Mean Field Multi-Agent Reinforcement Learning

- 摘要

作者提出 Mean Field 方法，该方法通过一个单智能体及其邻居智能体或者所有智能体的平均影响来估计智能体之间的相互作用。MA 环境中两个智能体的相互作用是相互增强的，不是独立于智能体的。作者提出两种均场方法的应用，一个是 Mean Field Q-learning，一个是 Mean Field AC 方法并讨论该算法可以收敛到纳什均衡点。

- 介绍

作者在大量智能体存在的环境中实现该算法。有限智能体的互动，任何一对智能体的互动都是透明的。应用均值场理论求解可伸缩性问题。通过一个智能体所包括的全局群体或者局部群体的平均影响作用来估计智能体的互动作用。

- 相关工作

零和博弈的 Max-Min Q-learning，多人随机博弈的纳什 Q-learning 和 FFQ 朋友敌人 Q-learning。

中心化训练和分散式执行算法有 BicNet、COMA 和 MADDPG。这些方法在智能体数量上就难以达到效果。作者提出的方法能解决这个问题，该方法是基于联合动作的，即将其他智能体的动作考虑进去。Q 函数的参数是与智能体数量无关的，因为它将多智能体之间的互动转化为一个智能体和它邻居智能体的互动，即一对多问题转换为一对一问题。该算法将独立于一个智能体考虑对于它自己最好的动作。该算法启发于 MFG 均场理论博弈，MFG 研究由个人决策的集合而产生的种群行为。**MFG 通过定义两个微分方程模拟个人价值函数的后向动力学和智能体群体的合计价值分布。尽管后向动力学等价于 bellman 方程，MFG 的主要目标是进行基于模型的规划，并推断个体密度随时间的变化。**该算法是 model-base 的，不同于 model-free，作者的目标是在 TD 框架下将 MFG 投入到可计算的 Q-learning。

- 方法

联合动作的维度与智能体的数量成比例。由于智能体战略性地执行动作，因此可能有竞争有合作，而同时评估带有联合动作的 Q 就变得不可行了，因为 Q 的估计复杂度很高。减少 Q 的复杂度计算的一个方法是将联合动作 Q 分解，因为交互可以分解为两个智能体之间的相互作用，所以对于智能体 i，该智能体的 Q 估计就是它与所有交互智能体的联合动作下的 Q 值的平均值。公式如下：

$$Q^j(s,\boldsymbol{a}) = \frac{1}{N^j}\sum_{k\in\mathcal{N}(j)} Q^j(s,a^j,a^k), \qquad (5)$$

其中 N 表示智能体 i 的邻居智能体的数量。该方法的分解能保证在成对交互中保留全局性。

1. Mean Field 估计

使用均场理论来估计公式 5 的成对 Q 函数。这里考虑离散动作空间，动作使用 one-hot 编码。对于智能体 j 的邻居智能体的动作，使用邻居智能体平均动作和一个小波动来表征。

$$a^k = \bar{a}^j + \delta a^{j,k}, \qquad \text{where } \bar{a}^j = \frac{1}{N^j}\sum_k a^k, \qquad (6)$$

公式如下：

第一项是邻居智能体的平均动作表征，第二项是小波动。平均动作可以将智能体 j 的邻居智能体动作分布显式表达。根据泰勒展开理论，成对 Q 函数如果二阶可微，则总 Q 函数可以泰勒展开：

$$
\begin{aligned}
Q^j(s,\boldsymbol{a}) &= \frac{1}{N^j}\sum_k Q^j(s,a^j,a^k) \\
&= \frac{1}{N^j}\sum_k \Big[Q^j(s,a^j,\bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s,a^j,\bar{a}^j)\cdot \delta a^{j,k} \\
&\qquad + \frac{1}{2}\delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}} Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k} \Big] \\
&= Q^j(s,a^j,\bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s,a^j,\bar{a}^j)\cdot \Big[\frac{1}{N^j}\sum_k \delta a^{j,k}\Big] \\
&\qquad + \frac{1}{2N^j}\sum_k \Big[\delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}} Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k}\Big] \quad (7) \\
&= Q^j(s,a^j,\bar{a}^j) + \frac{1}{2N^j}\sum_k R^j_{s,a^j}(a^k) \;\approx\; Q^j(s,a^j,\bar{a}^j), \quad (8)
\end{aligned}
$$

$$
\text{where } R^j_{s,a^j}(a^k) \triangleq \delta a^{j,k}\cdot \nabla^2_{\tilde{a}^{j,k}} Q^j(s,a^j,\tilde{a}^{j,k})\cdot \delta a^{j,k}, \quad \tilde{a}^{j,k} = \bar{a}^j + \epsilon^{j,k}\delta a^{j,k}
$$

其中小波动和为 0，即 $\sum_k \delta a^k = 0$，对于第二项，R 的输入变量 ak 由其他智能体决定，因此是另一个分布，因此本质是一个随机变量，事实上可以进一步证明 R 的边界在[-2M, 2M]，在成对 Q 函数是温和的条件下，即变化幅度不太大，R 是 M 阶可微的，因此 R 影响近乎不计接近 0。附录 B 具体展示了边界证明。简化假设所有智能体具有同质性和局部性，则 R 余数项互相抵消。

像图 1 所示那样，一个智能体 j 的均场估计可以被简化为一个中心智能体即网格中某一个点，和抽象出的虚拟邻居代表智能体，即网格中某一点附近的蓝色趋于邻居影响的交互。

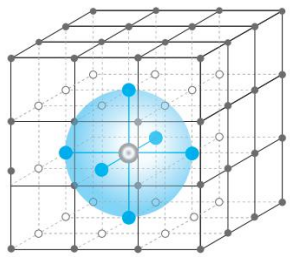

Figure 1: Mean field approximation. Each agent is represented as a node in the grid, which is only affected by the **mean** effect from its **neighbors** (the blue area). Many-agent interactions are effectively converted into two-agent interactions.

可以将交互简化成公式 8，同时均场 Q 函数的更新可以表达为：

$$
Q^j_{t+1}(s,a^j,\bar{a}^j) = (1-\alpha)Q^j_t(s,a^j,\bar{a}^j) + \alpha[r^j + \gamma v^j_t(s')], \quad (9)
$$

式中均场状态的价值函数由公式 10 确定：

$$
v^j_t(s') = \sum_{a^j} \pi^j_t(a^j|s',\bar{a}^j)\,\mathbb{E}_{\bar{a}^j(\boldsymbol{a}^{-j})\sim \boldsymbol{\pi}^{-j}_t}\Big[Q^j_t(s',a^j,\bar{a}^j)\Big], \quad (10)
$$

在博弈中，平均动作首先由前一个平均动作决定的参数化 Π 来计算的：

$$
\bar{a}^j = \frac{1}{N^j}\sum_k a^k,\; a^k \sim \pi^k_t(\cdot|s,\bar{a}^k_-), \quad (11)
$$

策略更新依赖于平均动作 a，因此新的玻尔兹曼策略通过以下公式确认：

$$\pi_t^j(a^j|s,\bar{a}^j)=\frac{\exp\left(-\beta Q_t^j(s,a^j,\bar{a}^j)\right)}{\sum_{a^{j'}\in\mathcal{A}^j}\exp\left(-\beta Q_t^j(s,a^{j'},\bar{a}^j)\right)}. \quad (12)$$

通过公式 11 和公式 12，对于所有智能体的平均动作及其策略就可以确认。虽然直观上没法说明智能体 j 的交互能够稳定下来，但是通过实验发现，上式通过几次迭代后收敛。

与纳什 V 函数不同的是，作者使用公式 10 的均场 V 函数代替纳什 V 函数，同时我们定义均场操作算子：

$$\mathscr{H}^{\mathrm{MF}}\boldsymbol{Q}(s,\boldsymbol{a})=\mathbb{E}_{s'\sim p}\left[\boldsymbol{r}(s,\boldsymbol{a})+\gamma\boldsymbol{v}^{\mathrm{MF}}(s')\right]. \quad (13)$$

事实上，我们可以证明该算子定义了一种压缩映射，那就是应用算子等价于迭代地更新 Q，在适当条件下，均场 Q 会收敛到纳什 Q。

2. 实现

在 MF - Q 中，loss fucntion 定义如下：

$$\mathscr{L}(\phi^j)=\left(y^j-Q_{\phi^j}(s,a^j,\bar{a}^j)\right)^2,\quad y^j=r^j+\gamma v_{\phi_-^j}^{\mathrm{MF}}(s')$$

其中 y 由目标网络确定，而 loss fucntion 的微分如下：

$$\nabla_{\phi^j}\mathscr{L}(\phi^j)=\left(y^j-Q_{\phi^j}(s,a^j,\bar{a}^j)\right)\nabla_{\phi^j}Q_{\phi^j}(s,a^j,\bar{a}^j), \quad (14)$$

Mean Field Q-learning 算法如下：

Algorithm 1 Mean Field Q-learning (MF-Q)

Initialise Q_{ϕ^j}, $Q_{\phi_-^j}$, and $\bar{a}^j$ for all $j\in\{1,\dots,N\}$
while training not finished **do**
  for $m=1,\dots,M$ **do**
    For each agent j, sample action a^j from Q_{ϕ^j} by Eq. (12), with the current mean action $\bar{a}^j$ and the exploration rate β
    For each agent j, compute the new mean action $\bar{a}^j$ by Eq. (11)
  Take the joint action $\boldsymbol{a}=[a^1,\dots,a^N]$ and observe the reward $\boldsymbol{r}=[r^1,\dots,r^N]$ and the next state s'
  Store $\langle s,\boldsymbol{a},\boldsymbol{r},s',\bar{\boldsymbol{a}}\rangle$ in replay buffer $\mathcal{D}$, where $\bar{\boldsymbol{a}}=[\bar{a}^1,\dots,\bar{a}^N]$
  for $j=1$ to N **do**
    Sample a minibatch of K experiences $\langle s,\boldsymbol{a},\boldsymbol{r},s',\bar{\boldsymbol{a}}\rangle$ from $\mathcal{D}$
    Sample action a_-^j from $Q_{\phi_-^j}$ with $\bar{a}_-^j\leftarrow\bar{a}^j$
    Set $y^j=r^j+\gamma v_{\phi_-^j}^{\mathrm{MF}}(s')$ by Eq. (10)
    Update the Q-network by minimizing the loss $\mathscr{L}(\phi^j)=\frac{1}{K}\sum\left(y^j-Q_{\phi^j}(s^j,a^j,\bar{a}^j)\right)^2$
  Update the parameters of the target network for each agent j with learning rate τ:

$$\phi_-^j\leftarrow\tau\phi^j+(1-\tau)\phi_-^j$$

而 MF - AC 算法的 actor 又可以通过以下公式训练：

$$\nabla_{\theta^j}\mathcal{J}(\theta^j)\approx\nabla_{\theta^j}\log\pi_{\theta^j}(s)Q_{\phi^j}(s,a^j,\bar{a}^j)\Big|_{a=\pi_{\theta^j}(s)}.$$

而 critic 的训练 loss function 可以通过公式 14 来确定。

Mean Field AC 算法如下：

Algorithm 2 Mean Field Actor-Critic (MF-AC)

Initialize Q_{ϕ^j}, $Q_{\phi^j_-}$, π_{θ^j}, $\pi_{\theta^j_-}$, and $\bar{a}^j$ for all $j \in \{1, \ldots, N\}$

while training not finished **do**

For each agent j, sample action $a^j = \pi_{\theta^j}(s)$; compute the new mean action $\bar{\boldsymbol{a}} = [\bar{a}^1, \ldots, \bar{a}^N]$

Take the joint action $\boldsymbol{a} = [a^1, \ldots, a^N]$ and observe the reward $\boldsymbol{r} = [r^1, \ldots, r^N]$ and the next state s'

Store $\langle s, \boldsymbol{a}, \boldsymbol{r}, s', \bar{\boldsymbol{a}} \rangle$ in replay buffer $\mathcal{D}$

for $j = 1$ to N **do**

Sample a minibatch of K experiences $\langle s, \boldsymbol{a}, \boldsymbol{r}, s', \bar{\boldsymbol{a}} \rangle$ from $\mathcal{D}$

Set $y^j = r^j + \gamma\, v^{\mathrm{MF}}_{\phi^j_-}(s')$ by Eq. (10)

Update the critic by minimizing the loss $\mathcal{L}(\phi^j) = \frac{1}{K} \sum \left(y^j - Q_{\phi^j}(s, a^j, \bar{a}^j)\right)^2$

Update the actor using the sampled policy gradient:

$$\nabla_{\theta^j} \mathcal{J}(\theta^j) \approx \frac{1}{K} \sum \nabla_{\theta^j} \log \pi_{\theta^j}(s') Q_{\phi^j_-}(s', a^j_-, \bar{a}^j_-)\Big|_{a^j_- = \pi_{\theta^j_-}(s')}$$

Update the parameters of the target networks for each agent j with learning rates τ_ϕ and τ_θ:

$$\phi^j_- \leftarrow \tau_\phi \phi^j + (1 - \tau_\phi)\phi^j_-$$
$$\theta^j_- \leftarrow \tau_\theta \theta^j + (1 - \tau_\theta)\theta^j_-$$

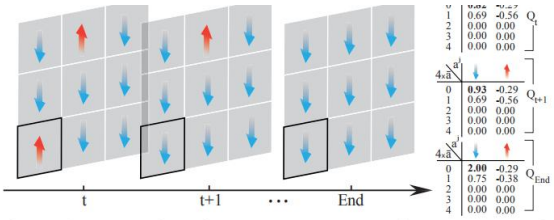

Figure 2: MF-Q iterations on a 3×3 stateless toy example. The goal is to coordinate the agents to an agreed direction. Each agent has two choices of actions: *up* $\uparrow$ or *down* $\downarrow$. The reward of each agent's staying in the same direction as its $[0, 1, 2, 3, 4]$ neighbors are $[-2.0, -1.0, 0.0, 1.0, 2.0]$, respectively. The neighbors are specified by the four directions on the grid with cyclic structure on all directions, *e.g.* the first row and the third row are adjacent. The reward for the highlighted agent j on the bottom left at time $t+1$ is 2.0, as all neighboring agents stay down in the same time. We listed the Q-tables for agent j at three time steps where $\bar{a}^j$ is the percentage of neighboring ups. Following Eq. 9, we have $Q^j_{t+1}(\uparrow, \bar{a}^j = 0) = Q^j_t(\uparrow, \bar{a}^j = 0) + \alpha[r^j - Q^j_t(\uparrow, \bar{a}^j = 0)] = 0.82 + 0.1 \times (2.0 - 0.82) = 0.93$. The rightmost plot shows the convergent scenario where the Q-value of staying down is 2.0, which is the largest reward in the environment.

图二为 MF - Q 的迭代过程示意图。

3. 收敛性证明

**

- 实验
  1. 环境
  2. 与非联合动作强化学习方法进行比较
  3. 其他智能体学习策略的影响
  4. 使用一组策略训练策略的影响

- 总结

  作者方法模拟了 MA 环境中的交互动力学。MF Q-learning 学习每一个智能体从其他智能

体得到的均场效应的最佳响应。作者提出将一对多问题转化为一对一，即该智能体与邻居智能体的问题。文中也证明了该算法收敛到 nash Q 值。