基于SAC的多智能体深度强化学习算法

肖 硕[1,2],黄珍珍[2],张国鹏[2],杨树松[3],江海峰[2],李天旭[2]
(1. 矿山数字化教育部工程研究中心,江苏徐州 221000;2. 中国矿业大学计算机科学与技术学院,江苏徐州 221000;
3. 宁波市轨道交通集团有限公司,浙江宁波 315000)

摘 要: 由于多智能体所处环境动态变化,并且单个智能体的决策也会影响其他智能体,这使得单智能体深度强化学习算法难以在多智能体环境中保持稳定.为了适应多智能体环境,本文利用集中训练和分散执行框架Cen-tralized Training with Decentralized Execution(CTDE),对单智能体深度强化学习算法Soft Actor-Critic(SAC)进行了改进,引入智能体通信机制,构建Multi-Agent Soft Actor-Critic(MASAC)算法.MASAC中智能体共享观察信息和历史经验,有效减少了环境不稳定性对算法造成的影响.最后,本文在协同以及协同竞争混合的任务中,对MASAC算法性能进行了实验分析,结果表明MASAC相对于SAC在多智能体环境中具有更好的稳定性.

关键词: 多智能体环境;集中训练;分散执行;多智能体深度强化学习
中图分类号: TP391 **文献标识码:** A **文章编号:** 0372-2112(2021)09-1675-07
电子学报URL: http://www.ejournal.org.cn DOI: 10.12263/DZXB.20200243

Deep Reinforcement Learning Algorithm of Multi-agent Based on SAC

XIAO Shuo[1,2], HUANG Zhen-zhen[2], ZHANG Guo-peng[2], YANG Shu-song[3], JIANG Hai-fei[2], LI Tian-xu[2]
(1. *Engineering Research Center of Mine Digitalization, Ministry of Education, Xuzhou, Jiangsu* 221000, *China*;
2. *School of Computer Sciences and Technology, China University of Mining & Technology, Xuzhou, Jiangsu* 221000, *China*;
3. *Operating Branch, Ningbo Rail Transit Group Co., LTD., Ningbo, Zhejiang* 315000, *China*)

Abstract: Due to the dynamic change of multi-agent environment, and the decision of single agent will affect other agents, it is difficult for the deep reinforcement learning algorithm of single agent to maintain stability in multi-agent environment. In order to adapt to multi-agent environment, this paper uses centralized training and decentralized execution framework (CTDE) to improve single agent deep reinforcement learning algorithm soft actor-critic (SAC). By introducing agent communication mechanism, in multi-agent soft actor-critic (MASAC), agents share observation information and historical experience, which effectively reduces the impact of environmental instability on the algorithm. Finally, in the task of cooperation and cooperation and competition, the performance of MASAC algorithm is analyzed experimentally. The results show that MASAC has better stability than SAC in multi-agent environment.

Key words: multi-agent environments; centralized training; decentralized execution; multi-agent deep reinforcement learning

1 引言

强化学习和深度学习相结合,形成的深度强化学习方法,近些年得到了快速的发展和应用.深度强化学习受到广泛关注源于DeepMind开发的围棋博弈人工智能AlphaGO[1]以较大优势战胜了人类的顶尖围棋选手,而在随后不到两年的时间里,升级后的AlphaGo Zero[2]就以100:0的战绩击败了AlphaGO.深度强化学习可以应用于众多领域,在视频图像处理领域,深度强化学习在物体检测[3]、视频对话[4]、图像视频[5,6]获取等方面取得了长足的发展.在机器人控制领域,将深度强化学习和机器人相结合[7],实现了机器人从视觉输入到动作输出的学习.

深度强化学习在单智能体领域的成功应用,也推动了其在多智能体环境中的发展.深度强化学习利用端到端的学习方法,可以帮助智能体在多智能体环境中,进行有效的决策.但是对于多智能体环境,如果直

收稿日期:2020-03-09;修回日期:2020-05-14;责任编辑:梅志强
基金项目:国家自然科学基金(No.62071470,No.U1934219,No.61971421);徐州市科技计划项目(No.KC19011,No.KC20167)

接将单智能体深度强化学习算法应用到每个智能体的决策，会使得智能体始终处于不稳定的环境中[8,9]. 所有智能体在环境中不断交互，导致环境对于每个智能体而言都是不断变化的. 为了解决这个问题，研究中通常利用集中训练和分散执行框架(CTDE)，将单智能体深度强化学习算法扩展到多智能体环境中. 文献[10]提出COMA算法，将actor-critic算法扩展到多智能体环境中. COMA使用集中的所有智能体共享的critic网络，在训练时critic网络的输入为所有智能体的行为信息和环境中可以获得的所有状态信息，而在执行时，actor网络只需要利用自身观察即可做出决策. Lowe等人通过每个智能体学习一个单独的critic网络解决了复杂混合环境中，智能体的学习问题，提出了MADDPG算法[11]. MADDPG基于DDPG算法，使用确定性策略函数作为actor网络. 独立的critic网络使得MADDPG在合作、竞争以及两者混合的环境中均有较好表现.

本文在深度强化学习算法Soft Actor-Critic(SAC)[12]的基础上展开研究，提出了多智能体深度强化学习算法MASAC. 不同于COMA，MASAC使用最大熵值的优化目标使得算法变得更加稳定. 它与COMA使用单一critic网络前向传播所有智能体动作的Q值有所区别，MASAC中每个智能体使用独立的critic网络计算所有状态和动作的Q值，使得算法可以应用在协同和竞争混合的环境中. MADDPG没有考虑智能体模型之间的通信，研究中MASAC使用独立的通信设备实现模型之间的通信，构建一种在CTDE基础上的通信模型. 同时，每个智能体可以使用GRU原理学习在通信设备中表示与环境交互的历史经验，以便分享给其它智能体. MADDPG使用确定性策略，而本文使用随机性策略拟合actor网络，随机性策略更容易训练，并且具有更高的稳定性. 在协同或协同和竞争混合的环境中，MASAC均有较好的表现.

2 Soft Actor-Critic模型

SAC不同于传统以最大化智能体回报为目标的传统算法，SAC同时最大化智能体回报和动作的熵值. 在策略函数和价值函数中最大熵具有不同的作用. 在策略函数中，它可以防止智能体的策略过早收敛于局部最优解. 在价值函数中，增加智能体动作的熵[13]，可以鼓励智能体探索，使得算法更加稳定. 最大化熵值算法SAC的优化目标为，

$$J(\pi)=\sum_{t=0}^{T}\mathrm{E}_{(s_t,a_t)\sim\rho^{\pi}}\left[r\left(s_t,a_t\right)+\lambda\mathcal{H}\left(\pi\left(\cdot\,|s_t\right)\right)\right] \tag{1}$$

其中，E为数学期望，T为智能体每轮与环境交互的总时间步数，s_t, a_t分别为t时刻智能体的状态和动作，ρ^{π}是策略π下轨迹$\left(s_t,a_t\right)$的分布，$r\left(s_t,a_t\right)$为智能体执行动作a_t获得的奖励. $\mathcal{H}(\cdot)$用于计算策略π的熵值，λ用于控制更关注熵还是奖励. SAC通过最小化KL散度更新actor网络，需要最小化如下目标：

$$J_{\pi}(\theta)=\mathrm{E}_{s_t\sim D,a_t\sim\pi_{\theta}}\left[\lambda\log\left(\pi_{\theta}\left(a_t|s_t\right)\right)-Q_{\beta}\left(s_t,a_t\right)\right] \tag{2}$$

s_t, a_t分别为t时刻智能体的状态和动作，θ为智能体actor网络π_{θ}的参数，β为critic网络Q_{β}的参数，D为存储训练样本的经验回放池. $J_{\pi}(\theta)$中的期望可以通过在经验回放池D中取出样本s_t进行近似计算. 为了提高算法的稳定性，$Q_{\beta}\left(s_t,a_t\right)$中$a_t$是经过对随机策略$\pi_{\theta}$进行采样得到的，因此整个过程对于参数$\theta$是不可微的，导致$\pi_{\theta}$的参数无法通过反向传播更新. 为了能够使用随机梯度方法最小化$J_{\pi}(\theta)$，可以利用重参数的方法将采样过程移出计算图. 当使用正态分布进行采样，通过输入服从某个分布的随机噪声ξ，原本的采样过程就变为：

$$\tilde{a}_{\theta}\left(s_t,\xi\right)=\tanh\left(\mu_{\theta}\left(s_t\right)+\sigma_{\theta}\left(s_t\right)\odot\xi\right) \tag{3}$$

$\tilde{a}_{\theta}\left(s_t,\xi\right)$为$\pi_{\theta}\left(\cdot|s_t\right)$得到的智能体动作概率分布中一个样本，$\mu_{\theta}$、$\sigma_{\theta}$分别为均值和标准差，tanh用于保证智能体动作的输出值被限制在一定范围. 经过重参数后，得到的优化目标为：

$$J_{\pi}(\theta)=\mathrm{E}\left[\lambda\log\left(\pi_{\theta}\left(\tilde{a}_{\theta}\left(s_t,\xi\right)|s_t\right)\right)-Q_{\beta}\left(s_t,\tilde{a}_{\theta}\left(s_t,\xi\right)\right)\right] \tag{4}$$

随机噪声ξ服从分布$\mathcal{N}$，根据最大熵原则，通过最小化贝尔曼误差$J_Q(\beta)$可以对critic网络进行更新：

$$J_Q(\beta)=\mathrm{E}_{s_t,a_t,r_t,s_{t+1}\sim D}\left[\frac{1}{2}\left(Q_{\beta}(s_t,a_t)-y\right)^2\right] \tag{5}$$

其中，

$$y=r(s_t,a_t)+A(s_{t+1},a_{t+1}) \tag{6}$$

$$A(s_{t+1},a_{t+1})=\gamma\mathrm{E}\left[Q_{\bar{\beta}}\left(s_{t+1},a_{t+1}\right)-\lambda\log\left(\pi_{\theta}\left(a_{t+1}|s_{t+1}\right)\right)\right] \tag{7}$$

$\left(s_t,a_t,r_t,s_{t+1}\right)$为从经验缓冲池$D$中取出的样本元组，动作$a_{t+1}$可以利用actor网络生成的概率分布得到，$a_{t+1}\sim\pi_{\theta}\left(\cdot\,|s_{t+1}\right)$，$\bar{\beta}$为目标critic网络$Q_{\bar{\beta}}$的参数.

3 基于SAC的MASAC网络模型设计

为了稳定环境，本文将单智能体深度强化学习算法SAC采用CTDE框架进行拓展，在训练中critic网络加入其它智能体的观察和动作作为额外信息，以保证环境对于每一个智能体都是稳定的，而在执行时，actor网络依然只需要使用智能体的私有观察进行决策.

假设环境中共有N个智能体，智能体的actor网络为$\pi=\left\{\pi_1,\cdots,\pi_N\right\}$，每个actor网络$\pi$的参数为$\theta=$

$\{\theta_1, \cdots, \theta_N\}$,使用$\pi_{\theta_i}$表示参数为$\theta_i$的actor网络$\pi_i$,同样,使用$Q^{\beta_i}$表示参数为$\beta_i$的critic网络$Q_i$. 那么根据CTDE框架和SAC,得到智能体$i$更新actor网络需要最小化目标$J_{\pi_i}(\theta_i)$:

$$J_{\pi_i}(\theta_i) = \mathrm{E}_{x\sim D, \tilde{a}\sim\pi_\theta}\left[\lambda \log\left(\pi_{\theta_i}(\tilde{a}_i|o_i)\right) - Q^{\beta_i}(x,\tilde{a})\right] \quad (8)$$

其中,经验回放池D用于存储智能体的状态x、行为a、奖励r和下一个状态x',使用四元组(x,a,r,x')作为D中的一条存储,其中,$x=(o_1,\cdots,o_N)$、$a=(a_1,\cdots,a_N)$、$r=(r_1,\cdots,r_N)$、$x'=(o_1',\cdots,o_N')$. 每个智能体使用一个独立的critic网络$Q^{\beta_i}(x,\tilde{a})$,输入为从经验回放池D中取样得到的所有智能体的观察x和动作$\tilde{a}=\{\tilde{a}_1,\ldots,\tilde{a}_N\}$,为了增加智能体训练的稳定性,使用当前智能体的策略生成$\tilde{a}_i, \tilde{a}_i\sim\pi_{\theta_i}(\cdot|o_i)$. 输出为智能体$i$动作$\tilde{a}_i$的$Q$值. 由于每个智能体分别具有一个$Q^{\beta_i}$,因此其可以根据环境的需要拥有不同形式的奖励函数,使其可以适应不同的环境,包括协同或协同和竞争两者相混合的环境. 分散执行的actor网络$\pi_{\theta_i}(\cdot|o_i)$仅需利用其自身的观察$o_i$,即可得到动作的概率分布,智能体通过对得到的概率分布采样获得具体执行的动作$\tilde{a}_i$. 同样利用重参数的采样方法可以得到该目标的可微形式:

$$J_{\pi_i}(\theta_i) = \mathrm{E}\left[\lambda \log\left(\pi_{\theta_i}(\tilde{a}_i|o_i)\right) - Q^{\beta_i}(x,\tilde{a})\right] \quad (9)$$

其中$x\sim D$,为了保证在进行采样后依然是可微的,设置$\tilde{a}_i=\tilde{a}_{\theta_i}(o_i,\xi), \xi\sim\mathcal{N}(0,1)$,对动作概率分布$\pi_{\theta_i}(\cdot|o_i)$进行采样操作. $J_{\pi_i}(\theta_i)$中的期望可以通过从经验回放池D采样并进行近似计算,采用随机梯度下降方法即可实现对智能体i的actor网络进行更新.

智能体i的critic网络参数β_i可以通过最小化智能体i的贝尔曼误差$J_Q(\beta_i)$进行更新:

$$J_Q(\beta_i) = \mathrm{E}_{x,a,r,x'\sim D}\left[\frac{1}{2}\left(Q^{\beta_i}(x,a) - y\right)^2\right] \quad (10)$$

其中,

$$y = r_i + \gamma \mathrm{E}_{a'\sim\pi_\theta}\left[Q^{\bar{\beta}_i}(x',a') - \lambda\log\left(\pi_{\bar{\theta}_i}(a_i'|o_i')\right)\right] \quad (11)$$

其中,$\bar{\beta}_i$为智能体i的目标critic网络$Q^{\bar{\beta}_i}$的参数,与SAC不同,为了稳定智能体的学习效果,在智能体中增加目标actor网络$\pi_{\bar{\theta}_i}$,$\bar{\theta}_i$为智能体i的目标actor网络$\pi_{\bar{\theta}_i}$的参数. 在经验回放池D中进行随机采样得到固定批量的样本对$J_Q(\beta_i)$进行近似计算,其中智能体i通过将下一个状态的观察o_i'输入目标actor网络$\pi_{\bar{\theta}_i}$,然后根据得到的动作概率分布进行采样得到智能体的下一个动作a_i'.

如图1所示,类似于DQN中的评估网络和目标网络,MASAC中智能体i具有4个深度神经网络,分别为actor网络和critic网络以及目标actor网络和目标critic网络. 在训练过程中,只对actor网络和critic网络进行训练,而目标actor网络和目标critic网络用于稳定actor网络和critic网络的学习效果. actor网络和目标actor网络分别利用智能体自身的当前观察o_i和下一个状态的观察o_i',生成当前动作和目标动作. critic网络的输入为当前所有智能体的观察x和动作a,输出为智能体i动作的Q值Q_i,目标critic网络的输入为下个状态智能体的观察x'和动作a',输出为智能体i目标动作的Q值T_Q_i. 同时,每次actor网络和critic网络参数更新后,需要对目标actor和目标critic网络进行软更新,用于保证算法的稳定运行:

$$\bar{\theta}_i = \tau\theta_i + (1-\tau)\bar{\theta}_i \quad (12)$$

$$\bar{\beta}_i = \tau\beta_i + (1-\tau)\bar{\beta}_i \quad (13)$$

其中,τ为控制目标网络软更新的超参数.

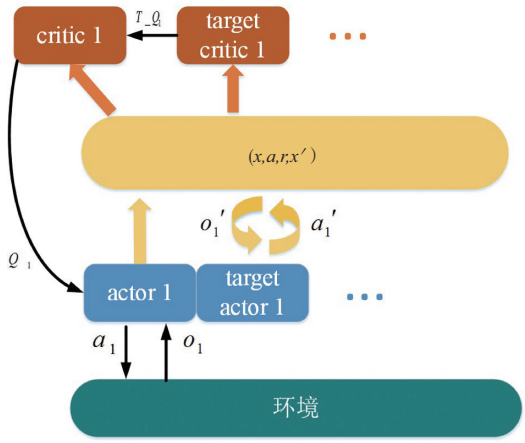


图1　MASAC网络模型

4　基于通信设备的MASAC通信模型设计

CTDE框架中,MASAC使用独立的critic网络,不涉及智能体的参数共享,因此无法利用智能体之间的反向传播,实现智能体之间的通信[14].

本文引入了一种可以在基于CTDE的MASAC中进行信息共享的通信机制. 该机制通过使用所有智能体可以共享的通信设备,解决智能体之间的通信问题. 智能体利用门机制和GRU的原理学习如何在通信设备中进行信息表示,将私有观察或历史经验有效地传达给其它智能体,同时需要对其它智能体存入设备的内容进行解码,帮助其改进自身策略[15,16].

4.1　MASAC中基于通信设备的通信机制

如图2所示,将通信设备定义为M,用于存储智能体的通信消息$\boldsymbol{m}$,其中$m\in R^M$. 每个智能体的actor网络

不再仅仅依赖于其自身的观察，而是同时输入通信消息 $\boldsymbol{m}$ 和私有观察，即 $o_i \times M \mapsto a_i$. 环境中每个智能体在执行动作之前，都需要进行读操作，访问设备 M，获取其它智能体留下的消息. 读取消息后，智能体执行写操作，根据自身观察和消息 $\boldsymbol{m}$ 更新设备 M 的内容，具体操作如图2所示.

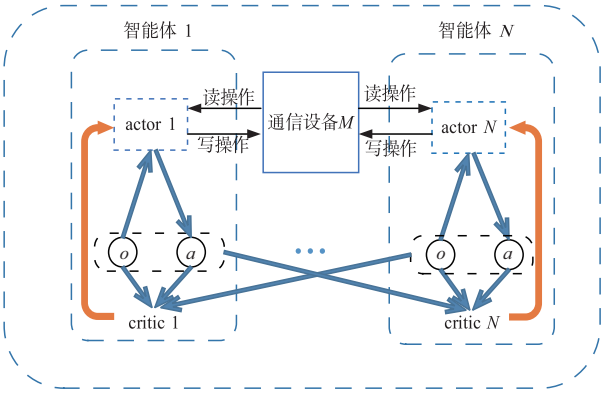


图2 MASAC中的通信模型

（1） 编码操作

智能体 i 在收到私有观察 o_i 后，对其进行编码操作：

$$\boldsymbol{e}_i = \varphi_{\theta_i^e}(o_i) \tag{14}$$

其中，$\varphi_{\theta_i^e}$ 是参数为 θ_i^e 的多层感知器（Multilayer Perceptron，MLP），编码操作是后续读操作和写操作的基础.

（2） 读操作

对当前观察进行编码后，智能体 i 执行读取操作，通过提取和解释 $\boldsymbol{m}$，获取之前智能体存入的相关信息. 由于在不同的状态中，智能体可以对编码 $\boldsymbol{e}_i$ 进行不同的解释，因此首先对编码在 $\boldsymbol{e}_i$ 中的信息进行解释和提取生成向量 $\boldsymbol{h}_i$，

$$\boldsymbol{h}_i = \boldsymbol{W}_i^h \boldsymbol{e}_i \tag{15}$$

其中 $\boldsymbol{W}_i^h$ 表示对编码 $\boldsymbol{e}_i$ 进行特征提取的需要进行学习的参数矩阵. 智能体 i 私有观察的编码 $\boldsymbol{e}_i$、编码解释向量 $\boldsymbol{h}_i$ 和当前 $\boldsymbol{m}$ 中分别包含不同类型的信息，这些信息共同用作学习选通机制 $\boldsymbol{k}_i$ 的输入：

$$\boldsymbol{k}_i = \sigma\left(\boldsymbol{W}_i^k[\boldsymbol{e}_i, \boldsymbol{h}_i, \boldsymbol{m}]\right) \tag{16}$$

式中，$\sigma(\cdot)$ 是Sigmoid函数，$[\boldsymbol{e}_i, \boldsymbol{h}_i, \boldsymbol{m}]$ 表示将三个向量连接在一起. $\boldsymbol{k}_i$ 的值用作权重提取并解释向量 $\boldsymbol{m}$ 的内容，

$$\boldsymbol{d}_i = \boldsymbol{m} \odot \boldsymbol{k}_i \tag{17}$$

其中 $\odot$ 表示哈达马积，$\boldsymbol{k}_i$ 取(0,1)中的值，对当前存储在 $\boldsymbol{m}$ 中的值分别赋予不同的权重，从而完成信息的读取.

从读操作中可以看出，每个智能体都拥有不同的可学习的参数矩阵 $\boldsymbol{W}_i^h$ 和 $\boldsymbol{W}_i^k$，因此每个智能体可以根据当前状态或观察对 $\boldsymbol{m}$ 进行不同的解释. 在读操作中，输出结果 $\boldsymbol{d}_i$ 依赖于 $\boldsymbol{e}_i$ 和 $\boldsymbol{m}$，将参数矩阵 $\boldsymbol{W}_i^h$ 和 $\boldsymbol{W}_i^k$ 归一化为参数 $\theta_i^d = \{\boldsymbol{W}_i^h, \boldsymbol{W}_i^k\}$，可以得到

$$\boldsymbol{d}_i = \varphi_{\theta_i^d}(\boldsymbol{e}_i, \boldsymbol{m}) \tag{18}$$

其中，$\varphi_{\theta_i^d}$ 为参数为 θ_i^d 的神经网络.

（3） 写操作

写入阶段，本文使用类似GRU中的方法更新 $\boldsymbol{m}$. 首先使用 $\boldsymbol{m}$ 和编码 $\boldsymbol{e}_i$ 获取控制重置门控 $\boldsymbol{r}_i$ 和控制更新门控 $\boldsymbol{z}_i$ 的状态，分别使用权重矩阵 $\boldsymbol{W}_i^r$ 和 $\boldsymbol{W}_i^z$ 生成 $\boldsymbol{r}_i$ 和 $\boldsymbol{z}_i$，

$$\boldsymbol{r}_i = \sigma\left(\boldsymbol{W}_i^r[\boldsymbol{m}, \boldsymbol{e}_i]\right) \tag{19}$$

$$\boldsymbol{z}_i = \sigma\left(\boldsymbol{W}_i^z[\boldsymbol{m}, \boldsymbol{e}_i]\right) \tag{20}$$

得到重置门控信号 $\boldsymbol{r}_i$ 之后，利用 $\boldsymbol{r}_i$ 重置 $\boldsymbol{m}$，得到 $\boldsymbol{r}_i'$，

$$\boldsymbol{r}_i' = \boldsymbol{r}_i \odot \boldsymbol{m} \tag{21}$$

通过将 $\boldsymbol{r}_i'$ 和 $\boldsymbol{e}_i$ 连接，通过tanh激活函数将数据放缩到(−1,1)之间，

$$\tilde{\boldsymbol{m}} = \tanh\left(\boldsymbol{W}_i^{\tilde{m}}[\boldsymbol{r}_i', \boldsymbol{e}_i]\right) \tag{22}$$

$\boldsymbol{W}_i^{\tilde{m}}$ 为可学习的权重矩阵，用于选取需要记忆的内容. 更新记忆阶段，使用控制更新的门控 $\boldsymbol{z}_i$ 通过如果如下方式对 $\boldsymbol{m}$ 进行更新：

$$\boldsymbol{m}' = \boldsymbol{z}_i \odot \boldsymbol{m} + (1 - \boldsymbol{z}_i) \odot \tilde{\boldsymbol{m}} \tag{23}$$

如图3所示，不同于LSTM这里 $\boldsymbol{z}_i$ 既充当了遗忘门，又充当了记忆门的功能，当 $\boldsymbol{z}_i$ 的值接近1时，代表记忆下来的数据越多，当 $\boldsymbol{z}_i$ 接近于0时，代表遗忘的越多. $\boldsymbol{z}_i \odot \boldsymbol{m}$ 表示对历史状态的选择性遗忘，这里 $\boldsymbol{z}_i$ 就充当了遗忘门的作用. $(1 - \boldsymbol{z}_i) \odot \tilde{\boldsymbol{m}}$ 表示对当前状态 $\tilde{\boldsymbol{m}}$ 的选择性记忆，$(1 - \boldsymbol{z}_i)$ 充当了记忆门的作用.

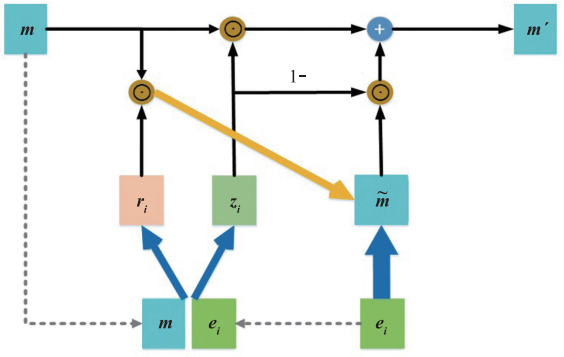


图3 写操作模型

每个智能体根据各自的权重矩阵 $\boldsymbol{W}_i^r$、$\boldsymbol{W}_i^z$ 和 $\boldsymbol{W}_i^{\tilde{m}}$ 生成的 $\boldsymbol{m}'$，用于更新通信设备 M，从而使得其它智能体都能够获得该更新信息. 新的 $\boldsymbol{m}'$ 依赖于智能体的编码 $\boldsymbol{e}_i$ 和 $\boldsymbol{m}$，因此将权重矩阵 $\boldsymbol{W}_i^r$、$\boldsymbol{W}_i^z$ 和 $\boldsymbol{W}_i^{\tilde{m}}$ 归一化为 $\theta_i^w = \{\boldsymbol{W}_i^r, \boldsymbol{W}_i^z, \boldsymbol{W}_i^{\tilde{m}}\}$，写操作可表示为

$$\boldsymbol{m}' = \varphi_{\theta_i^w}(\boldsymbol{e}_i, \boldsymbol{m}) \tag{24}$$

$\varphi_{\theta_i^w}$为参数为θ_i^w的神经网络.

4.2 基于通信机制的actor网络结构

智能体在完成读操作和写操作之后,便可以利用读操作生成的$\boldsymbol{d}_i$、写操作对$\boldsymbol{m}$更新后得到的$\boldsymbol{m}'$以及智能体私有观察的编码$\boldsymbol{e}_i$,生成动作的概率分布,即

$$\pi_{\theta_i}\left(a_i \middle| o_i, \boldsymbol{m}\right) = \varphi^a_{\theta_i^a}\left(\boldsymbol{d}_i, \boldsymbol{m}', \boldsymbol{e}_i\right) \tag{25}$$

其中$\varphi^a_{\theta_i^a}$是参数为θ_i^a的神经网络. 智能体读取得到的内容$\boldsymbol{d}_i$和$\boldsymbol{m}$更新后得到的内容$\boldsymbol{m}'$均依赖于$\boldsymbol{m}$和$\boldsymbol{e}_i$,而$\boldsymbol{e}_i$依赖于o_i,因此可以得到actor网络π_{θ_i}的表达式为,

$$\begin{aligned}&\pi_{\theta_i}\left(a_i \middle| o_i, \boldsymbol{m}\right)\\ &= \varphi^a_{\theta_i^a}\left(\boldsymbol{d}_i, \boldsymbol{m}', \boldsymbol{e}_i\right)\\ &= \varphi^a_{\theta_i^a}\left(\varphi_{\theta_i^d}\left(\boldsymbol{e}_i, \boldsymbol{m}\right), \varphi_{\theta_i^w}\left(\boldsymbol{e}_i, \boldsymbol{m}\right), \boldsymbol{e}_i\right)\end{aligned} \tag{26}$$

构建完成智能体之间通信后,可以得到完整的MASAC算法.

5 仿真实验

为了评价MASAC算法在多智能体环境中的表现,研究中利用多智能体深度强化学习环境Grounded Communication Environment(GCE)进行仿真实验,采用基于GCE多个包含协同或协同和竞争两者混合的环境对MASAC进行评价.

5.1 实验设置

本文实验采用的深度学习框架为pytorch. actor中编码操作采用由256个神经单元组成的单层MLP完成,$\boldsymbol{m}$和$\boldsymbol{h}_i$的维度为200. critic网络的隐藏层共3层,每层的神经元个数为128. 实验中采用的训练优化器为AdamOptimizer,actor网络和critic网络的学习速率设置为0.001,更新目标网络的超参数τ设置为0.01,奖励折扣γ设置为0.95,参数λ设置为0.2.

5.2 MASAC在不同环境中的稳定性分析

单智能体深度强化学习算法SAC在多智能体环境中独立地训练每个智能体,而不考虑环境的动态性,而MASAC利用CTDE中集中训练方式和通信机制,可以提高算法稳定性. 实验中主要用到环境为协同通信环境(Cooperative Communication,CC)和同时包含协同和竞争的环境physical deception.

CC是一个多智能体合作的环境,环境中主要包含两个智能体,speaker和listener,以及3个不同颜色的地标组成. speaker可以获得和listener相同颜色的地标(目标地标)位置,并学习以通信的方式将listener引领到该位置,两者通过listener到目标地标距离的远近获得共享奖励.

使用集中学习算法MASAC和分散学习算法SAC在协同环境CC中对比,智能体训练过程中获得的奖励如图4所示.

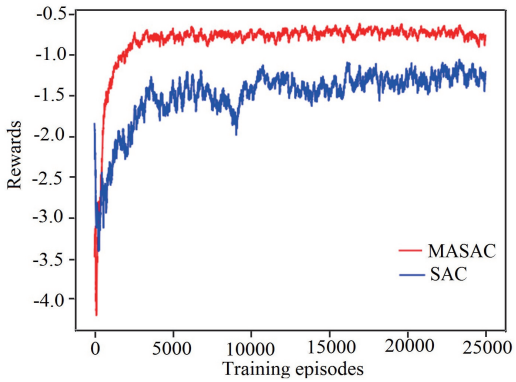


图4　CC环境不同算法获得的奖励

在CC环境中,智能体共训练了25000轮. 如图4所示,MASAC在训练到5000轮时,已经达到收敛状态,而SAC在训练17000轮后逐渐趋于稳定. 无论是算法收敛效果,还是智能体获得的奖励,集中学习的算法MASAC均优于分散学习的算法SAC.

实验中对训练25000轮后的智能体,在CC环境中测试了1000轮,并对结果进行了统计,包括智能体到达目标的概率和距离目标的平均距离.

如表1所示,由MASAC训练的listener虽然掌握了目标地标的位置信息,但由于listener的动作是经过对随机策略函数生成的概率分布随机采样得到的,所以并不是每次都朝着正确的地标方向移动,因此到达目标的概率为87.5%. 同样对于SAC,由于相对随机的动作,智能体虽然不知道目标位置,到达目标的概率仍有19.3%,但与MASAC到达目标的概率相差68.2%.

表1　智能体到达目标的概率和距离目标的平均距离

算法	到达目标的概率	平均距离
MASAC	87.5%	0.126
SAC	19.3%	0.437

通过有效的集中训练和通信机制,同样可以使MASAC在同时包含竞争和合作的更加复杂的环境中进行训练. 实验中引入同时包含竞争和合作的环境physical deception测试算法性能. 如图5所示,physical deception中的2个蓝色智能体学习通过合作到达2个地标(绿色和黑色小球)中的其中一个,环境中的另一个粉色智能体也希望到达该目标,从而获得奖励,但是粉色智能体不知道哪个地标是目标地标,当粉色智能体接近目标地标时,蓝色智能体会受到惩罚.

在physical deception中,分别利用MASAC训练蓝色智能体和粉色智能体. 如图5,分别截取了在训练达到20000轮t=5和t=30时实验环境的截图,在实验环境

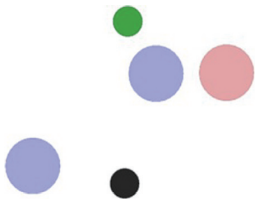
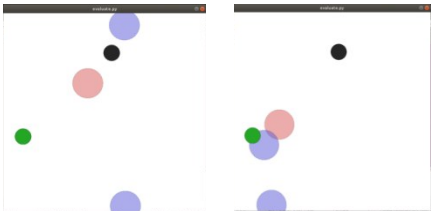


图5 physical deception环境示意图

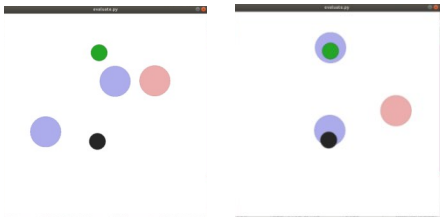
中,粉色智能体和蓝色智能体有相同的目标,蓝色智能体知道目标地标的颜色,因此粉色智能体需要学会根据蓝色智能体的位置判断目标地标颜色. 如图6所示,在训练20000轮后,蓝色智能体已经学会利用占据目标地标的方法,减少粉色智能体到达目标地标的概率,提高其身获得的奖励.



(a) t=5 (b) t=30

图6 physical deception训练20000轮效果图

蓝色智能体获得的奖励会随着粉红色智能体靠近目标地标而变少,可以看到此时两个蓝色智能体并未真正实现协同. 随着训练的进行,当训练次数达到60000轮时,分别截取了t=5和t=30的实验画面.



(a) t=5 (b) t=30

图7 physical deception训练60000轮效果图

如图7,粉色智能体靠近目标地标时,蓝色智能体会受到惩罚,而蓝色智能体的奖励为其中一个智能体到达目标地标即可获得最大奖励,并且粉色智能体不知道目标地标的位置,因此蓝色智能体逐渐在MASAC的训练中学会协同方法,通过覆盖环境中的两个地标,迷惑粉色对手,使其无法分辨出目标地标位置,从而减少惩罚,获得更多奖励.

以上的实验作为测试MASAC在不同环境中的稳定性,可以看到,相比于SAC,MASAC在CTDE框架和有效的通信机制的作用下,在不同环境中均具有较好的表现.

6 总结

本文将深度强化学习算法SAC通过CTDE框架扩展为多智能体深度强化学习算法MASAC,并且构建了智能体之间的通信机制. MASAC帮助智能体实现了观察信息和历史经验的共享,使其能够在多智能体环境中取得更好的表现,同时有效减少了多智能体环境的不稳定性对算法造成的影响. 实验表明,在协同或者协同和竞争两者兼有的环境中,MASAC均比SAC取得了更好的训练效果.

参考文献

[1] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature, 2016, 529(7587): 484 – 489.

[2] Silver D, Schrittwieser J, Simonyan K, et al. Mastering the game of Go without human knowledge[J]. Nature, 2017, 550 (7676): 354 – 359.

[3] 周沛, 陈后金, 于泽宽, 等. 跨模态医学图像预测综述[J]. 电子学报, 2019, 47(1):220 – 226.
Zhou P, Chen H J, Yu Z K, et al. A review of multimodal medical image prediction [J]. Acta Electronica,Sinica,2019, 47 (1): 220 – 226. (in Chinese)

[4] Lowe R, Foerster J, Boureau Y L, et al. On the pitfalls of measuring emergent communication[A]. Proceedings of the 18th International Conference on Autonomous Agents and Multi-Agent Systems[C]. Montreal: ICAAMS, 2019. 693 – 701.

[5] Wang X, Chen W, Wu J, et al. Video captioning via hierarchical reinforcement learning[A]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition [C]. Hawaii: CVPR, 2018. 4213 – 4222.

[6] 郑兴华, 孙喜庆, 吕嘉欣,等. 基于深度学习和智能规划的行为识别[J]. 电子学报, 2019, 47(8):1661 – 1668.
Zheng X H, Sun X Q, LV J X, et al. Behavior recognition based on deep learning and intelligent planning [J]. Acta Electronica Sinica, 2019, 47 (8): 1661 – 1668. (in Chinese)

[7] Schulman J, Levine S, Abbeel P, et al. Trust region policy optimization[A]. International Conference on Machine Learning[C]. Lille: ICML, 2015. 1889 – 1897.

[8] 闻佳, 王宏君, 邓佳, 等. 基于深度学习的异常事件检测[J]. 电子学报, 2020,48(2):308 – 313.
Wen J, Wang H J, Deng J, et al. Abnormal event detection based on deep learning [J]. Acta Electronica Sinica, 2020, 48 (2): 308 – 313. (in Chinese)

[9] Abdallah S, Kaisers M. Addressing environment non-stationarity by repeating Q-learning updates[J]. The Journal of Machine Learning Research, 2016, 17(1): 1582 – 1612.

[10] Foerster J N, Farquhar G, Afouras T, et al. Counterfactual multi-agent policy gradients[A]. Thirty-second AAAI Conference on Artificial Intelligence[C]. New Orleans: AAAI, 2018. 2974 – 2982.

[11] Lowe R, Wu Y, Tamar A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments[A]. Advances in Neural Information Processing Systems[C]. Long Beach: NIPS, 2017. 6379 – 6390.

[12] Haarnoja T, Zhou A, Abbeel P, et al. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor[A]. International Conference on Machine Learning[C]. Stockholm: ICML, 2018. 1856 – 1865.

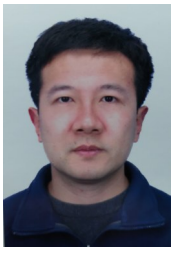
[13] Haarnoja T, Tang H, Abbeel P, et al. Reinforcement learning with deep energy-based policies[A]. Proceedings of the 34th International Conference on Machine Learning[C]. Sydney: ICML, 2017. 1352 – 1361.

[14] Das A, Kottur S, Moura J M F, et al. Learning cooperative visual dialog agents with deep reinforcement learning[A]. Proceedings of the IEEE International Conference on Computer Vision[C]. Venice: ICCV, 2017. 2951 – 2960.

[15] 曹源,唐涛,徐田华,穆建成.形式化方法在列车运行控制系统中的应用[J].交通运输工程学报, 2010, 10(1):112 – 126.
Cao Yuan, Tang Tao, Xu Tianhua, Mu Jiancheng. Application of formal method in train operation control system [J]. Journal of Transportation Engineering, 2010, 10 (1): 112 – 126. (in Chinese)

[16] 吴胜权,黄振晖,曹源. 有轨电车路权配置与信号系统选择[J]. 中国铁路, 2014, (8):97 – 99.
Wu Shengquan, Huang Zhenhui, Cao Yuan. Tram right of way configuration and signal system selection [J]. China Railway, 2014, (8): 97 – 99. (in Chinese)

作者简介



肖　硕　男,1981年9月生于江苏徐州。中国矿业大学副教授,硕士生导师,研究方向为人工智能、物联网等。
E-mail:sxiao@cumt.edu.cn

黄珍珍　女,1981年8月生于江苏徐州。中国矿业大学副研究馆员,研究方向为计算机网络、人工智能、物联网等。
E-mail:huangzhenzhen@cumt.edu.cn

张国鹏　男,1978年6月生于江苏徐州。中国矿业大学教授,博士生导师,研究方向为人工智能、物联网等。
E-mail:gpzhang@cumt.edu.cn