Learning Reward Machines for Partially Observable Reinforcement Learning

Rodrigo Toro Icarte, Ethan Walde, Torn Q. Klassen, Richard Valenzano, Margarita P. Castro, Sheila A. McIlraith

Abstract

Reward Machines (RMs), originally proposed for specifying problems in RL, provide a structured, automata-based representation of a reward function that allows an agent to decompose problems into subproblems that can be efficiently learned using off-policy learning. Here we show that RMs can be learned from experience, instead of being specified by the user, and that the resulting problem decomposition can be used to effectively solve partially observable RL problems. We pose the task of learning RMs as a discrete optimization problem where the objective is to find an RM that decomposes the problem into a set of subproblems such that the combination of their optimal memoryless policies is an optimal policy for the original problem. We show the effectiveness of this approach on three partially observable domains, where it significantly outperforms A3C, PPO, and ACER.

The Cookie Domain

RMs start at state \( s_0 \) and moves according to the edges. Each edge has an \( s_1 \)-like condition over high-level detectors and a reward.

Learning policies given an RM:
- **Standard**: Learn a policy \( \pi(o, a) \) that selects the action \( a \) considering the current observation \( o \) and RM state \( u \).
- **QRM**: Learn one policy per RM state and improve each policy in parallel using off-policy learning.

Main contributions

1. First approach for learning RMs from experience.
2. Extending RMs to work under partial observability.
3. Developing a theory for 1 and 2.

Learning Reward Machines

Problem setting: Find a policy that maximizes the collected external reward given by a partially observable environment \( \mathcal{E} \).

Assumptions: The agent has access to a set of high-level binary classifiers (e.g., \( (u, 0), (u, 1) \)) for the cookie domain.

Key insight: Learn an RM such that its internal state can be effectively used as external memory by the agent to solve the task.

Perfect Reward Machines

Perfect RMs make the environment Markovian w.r.t. \( O \times U \), i.e.:
\[ P(o_{t+1}, r_{t+1} | o_t, a_t, a_{t-1}) = P(o_{t+1}, r_{t+1} | o_t, a_t) \]
for every possible trace \( o_0, a_0, \ldots, a_{t-1} \) generated by any policy.

Properties:
- If the set of belief states of \( \mathcal{E} \) is finite, then there exists a perfect RM for \( \mathcal{E} \).
- Optimal policies over \( O \times U \) for perfect RMs are also optimal for \( \mathcal{E} \).

How to Learn Perfect Reward Machines

Ideally, for every possible RM, collect a set of traces, fit a predictive model for \( P(o_t, r_t | o, a) \), and pick the RM that makes better predictions. This is prohibitively expensive.

Alternative: Optimize over the necessary condition that perfect RMs must correctly predict—at least—what is possible and impossible in the environment at the level of the binary classifiers.

Motivation

Why is the cookie domain so challenging for Deep RL agents?
- It requires understanding of long-term dependencies.
- Solving this problem requires two levels of reasoning.

High-level: learning policies for navigating the map.

Low-level: reasoning about memory at the level of objects.

Idea: Let’s give the agent a set of high-level binary classifiers to detect—from the current observation—the color of the room, \( u \). Whether it sees a cookie \( u \), and whether it just ate a cookie \( u \) or pressed the button \( u \).

Can we do better now?

Example: The states from the “perfect RM” keep track of the possible location of the cookies—which is relevant for solving the task.

Learning RMs and Policies

1. Collect a training set \( T \) composed of random traces.
2. Learn an RM using \( T \).
3. Learn policies for the learned RM using RL.
4. If the RM predictions are incorrect, augment \( T \) and goto 2.

Summary of the Results

- Our LRM approaches outperform the baselines.
- DQRM pays off in domains with sparse rewards.
- LRM works because Tabu search finds high-quality RMs.

Discussion

This work represents a building block for creating RL agents that can solve cognitively challenging partially observable tasks. RL learning provided the agent with memory, but more importantly with discrete reasoning capabilities that operated at a higher level of abstraction (i.e., Tabu search) while leveraging deep RL’s abilities to learn policies from low-level inputs (i.e., DQRM).

References