Value-Decomposition Networks For Cooperative Multi-Agent Learning

- 摘要

作者提出的方法适用于协作多智能体环境下带有单一全局 reward 信号。该种问题是很难的，因为大量的联合动作组合。研究发现，使用全局 reward 总会出现 lazy 智能体的问题，就是 lazy 智能体分配到一个假的、几乎没有的 reward 使得它不能学习。

因此作者专门针对这个问题提出该算法，该算法训练每个智能体，是将全局 reward 通过框架分解为每个智能体独立的 reward，实验效果很好，全局 reward 又叫做 team reward 通过时间积累。

- 介绍

每个智能体自己选择独立的动作。在独立训练每个智能体，又容易因为局部观测导致智能体会从队友的行为中获得不很真实的 reward 信号。

作者说别人怎么怎么不好，以后说自己的方法是在独立学习者方法中加入将 team reward 进行分解，使得每个智能体得到的 reward 都和自己的局部观测相关，该方法是集中式学习方法。

作者提出在智能体之间神奇的已学习额外的值分解方法。隐式的，该值分解方法的目标是学习最优的线性值分解函数，该函数通过 Q 梯度反向传播，而 Q 梯度由 NN 表达，NN 表达的是个人部分的值函数。作者使用训练时以集中的方式有效地学习，而代理可以单独部署的框架方法。

作者也评估了，权值共享、角色信息及其信道等加成，表明可以改善性能。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。

作者考虑双人环境的协作。

其他解决协作下的 team reward 是基于**不同 reward**，衡量智能体的动作对于整个系统的影响。这种 reward 具有很好的性质，但可能不实用，因为它需要考虑关于状态的知识。

- 背景
- 方法

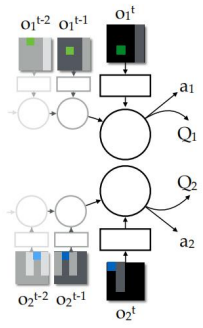

Figure 1: Independent agents architecture showing how local observations enter the networks of two agents over time (three steps shown), pass through the low-level linear layer to the recurrent layer, and then a dueling layer produces individual Q-values.

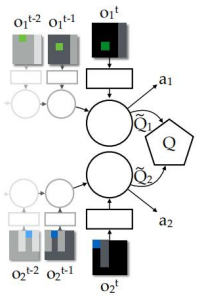

Figure 2: Value-decomposition individual architecture showing how local observations enter the networks of two agents over time (three steps shown), pass through the low-level linear layer to the recurrent layer, and then a dueling layer produces individual "values" that are summed to a joint Q-function for training, while actions are produced independently from the individual outputs.

上图图 1 表示纯粹 DQN，图二表示有值分解的 DQN，每个智能体产生的 Q 通过联合 Q-function 进行求和为了训练，同时动作又是独立输出。

作者使用的假设是联合 Q 可以被加性分解为每个智能体的个人 Q，即如下：

$$Q((h^1, h^2, ..., h^d), (a^1, a^2, ..., a^d)) \approx \sum_{i=1}^{d} \tilde{Q}_i(h^i, a^i)$$

其中 Q 表示智能体 i 的独立 Q，只和自己的局部观测有关。智能体 i 的 Q 通过使用全局 reward 的 Q 规则下得到的梯度求得的，因此智能体 i 的 Q 是隐式学习得到的，不是加特定约束、特定 reward 学习的。该算法中每个智能体的 Q 都是独立的，因此每个智能体贪婪选择自己的 Q，才能使联合 Q 最大化，即可以理解为联合 Q 是采取联合动作，在联合状态下进行的。

首先 reward 有，该分解设定一个参数来调优，设定完以后各自隐式学习 Q_ni 然后求和：

$$r(\mathbf{s}, \mathbf{a}) = r_1(o^1, a^1) + r_2(o^2, a^2),$$

因此联合 Q 使用以下公式表达：

$$Q^{\pi}(\mathbf{s}, \mathbf{a}) = \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r(\mathbf{s}_t, \mathbf{a}_t) | \mathbf{s}_1 = \mathbf{s}, \mathbf{a}_1 = \mathbf{a}; \pi]$$

$$= \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_1(o_t^1, a_t^1) | \mathbf{s}_1 = \mathbf{s}, \mathbf{a}_1 = \mathbf{a}; \pi] + \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_2(o_t^2, a_t^2) | \mathbf{s}_1 = \mathbf{s}, \mathbf{a}_1 = \mathbf{a}; \pi]$$

$$=: \bar{Q}_1^{\pi}(\mathbf{s}, \mathbf{a}) + \bar{Q}_2^{\pi}(\mathbf{s}, \mathbf{a})$$

联合 Q 是两个智能体的 Q 求和得到的，这样每个智能体可以指关注自己的状态和动作，而要是该时间戳下的状态和动作不够的话，可以增加以前时间戳的信息或者其他智能体的信息，通过 LSTM 等。Permutation 序列

Definition 1 (Agent Invariance). *If for any permutation (bijection)* $p : \{1, ..., d\} \to \{1, ..., d\}$,

$$\pi(p(\bar{h})) = p(\pi(\bar{h}))$$

we say that π *is* agent invariant.

作者也使用参数共享方法解决 lazy 智能体。

并不总是希望有智能体不变性，例如，需要专门的角色来优化特定的系统。在这种情况下，我们向每个智能体提供角色信息，或标识符。角色信息被提供给智能体，作为他们的身份的 1 热编码与第一层的每个观察连接。当智能体共享所有网络权重时，它们就是拥有条件智能体不变性，即。只有在以相同的角色为条件时，才有相同的政策。我们还考虑了智能体网络之间的信息通道，即可微连接智能体网络模块之间。这些具有共享权重的体系结构满足智能体不变性。

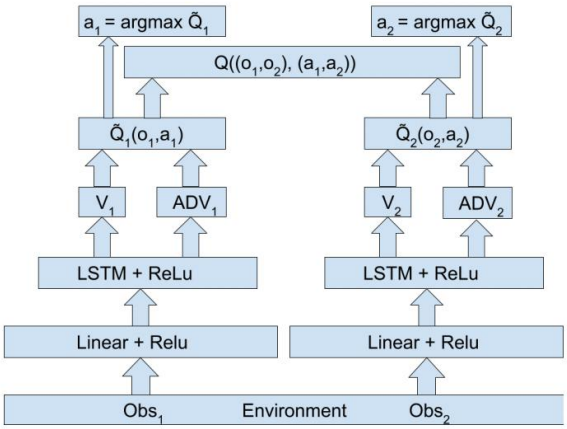

Figure 15: Value-Decomposition Individual Architecture

- 实验

Agent	V.	S.	Id	L.	H.	C.
1						
2	✓					
3	✓	✓				
4	✓	✓	✓			
5	✓	✓	✓	✓		
6	✓	✓	✓		✓	
7	✓	✓	✓	✓	✓	
8	✓					✓
9						✓

Table 1: Agent architectures. V is value decomposition, S means shared weights and an invariant network, Id means role info was provided, L stands for lower-level communication, H for higher-level communication and C for centralization. These architectures were selected to show the advantages of the independent agent with value-decomposition and to study the benefits of additional enhancements added in a logical sequence.

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 VDN 是自动值分解成局部的一步，未来作者考虑值分解方法的范围问题，怎么扩展更多的 team size。这样的话，智能体更会趋向于从其他智能体动作中获取 reward，作者也希望能用非线性函数进行分解。