Bidirectional End-to-End Framework for Transfer from Abstract Models in Non-Markovian Reinforcement Learning

Mahyar Alinejad[1],Precious Nwaorgu[1],Chinwendu Enyioha[1],Ỹue Wang[1],Ãlvaro Velasquez[2],George Atia[1]

[1]Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL
[2]Department of Computer Science, University of Colorado Boulder, CO
NEUS 2025

Abstract

We propose a bidirectional reinforcement learning (RL) framework for non-Markovian tasks in discrete and continuous environments. A simplified teacher model is learned from offline trajectories, alongside a Deterministic Finite Automaton (DFA) inferred via RPNI to capture task dependencies. Policies are trained in the teacher environment and transferred to the original domain through automaton distillation, enhancing efficiency over direct RL. The framework integrates DQN for discrete tasks and DDPG/TD3 for continuous settings, achieving faster convergence and better generalization, as shown in empirical results.

Introduction

Non-Markovian tasks, where rewards depend on state-action histories, challenge standard RL due to temporal dependencies. We address this by:

- Constructing a surrogate teacher environment from offline data.
- Inferring a DFA to encode task structure using RPNI.
- Transferring knowledge via automaton distillation to the original environment.

This bidirectional approach improves learning efficiency in complex discrete and continuous settings without predefined task structures.

Problem Formulation

We address a Non-Markovian Reward Decision Process (NMRDP):

$$\mathcal{M} = (S, A, T, R, \gamma),$$

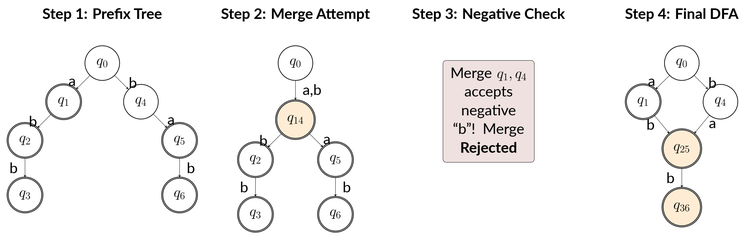
where $R : (S \times A)^* \to \mathbb{R}$ depends on state-action histories. Given an offline dataset $\mathcal{D} = \{(\tau_i, y_i)\}$, with trajectories τ_i and labels $y_i \in \{+1, -1\}$, we:

- Infer a DFA $\mathcal{A} = (\Omega, \Sigma, \delta, \omega_0, F)$ to encode task dependencies.
- Construct a surrogate teacher model $\widehat{\mathcal{M}} = (\widehat{S}, \widehat{A}, \widehat{T}, \widehat{R}, \gamma)$.
- Form a product MDP $\widehat{\mathcal{M}} \otimes \mathcal{A}$ with states (s, ω) to make rewards Markovian.

The goal is to learn an optimal policy π^* maximizing $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(\tau)\right]$ via bidirectional transfer.

Illustrative Example: RPNI Process and Product MDP

Alphabet: $\Sigma = \{a, b\}$, **Positives:** $\{a, ab, abb, ba, bab\}$, **Negatives:** $\{b, aa, bb\}$



- **Step 1:** Build Prefix Tree from positive examples.
- **Step 2:** Attempt merging states q_1 and q_4.
- **Step 3:** Reject merge if negative examples are accepted.
- **Step 4:** Final minimal DFA with valid merges.

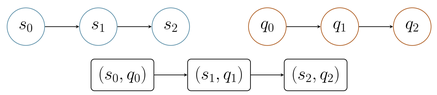


Figure 1. Product MDP: Combines environment state and DFA state to track task progress.

Methodology (Bidirectional Framework)

1. **Student-to-Teacher:**
   - Construct surrogate teacher model $\widehat{\mathcal{M}}$ from offline trajectories.
   - Infer DFA $\mathcal{A}$ using RPNI to capture task structure.
2. **Teacher Training:**
   - Form product MDP $\widehat{\mathcal{M}} \otimes \mathcal{A}$.
   - Train policy π_{teacher} using Q-learning/DQN (discrete) or DDPG/TD3 (continuous).
3. **Teacher-to-Student:**
   - Transfer knowledge via automaton distillation, guiding student policy in original environment.

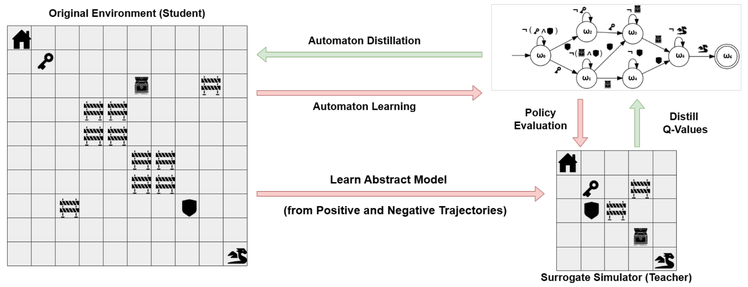


Figure 2. From offline data to teacher model and DFA, policy training and distillation back to student.

Teacher Model Abstraction

The surrogate teacher model $\widehat{\mathcal{M}} = (\widehat{S}, \widehat{A}, \widehat{T}, \widehat{R}, \gamma)$ is constructed from an offline dataset $\mathcal{D} = \{(\tau_i, y_i)\}$, where $\tau_i = (s_0, a_0, s_1, \dots)$ are trajectories and $y_i \in \{+1, -1\}$ are labels. Two cases are considered:

- **Discrete-to-Discrete:** For a discrete environment $\mathcal{M} = (S, A, T, R, \gamma)$, extract reduced sets $\widehat{S} \subseteq S$, $\widehat{A} \subseteq A$. Estimate transitions with Laplace smoothing ($\alpha > 0$):

$$\widehat{T}(s, a, s') = \frac{|\{(s, a, s') \in \mathcal{D}\}| + \alpha}{\sum_x (|\{(s, a, x) \in \mathcal{D}\}| + \alpha)}.$$

- **Continuous-to-Discrete:** For continuous states $s_c \in \mathbb{R}^n$, actions $a_c \in \mathbb{R}^m$, discretize via mappings $\phi_s : S \to \widehat{S}$, $\phi_a : A \to \widehat{A}$. Estimate transitions:

$$\widehat{T}(s_d, a_d, s'_d) = \frac{|\{(s_d, a_d, s'_d) \in \mathcal{D}\}| + \alpha}{\sum_x (|\{(s_d, a_d, x) \in \mathcal{D}\}| + \alpha)}.$$

The product MDP $\widehat{\mathcal{M}}' = \widehat{\mathcal{M}} \otimes \mathcal{A} = (\widehat{S} \times \Omega, \widehat{A}, \widehat{T}', \widehat{R}', \gamma)$ incorporates DFA states $\omega \in \Omega$, with rewards:

$$\widehat{R}'(s, \omega, a) = \begin{cases} +1, & \text{if } \omega \in F, \\ r(\omega, \omega'), & \text{otherwise,} \end{cases}$$

where $r(\omega, \omega')$ encourages progress in positive trajectories. This simplifies $\mathcal{M}$ for efficient policy training.

Automaton Distillation Process

Knowledge is transferred from the teacher to the student via automaton distillation:

- **Teacher Q-values:** $Q_{\text{teacher}}(s_d, \omega, a_d)$ learned in product MDP.
- **Distillation:** Compute average Q-values for automaton transitions:

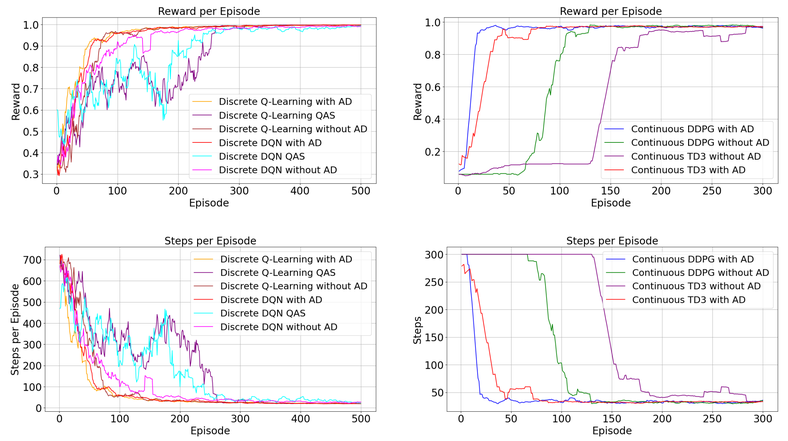
$$Q^{\text{avg}}_{\text{teacher}}(\omega, \sigma) = \frac{1}{N_{\omega,\sigma}} \sum_i Q_{\text{teacher}}(s^i_d, a^i_d).$$

- **Student Update:** Blend teacher knowledge with student learning:

$$Q_{\text{student}}(s, \omega, a) \leftarrow Q_{\text{student}}(s, \omega, a) + \alpha \big[\beta(\omega, \sigma) Q^{\text{avg}}_{\text{teacher}}(\omega, \sigma) + (1 - \beta(\omega, \sigma))(r + \gamma \max_{a'} Q_{\text{student}}(s', \omega', a')) - Q_{\text{student}}(s, \omega, a)\big].$$

Experimental Results

Environments: Dungeon Quest, Blind Craftsman, OfficeWorld, Minecraft Bridge.
Baselines: QAS (augmented state Q-learning/DQN) and Standard Q-learning/DQN (Discrete), Standard DDPG/TD3 (continuous).
Metrics: Reward per episode, steps per episode, reward per step.



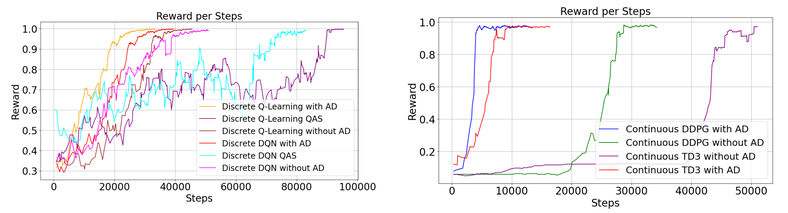


Figure 3. Student performance in the original Dungeon Quest environment after knowledge transfer. (**Left column**): Discrete-to-discrete mode; (**Right column**): Continuous-to-discrete mode. Rows show (**top**) reward per episode, (**middle**) steps per episode, and (**bottom**) reward per step.

Key Findings:

- **Faster Convergence:** AD accelerates learning by leveraging teacher knowledge.
- **Better Generalization:** Effective in both discrete and continuous settings.
- **Lower Sample Complexity:** Structured transfer reduces exploration needs.

Conclusion

Our bidirectional RL framework:

- Combines surrogate model construction, RPNI-based DFA inference, and automaton distillation.
- Handles non-Markovian tasks in discrete and continuous environments.
- Achieves faster convergence and better generalization than baselines.

Acknowledgment: Supported by DARPA (HR0011-24-9-0427) and NSF (CCF-2106339).

References

[1] J. Oncina et al., "Identifying Regular Languages in Polynomial Time," *ALT*, 1992.
[2] V. Mnih et al., "Human-Levelcar Control through Deep RL," *Nature*, 2015.
[3] T. Lillicrap et al., "Continuous Control with Deep RL," *ICLR*, 2016.
[4] S. Fujimoto et al., "Addressing Function Approximation Error in Actor-Critic Methods," *ICML*, 2018.