Predicting Future Actions of Reinforcement Learning Agents

Stephen Chung[1] Scott Niekum[2] David Krueger[3]

[1]University of Cambridge
[2]University of Massachusetts Amherst
[3]Mila

Introduction

Introduction

- Increasing application of RL in real world raises need to predict future agent actions
    - **Intervention for Dangerous Behavior**: Predicting an autonomous vehicle about to run a red light enables timely intervention
    - **Improve Human-Agent Interaction**: Helpful for passengers and other drivers to know if a nearby autonomous vehicle will turn left or right
- Goal: Predict future actions and associated events for a trained policy
- Orthogonal to safety RL, where goal is to train a safe policy

Problem Setting

- Notation: Let $H_t = (S_t, A_t, R_t)$ denote the transition at step t, and let $H_{t:T} = \{H_t, H_{t+1}, \ldots, H_T\}$
- Task: for some function f, we are interested in estimating the distribution of a random variable $f(H_{t:T})$ conditioned on the current state and action:
$$\mathbb{P}(f(H_{t:T}) \mid S_t, A_t). \tag{1}$$
- **Action Prediction**: $f(H_{t:T}) = (A_{t+1}, A_{t+2}, \ldots, A_{t+L})$ - interested in the action distribution in the next L steps
- **Event Prediction**: $f(H_{t:T}) = \max\{g(S_{t+k}, A_{t+k})\}_{k=1,\ldots,L}$ for some indicator function g, which is equivalent to estimating:
$$\mathbb{P}\left(\bigcup_{k=1}^{L} g(S_{t+k}, A_{t+k}) = 1 \,\middle|\, S_t, A_t\right), \tag{2}$$
which is the probability of an event defined by g to occur within L steps.
- Assume the policy is already a trained policy and fixed, and we have transitions generated by this policy. We do not limit the type of RL algorithm on which the policy is trained.

Agent Type

- Vanilla approach: treat as a supervised learning problem, using (S_t, A_t) as input and $f(H_{t:T})$ as the target output
- Additional information beyond the state-action pair is useful for prediction, such as hidden layer activations of a neural-network-based policy or other internal computations depending on the RL algorithm
- Useful to characterize agent type by their RL algorithm:
  - **Explicit planning agents**: Use an environment simulator or world model explicitly for planning. Examples: MuZero, Thinker
  - **Implicit planning agents**: No learned world model or explicit planning algorithm, yet these agents still exhibit planning-like behavior. Example: DRC
  - **Non-planning agents**: Neither a learned world model nor an explicit planning algorithm, and do not exhibit planning-like behavior. Example: IMPALA (a variant of actor-critic)
- We will focus on MuZero, Thinker, DRC and IMPALA policy in this work

Inner-state Approach

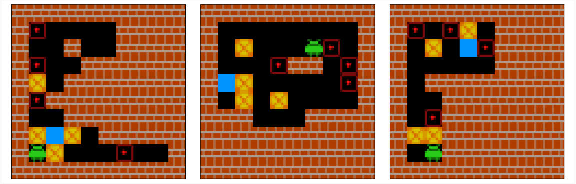
- **Inner state**: All intermediate computations required to compute the action A_t
- We provide the inner state as additional input along with (S_t, A_t) and train a predictor using a standard supervised learning method
- The following inner states are selected in this work:
    - **MuZero**: The most visited rollout during imagination
    - **Thinker**: All rollouts during imagination
    - **DRC**: The hidden states of the LSTM-based network
    - **IMPALA**: The final layer of the convolutional network
- Motivation: These inner states should contain additional useful information for the prediction problem

Simulation-based Approach

- Train a world model and simulate the agent in this model to generate rollouts, which are provided as additional input along with (S_t, A_t)
- In the ideal case where the world model perfectly matches environment dynamics, the empirical distribution of f in these rollouts converges to the target distribution
- However, this assumption is impractical in most settings, especially in real-world applications
- **Analogies**:
    - **Inner-state Approach**: Observe an animal's neural activations to understand its plan and predict its future actions
    - **Simulation-based Approach**: Place the animal in a controlled, matrix-like environment to observe its actions and use it to predict future actions

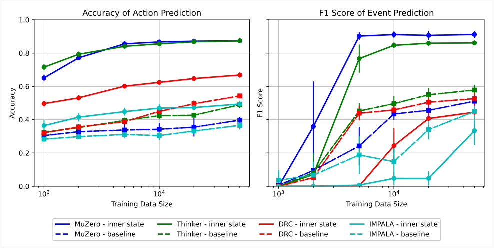
Conjecture: The inner-state approach will likely outperform the simulation-based approach unless the world model closely approximates the true environment dynamics.

Experiments Setup


Example levels of Sokoban

- We aim to predict the agent's action within $L = 5$ steps (action prediction) or determine if the agent will step on the blue tile within $L = 5$ steps (event prediction)
- We generate a fixed number of trained agents for this environment and train a deep predictor to match the target distribution; the predictor is evaluated on a separate test-level dataset

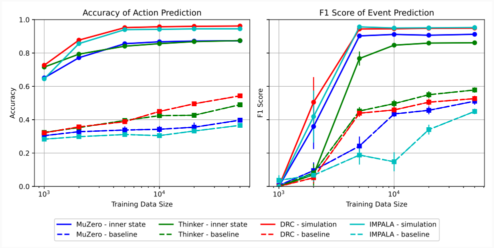
Experiment Results - Inner-state Approach



Final accuracy of action prediction and F1 score of event prediction with inner state approach on the testing dataset.

- **Action Prediction**: The inner-state of explicit planning agents significantly improves prediction accuracy; moderate improvement for implicit planning agents and minimal improvement for non-planning agents, consistent with the degree of planning in each type
- **Event Prediction**: The inner-state of explicit planning agents significantly improves prediction accuracy; no improvement for implicit and non-planning agents as they likely ignore the defined event

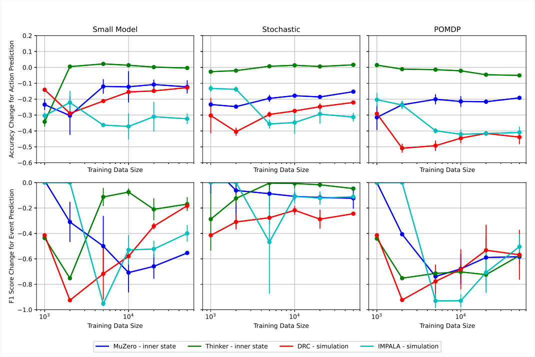
Experiment Results - Simulation-based Approach



Final accuracy of action prediction and F1 score of event prediction with simulation-based approach (DRC and IMPALA) on the testing dataset.

- The simulation-based approach for implicit planning and non-planning agents performs well and even outperforms the inner-state approach for explicit planning agents.

Experiment Results - World Model Ablation



Change in the final accuracy of action prediction and F1 score of event prediction for the world model ablation settings.

- We consider world model ablation such as a small world model, a stochastic environment, and a POMDP variant
- Generally, the inner-state approach is more robust to these ablations. Explicit planning agents, even with an imperfect world model, can adapt to inaccuracies, while other agent types are not trained to handle an inaccurate world model.

Conclusion

- Both inner-state and simulation-based approaches are useful for predicting future actions and events
- The inner-state approach performs best with explicit planning agents, followed by implicit planning agents, and then non-planning agents; explicit planning provides more useful information than implicit planning or learned state features
- The simulation-based approach works very well when an accurate world model is available
- The inner-state approach is more robust to world model quality than the simulation-based approach
- Event prediction is generally more challenging than action prediction, as defined events may not be relevant to the agent and can therefore be ignored