ETH zürich

Robust Knowledge Transfer in Tiered Reinforcement Learning

Jiawei Huang, Niao He

Department of Computer Science, ETH Zurich

D INFK

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Tiered RL Setting [1]**
  - Target/High-Tier task M_{Hi} + Source/Low-Tier task M_{Lo} learning in parallel
  - Knowledge transfer from M_{Lo} to M_{Hi}

- **Scenarios in Practice**
  - User Interaction Applications [1]
    - Users with higher risk tolerance: 
    - Users with lower risk tolerance: 

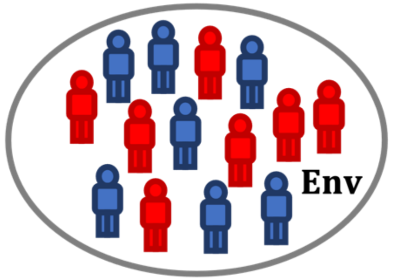

Figure from [1]

  - Robotics
    - Multiple robots learning in parallel
    - Some are more vulnerable than others

Figure from [2]

[1] Huang et. al., Tiered Reinforcement Learning: Pessimism in the Face of Uncertainty and Constant Regret. *NeurIPS 2022*
[2] Karol Hausman, *Research Blog*. https://blog.research.google/2021/04/multi-task-robotic-reinforcement.html?m=1

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Main Objective in Tiered RL Setting**
  - $\mathrm{Regret}(M_{\mathrm{Lo}})$: always near-optimal regret → Source tasks are also important in many cases
  - $\mathrm{Regret}(M_{\mathrm{Hi}})$:
    - **If tasks are similar:** better than optimal regret; → No negative transfer
    - **Otherwise:** keep near-optimal

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Main Objective in Tiered RL Setting**
  - $\mathrm{Regret}(M_{\mathrm{Lo}})$: always near-optimal regret → Source tasks are also important in many cases
  - $\mathrm{Regret}(M_{\mathrm{Hi}})$:
    - **If tasks are similar:** better than optimal regret;
    - **Otherwise:** keep near-optimal

    → No negative transfer
- **Limitation of Existing Knowledge Transfer Frameworks**

	Transfer RL	Multi-Task RL	Parallel Transfer RL (ours; [1])
Guarantees on low-tier/source task?	✗	✓	✓
Tasks learning in parallel?	✗	✓	✓
Distinguish high-tier/target and low-tier/source tasks?	✓	✗	✓

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Main Objective in Tiered RL Setting**
  - $\mathrm{Regret}(M_{\mathrm{Lo}})$: always near-optimal regret → Source tasks are also important in many cases
  - $\mathrm{Regret}(M_{\mathrm{Hi}})$:
    - **If tasks are similar:** better than optimal regret;
    - **Otherwise:** keep near-optimal → No negative transfer
- **Limitation of Existing Knowledge Transfer Frameworks**

	Transfer RL	Multi-Task RL	Parallel Transfer RL (ours; [1])
Guarantees on low-tier/source task?	❌	✅	✅
Tasks learning in parallel?	❌	✅	✅
Distinguish high-tier/target and low-tier/source tasks?	✅	❌	✅

- **Limitation of Existing Tiered RL Literature [1]**
  - Strong prior knowledge: $M_{\mathrm{Hi}} = M_{\mathrm{Lo}}$

[1] Huang et. al., Tiered Reinforcement Learning: Pessimism in the Face of Uncertainty and Constant Regret. *NeurIPS 2022*

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Setting**
  - Tabular MDP with finite horizon H
  - M_{Hi} shares state-action space with M_{Lo}
  - No prior knowledge about similarity between M_{Hi} and M_{Lo}

[1] Golowich et. al., Can Q-Learning be Improved with Advice? *COLT 2022*
[2] Gupta et. al., Unpacking Reward Shaping: Understanding the Benefits of Reward Engineering on Sample Complexity. *NeurIPS 2022*

Tiered Reinforcement Learning: A Parallel Knowledge Transfer Framework

- **Setting**
  - Tabular MDP with finite horizon H
  - M_{Hi} shares state-action space with M_{Lo}
  - No prior knowledge about similarity between M_{Hi} and M_{Lo}

- **Main Assumption**
  - Optimal Value Dominance:
    - $\forall h, s_h, V^*_{\mathrm{Lo}}(s_h) \geq V^*_{\mathrm{Hi}}(s_h)$
    - Similar assumptions in [3,4]
    - Theorem 3.1 [Lower bound]: negative transfer is unavoidable if violated

[1] Golowich et. al., Can Q-Learning be Improved with Advice? *COLT 2022*
[2] Gupta et. al., Unpacking Reward Shaping: Understanding the Benefits of Reward Engineering on Sample Complexity. *NeurIPS 2022*

Main Results

- **Bandit & RL Setting with Single Source Task**
  - $\mathrm{Regret}(M_{\mathrm{Hi}}, K) = O\left(SH \sum_h \sum_{s_h, a_h \notin \mathrm{Transferable}(M_{\mathrm{Hi}}, M_{\mathrm{Lo}})} \frac{1}{\Delta(s_h, a_h) \vee \frac{\Delta_{\min}}{H}} \log K\right)$

Main Results

- **Bandit & RL Setting with Single Source Task**
  - $\mathrm{Regret}(M_{\mathrm{Hi}}, K) = O(SH \sum_h \sum_{s_h, a_h \notin \mathrm{Transferable}(M_{\mathrm{Hi}}, M_{\mathrm{Lo}})} \frac{1}{\Delta(s_h, a_h) \vee \frac{\Delta_{\min}}{H}} \log K)$

Transferable states in single source task setting:

$$d^*_{\mathrm{Lo}}(s_h) > 0;$$
$$V^*_{\mathrm{Lo}}(s_h) \leq V^*_{\mathrm{Hi}}(s_h) + O\left(\frac{\widetilde{\Delta}}{H}\right);$$
$$\pi^*_{\mathrm{Lo}}(s_h) = \pi^*_{\mathrm{Hi}}(s_h)$$

Main Results

- **Bandit & RL Setting with Single Source Task**
  - $\text{Regret}(M_{\text{Hi}}, K) = O(SH \sum_h \sum_{s_h, a_h \notin \text{Transferable}(M_{\text{Hi}}, M_{\text{Lo}})} \frac{1}{\Delta(s_h, a_h) \vee \frac{\Delta_{\min}}{H}} \log K\ )$

Transferable states in single source task setting:

$$d^*_{\text{Lo}}(s_h) > 0;$$
$$V^*_{\text{Lo}}(s_h) \leq V^*_{\text{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right);$$
$$\pi^*_{\text{Lo}}(s_h) = \pi^*_{\text{Hi}}(s_h)$$

- **Bandit & RL Setting with Multiple Source Tasks**
  - W-Source Tasks: $M^1_{\text{Lo}}, \dots, M^W_{\text{Lo}}$
  - $\text{Regret}(M_{\text{Hi}}, K) = O(SH \sum_h \sum_{s_h, a_h \notin \text{Transferable}(M_{\text{Hi}}, M^1_{\text{Lo}}, \dots, M^W_{\text{Lo}})} \frac{1}{\Delta(s_h, a_h) \vee \frac{\Delta_{\min}}{H}} \log WK\ )$

Transferable states in single source task setting:

$$d^*_{\text{Lo}}(s_h) > 0;$$
$$\exists w \in [W]\ V^*_{\text{Lo},w}(s_h) \leq V^*_{\text{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right),$$
$$\pi^*_{\text{Lo},w}(s_h) = \pi^*_{\text{Hi}}(s_h)$$

Overview of Robust Knowledge Transfer Mechanism

- **Single Source Task Setting:**
  - Key idea: separation between transferable & non-transferable states
    - If transferable: $V^*_{\mathrm{Lo}}(s_h) \leq V^*_{\mathrm{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right) = Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right)$
    - Otherwise: $V^*_{\mathrm{Lo}}(s_h) \geq V^*_{\mathrm{Hi}}(s_h) \geq Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) + O(\frac{H-1}{H}\Delta_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}))$

Overview of Robust Knowledge Transfer Mechanism

- **Single Source Task Setting:**
  - Key idea: separation between transferable & non-transferable states
    - If transferable: $V^*_{\mathrm{Lo}}(s_h) \leq V^*_{\mathrm{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right) = Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right)$
    - Otherwise: $V^*_{\mathrm{Lo}}(s_h) \geq V^*_{\mathrm{Hi}}(s_h) \geq Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) + O\left(\frac{H-1}{H}\Delta_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}})\right)$
    - Checking condition:
      - $Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) \geq V^*_{\mathrm{Lo}}(s_h)$

Overview of Robust Knowledge Transfer Mechanism

- **Single Source Task Setting:**
  - Key idea: separation between transferable & non-transferable states
    - If transferable: $V^*_{\mathrm{Lo}}(s_h) \leq V^*_{\mathrm{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right) = Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right)$
    - Otherwise: $V^*_{\mathrm{Lo}}(s_h) \geq V^*_{\mathrm{Hi}}(s_h) \geq Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) + O(\frac{H-1}{H}\Delta_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}))$
    - Checking condition:
      - $\overline{Q}^*_{\mathrm{Hi}}(s_h, \underline{\pi}^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) \geq \underline{V}^*_{\mathrm{Lo}}(s_h)$

Estimated by UCB in M_{Hi}

Estimated by LCB in M_{Lo}

Overview of Robust Knowledge Transfer Mechanism

- **Single Source Task Setting:**
  - Key idea: separation between transferable & non-transferable states
    - If transferable: $V^*_{\mathrm{Lo}}(s_h) \leq V^*_{\mathrm{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right) = Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right)$
    - Otherwise: $V^*_{\mathrm{Lo}}(s_h) \geq V^*_{\mathrm{Hi}}(s_h) \geq Q^*_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) + O\left(\frac{H-1}{H}\Delta_{\mathrm{Hi}}(s_h, \pi^*_{\mathrm{Lo}})\right)$
    - Checking condition:
      - $\overline{Q}^*_{\mathrm{Hi}}(s_h, \underline{\pi}^*_{\mathrm{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) \geq \underline{V}^*_{\mathrm{Lo}}(s_h)$

Estimated by UCB in M_{Hi}

Estimated by LCB in M_{Lo}

    - Avoid negative transfer
      - Every negative transfer will result in tighter estimation of Q^*_{Hi} and V^*_{Lo}

Overview of Robust Knowledge Transfer Mechanism

- **Single Source Task Setting:**
  - Key idea: separation between transferable & non-transferable states
    - If transferable: $V^*_{\text{Lo}}(s_h) \leq V^*_{\text{Hi}}(s_h) + O\left(\frac{\tilde{\Delta}}{H}\right) = Q^*_{\text{Hi}}(s_h, \pi^*_{\text{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right)$
    - Otherwise: $V^*_{\text{Lo}}(s_h) \geq V^*_{\text{Hi}}(s_h) \geq Q^*_{\text{Hi}}(s_h, \pi^*_{\text{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) + O(\frac{H-1}{H}\Delta_{\text{Hi}}(s_h, \pi^*_{\text{Lo}}))$
    - Checking condition:
      - $\overline{Q}^*_{\text{Hi}}(s_h, \underline{\pi}^*_{\text{Lo}}) + O\left(\frac{\tilde{\Delta}}{H}\right) \geq \underline{V}^*_{\text{Lo}}(s_h)$

Estimated by UCB in M_{Hi}

Estimated by LCB in M_{Lo}

Estimation error tolerance

    - Avoid negative transfer
      - Every negative transfer will result in tighter estimation of Q^*_{Hi} and V^*_{Lo}

Overview of Robust Knowledge Transfer Mechanism

- **Multiple Source Tasks Setting:**
  - **New issue**: how to select transferable tasks from task set?
  - **Solution**: A novel task selection mechanism: "***Trust till Failure***"
    - For each state:
      - Maintain a feasible task set $\mathcal{M}_{s_h}$
      - Pick $\mathrm{M}_{\mathrm{Trust}} \in \mathcal{M}_{s_h}$ to trust until it is no longer feasible
      - When selecting the next task to trust:
        - Priorly select the feasible task recommending the same action

Experiments

- **Setting**
  - Toy tabular MDP example;
  - 5 source tasks at most;
  - Different tasks created by permuting transition matrix
- **Results**

Thank you!