Multi-task Actor-Critic with Knowledge Transfer via a Shared Critic

Gengzhi Zhang GENGZHIZHANG@QQ.COM
ChongQing University

Liang Feng[*] LIANGF@CQU.EDU.CN
ChongQing University

Yaqing Hou HOUYQ@DLUT.EDU.CN
Dalian University of Technology

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

Multi-task actor-critic is a learning paradigm proposed in the literature to improve the learning efficiency of multiple actor-critics by sharing the learned policies across tasks while the reinforcement learning progresses online. However, existing multi-task actor-critic algorithms can only handle reinforcement learning tasks within the same problem domain, they may fail in cases where tasks possessing diverse state-action spaces. Taking this cue, in this paper, we embark a study on multi-task actor-critic with knowledge transfer via a share critic to enable the multi-task learning of actor-critic in heterogeneous state-action environments. Further, for efficient learning of the proposed multi-task actor-critic, a new formula for calculating the gradient of the actor network is also presented. To evaluate the performance of our approach, comprehensive empirical studies on continuous robotic tasks with different numbers of links. The experimental results confirmed the effectiveness of the proposed multi-task actor-critic algorithm.

Keywords: Multi-task actor-critic, Cross-domain transfer, Transfer reinforcement learning, Reinforcement learning

1. Introduction

Reinforcement learning (RL) models how agents learn from environment through trial-and-error, which has attracted much attention in both academia and industry in the last decades Sutton (1988); Watkins and Dayan (1992); Sutton and Barto (2018); Li (2017). In particular, RL is a learning paradigm which has been defined as a Markov decision process (MDP) that learns by interacting with an environment via a sense, act, and learn cycle. During the learning process, an RL agent receives a numerical reward which is the feedback from the environment to evaluate the performance of the agent. The RL agent seeks better actions autonomously with the aim of maximizing the accumulated reward over time. In recent years, RL has been successfully applied in many complex real-world applications, such as robotics Schulman et al. (2015); Kober et al. (2013), games Mnih et al. (2013), and biology Khamassi et al. (2005).

[*] Corresponding author

Actor-critic method is one of the most popular reinforcement learning algorithms investigated in the literature Barto et al. (1983); Sutton (1985), which possesses separate structures for policy learning and value function approximation explicitly. The policy structure is known as the actor, which is used to select actions, while the estimated value function is known as the critic that criticizes the actions made by the actor. Due to these structures, actor-critic is very efficient in action selection, and is able to learn an explicitly stochastic policy. It also makes it easy to impose domain-specific constraints on the set of allowed policies. Over the years, many actor-critic algorithms have been developed in the literature, such as Advantage Actor Critic (A2C), Asynchronous Advantage Actor Critic (A3C) and Soft Actor-critic Mnih et al. (2016); Haarnoja et al. (2018).

Despite the success enjoyed by the actor-critic approach, it is worth noting that actor-critic possesses a high sample complexity. As on-policy learning methods, actor-critic algorithms require collections of new samples for each gradient step. It thus becomes extravagantly expensive to learn an effective policy via actor-critic in cases that the task complexity increases. To improve the efficiency of actor-critic algorithms, in the literature, multi-task actor-critic learning has been proposed to leverage knowledge across related RL tasks to accelerate the learning process. For instance, Macua et al. (2017) presented a distributed multi-task actor-critic algorithm, in which multiple agents collect data independently and transfer the learned policy between neighbors to accelerate the convergence to a common policy. Teh et al. (2017) proposed a joint training framework called Distral, for multi-task actor-critic. It shares a distilled policy between tasks to make the learning process stable and efficient. However, as these algorithms assume all the RL tasks share common state and action spaces, they may fail when different tasks have diverse state-action space and environment dynamics. To the best of our knowledge, there is little work focusing on multi-task actor-critic learning with tasks involving heterogeneous state-action spaces. In particular, Dewangan et al. (2018) proposed to learn a compound policy network of multiple tasks sharing a set of actions. Nevertheless, if the tasks are not related and have different policy distributions, the gradient from different tasks will interfere negatively, which makes the learning process unstable and even less data efficient.

Keeping the above in mind, in this paper, we propose a new multi-task actor-critic algorithm by considering tasks with different state-action spaces. The proposed algorithm allows multiple tasks to be learned simultaneously and contains knowledge transfer between tasks to accelerate the learning process of actor-critic. In contrast to existing approach which learns a compound policy, each task in the proposed algorithm is solved by independent actor-critic network with the aim of avoiding negative interference between different tasks. Further, a centralized critic, namely shared critic, is proposed to extract common features from all tasks to guide each independent actor-critic network to explore the policy space. To achieve the effective exploration with the guidance of shared critic, we present a new formula for the gradient step in actor network to enable agent to learn policy from the transferred knowledge. In this way, knowledge transfer can be conducted between tasks with different state-action spaces towards enhanced learning performance. The shared critic can not only help agents overcome random exploration, but also prevent the search of policy from trapping in a local optimum.

To demonstrate the efficacy of the proposed algorithm, comprehensive empirical studies on continuous robotic control tasks built on MuJoCo platform[1] have been conducted. The tasks differ in terms of state space, action space, and environmental dynamics. The obtained experimental results indicate that our method can accelerate the learning process with improved RL performance over recently proposed multi-task actor-critic approaches and single-task actor-critic. The rest of this paper is organized as follows. Section 1.1 and 1.2 discuss the related works and actor-critic algorithm. Section 2 present the details of our proposed method. Section 3 contains the introduction of tasks, experimental discussions, and results. Section 4 concludes this paper.

1.1. Related Work

In the literature, there are generally two ways of conducting multi-task actor-critic reinforcement learning to reduce the sample complexity and accelerate convergence. The first one is to learn multiple tasks simultaneously and transfer the learned policies across tasks while the learning progresses online. For instance, besides the works introduced in section 1, Yang et al. (2017) proposed a multi-actor and single-critic architecture to enable the robot to learn multiple skills. It reduces the number of parameters by sharing visual features and achieved stable learning with superior performance via multi-task training. On the other hand, the other approach is continual learning where the agent is trained on multiple tasks sequentially with knowledge transfer over tasks. For instance, Ammar et al. (2014) presented a multi-task policy gradient method based on deep deterministic policy gradient (DDPG). In this work, the RL tasks are solved consecutively with transfer knowledge over tasks to improve the sample efficiency and learning speed of actor-critic. In these works, as aforementioned, the tasks considered in the multi-task learning scenario share common state-action spaces.

In order to enable multi-task actor-critic with tasks having different state-action spaces and environment dynamics, besides the learning of compound policy in Dewangan et al. (2018), Ammar et al. (2015) proposed a lifelong policy gradient algorithm which allows cross-domain transfer between consecutive RL tasks. In this work, the authors proposed to learn a repository of shared knowledge and project the shared knowledge to the task-specific domains through project matrix. However, as the agent has to learn tasks consecutively over time in a lifelong learning scenario, it is computationally expensive compared to the scenario of learning multiple tasks simultaneously.

Moreover, another research topic related to the algorithm proposed in this paper is multi-agent reinforcement learning (MARL) with centralized training and decentralized executing. In MARL, actor-critic is widely considered as the RL agent, since the decoupled architecture of actor-critic is easy to be expanded to multiple policies and centralized critic networks. For instance, Lowe et al. (2017) and Foerster et al. (2017) proposed actor-critic based methods to learn policies that require multi-agent coordination or competition. The centralized critic contains the information of multiple agents, while actors execute actions only using local information. However, there are fundamental differences between these methods and our proposed multi-task actor-critic algorithm. First of all, the centralized critic in MARL is to estimate the value function of global state which integrates the states

[1] https://www.roboti.us/

of all agents, while ours aim is to extract the common feature of value function across multiple tasks. Next, the purpose of knowledge transfer in MARL is to recognize changes of environment and identify the impact of each agent to the environment, while in the proposed algorithm, the transfer of knowledge across tasks is to enhance the efficiency and stability of the RL process.

1.2. Background

In the literature, RL has been modeled by a Markov decision process (MDP), which can be represented by the tuple $< X, U, P, \rho >$, where X denotes the state space, U gives the action space, P is state transition function $P : X \times U \to X$, and ρ is the reward function $\rho : X \times U \times X \to R$. For each time step, an RL agent executes an action $a \in U$, selected according to the policy π, which thus induces a transition in the environment based on the state transition function $P(s'|s,a) \to [0,1]$, $s', s \in X$. After the transition, the RL agent receives feedback r from the environment, which is given by the reward function $r = \rho(s,a)$.

Generally, the goal of policy-based RL approaches is to find the optimal policy π, which map the action for each state directly, with the aim of maximizing a discounted sum of future rewards $G_t = \sum_{i=t}^{+\infty} \gamma^{i-t} r_i$, where $\gamma \in (0,1)$ is a discounting factor. The policy $\pi_\theta(a|s)$ with parameter θ is updated using the gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}[G_t^\pi \nabla_\theta log_\theta \pi_t(a_t|s_t)] \tag{1}$$

The cumulative reward function can be replaced by V function $V^\pi(s) = \mathbb{E}[r_t^\gamma | S_t = s; \pi]$, which denotes the estimation of cumulative discounted future reward, and the Q function $Q^\pi(s,a) = \mathbb{E}[r_t^\gamma | S_t = s, A_t = a; \pi]$, which is the discounted cumulative reward from state-action pair (s,a). However, such functions may lead to high variance in the estimate of the gradient, which thus result in slow learning process.

To reduce variance of gradient estimation, actor-critic methods jointly learn policy network (actor) with parameter θ and value network (critic) with parameter η. For policy network, the advantage function $A(s_t, a_t) = Q(s_t, a_t) - V_\eta(s_t)$ is often considered, which measures the success of current action against the average of actions that would have been taken at the current state. Further, the temporal difference (TD) error δ_η can be employed as an unbiased estimate of the advantage function to reduce the number of required networks. In this way, the policy gradient with the TD error becomes:

$$\nabla_\theta J(\theta) = \mathbb{E}[log_\theta \pi_\theta(a|s)\delta_\eta] \tag{2}$$

$$\delta_\eta = r + \gamma V_\eta(s') - V_\eta(s) \tag{3}$$

Moreover, in this study, we adopt the n-step return Sutton and Barto (2018) to compute advantage function to adjust the tradeoff between variance brought by the immediate reward, and bias introduced by the value estimation. Therefore, δ_η is computed as:

$$\delta_\eta = r_{t+1} + \gamma r_{t+2} + ... + \gamma^{n-1} V_\eta(s_{t+n}) - V_\eta(s_t) \tag{4}$$

Next, the value or critic network is updated by minimizing the squared loss of TD error. The loss function of critic network is as follow:

$$\begin{gathered} L(\theta) = E_{(s,a,r,s') \sim \pi_\theta} (V_\eta(s) - y)^2 \\ \text{where } y = r_{t+1} + \gamma r_{t+2} + ... + \gamma^{n-1} V_\eta(s_{t+n}) \end{gathered} \tag{5}$$

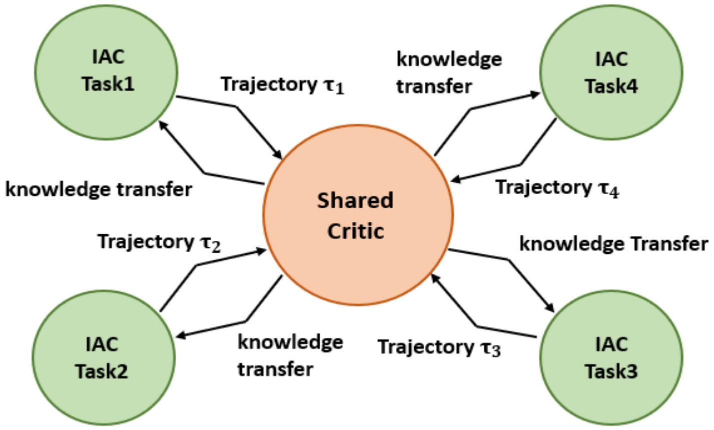


Figure 1: Overall architecture of the proposed multi-task actor-critic with a shared critic network

In this paper, this advantage actor-critic with n-step return is considered as the basic RL algorithm in our proposed multi-task actor-critic algorithm.

2. Proposed Method

In this section, we present the details of the proposed multi-task actor-critic with a shared critic network. In particular, as illustrated in Fig. 1, the overall architecture of the proposed algorithm contains independent actor-critic (IAC) agents performing self-learning on each RL simultaneously, and a centralized critic, i.e., shared critic, for knowledge sharing between different tasks. The proposed architecture is similar to the Distral framework Teh et al. (2017) discussed in section 1 which has a distilled policy for sharing knowledge between RL tasks. However, there are two significant differences between our proposed algorithm and Distral. First of all, the knowledge transfer in Distral is in the form of policy, while our proposed algorithm intends to transfer advantage values via the shared critic. Secondly, Distral focuses on the multi-task learning with tasks possessing common state-actions spaces. However, in this paper, we considers the more complex multi-task scenarios that different task contains heterogeneous state-action spaces.

In what follows, the details of the shared critic network and the independent actor-critic are presented.

2.1. Shared Critic Network

As depicted in Fig. 2, the proposed shared critic collect the trajectories from all tasks to learn a common value function distribution over all tasks and helps each agent to accelerate the learning by biasing the search direction of task-specific policies via sharing advantage value.

In particular, the shared critic network is parameterized by η_0. It collects trajectories $\tau_1, \tau_2, .., \tau_n$ sampled from the m task-specific actors in each task. There are two functions for

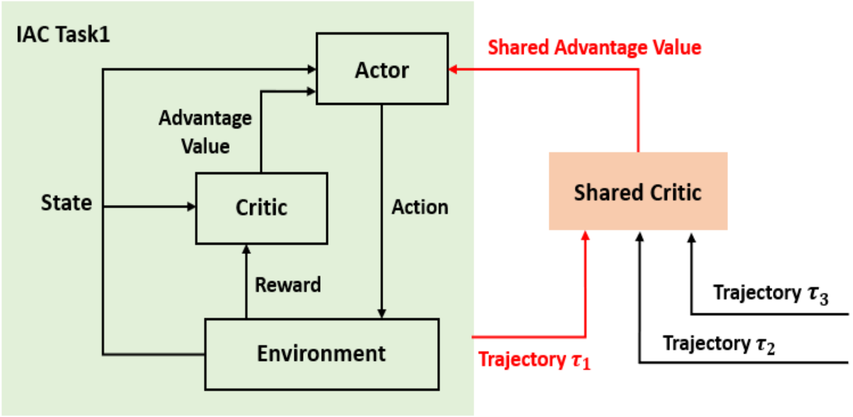


Figure 2: Knowledge transfer between the shared critic network and independent actor-critic.

estimating the critic, which are $V(s)$ and $Q(s, a)$. Here we consider V function $V_0(s)$ as the objective function of the shared critic network. This is because the action value of similar behaviors of the agent could very different, since the tasks may have different state-action spaces. Therefore, the objective function of the shared critic network is to minimize the square loss of V value and the discounted reward of all the tasks, which is given by:

$$L(\eta_0) = \sum_{i=1}^{N}\sum_{t=1}^{T}(V_{\eta_0}(s_{i,t}) - y_{i,t})^2 \tag{6}$$

where target $y_{i,t}$ is the cumulative discounted future reward of state s at the t-th time step in task i, which is given by:

$$\begin{aligned} y_{i,t} = r_{i,t+1} + \gamma r_{i,t+2} + ... \\ + \gamma^{n-1} V_{\eta_0}(s_{i,t+n}) \end{aligned} \tag{7}$$

The shared critic network V_{η_0} learns from all the state spaces of multiple tasks and estimate the value according to the data of multiple tasks. Note that the dimension of state in different tasks could be different, in this study, we pad state vectors with zero to make state vectors with a common dimension before shared critic network training. The padded state vector should remain structurally uniform. For instance, as illustrated in Fig. 3, the state of the robot arm includes the angle of each joint (2 dimension), the coordinates of the target point (2 dimension), and the distance from the fingertip to the target point (3 dimension). When a two-link arm and a three-link arm undergo RL concurrently using the proposed algorithm, the state of the two-link arm will be padded with two zeros in joint angle to make the dimension consistent with the state of the three-link arm.

Furthermore, the shared advantage value A_0 computed by shared critic will be used for knowledge transfer across different tasks, which is as follows:

$$\begin{aligned} A_0(s_{i,t}, a_{i,t}) = \delta_{\eta_0} = r_{i,t+1} + \gamma r_{i,t+2} + ... \\ + \gamma^{n-1} V_{\eta_0}(s_{i,t+n}) - V_{\eta_0}(s_{i,t}) \end{aligned} \tag{8}$$

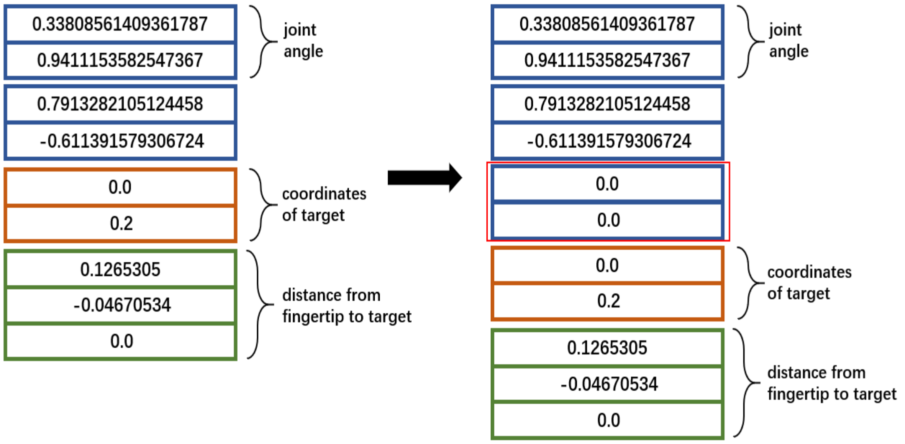


Figure 3: Transform the state vector of a two-link robot arm to be consistent with that of a three-link robot arm.

As discussed in section 1.2, the advantage value guides the search direction of policy optimization, since it measures how good it is to make the transition from the current state to the next. Therefore, the transfer of the shared advantage value is to bise the direction of policy optimization in each RL task via the common value distribution learned by the shared critic network on all the tasks.

2.2. Independent Actor-Critic

Due to the discrepancy of different tasks, independent actor-critic (IAC) agent is considered to learn each tasks simultaneously. Moreover, in this study, each IAC possesses independent parameters, and do not share a common set of parameter as in MARL Foerster et al. (2017).

Next, as mentioned in section 1.2, the advantage actor-critic with n-step return (A2C) is considered as the IAC agent in this paper. Further, as illustrated in Fig. 2, we propose to calculate the advantage value of the IAC by considering both the advantage value $A_i(s_{i,t}, a_{i,t})$ computed by its own critic network and the advantage value $A_0(s_{i,t}, a_{i,t})$ obtained by shared critic network, where i, t represent the state and the action for task i at t-th time step. The proposed formula is given by:

$$\begin{aligned} A_i^{total}(s_{i,t}, a_{i,t}) =& (1-\alpha) A_i(s_{i,t}, a_{i,t}) \\ &+ \alpha A_0(s_{i,t}, a_{i,t}) \end{aligned} \tag{9}$$

$$\begin{aligned} A_i(s_{i,t}, a_{i,t}) = \delta_{\eta_i} &= r_{i,t+1} + \gamma r_{i,t+2} + ... \\ &+ \gamma^{n-1} V_{\eta_i}(s_{i,t+n}) - V_{\eta_i}(s_{i,t}) \end{aligned} \tag{10}$$

where α is a scalar to balance the tradeoff between self-learning and knowledge transferred from the shared critic[2].

[2] In this study, α is set as 0.5 in all the experiments.

Algorithm 1 Multi-task actor-critic with a shared critic

Input: State s: Reward r

Parameter: Task number $i = 1, 2...M$; Number of Episode E; Maximum steps of task per episode T; Transfer weight α;

Output: Critic $\eta_{i,E}$ actor $\theta_{i,E}$;

```
Randomly initialise actor $\pi_{\theta_i}$ and critic network $V_{\eta_{ni}}$ for task $i$, $i = 1, 2...m$;
Initialize episode counter $e$, $e = 0$
while $e_i < E_i$ do
  for each task $i$ do
    Set step counter $t = 0$
    while $t < T_i$ and not terminal state do
      Select action $a_{i,t} \sim \pi_{\theta_i}$.
      Execute $a_{i,t}$ and state $r_{i,t+1}$ and $s_{i,t+1}$
      Store tuple $(s_{i,t}, a_{i,t}, r_{i,t+1}, s_{i,t+1})$
      Update step counter: $t \leftarrow t + 1$
    end while
    for each sample do
      Compute advantage value using Eq.10
      Compute shared advantage value using Eq.8
    end for
    Compute critic gradient using Eq.12
    Compute actor gradient using Eq.11
    Compute shared critic gradient using Eq.6
  end for
  Update episode counter $e \leftarrow e + 1$
end while
return Critic and actor weights: $\theta_{i,E}, \eta_{i,E}$;
```

Subsequently, with the obtained advantage value, actor network parameterized by θ_i is updated by:

$$\triangle\theta_i \propto A_i^{total}(s_{i,t}, a_{i,t})\nabla_{\theta_i} log(\pi_i(a_{i,t}|s_{i,t})) \tag{11}$$

The optimization of the critic network V_{η_i} is proceeded as routine, which is updated by:

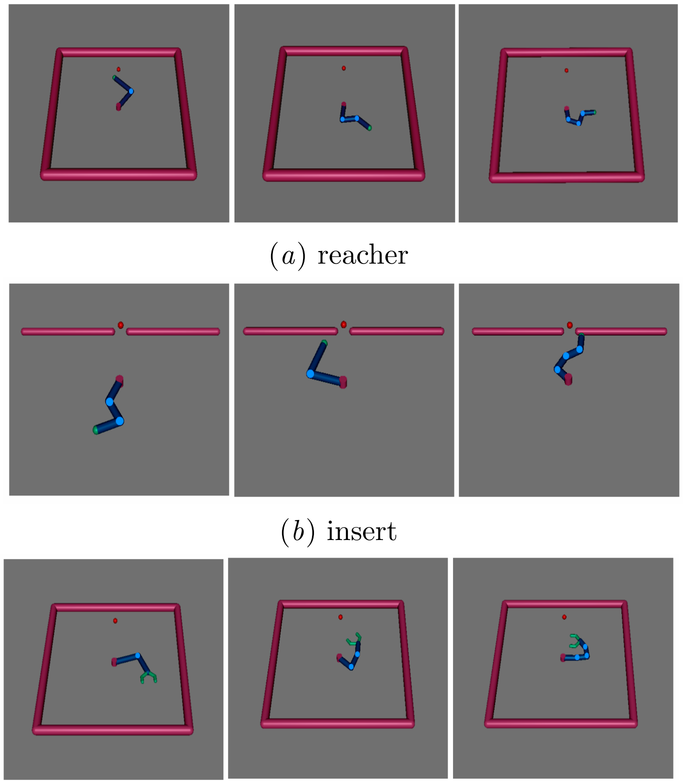
$$L(\eta_i) = \sum_{t=1}^{T}(V_{\eta_i}(s_{i,t}) - y_{i,t})^2 \tag{12}$$

where target $y_{i,t}$ is the discounted reward, which has been given in Eq.7.

Lastly, the pseudo code of the proposed multi-task actor-critic with a share critic network is summarized in Algorithm 1.

3. Experiments

In this section, comprehensive empirical studies are conducted to evaluate the performance of the proposed multi-task actor-critic. The well-known robot tasks with different numbers



(a) reacher

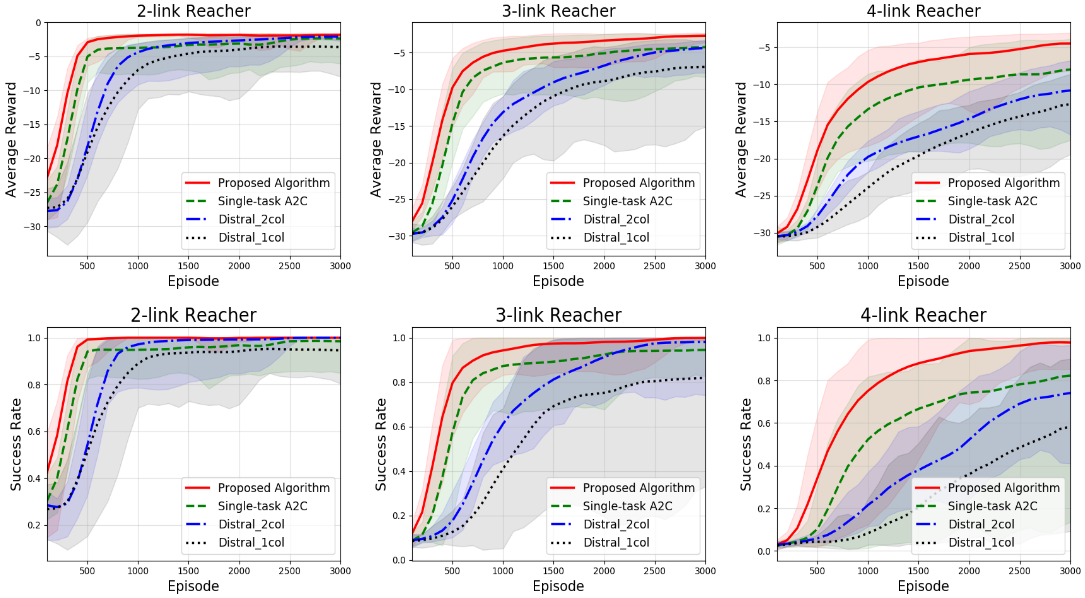
(b) insert

(c) grasp

Figure 4: Three sets of tasks

of links built on the MuJoCo physics simulator is considered here to construct the heterogeneous mutli-task RL scenarios. In particular, according to Gupta et al. (2017), three sets of tasks which are reacher, insert and grasp, are investigated in this study. As shown in Fig. 4, in each task, there are three types of robots/agents, i.e. 2 links, 3 links and 4 links. The state of a robot includes joint angles, joint position and goal position. The action is a set of driving forces, which change the angular velocities of the robot arm. Therefore, the dimension of state and action are different across robot arms with different links. For each set of tasks, the target positions are the same. The position of robot arm is randomly generated at the beginning of a learning episode. A negative reward is given to the agent which is proportional to the distance between the end effector and the target during the RL process. When an agent successfully completes the given task, it will receive a positive reward 1.

Subsequently, for comparison, the single task actor-critic without knowledge transfer is considered as the baseline algorithm. Moreover, as mentioned in section 2, since the proposed algorithm has similar architecture with Distral proposed in Teh et al. (2017), two instantiations of the Distral, labeled as Distral_1col and Distral_2col are also compared in this study. Distral_1col and Distral_2col have the same way to transfer policy, but have different forms in the representation of policies. In Distral_1col, the policy is parameterized by policy network with the structure of one column. In Distral_2col, the policy is parameterized



(a) 2-link Reacher (b) 3-link Reacher (c) 4-link Reacher

Figure 5: Reacher

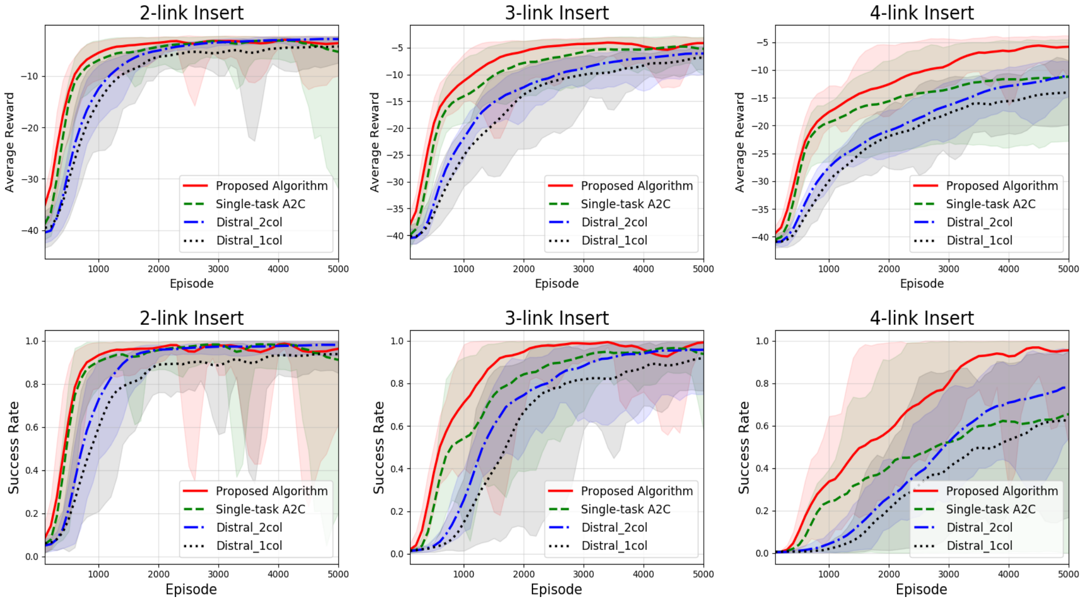
by two column of networks, of which one column is distilled policy, the other column adjusts the distilled policy to specializing to one task. Further, to enable the learning of policy in different state-action spaces in Distral, the operation of padding zeros (see Fig. 3) used in our proposed algorithm is applied here again to ensure a fair comparison. Lastly, in all the baseline algorithms, the advantage actor-critic with N-step returns (A2C) is employed as the base agent, and the policies and value functions are configured as 3 layer neural networks with 40 hidden units and RELU activation, that are trained by the standard back propagation using the ADAM optimizer Kingma and Ba (2014) with learning rate 0.001.

3.1. Reacher

The first investigation is based on the Reacher task. In this task, the agent aims to research a pre-defined target point which is set as the farthest distance that the agent can reach. Fig. 5 presents the convergence curves of averaged rewards and averaged success rates[3] obtained by all the compared algorithms over 20 independent runs. In the figure, the Y-axis gives the obtained values of the averaged reward or success rate, while the X-axis denotes the episode incurred by the corresponding algorithm so far.

As can be observed in the figure, the proposed multi-task actor-critic algorithm achieves the superior learning speed in contrast to both the single-task A2C and the multi-task Distral approaches, i.e., Distral_1col and Distral_2col, on all the three Reacher tasks. In particular, on the 2-link Reacher task, the proposed algorithm uses only 500 episodes to arrive the competitive success rate obtained by the Distral_2col in round 2000 episodes,

[3] Success rate denotes the ratio between the number of times that the agent completes the task successfully and the total number of times conducted on this task.



(a) 2-link insert

(b) 3-link insert

(c) 4-link insert

Figure 6: Insert

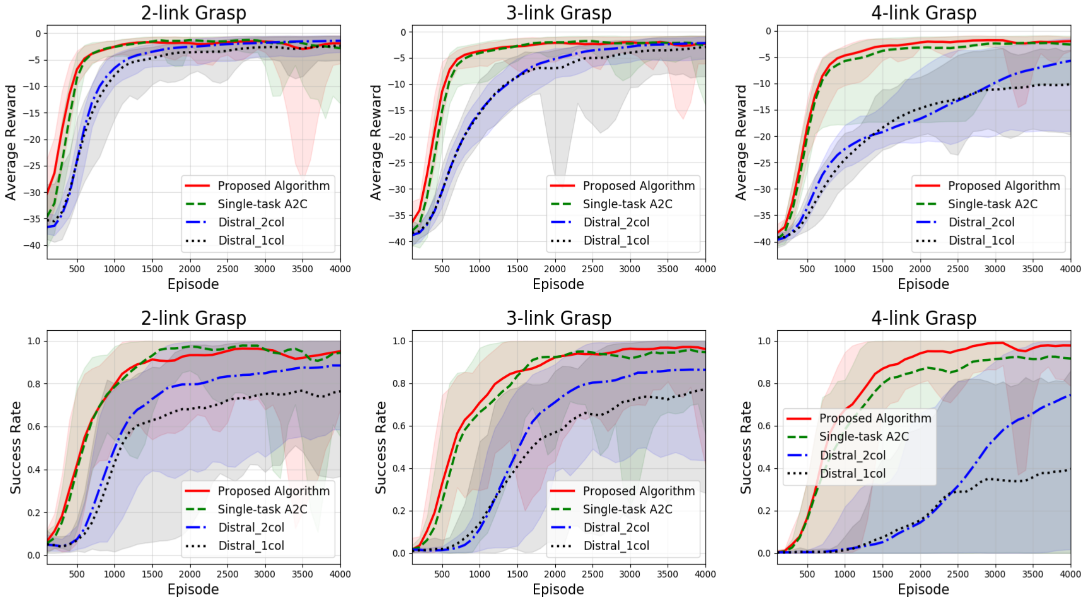
and single-task A2C and distral_1col did not learn the optimal policy in 3000 episodes. Further, with the number of links increases, the Reacher task becomes more complex since the volume of the sate-action space increases. Therefore, from 2-link Reacher task (Fig. 5(a)subfigure) to 4-link Reacher task (Fig. 5(c)subfigure), the success rates obtained by the single-task A2C decrease accordingly. However, with proper knowledge sharing across tasks, the proposed algorithm obtains consistent superior rewards and success rates, and brings increased speedups over the baseline algorithms from 2-link to the 4-link Reacher tasks.

Moreover, in Fig. 5, it is also observed that the multi-task Distral algorithms, i.e., Distral_1col and Distral_2col, obtain deteriorated performance in terms of both averaged reward and success rate even against the single-task A2C without knowledge transfer. This on one hand, shows that the policy based knowledge transfer across tasks may not be suitable for tasks with diverse state-action spaces. On the other hand, this again confirms the effectiveness of the proposed multi-task actor-critic with a shared critic across heterogeneous RL tasks.

3.2. Insert

Moreover, we investigate the performance of our proposed algorithm on Insert task. In this task, the robot arm has to go through a narrow opening to reach the predefined target. In contrast to the Reacher task, this task contains more actions, and requires sophisticated policies to complete the predefined goals.

Fig. 6 presents the averaged rewards and averaged success rates obtained by all the compared algorithms on the 2-link, 3-link, and 4-link Insert task over 5000 episodes. As



(a) 2-link Grasp (b) 3-link Grasp (c) 4-link Grasp

Figure 7: Grasp

can be observed, the proposed algorithm converges faster than single-task A2C, Distral_2col and Distral_1col in terms of average reward and success rate in all three Insert tasks. When the number of links increases, the tasks become more complex and the improvements of performance brought by our proposed algorithm becomes more significant. In 4-link Insert task,while single-task A2C obtains 60 percent success rate, our proposed method arrives the success rate of 95 percent.

For multi-task Distral approaches, Distral_1col always has negative effect on the learning of tasks, and Distral_2col always disturbs the learning in the early stage and causes the decrease of learning speed. The comparison results between multi-task Distral approaches and the proposed algorithm demonstrate that our proposed algorithm can effectively avoid the interference between tasks with diverse state-action spaces.

3.3. Grasp

In the Grasp task, the robot arm is required to grasp objects with an appropriate angle. Therefore, in these Grasp tasks, knowledge learned in one scenario is often task-specific. In contrast to Reacher task and Insert task, knowledge transfer in this task may not be helpful and even lead to negative transfer easily.

As can be observed, fig 7 shows that knowledge transferred by Distral_2col and Distral_1col have a tremendously negative effects on the performance of task, while our proposed algorithm avoids negative transfer and obtains similar results compared to single-task A2C. In 4-link Grasp task, the proposed algorithm obtains better final success rate than the single-task A2C. This experiment further confirms the efficacy of the proposed algorithm for heterogeneous RL tasks.

4. Conclusion

In this paper, we have proposed a new multi-task actor-critic algorithm to enable knowledge transfer across tasks possessing heterogeneous state-action spaces. In particular, a centralized or shared critic network has been proposed to learn a common value distribution over all tasks, which is then used to guide the learning of independent actor-critic networks. To evaluate the performance of the proposed algorithm, comprehensive empirical studies have been conducted using the continuous robotic tasks, i.e., Reacher, Insert, and Grasp, over both single-task actor-critic and existing multi-task actor-critic algorithms. The obtained results confirms the efficacy of the proposed algorithm for multi-task actor critic.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61876025.

References

Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, and Matthew Taylor. Online multi-task learning for policy gradient methods. In *International conference on machine learning*, pages 1206–1214, 2014.

Haitham Bou Ammar, Eric Eaton, José Marcio Luna, and Paul Ruvolo. Autonomous cross-domain knowledge transfer in lifelong policy gradient reinforcement learning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

Parijat Dewangan, S Phaniteja, K Madhava Krishna, Abhishek Sarkar, and Balaraman Ravindran. Digrad: Multi-task reinforcement learning with shared actions. *arXiv preprint arXiv:1802.10463*, 2018.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926*, 2017.

Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Mehdi Khamassi, Loïc Lachèze, Benoît Girard, Alain Berthoz, and Agnès Guillot. Actor–critic models of reinforcement learning in the basal ganglia: from natural to artificial rats. *Adaptive Behavior*, 13(2):131–148, 2005.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pages 6379–6390, 2017.

Sergio Valcarcel Macua, Aleksi Tukiainen, Daniel García-Ocaña Hernández, David Baldazo, Enrique Munoz de Cote, and Santiago Zazo. Diff-dac: Distributed actor-critic for average multitask deep reinforcement learning. *arXiv preprint arXiv:1710.10363*, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

Richard S Sutton. Temporal credit assignment in reinforcement learning. 1985.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4496–4506, 2017.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Zhaoyang Yang, Kathryn E Merrick, Hussein A Abbass, and Lianwen Jin. Multi-task deep reinforcement learning for continuous action control. In *IJCAI*, pages 3301–3307, 2017.