sigma^1 \succ \sigma^2\right] + \mu(2) \log \hat{P}\left[\sigma^2 \succ \sigma^1\right].$$

Method – Selecting Queries

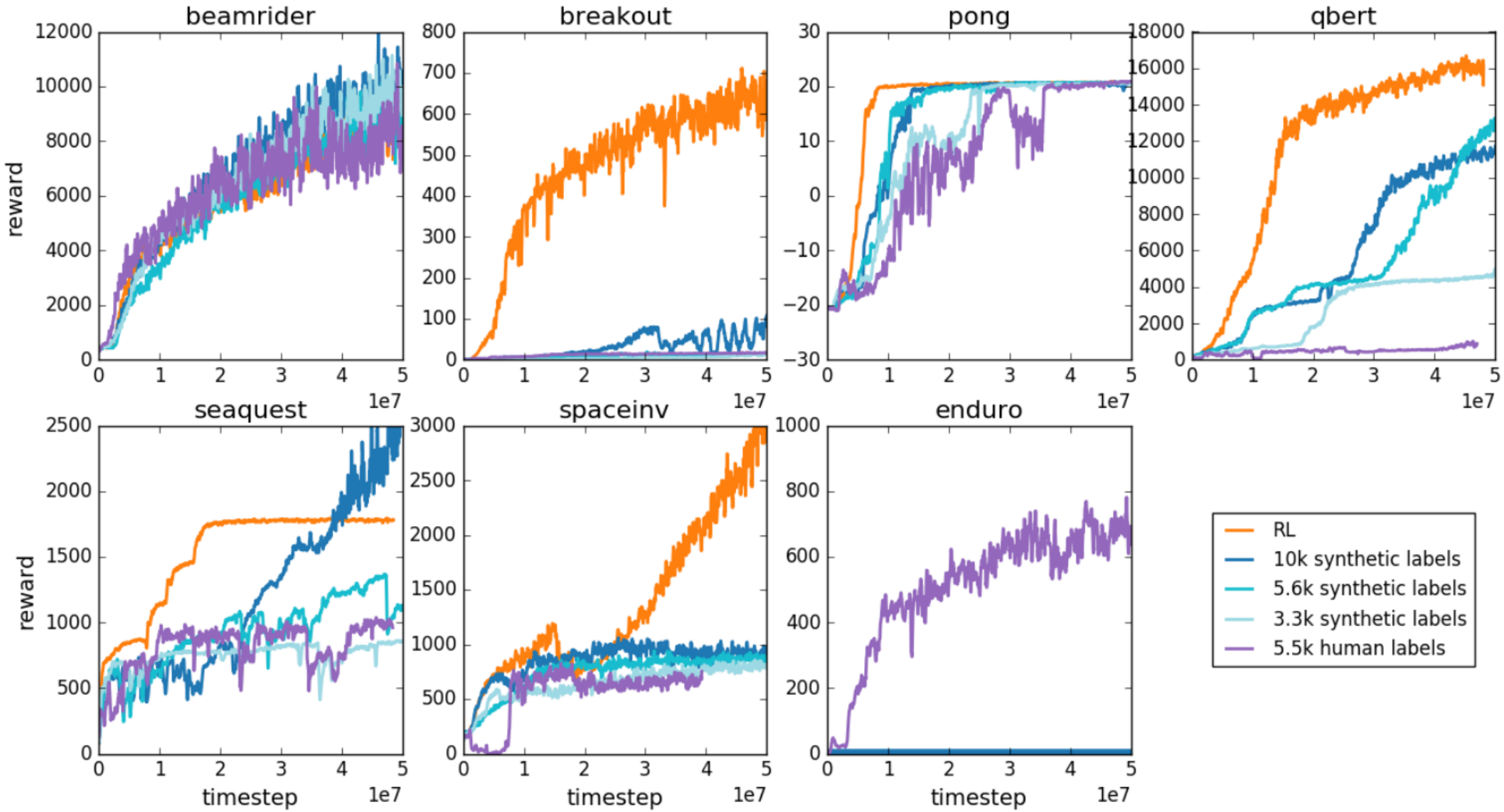
Crude approximation & Impairs performance

- Querying preferences based on an approximation to the uncertainty in the reward function estimator:
  - We sample a large number of pair of trajectory segments.
  - Using reward function estimator to predict which segment will be preferred from each pair.
  - Select those trajectories for which the predictions have the highest variance across ensemble members.
- Expectation:
  - Ideally, we would want to query based on the expected value of information of the query but we leave it to future work to explore this direction further.

Experiment – Simulated Robotics



Experiment – Atari



Experiment – Novel behaviors

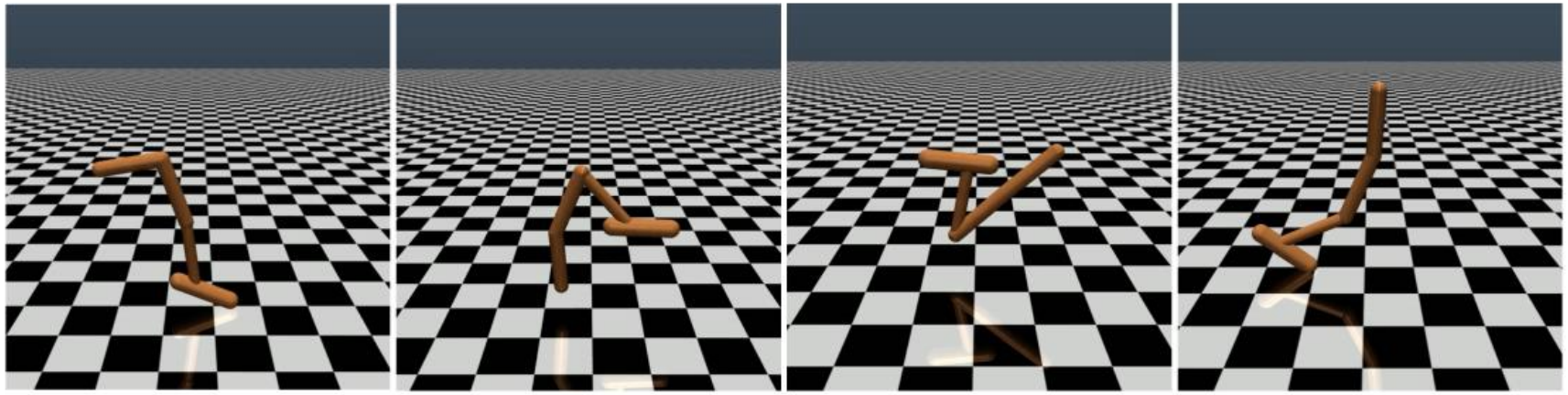
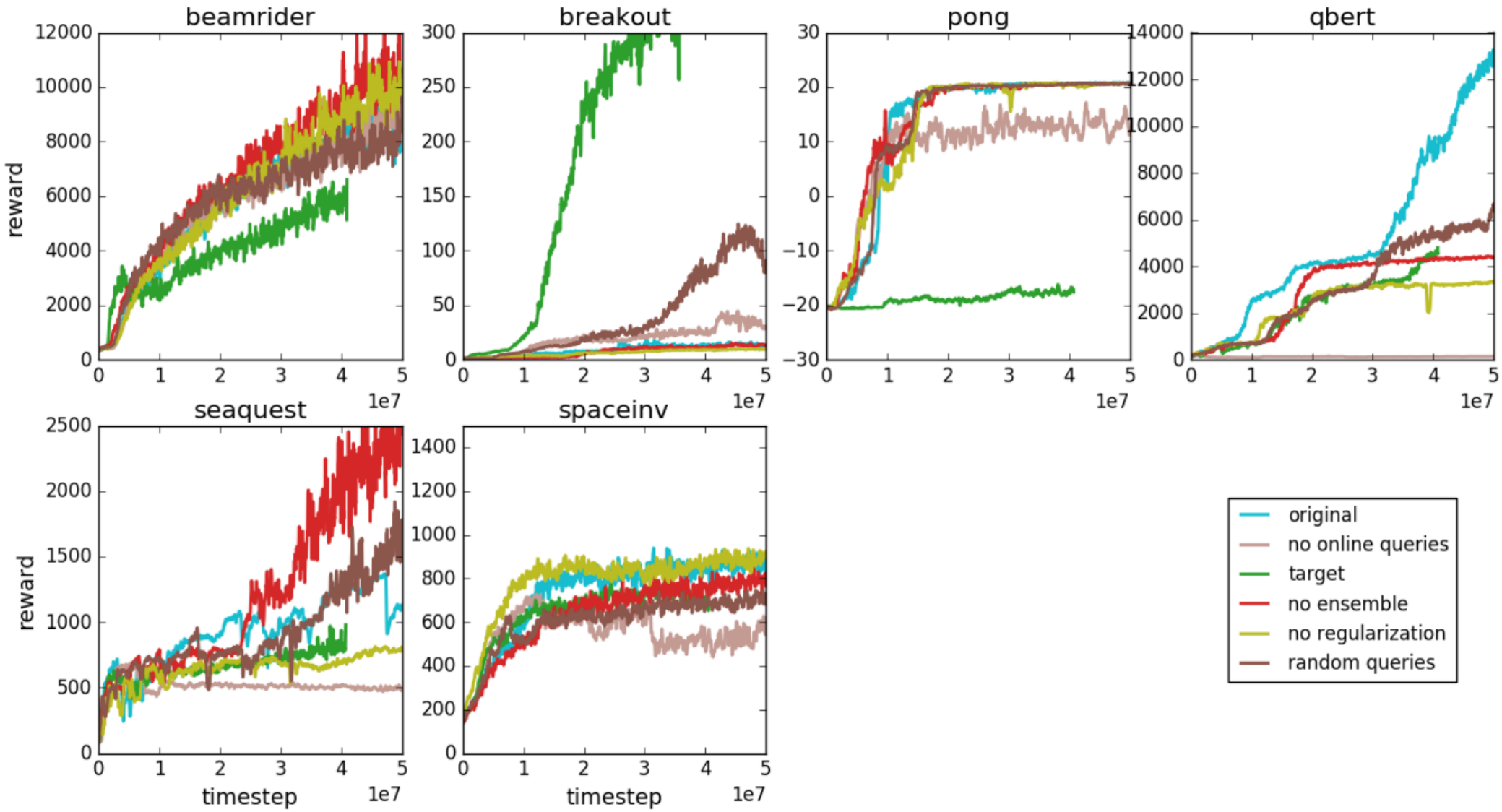


Figure 4: Four frames from a single backflip. The agent is trained to perform a sequence of backflips, landing upright each time. The video is available at `https://goo.gl/MhgvIU`.

- **Using the same parameters as in the previous experiments, we show that our algorithm can learn novel complex behaviors.**

Experiment – Query strategy



Conclusions

- We show that by learning a separate reward model using supervised learning, it is possible to **reduce the interaction complexity by roughly 3 orders of magnitude**.

- We provide the first evidence that these techniques can be economically **scaled up to state-of-the-art reinforcement learning systems**.

- Ensuring that powerful RL systems can be **applied in the service of complex human values** rather than low-complexity goals.