Université de Lille

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

Inria

Programming Markovian Policies by Human Feedback

Riad Akrour

November 26, 2024

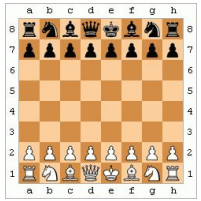
Personal Background

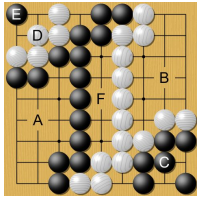
- PhD (2014) at Inria Saclay (TAU team) w/ Michèle Sebag and Marc Schoenauer
  - Topic: Preference-based Reinforcement Learning
- Postdoc 1 (6+ years) at TU Darmstadt (Germany) w/ Jan Peters and Gerhard Neumann
  - Topic: Entropy regularization in RL + Robotics
- Postdoc 2 (6 months) at Aalto University (Finland) w/ Joni Pajarinen
  - Topic: Reinforcement Learning and Computer Vision
- Researcher (ISFP, since 2022) at Inria Lille (Scool team) headed by Philippe Preux
  - Topic: Deep Reinforcement Learning

Personal Background

- PhD (2014) at Inria Saclay (TAU team) w/ Michèle Sebag and Marc Schoenauer
  - Topic: **Preference-based Reinforcement Learning (PbRL)**
- Postdoc 1 (6+ years) at TU Darmstadt (Germany) w/ Jan Peters and Gerhard Neumann
- Postdoc 2 (6 months) at Aalto University (Finland) w/ Joni Pajarinen
- Researcher (ISFP, since 2022) at Inria Lille (Scool team) headed by Philippe Preux
  - ANR project NeuRL: **Neuro-Incremental Reinforcement Learning from Human Preferences**
    - PbRL applied to crop management and precision agriculture
    - Research engineer hired in May 2024
    - PhD student started in October 2024

Successes of Deep Reinforcement Learning



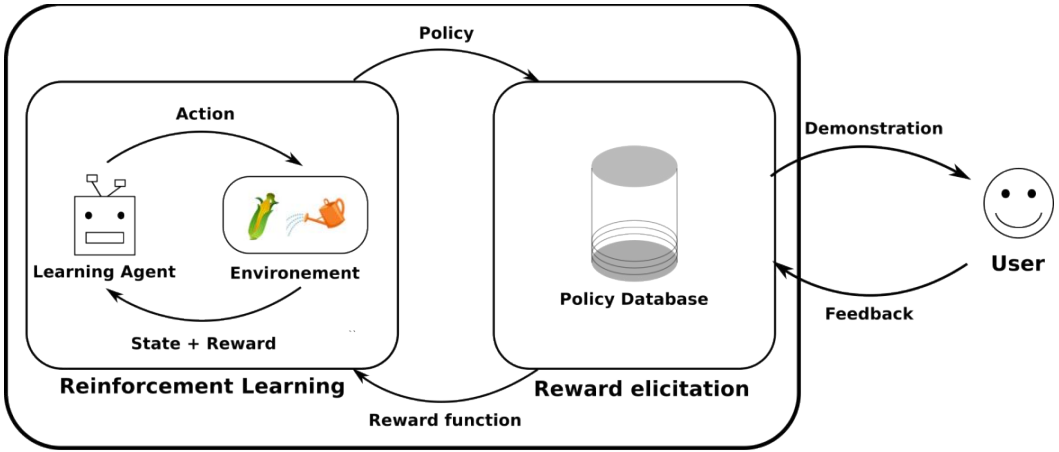




- Deep RL has known several successes, especially in game domains
- Specifying a task in RL = defining a reward function

$$\arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R\left(s_t, a_t\right) \mid a_t = \pi\left(s_t\right)\right]$$

Preference-based deep RL (**this talk**)

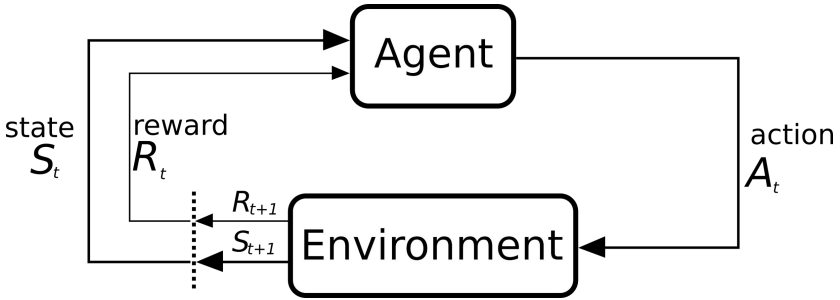
- Program the agent through higher level human feedback



Outline

- Deep RL in a nutshell
- RL with a human in the loop
- Deep Preference-based RL (PbRL) and key challenges
- ANR project NeuRL, PbRL applied to **crop management**
- Perspectives

Objective of Reinforcement Learning



- Objective in RL: maximize the policy return

$$\arg\max_{\pi} J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 \sim p_0,\ a_t = \pi(s_t), s_{t+1} \sim p(s_t, a_t)\right]$$

- Environment modelled as a Markov Decision Process (MDP): $(\mathcal{S},\ \mathcal{A},\ R, p,\ p_0, \gamma)$

Useful Definitions in RL

- Q-function $Q^{\pi}(s, a)$: expected cumulative rewards when doing *"a"* in *"s"* then following *π*

$$Q^{\pi}(s,a) = R(s,a) + \gamma \mathbb{E}\left[Q^{\pi}\left(s', a'\right)\right] \\ \stackrel{\Delta}{=} \left(T^{\pi} Q^{\pi}\right)(s,a)$$

- Policy improvement: if $Q^{\pi}\left(s, \pi'(s)\right) \geq Q^{\pi}(s, \pi(s))$ for all $s \in \mathcal{S}$ then $J(\pi') \geq J(\pi)$

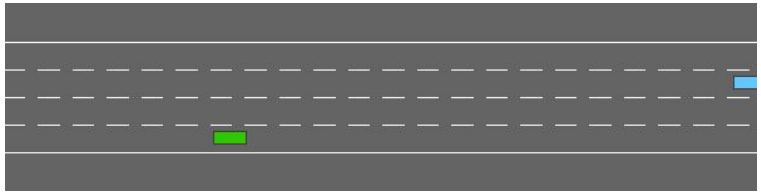
- T^{π} called the Bellman operator
  - It is a contraction with Q^{π} its unique fixed point
    - Let $Q_0 : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ an arbitrary function of the state-action space
    - and $Q_k = T^{\pi} Q_{k+1}$, then $\lim_{k \to \infty} Q_k = Q^{\pi}$

Deep RL in a Nutshell

- Randomly initialized neural network Q_0
- Repeat
  - Set $\pi_k(s) = \arg\max_{a \in \mathcal{A}} Q_k(s,a)$ or $\pi_k = \arg\max_{\pi} \mathbb{E}\left[Q_k(s, \pi(s))\right]$ # Policy improvement
  - Fit $Q_{k+1} = \arg\min_{Q} \mathbb{E}\left[\left\| R(s,a) + \gamma Q_k\left(s', \pi_k\left(s'\right)\right) - Q(s,a) \right\|_2^2\right]$ # Apply (empirical) Bellman operator $\hat{T}^{\pi}$
- <u>Deep RL in practice</u>
  - Can work well and solve decision making tasks with high dimensional state-action spaces
  - Can be unreliable (large variance across seeds, large performance oscillations/collapse)
  - Slow (can take several hours to several days on current GPUs)

RL with a Human in the Loop

Reward Specification in RL (1/2)



highway-v0, E. Leurent et al. 2018

- **Reward shaping**: rewards should be frequent enough to enable learning
- **Multiple objectives**: rewards trade objectives that are often conflicting
- **Sensitivity and threshold effect**: slight reward modifications can lead to large behavioral changes
    - Tuning the rewards is often a trial and error process

Reward Specification in RL (2/2)

- **Personalization**: Some RL tasks call for personalized rewards
  - E.g. RL in healthcare

> "(...) specifying such a reward function precisely is not only **difficult** but sometimes even **inadequate** and **misleading**. Several threshold and weighting parameters are needed to provide a way for **trading off efficacy** and **toxicity**, which heavily rely on **personal experience** that varies from one to another."

Reinforcement Learning in Healthcare: A Survey, Yu et al., 2021

- In summary: Some tasks have a clearly defined reward function (rewards -> behavior)
  - In other tasks reward specification is a **reverse problem** (behavior -> rewards)

Learning from Expert Demonstrations (1/2)

- Remainder of the talk: solving **MDP\R** problems: $(\mathcal{S},\ \mathcal{A}, p,\ p_0, \gamma)$
- **Approach 1**: Expert provides trajectories -> AI extracts policy reproducing these trajectories

P. Abbeel et al.; Autonomous Helicopter Aerobatics through Apprenticeship Learning; 2010

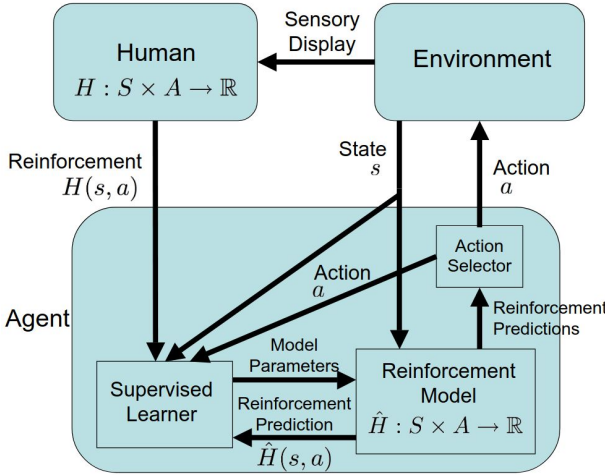
Learning from Expert Demonstrations (2/2)

1. Behavioral Cloning
    - Demonstrations -> Supervised Learning -> Policy; but in general training data ≠ test data
2. Inverse Reinforcement Learning
    - Demonstrations -> Rewards -> RL -> Policy; can be an ill-conditioned problem
3. Imitation Learning
    - Behavioral Cloning++ with typically more interactions on MDP\R

- Summary: Learning from Demonstrations useful for automating tasks we know how to solve
    - Leaves little room for the AI to find novel solution

Learning from Granular Human Feedback

- Human feedback on state-action pairs interactively during training
- What type of feedback, and how to interpret it?
  - Reward or Q-function? [Knox & Stone, 10; Warnell et al., 18]
  - Action correction? [Cederborg et al., 15; Chisari et al. 22]
    - "Do this action in this state"
  - Relative action correction? [Celemin et al., 22]
    - "Increase the speed in this state"

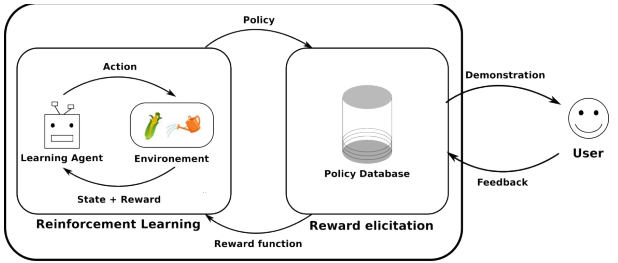


Interactively Shaping Agents via Human Reinforcement; Knox & Stone, 2009

- Due to its granularity, often combined to other types of feedback: demonstration, reward function…

Preference-based RL (PbRL)

- Preferences on trajectories the learner chose to demonstrate
  - Preferences -> Rewards -> RL [Akrour et al., 14; Christiano et al., 17; Hu et al. 24]
  - Preferences -> Direct Preference Optimization [Fürnkranz et al., 12; Rafailov et al. 23]



Schematic view of PbRL

- PbRL assumes very little expertise from the user [Cakmak & Thomaz, 12] but
  - Current (deep) PbRL methods require an unrealistic amount of feedback (500+)
  - Research for the most part focused on simulated users
- PbRL a.k.a. RLHF; Latter mostly used for training LLMs [e.g. Rafailov et al. 23]; Not the focus of this talk

Preference-based Reinforcement Learning

PbRL: General Overview (1/2)

- Focus on PbRL of type Preferences -> Rewards -> RL (most common [Wirth et al., 17; Celemin et al., 22])
  - **Select policies** π_1, π_2 maximizing an exploration/exploitation trade-off of the probabilistic reward model
  - **Request preference** feedback on pair of trajectories τ_1, τ_2 sampled from π_1, π_2 resp.
  - **Update** probabilistic reward **model** from preference feedback
- Exploration/exploitation of reward model
  - Exploitation, maximize the expected reward
  - Exploration, reduce uncertainty of the reward model
  - It is important to consider a trade-off of the two: we do not need to perfectly know the user's reward model to maximize it

PbRL: General Overview (2/2)

- **Key question** in PbRL: what policy selection criterion to find best policy with minimal user feedback
  - Similar questions as in preference elicitation, duelling bandits, Bayesian optimization,
- Practical challenges
  - How to **model** the rewards and their uncertainty in high-dimensional spaces?
  - How to **maximize** the policy selection criterion? RL?
  - How to deal with the sample inefficiency of deep RL?
- Lesser discussed challenges
  - Most PbRL benchmarks measured with synthetic (simulated) users
    - Understandable for reproducibility of experiments, but little is known on adequacy with ground truth

User and Reward Modelling

- The user response models the probability of the user preferring τ_1 over τ_2 which we denote $\tau_1 \succ \tau_2$
- Most common choice is the Bradley-Terry model [Bradley et al., 51]

$$P\left(\tau_1 \succ \tau_2; \hat{R}\right) = \frac{s\left(\tau_1; \hat{R}\right)}{s\left(\tau_1; \hat{R}\right) + s\left(\tau_2; \hat{R}\right)}$$

- s is a score function that maps trajectories to positive reals

$$s\left(\tau_1 = (s_0, a_0, \ldots, s_{H-1}, a_{H-1}); \hat{R}\right) = \exp\left(\sum_{t=0}^{H-1} \gamma^t \hat{R}\left(s_t, a_t\right)\right)$$

- $\hat{R}$ is a reward model mapping the state-action space to real values, e.g. can be a neural network

Training the Reward Model

- Given a dataset of trajectory pairs and associated ground truth probabilities

$$\mathcal{D}_k = \left\{ \left(\tau_1, \tau_1', \mu_1\right), \ldots, \left(\tau_k, \tau_k', \mu_k\right) \right\} \text{ with } \mu_i = \begin{cases} 1 & \tau_i \succ \tau_i' \\ 0 & \tau_i' \succ \tau_i \end{cases}$$

- Reward model $\hat{R}$ is then trained by minimizing the cross-entropy loss

$$\ell\left(\hat{R}, D_k\right) = -\frac{1}{k} \sum_{i=1}^{k} \mu_k \log P\left(\tau_i \succ \tau_i'; \hat{R}\right) + (1 - \mu_k) \log P\left(\tau_i' \succ \tau_i; \hat{R}\right)$$

- We also want to model reward uncertainty
  - Use an ensemble of neural networks [Christiano et al. 17; Lee et al. 21]
  - Bayesian modelling with MCMC sampling [Biyik et al., 19]

Query Selection in PbRL (1/3)

- Given a distribution over reward models, what two policies should we show the user? With what objective?
- Preference elicitation view: maximize the Expected Posterior Utility (EPU) [Viappiani & Boutilier 20]
  - Let $U = \{\tau_1, \ldots, \tau_N\}$ be a set of trajectories we can choose from
  - Preference elicitation aims at finding the most preferred item in the set U
  - Let P_k be the probability distribution over reward models after observing *k* user preferences
  - Define the Expected Utility (EU) and EU* as

$$\mathrm{EU}(\tau; P_k) = \sum_i P_k\left(\hat{R}_i\right) s\left(\tau; \hat{R}_i\right) \qquad \mathrm{EU}^\star(P_k) = \arg\max_\tau \mathrm{EU}(\tau; P_k)$$

  - Let $q = (\tau, \tau')$ be a query, $r \in \{\succ, \prec\}$ a user response and $P_k^{q,r}$ the belief after observing (*q,r*)
  - Then the Expected Posterior Utility (EPU) of a query is

$$\mathrm{EPU}(q) = \sum_i \sum_r P_k\left(\hat{R}_i\right) P\left(q, r; \hat{R}_i\right) \mathrm{EU}^\star\left(P_k^{q,r}\right)$$

Query Selection in PbRL (2/3)

- Maximizing EPU is myopically (one-step ahead) optimal
  - The expected utility will be the highest possible after one query, but expensive to compute
- Let $r\left(q\right)$ be the selected (preferred) trajectory in the query
- Expected Utility of Selection (EUS) of a query defined by

$$\mathrm{EUS}\left(q\right)=\sum_{i}\sum_{r}P_{k}\left(\hat{R}_{i}\right)P\left(q,r;\hat{R}_{i}\right)s\left(r\left(q\right);\hat{R}_{i}\right)$$

  - EUS quantifies the expected utility of trajectories in *q* (recommendation set)
- For some user response models maximizing EUS is equivalent to maximizing EPU [Viappiani & Boutilier 10]
  - Query set = Recommendation set
  - Intuition: a good recommendation set needs to contain diverse elements which makes for a good query
  - EUS much easier to maximize than EPU
  - If we maximize over set of policies instead of trajectories, can only maximize a lower bound of EUS

Query Selection in PbRL (3/3)

- Query selection in PbRL can alternatively be based on Posterior Sampling [Novoseller et al., 20; Wu et al., 24]
  - Sample reward functions $\hat{R} \sim P_k$, $\hat{R}' \sim P_k$ from current belief
  - Find policy maximizing the policy return $\pi = \arg\max_{\pi} J\left(\pi; \hat{R}\right)$ and similarly for π'
  - Sample trajectories $\tau \sim \pi$, $\tau' \sim \pi'$ and present query $q = \left(\tau, \tau'\right)$ to the user

- Theoretical guarantees of finding the most preferred policy [Novoseller et al., 20]
- Can use the previous policy for τ to solve only one RL problem between queries [Wu et al., 24]
- Solving an RL problem $\pi = \arg\max_{\pi} J\left(\pi; \hat{R}\right)$ between each query is challenging in a deep RL setting

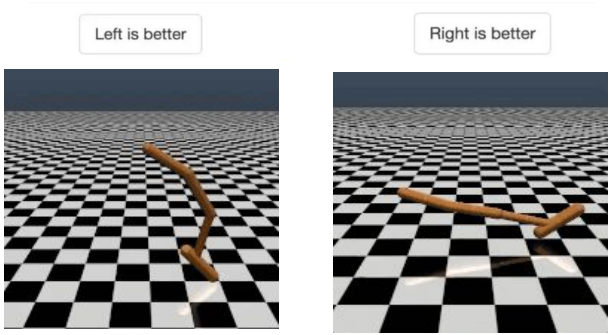
Deep PbRL: Quick Overview

Deep PbRL: Main concessions

- [Christiano et al., 17] is one of the first deep PbRL implementation
  - Also included a large scale study with non-expert human users (contractors)
- Main concessions to scale PbRL to the deep setting:
  - **Passive query set generation**: stochastic policy maximizing average reward, store trajectories in a replay buffer, queries sampled by picking trajectories from the buffer. Enough diversity?
  - **Uncertainty quantification**: fit an ensemble of three neural reward models on same data, difference in prediction stemming from difference in initialization. Capturing reward uncertainty?
  - **Simple query selection**: select a batch of trajectories that maximize probability disagreement, i.e. variance of $P\left(\tau \succ \tau'; \hat{R}_i\right)$ for different reward models. Over explorative?

Deep PbRL: User Study

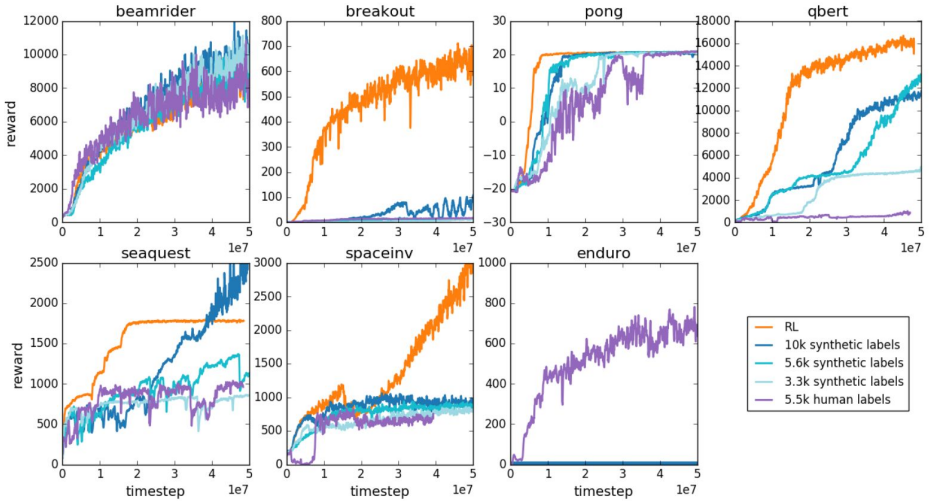
- In [Christiano et al., 17], contractors presented with task instructions or asked to play an Atari game for 5mn
  - Users periodically shown short trajectory clips
  - Instructions insist that feedback should be about actions in the clip, not situation agent is in (e.g. number of lifes in Atari tasks)
  - Takes users 3-5 seconds to provide one feedback
  - Training required 30mn to 5 hours of human time per task
  - User model: Bradley-Terry + constant noise



Deep Reinforcement Learning from Human Preferences [Christiano et al., 17]: teaching a task (backflip) from feedback that is hard to describe with a manually defined reward function

Deep PbRL: Experiment Results

- User feedback nearly as good as synthetic feedback with true reward model
- On *enduro* human feedback improves reward shaping
- PbRL + human feedback failed on *qbert* because game is too abstract



PbRL results on Atari games [Christiano et al., 17]

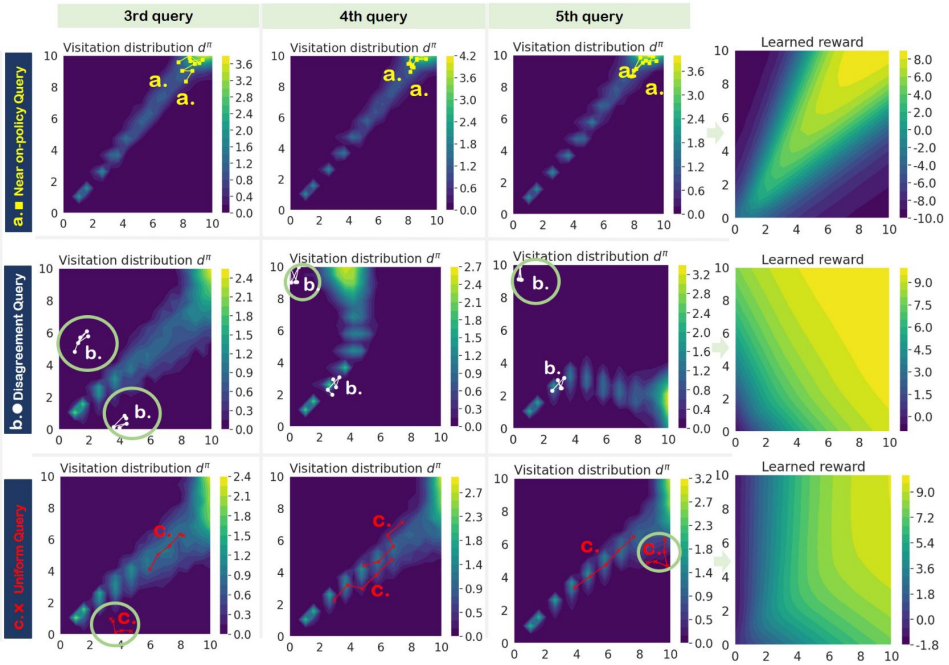
- Preference complexity likely too high in practice, was distributed among several users defeating the purpose of personalisation

Pre-training with Intrinsic Motivation

- One of the main concession of deep PbRL is that queries are not actively generated but sampled from a buffer
  - Quality of queries might decrease if buffer lacks diversity (especially true initially)
- Instead of starting PbRL from scratch, we can let the AI autonomously explore the MDP
  - Use **artificial curiosity** [Oudeyer et al., 07; Schmidhuber et al., 10] to fill the buffer with diverse trajectories in the absence of reward function
- [Lee et al., 21] proposed to **maximize state entropy** in the buffer as an intrinsic reward
  - Use intrinsic reward $r(s) = \log \left|\left|s - s^k\right|\right|$, where s^k is the nearest center following a K-NN clustering
  - Shown to improve feedback efficiency compared to baseline on several robot locomotion tasks

On-policy Query Selection

- Probability disagreement is **too explorative**: the goal is not to learn a uniformly accurate reward model
- [Hu et al., 24] propose to limit the query buffer to the most recent trajectories
- Query selected randomly from the buffer
- Single reward model learned (uncertainty not needed)
- Performance improves over baselines
- Corresponds to **pure exploitation**, noise in the query resulting purely from policy noise
- Queries hard to rank by human?



Query-Policy Misalignment in Preference-based Reinforcement Learning [Hu et al., 24]

PbRL for Crop Management (ANR NeuRL Project)

RL Environment for Crop Management

- [Gautron et al., 22] developed gym-DSSAT, an RL environment built on top of the DSSAT simulator
  - **DSSAT** simulates crop growth for 42 crops and is in use for over 30 years
  - Gym wrapper: allows the use of off-the-shelves deep RL algorithms for precision agriculture tasks
  - Collaboration between **Inria Scool** and **Cirad** (a French Agricultural Research Center)
- Precision agriculture tasks over several crop types and weather conditions gathered from real data
  - Actions include deciding when to plant, and when/how much to water and fertilize the crop
  - Can be used as a decision support system, but what is the reward function?

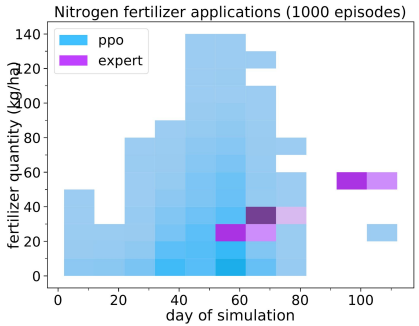
PbRL for Personalized Crop Management

- Some performance indicators in gym-DSSAT that can be used in the reward function [Gautron et al., 22]

variable	definition	comment
grnwt	grain yield (kg/ha)	quantitative objective to be maximized
pcngrn	massic fraction of nitrogen in grains	qualitative objective to be maximized
cumsumfert	total fertilization (kg/ha)	cost to be minimized
–	application number	cost to be minimized
–	nitrogen use efficiency (kg/kg)	agronomic criteria to be maximized
cleach	nitrate leaching (kg/ha)	loss/pollution to be minimized

expert	PPO
3686.5 (1841.0)	3463.1 (1628.4)
1.7 (0.2)	1.5 (0.3)
115.8 (5.2)	82.8 (15.2)
3.0 (0.1)	5.7 (1.6)
22.0 (14.1)	**28.3** (16.7)
18.0 (12.0)	18.3 (11.6)

- With canonical gym-DSSAT reward and a deep RL method we obtain the following compromise? Is it desirable?
- Beyond supporting decision making, studies have shown that farmers can internalize new knowledge from a DSS [Evans et al., 17]
- PbRL can personalize the decision support system and generate knowledge by exploring new strategies

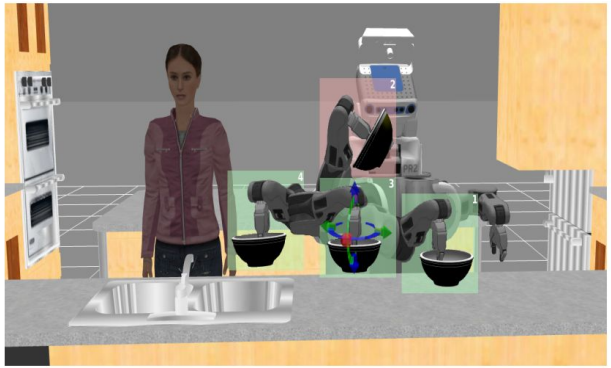


Towards Feedback Efficient Deep PbRL

- Current deep PbRL methods need 500+ human feedbacks to work — likely too much for our task
  - Orders of magnitude higher than non-deep PbRL with active query generation [Wirth et al., 17]
    - Need reliable and efficient deep RL algorithms to optimize query selection objective
    - **Reliable**: entropy regularized deep RL with growing neural nets = avoid catastrophic forgetting?
    - **Sample efficient**: reuse transition data with model-based RL?

Richer Human Feedback with Interpretable RL?

- Beyond preferences over trajectories users can provide feedback directly at policy level
- In some tasks (e.g. robot object manipulation on the side), policies have an interpretable structure (e.g. a set of waypoints) that can be easily modified by a user
- Interpretable RL could be a general way of learning compressed, human readable policies on which feedback can be given
    - E.g. “I’d do/avoid these actions in these situations”



Learning Preferences for Manipulation Tasks from Online Coactive Feedback [Jain et al., 15]