FedHQL: Federated Heterogeneous Q-Learning

Extended Abstract

Flint Xiaofeng Fan
National University of Singapore
xfxf@u.nus.edu

Yining Ma
National University of Singapore
yiningma@u.nus.edu

Zhongxiang Dai
National University of Singapore
daizhongxiang@comp.nus.edu.sg

Cheston Tan
I2R, A*STAR
cheston-tan@i2r.a-star.edu.sg

Bryan Kian Hsiang Low
National University of Singapore
lowkh@comp.nus.edu.sg

ABSTRACT

This study introduces the problem setting of Federated Reinforcement Learning with Heterogeneous And bLack-box agEnts (FedRL-HALE), in which multiple RL agents with varying policy parameterizations, training configurations, and exploration strategies work together to optimize their policies through the proposed Federated Heterogeneous Q-Learning (FedHQL) algorithm. Empirical results demonstrate the effectiveness of FedHQL in improving system performance and increasing the sample efficiency of individual agents with high confidence.

KEYWORDS
Federated learning; Q-learning; Federated reinforcement learning

1 INTRODUCTION

Leveraging on the growing literature of federated learning (FL) [7, 9, 12, etc.], federated reinforcement learning (FedRL) [18] has emerged as a promising approach to improve the sample efficiency of RL agents in real-world environments. FedRL achieves collective intelligence [16] from distributed agents without requiring access to the raw trajectories of agent-environment interactions.

Despite their promising theoretical results [2, 3, 6, 8] and practical applications [5, 10, 11, 13, 15, 17, etc.], current FedRL algorithms have a limitation that they presume that all participants are homogeneous. This means that all agents must have the same policy parameterization (e.g., the architecture of the policy neural network, including the number of layers, the activation function, etc.) and the same training configurations for the policy (e.g., the learning rate). Such an assumption can be a significant limitation in real-world applications where agents are often heterogeneous, due to various disagreements such as computational budgets, assessments of the task’s difficulty, etc.

2 PROBLEM FORMULATION

Consider the task of federatively solving a sequential decision-making problem represented by the Markov Decision Process $M \triangleq \{S, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho, T\}$ [14]. Let the set $B \triangleq \{Q_n(a|s; D_n, \omega_n)\}_{n=1}^N$ denote a group of $N$ distributed heterogeneous and black-box agents. Each agent $B_n$ independently operates in a separate copy of the underlying MDP $M$ following its policy $\pi_n$, and generates its private experience data $D_n \triangleq \{(s, a, s', r)\}_{i\in\mathcal{D}_n}$. Each action valuator $Q_n(a|s; D_n, \omega_n)$, which we will use $Q_n(s, a)$ to denote, consists of a non-linear function $f_n(s; D_n, \omega_n)$ which predicts the value of action $a$ given a state $s$. The non-linear function $f_n(s; D_n, \omega_n)$ is parameterized by a neural network with parameters $\omega_n$ and learned using the private experience data $D_n$.

Due to the heterogeneity among agents, different agents may choose different neural network architectures and employ different optimization methods to train their networks. To facilitate knowledge aggregation, we let the central server broadcast query state(s)

![Figure 1: Graphical illustration of FedHQL.](image-url)
3 FEDHQL

Here we discuss the key components of FedHQL illustrated in Fig. 1.

**Federated Q-learning.** At the core of FedHQL is the federated version of Q-learning with \( N \) heterogeneous and black-box agents. Each agent \( B_n \) independently interacts with its own copy of the MDP using its preferred intra-agent exploration strategy. Each agent \( B_n \) updates its current estimation of action values \( Q_n(s, a) \) through Q-learning as follows:

\[
Q_n(s, a) \leftarrow Q_n(s, a) + \alpha_n [R(s_t+1, a_t+1) + \gamma \max_a Q_n(s_{t+1}, a) - Q_n(s, a)],
\]

where \( \alpha_n \) is the learning rate, \( \gamma \) is the discount factor, and \( R(s_t+1, a_t+1) \) is the reward obtained at the next state.

**Federated Upper Confidence Bound (FedUCB).** FedUCB begins with the following corollary:

**Corollary 3.1 (FedUCB).** Under the same assumptions and notations defined in Theorem 4.2 in [4], for any \( c > 0 \), with probability at least \( 1 - 3e^{-c} \), we have

\[
\mu_s, a \leq Q^*(s, a) \leq Q^{UCB}(s, a) \leq Q(s, a) + \frac{2\sqrt{V_{s, a}}}{N} + \frac{3c}{N}.
\]

Corollary 3.1 suggests that the optimal value of action \( a \) at state \( s \), \( Q^*(s, a) \), is upper-bounded by \( Q^{UCB}(s, a) \) defined above with high confidence. Inspired by Corollary 3.1, we develop our practical FedUCB algorithm for the knowledge aggregation in FedRL-HALE, which firstly calculates (for any \( s, a \)):

\[
\hat{Q}(s, a) = \frac{1}{N} \sum_{n=1}^{N} Q_n(s, a),
\]

\[
Q^{std}(s, a)^2 = \frac{1}{N} \sum_{n=1}^{N} (Q_n(s, a) - \hat{Q}(s, a))^2,
\]

\[
Q^{UCB}(s, a) = \hat{Q}(s, a) + \lambda \frac{Q^{std}(s, a)}{\text{exploration}} + \frac{\lambda}{\text{exploitation}}.
\]

where the degree of exploration is controlled by the parameter \( \lambda \), which we will refer to as inter-agent exploration coefficient, such that a larger \( \lambda \) encourages the selection of more exploratory actions.

**Federated Temporal Difference (FedTD).** With the FedUCB derived above, the server is able to optimistically select an action that leads to high rewards with high probability. Inspired by Fan et al. [3], we let the server operate in another separate copy of the underlying MDP and execute the selected action \( a \), hence generating a new sample \((s_t, a_t, s_{t+1}, r_t)\). This new sample will then be used to perform a federated version of Temporal Difference (FedTD) learning:

\[
\hat{Q}(s_t, a_t) \leftarrow \hat{Q}(s_t, a_t) + \alpha_n (r_t + \gamma \max_b \hat{Q}(s_{t+1}, b) - \hat{Q}(s_t, a_t)),
\]

where \( \hat{Q}(s_t, a_t) \) is the current estimation of action value, \( \alpha_n \) is the learning rate, \( \gamma \) is the discount factor, and \( r_t \) is the reward obtained at the next state.

**Individual Improvement.** After the FedTD target \( \hat{Q}(s_t, a_t) \) is updated, we let the server broadcast the updated \( \hat{Q}(s_t, a_t) \) back to all agents. An agent \( B_n \) will then update its own action value estimation \( \hat{Q}(s_t, a_t) \) using the following regression loss:

\[
L_n = ||\hat{Q}(s_t, a_t) - Q_n(s_t, a_t)||^2,
\]

where \( \hat{Q}(s_t, a_t) \) is the updated target value, \( Q_n(s_t, a_t) \) is the current action value, \( \alpha_n \) is the learning rate, and \( \gamma_n \) is a step-size hyper-parameter. This loss essentially helps the agent to improve its knowledge about action \( a_t \) at state \( s_t \) using the knowledge aggregated by FedUCB and updated by FedTD.

**4 EMPIRICAL EVALUATION**

We investigate the efficacy of FedHQL in improving the overall system performance with 5 heterogeneous agents depicted in Tab. 1. Given the fixed budget of each agent, we examine the average performance of agents versus the average consumption of the budget per agent. The results in both tasks are plotted in Fig. 2. The figures show that FedHQL with different choices of inter-agent exploration coefficients, FedHQL (\( \lambda = 0, 1, 3, 5, 10 \)), significantly improves the average performance per agent over that of independent self-learning, DQN (w.o. Fed). For example, in the LunarLander task, an agent is expected to consume at least 40% of its budget (i.e., total 1.6m = 4 × 10^6 × 0.4 interactions) on average to receive positive returns while an agent in FedHQL (\( \lambda = 1 \)) can achieve a performance close to 100 using only about 20% of its budget (i.e., total 0.8m = 4 × 10^6 × 0.2 interactions). More experimental results and analysis can be found in [4].
REFERENCES


