Deep Multi-Agent Reinforcement Learning

Chongjie Zhang
Institute of Interdisciplinary Information Sciences
Tsinghua University
August 6, 2020
Recent AI Breakthrough
What’s Next for AI

- Shifting from pattern recognition to decision-making/control
- Shifting from single-agent to multi-agent settings

Drone Delivery
Smart Grids
Home Robots
Autonomous Vehicles
Multi-robot assembly
Video Games
Types of Multi-Agent Systems

- **Cooperative**
  - Working together and coordinating their actions
  - Maximizing a shared team reward

- **Competitive**
  - Self-interested: maximizing an individual reward
  - Opposite rewards
  - Zero-sum games

- **Mixed**
  - Self-interested with different individual rewards (not opposite)
  - General-sum games
Cooperative Multi-Agent Systems (MAS)

- A group of agents work together to optimize team performance
- Model: decentralized partially observable Markov decision process (Dec-POMPD)
  - Multi-agent sequential decision-making under uncertainty
  - Extension of MDPs and POMDPs
- At each step, each agent $i$ takes an action and receives:
  - A local observation $o_i$
  - A joint immediate reward $r$
Dec-POMDP

- **Model**
  - Agent: $i \in I = \{1, 2, ..., N\}$
  - State: $s \in S$
  - Action: $a_i \in A, \ a \in A^N$
  - Transition function: $P(s' | s, a)$
  - Reward: $R(s, a)$
  - Observation: $o_i \in \Omega$
  - Observation function: $o_i \in \Omega \sim O(s, i)$
Dec-POMDP

- **Objective:** to find policies for agents to jointly maximize the expected cumulative reward

- A local policy $\pi_i$ for each agent $i$: mapping its observation-action history $\tau_i$ to its action
  - **Action-observation history:** $\tau_i \in T = (\Omega \times A)^*$
  - State is unknown, so beneficial to remember the history

- Joint policy $\pi = < \pi_1, \ldots, \pi_n >$

- **Value function:** $Q^\pi_{tot}(\tau, \alpha) = \mathbb{E} \left[ \sum_{t=0}^{\infty} y^t r_t \mid s_0 = s, a_0 = \alpha, \pi \right]$

- Policy $\pi(\tau) = \text{argmax}_\alpha Q^\pi_{tot}(\tau, \alpha)$
Multi-Agent Reinforcement Learning (MARL)

- MARL is promising for solving Dec-POMDP problems
  - The environment model is often unknown

- MARL: learning policies for multiple agents
  - Where agents are interacting
  - Learning by interacting with other agents and the environment
Outline

- **Value-Based Methods**
  - Paradigm: Centralized Training and Decentralized Execution
  - Basic methods: VDN, QMIX, QPLEX
  - Theoretical analysis
  - Extensions

- **Policy Gradient Methods**
  - Paradigm: Centralized Critic and Decentralized Actors
  - Method: Decomposable Off-Policy Policy Gradient (DOP)

- **Goals:**
  - To give a brief introduction to (cooperative) MARL
  - To excite you about MARL
Multi-Agent Reinforcement Learning (MARL)

- How to learn joint policy $\pi$ or value function $Q_{tot}$?

**Centralized Value Functions**

- **Joint Network**: $Q_{tot}$
  - $\tau_1, a_1$, $\ldots$, $\tau_n, a_n$

**Decentralized Value Functions**

- **Agent 1**: $Q_1$
  - $\tau_1, a_1$
  - **Agent n**: $Q_n$
  - $\tau_n, a_n$

**Factorized Value Functions**

- **Mixing Network**: $Q_{tot}$
  - $Q_1$
  - $Q_n$
  - **Agent 1**: $\tau_1, a_1$
  - **Agent n**: $\tau_n, a_n$

**Scalability**

- Non-stationarity
- Credit assignment

**Centralized training**

- Decentralized execution
Factorized Value Function Learning

- Paradigm: centralized training with decentralized execution

\[ Q_{tot}(\tau, a) \]  
\[ TD \text{ Loss} \]  
\[ \mathcal{L}(\theta) = \mathbb{E}_{(\tau, a, r, \tau') \in \mathcal{D}} \left[ (r + \gamma V(\tau'; \theta^-) - Q(\tau, a; \theta))^2 \right] \]

\[ V(\tau'; \theta^-) = \max_{a'} Q(\tau', a'; \theta^-) \]

- Individual-Global Maximization (IGM) Principle
  - Consistent action selection between joint and individuals
  - \[ \arg\max_a Q_{tot}(\tau, a) = \left( \arg\max_{a_1} Q_1(\tau_1, a_1), \ldots, \arg\max_{a_n} Q_n(\tau_n, a_n) \right) \]
Value Decomposition Networks (VDN)

- **VDN**: $Q_{tot}(\tau, \alpha) = \sum_i Q_i(\tau_i, a_i)$
- Sufficient for IGM constraint
  - $\underset{\alpha}{\text{argmax}} Q_{tot}(\tau, \alpha) = \left( \underset{a_1}{\text{argmax}} Q_1(\tau_1, a_1) \right) \ldots \left( \underset{a_n}{\text{argmax}} Q_n(\tau_n, a_n) \right)$
- No specific reward for each agent
- Implicit credit assignment through gradient backpropagation

[Sunehag et. al., 2017]
Learned Kiting Strategy in Starcraft II
Why VDN Works?

- Scalable maximization operator for action selection
  - Because of the consistency of individual-global maximization
- Parameter sharing among agents
- Implicit credit assignment

Cons:
- Not necessary for IGM
- Limited representation
QMIX: A Monotonic Mixing Network

- Monotonic function: $\frac{\partial Q_{tot}}{\partial Q_i} > 0$
- Weights of the mixing network are restricted to be non-negative

[Rashid et. al., 2018]
Representational Complexity

- QMIX’s representation is also limited
  - Monotonic condition is sufficient, but not necessary for IGM
- Sketch Illustration

![Ground Truth](image1)
![VDN](image2)
![QMIX](image3)
Empirical Results on Matrix Games

<table>
<thead>
<tr>
<th></th>
<th>$a_1$</th>
<th>$A^{(1)}$</th>
<th>$A^{(2)}$</th>
<th>$A^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A^{(1)}$</td>
<td></td>
<td><strong>8</strong></td>
<td>-12</td>
<td>-12</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(a) Payoff of matrix game.

(b) $Q_{tot}$ of VDN.

(c) $Q_{tot}$ of QMIX.
Three Modules of QPLEX

- Individual Action-Value Function
- Transformation
- Duplex Dueling

\[ Q_i(\tau, a_i) = w_i(\tau)Q_i(\tau, a_i) + b_i(\tau) \]
QPLEX: Duplex Dueling Mixing Network

- **Input:** \( \{Q_1(\tau, \cdot), \ldots, Q_n(\tau, \cdot)\} \)

- **Individual Dueling:**
  - \( V_i(\tau) = \max_{a_i'} Q_i(\tau, a_i') \)
  - \( A_i(\tau, a_i) = Q_i(\tau, a_i) - V_i(\tau) \)

- **Joint Dueling:**
  - \( V_{tot}(\tau) = \sum_{i=1}^{n} V_i(\tau) \)
  - \( A_{tot}(\tau, a) = \sum_{i=1}^{n} \lambda_i(\tau, a) A_i(\tau, a_i), \text{ where } \lambda_i(\tau, a) > 0 \)
  - \( Q_{tot}(\tau, a) = V_{tot}(\tau) + A_{tot}(\tau, a) = \sum_i V_i(\tau) + \sum_{i=1}^{n} \lambda_i(\tau, a) A_i(\tau, a_i) \)
Theorem: The joint action-value function class that QPLEX can realize is equivalent to what is induced by the IGM principle.

Individual-Global Maximization Principle:

$$\arg\max_{a} Q_{tot}(\tau, a) = \begin{pmatrix} \arg\max_{a_1} Q_1(\tau_1, a_1) \\ \vdots \\ \arg\max_{a_n} Q_n(\tau_n, a_n) \end{pmatrix}$$
QPLEX Realizes the IGM Constraint

- **Sufficient Condition for IGM**

\[
\begin{align*}
\arg\max_a Q_{tot}(\tau, a) &= \arg\max_a \sum_i V_i(\tau) + \sum_{i=1}^n \lambda_i(\tau, a)A_i(\tau, a_i) \\
&= \arg\max_a \sum_{i=1}^n \lambda_i(\tau, a)A_i(\tau, a_i) \\
&= (\arg\max_{a_1} A_1(\tau, a_1), ... , \arg\max_{a_n} A_n(\tau, a_n)) \quad (\forall i, \lambda_i(\tau, a) > 0, A_i(\tau, a_i) \leq 0, \text{ and } \max_{a_i} A_i(\tau, a_i) = 0) \\
&= (\arg\max_{a_1} Q_1(\tau, a_1), ... , \arg\max_{a_n} Q_n(\tau, a_n)) \quad (\forall i, Q_i(\tau, a_i) = V_i(\tau) + A_i(\tau, a_i)) \\
&= (\arg\max_{a_1} w_1(\tau)Q_1(\tau_1, a_1) + b_1(\tau), ... , \arg\max_{a_n} w_n(\tau)Q_n(\tau_n, a_n) + b_n(\tau)) \\
&= (\arg\max_{a_1} Q_1(\tau_1, a_1), ... , \arg\max_{a_n} Q_n(\tau_n, a_n)) \quad (\forall i, w_i(\tau) > 0)
\end{align*}
\]
QPLEX Realizes the IGM Constraint

- **Necessary Condition for IGM**

  for \( Q_{tot}(\tau, a) \), \( \exists \{Q_1(\tau_1, a_1), ..., Q_n(\tau_n, a_n)\} \), s. t.

  \[
  \text{argmax}_a Q_{tot}(\tau, a) = (\text{argmax}_{a_1} Q_1(\tau_1, a_1), ..., \text{argmax}_{a_n} Q_n(\tau_n, a_n))
  \]

  \[
  Q_{tot}(\tau, a) = V_{tot}(\tau) + A_{tot}(\tau, a)
  \]

  \[
  = \sum_i V_i(\tau) + \sum_{i=1}^n \lambda_i(\tau, a) A_i(\tau, a_i)
  \]

  \[
  (\exists \{Q_1(\tau, a_1), ..., Q_n(\tau, a_n)\}, \ \exists \lambda_i(\tau, a) > 0 \\ V_i(\tau) = \text{max}_{a_i} Q_i(\tau, a_i), \ \ A_i(\tau, a_i) = Q_i(\tau, a_i) - V_i(\tau))
  \]

  \[
  \forall Q_i(\tau, a_i), \ \exists w_i(\tau) > 0 \text{ and } b_i(\tau), \text{ s. t. } Q_i(\tau, a_i) = w_i(\tau) Q_i(\tau_i, a_i) + b_i(\tau)
  \]

  In QPLEX, \( \lambda_i(\tau, a), w_i(\tau) > 0 \) and \( b_i(\tau) \) are represented by neural networks, and thus \( Q_{tot}(\tau, a) \) can be realized by QPLEX with \( \{Q_1(\tau_1, a_1), ..., Q_n(\tau_n, a_n)\} \).
Representational Complexity

- Sketch Illustration

Ground Truth  VDN  QMIX  QPLEX
Empirical Results on Matrix Games

(a) Payoff of matrix game.

<table>
<thead>
<tr>
<th>$a_2$</th>
<th>$A^{(1)}$</th>
<th>$A^{(2)}$</th>
<th>$A^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1)}$</td>
<td>8</td>
<td>-12</td>
<td>-12</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>-12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>-12</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(a) $Q_{tot}$ of QPLEX.

<table>
<thead>
<tr>
<th>$a_2$</th>
<th>$A^{(1)}$</th>
<th>$A^{(2)}$</th>
<th>$A^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1)}$</td>
<td>8.0</td>
<td>-12.1</td>
<td>-12.1</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>-12.2</td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>-12.1</td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
</tbody>
</table>

(b) $Q_{tot}$ of VDN.

<table>
<thead>
<tr>
<th>$a_2$</th>
<th>$A^{(1)}$</th>
<th>$A^{(2)}$</th>
<th>$A^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1)}$</td>
<td>-6.5</td>
<td>-5.0</td>
<td>-5.0</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>-5.0</td>
<td>-3.5</td>
<td>-3.5</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>-5.0</td>
<td>-3.5</td>
<td>-3.5</td>
</tr>
</tbody>
</table>

(c) $Q_{tot}$ of QMIX.

<table>
<thead>
<tr>
<th>$a_2$</th>
<th>$A^{(1)}$</th>
<th>$A^{(2)}$</th>
<th>$A^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1)}$</td>
<td>-7.8</td>
<td>-7.8</td>
<td>-7.8</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>-7.8</td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>-7.8</td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
</tbody>
</table>
StarCraft II Benchmark: Online Learning

(a) 5s10z

(b) 1c3s5z

(c) 3s5z

(a) 1c3s8z_vs_1c3s9z

(b) 7sz

(c) 3s_vs_5z
StarCraft II Benchmark: Offline Learning

Data collected by a behavior policy learned by QMIX
Outline

- **Value-Based Methods**
  - Paradigm: Centralized Training and Decentralized Execution
  - Basic methods: VDN, QMIX, QPLEX
  - Theoretical analysis
  - Extensions

- **Policy Gradient Methods**
  - Paradigm: Centralized Critic and Decentralized Actors
  - Method: Decomposable Off-Policy Policy Gradient (DOP)
Theoretical Analysis on Linear Value Factorization

- **Linear Value Factorization**
  - Example methods: VDN, Qatten, ...

- **A theoretical analysis framework**
  - Multi-agent fitted Q-iteration

- **Analysis**
  - Implicit credit assignment
  - Convergence

\[
Q_{tot}(\tau, \alpha) \rightarrow \text{TD Loss}
\]

\[
\sum Q(t, a) = Q_{tot}(\tau, \alpha)
\]

\[
Q_1(\tau_1, a_1) \rightarrow \cdots \rightarrow Q_n(\tau_n, a_n)
\]

[Wang et. al., 2020]
Fitted Q-Iteration (FQI) Framework

- Multi-Agent MDP (MMDP): assume full observability
- Fitted Q-Iteration (FQI) for MMDP
  - Bellman optimality operator $\mathcal{T}$:
    $$(\mathcal{T}Q)_{tot}(s, a) = r(s, a) + \gamma \mathbb{E}_{s'} \left[ \max_{a'} Q_{tot}(s', a') \right]$$
  - Given a dataset $D = \{(s, a, r, s')\}$.
  - Iteratively optimization (empirical Bellman error minimization):
    $$Q^{(t+1)} \leftarrow \arg\max_Q \mathbb{E}_{(s, a, r, s') \sim D} \left[ \left( r + \gamma \max_{a'} Q^{(t)}_{tot}(s', a') - Q_{tot}(s', a') \right)^2 \right]$$
Basic Assumptions

- **Assumption 1: deterministic dynamics**
  - \( P(\cdot \mid s, a) \) is deterministic

- **Assumption 2: adequate and factorizable dataset**
  - Empirical probability is factorizable
  - \( p_D(a \mid s) = \prod_i p_D(a_i \mid s), \sum_{a_i} p_D(a_i \mid s) = 1, p_D(a_i \mid s) > 0 \)
Fitted Q-Iteration with Linear Value Factorization

- The action-value function class $Q^{LVD}$:
  - $Q^{LVD} = \{ Q \mid Q_{tot}(\cdot, a) = \sum_{i=1}^{n} Q_{i}(\cdot, a_{i}), \forall a \text{ and } [\forall Q_{i}]_{i=1}^{n} \}$

- Given an adequate and factorizable dataset $D$.

- Iteratively optimization framework:
  - $Q^{t+1} \leftarrow \arg\max_{Q \in Q^{LVD}} \sum_{(s,a)} p_{D}(a|s) \left( y^{(t)}(s, a) - \sum_{i=1}^{n} Q_{i}(s, a_{i}) \right)^{2}$
  - $y^{(t)}(s, a) = r + \gamma \max_{a'} Q_{tot}^{(t)}(s', a')$
Theoretical Analysis of Linear Value Factorization

Theorem 1. (Closed-form solution of FQI-LVD)

- A single iteration of empirical Bellman operator $Q^{(t+1)} = T_D^{LVD} Q^{(t)}$:
  - $Q_i^{(t+1)}(s, a_i) = \mathbb{E}_{a'_{-i}} [y^{(t)}(s, a_i \oplus a'_{-i})] - \frac{n-1}{n} \mathbb{E}_{a'} [y^{(t)}(s, a')] + w_i(s)$
  - $a_i \oplus a'_{-i} = a' = (a'_1, ..., a'_{-1}, a_i, a'_{i+1}, ..., a'_n)$
  - $\forall w = [w_i]_{i=1}^n$ s.t. $\sum_{i=1}^n w_i(s) = 0$

Proof sketch:

- regarded as a weighted linear least squares problem with $n|S||A|$ variables and $|S||A|^n$ data points.
  - $Q^{(t+1)} \leftarrow \arg\max_{Q \in Q^{LVD}} \sum_{(s,a)} p_D(a|s) \left( y^{(t)}(s, a) - \sum_{i=1}^n Q_i(s, a_i) \right)^2$
  - Construct a solution and use pseudo inverse to prove
Theorem (Closed-form solution of FQI-LVD)

A single iteration of empirical Bellman operator $Q^{(t+1)} = T_D^{LVD} Q^{(t)}$:

- $Q_i^{(t+1)}(s, a_i) = \mathbb{E}_{a'_{-i}}[y(t)(s, a_i \oplus a'_{-i})] - \frac{n-1}{n} \mathbb{E}_{a'}[y(t)(s, a')] + w_i(s)$
- $\forall w = [w_i]_{i=1}^n \text{ s.t. } \sum_{i=1}^n w_i(s) = 0$

1: Will be eliminated when considering the joint action-value function.

2: Do not change the greedy action selection.
Implicit counterfactual credit assignment mechanism of FQI-LVD

- A single iteration of empirical Bellman operator $Q^{(t+1)} = \mathcal{T}_{D}^{LVD} Q^{(t)}$:
  
  \[
  Q_{i}^{(t+1)}(s, a_{i}) = \mathbb{E}_{a'_{i}}[y^{(t)}(s, a_{i} \oplus a'_{-i})] - \frac{n-1}{n} \mathbb{E}_{a'}[y^{(t)}(s, a')] \\
  \text{Evaluation of } a_{i} \quad \text{Baseline}
  \]
Convergence Analysis of Linear Value Factorization

- Theorem 2. With uniform data distribution, there exists MMDPs, FQI-LVD diverges to infinity from any arbitrary initialization.

- Theorem 3. With $\epsilon$-greedy exploration, FQI-LVD has local convergence when $\epsilon$ is sufficient small.

- Multi-agent Q-learning with linear value decomposition structure requires on-policy samples to maintain numerical stability.
Offline Learning on Starcraft Benchmark

Data collected by a behavior policy learned by QMIX
Outline

- **Value-Based Methods**
  - Paradigm: Centralized Training and Decentralized Execution
  - Basic methods: VDN, QMIX, QPLEX
  - Theoretical analysis
  - Extensions

- **Policy Gradient Methods**
  - Paradigm: Centralized Critic and Decentralized Actors
  - Method: Decomposable Off-Policy Policy Gradient (DOP)
Challenges of Value Factorization Learning

- Uncertainty
  - Full value factorization ➞ miscoordination

\[ Q_{tot}(\tau, a) \]
\[ Q_1(\tau_1, a_1) \quad \ldots \quad Q_n(\tau_n, a_n) \]
Limitations of Full Value Factorization

- Can cause miscoordinations during execution
  - Need Communication!
Nearly Decomposable Q-Value Learning (NDQ)

- Allowing communication, but minimized
- Learn when, what, and with whom to communicate
NDQ Framework: Communication Optimization

Mutual Information Loss $\uparrow$

Entropy Loss $\downarrow$

Expressiveness

Succinctness

$Q_{tot}(\tau, a)$

$TD\ Loss$

Mixing Network

$Q_1(\tau_1, a_1, m_1^{in})$

$\cdots$

$Q_n(\tau_n, a_n, m_n^{in})$

Message $M_{ji}$

$[Wang\ et.\ al.,\ 2020]$
Learning Optimal Communication Protocol
Experiments: Micro-Management in Starcraft II

(a) 3b_vs_1h1m  (b) 3s_vs_5z  (c) lo2r_vs_4r
(d) 5z_vs_1ul  (e) 1o10b_vs_1r  (f) MMM

https://sites.google.com/view/ndq
SC2 benchmark: without Message Drop

3b vs 1h1m

3s vs 5z

1o2r vs 4r

5z vs 1ul

1o10b vs 1r

MMM
SC2 benchmark: 80% Messages Dropped

3b vs 1h1m

3s vs 5z

1o2r vs 4r

5z vs 1ul

1o10b vs 1r

MMM
Challenges of Value Factorization Learning

- **Uncertainty**
  - Full value factorization ➔ miscoordination

- **Complex tasks require diverse or heterogeneous agents**
  - Shared value network ➔ ineffective learning

\[
Q_1(\tau_1, a_1) \rightarrow Q_{tot}(\tau, a) \rightarrow \ldots \rightarrow Q_n(\tau_n, a_n)
\]
Why dynamic shared learning?

- Complex cooperative tasks require diverse behaviors among agents

- Learning a single shared policy network for agents\([1-4]\)
  - Lack of diversity and requiring a high-capacity neural network
  - May result in slow, ineffective learning

- Learning independent policy networks is not efficient
  - Some agents perform similar sub-tasks, especially in large systems

ROMA: Multi-Agent Reinforcement Learning with Emerging Roles

- Agents with similar roles have similar policies and share their learning
  - Similar roles ↔ similar subtasks ↔ similar behaviors

- Inferring an agent’s roles based on the local observations and execution trajectories

- Conditioning agents’ policies on their roles

- An agent can change its roles in different situations

[Wang et. al., 2020]
ROMA Framework

[Wang et. al., 2020]
The SMAC Challenge in Starcraft II

https://sites.google.com/view/romarl
Starcraft II: 27 Marines vs 30 Marines
Specialized Roles

(a) Strategy: sacrificing Zealots 9 and 7 to minimize Banelings’ splash damage.

(b) Strategy: forming an offensive concave arc quickly

(c) Strategy: green Zerglings hide away and Banelings kill most enemies by explosion.
Outline

- **Value-Based Methods**
  - Paradigm: Centralized Training and Decentralized Execution
  - Basic methods: VDN, QMIX, QPLEX
  - Theoretical analysis
  - Extensions

- **Policy Gradient Methods**
  - Paradigm: Centralized Critic and Decentralized Actors
  - Method: Decomposed Off-Policy Policy Gradient (DOP)
Value-Based Methods

▪ Significantly contribute to the recent progress of MARL.
  ▪ VDN, QMIX, QPLEX, QTRAN, Qatten, NDQ, ROMA, …

▪ Drawbacks:
  ▪ Lack of stability;
  ▪ Limited in discrete action spaces.

▪ Policy gradient methods hold a promise
▪ Paradigm: centralized critic with decentralized actors
Centralized Critic with Decentralized Actors

Actor 1

Critic

Actor 2

Environment

\(g_1^t\)

\(a_1^t\)

\(o_1^t\)

\(s_t\)

\(r_t\)

\(a_2^t\)

\(o_2^t\)

\(g_2^t\)
Centralized Critic with **Decentralized Actors**
Centralized Critic with Decentralized Actors

- Single-agent
  \[ g = \mathbb{E}_\pi \left[ Q^\pi(s, a) \nabla_\theta \log \pi_\theta(a|s) \right] \]

- Multi-agent (Centralized Critics with Decentralized Actors)
  \[ g = \mathbb{E}_\pi \left[ \sum_i Q^\pi(s, a) \nabla_\theta \log \pi_i(a_i|\tau_i) \right] \]

- Add a baseline to reduce variance
  \[ g = \mathbb{E}_\pi \left[ \sum_i (Q^\pi(s, a) - b(s, a_{-i})) \nabla_\theta \log \pi_i(a_i|\tau_i) \right] \]
  Independent of \( a_i \)
Counterfactual Baseline to Assign Credit

- Gradient: \( g = \mathbb{E}_\pi [\sum_i (Q^\pi(s, a) - b(s, a_{-i})) \nabla_\theta \log \pi_i(a_i | \tau_i)] \)

- A problem
  - \( Q^\pi(s, a) \) is an estimation of the global return;
  - Not tailored to a specific agent

- COMA method: counterfactual baseline:
  - \( b(s, a_{-i}) = \sum_{a_i} \pi_i(a_i | o_i) Q^\pi(s, a_{-i}, a_i) \)
  - Simultaneously achieving
    - Variance reduction
    - Credit assignment

[Foerster et. al., 2018]
Policy-Based Methods

- Centralized Critic with Decentralized Actors
  - COMA\[^6\] (stochastic policy gradients)
  - MADDPG\[^7\] (deterministic policy gradients)
  - Some extensions
    - MAAC\[^8\]

---

Unsatisfactory Performance
Why?

- On-policy learning of stochastic PG
  - E.g., COMA

- Lack credit assignment mechanism of deterministic PG
  - E.g., MADDPG, MAAC

- Centralized-Decentralized Mismatch (CDM)
  - Centralized critics introduce the influence of other agents.
Centralized-Decentralized Mismatch (CDM)

\[ g = \mathbb{E}_\pi \left[ \sum_i Q^\pi_{tot}(\tau, a_1, a_2, \cdots, a_n) \nabla_{\theta_i} \log \pi_i(a_i|\tau_i) \right] \]

- The joint critic introduces the influence from other agents.
  - Assume the optimal action under \( \tau \) is \( a^* = (a_1^*, \cdots, a_n^*) \).
  - It is possible that \( \mathbb{E}_{\pi_{-i}}[Q^\pi_{tot}(\tau, a_{-i}, a_i^*)] < 0 \), leading to the decrease of \( \pi_i(a_i^*|\tau_i) \). (e.g. when \( \pi_{-i} \) is suboptimal)
  - Suboptimality reinforces each other by propagating though the joint critic.
  - Leading to large variance
10 agents, 10 actions. Rewarded 10 if all agents take the first action; otherwise -10.
DOP: Off-Policy Decomposed Policy Gradient

- Introducing linearly decomposed critic
  - $Q^\pi_{tot}(\tau, \cdot) = \sum_i k_i(\tau)Q_i(\tau, \cdot) + b(\tau)$
  - where $k_i(\tau) > 0$

- Benefits:
  - Simple policy update rules
  - Tractable off-policy learning
  - Convergence guarantees
  - Addressing centralized-decentralized mismatch (CDM)
DOP Policy Gradient Theorem

Policy gradient theorem

\[ \nabla J(\theta) = \mathbb{E}_\pi \left[ \sum_i \nabla_\theta \log \pi_i(a_i | \tau_i) k_i(\tau) Q_i(\tau, a_i) \right] \]

Proof

\[ U_i(\tau, a_i) = Q^\pi_{tot}(\tau, a) - \sum_x \pi_i(x | \tau_i) Q^\pi_{tot}(\tau, (x, a_{-i})) \]
\[ = \sum_j k_j(\tau) Q_j(\tau, a_j) - \sum_x \pi_i(x | \tau_i) [\sum_{j \neq i} k_j(\tau) Q_j(\tau, a_j) + k_i(\tau) Q_i(\tau, x)] \]
\[ = k_i(\tau) [Q_i(\tau, a_i) - \sum_x \pi_i(x | \tau_i) Q_i(\tau, x)] \]

\[ \nabla J(\theta) = \mathbb{E}_\pi [\sum_i \nabla_\theta \log \pi_i(a_i | \tau_i) U_i(\tau, a_i)] \]
\[ = \mathbb{E}_\pi [\sum_i \nabla_\theta \log \pi_i(a_i | \tau_i) k_i(\tau)(Q_i(\tau, a_i) - \sum_x \pi_i(x | \tau_i) Q_i(\tau, x))] \]
\[ = \mathbb{E}_\pi [\sum_i \nabla_\theta \log \pi_i(a_i | \tau_i) k_i(\tau) Q_i(\tau, a_i)] \]
Off-Policy Learning

- Evaluate value functions using off-policy data:
  - Estimate $Q^\pi_{tot}(\tau, a)$ using data collected by a behavior policy $\beta$.

- Popular techniques in single-agent settings:
  - Importance sampling:
    - Require computing $\prod_i \frac{\pi_i(a_i|\tau_i)}{\beta_i(a_i|\tau_i)}$ in multi-agent settings.
    - The variance grows exponentially with the number of agents.
Off-Policy Evaluation with Linear Decomposition

- Evaluate value functions using off-policy data:
  - Estimate $Q_{tot}^{\pi}(\tau, a)$ using data collected by a behavior policy $\beta$.

- Popular techniques in single-agent settings:
  - Tree-backup (Sutton & Barto (2018 edition), section 7.5):
    - Require computing $\mathbb{E}_{\pi}[Q_{tot}^{\pi}(\tau, \cdot)]$ in multi-agent settings;
    - Need a summation for every joint action; the complexity is exponential;
    - Fortunately, using linearly decomposed critics, this expectation can be computed in **linear time**:
      $$\mathbb{E}_{\pi}[Q_{tot}^{\pi}(\tau, \cdot)] = \sum_{i} k_i(\tau)\mathbb{E}_{\pi_i}[Q_i(\tau, \cdot)] + b(\tau)$$
Other Properties of DOP

▪ Policy Improvement Theorem

Theorem 2. [Stochastic DOP policy improvement theorem] For any pre-update policy $\pi^o$ which is updated by Eq. 9 to $\pi$, let $\pi_i(a_i|\tau_i) = \pi_i^o(a_i|\tau_i) + \beta_{a_i,\tau}\delta$, where $\delta > 0$ is a sufficiently small number. If it holds that $\forall\tau, a'_i, a_i, Q_i^{\phi_i}(\tau, a_i) > Q_i^{\phi_i}(\tau, a'_i) \iff \beta_{a_i,\tau} \geq \beta_{a'_i,\tau}$, then we have

$$J(\pi) \geq J(\pi^o),$$

i.e., the joint policy is improved by the update.

▪ Attenuating Centralized-Decentralized Mismatch

Theorem 3. Denote r.v.s $g_1 = \nabla_{\theta_i} \log \pi_i(a_i|\tau_i; \theta_i) Q_i^{\phi_i}(\tau, a_i)$, $g_2 = k_i(\tau) \nabla_{\theta_i} \log \pi_i(a_i|\tau_i; \theta_i) Q_i^{\phi_i}(\tau, a_i)$, under any $\tau$ we have

$$\frac{\text{Var}_{\pi_i}(g_2)}{\text{Var}_{\pi}(g_1)} = O(\frac{1}{n}).$$ (11)
A similar theorem can be derived for the deterministic case:

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim D} \left[ \sum_i \nabla_{\theta} \pi_i(\tau_i) \nabla_{\theta} k_i(\tau) Q_i(\tau, a_i) | a_i = \pi_i(\tau_i) \right]$$

**Proof.** Drawing inspirations from the single-agent case [silver et. al, 2014].

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim D} \left[ \nabla_{\theta} Q_{tot} (\tau, a) \right]$$

$$= \mathbb{E}_{\tau \sim D} \left[ \sum_i \nabla_{\theta} k_i(\tau) Q_i(\tau, a_i) | a_i = \pi_i(\tau_i) \right]$$

$$= \mathbb{E}_{\tau \sim D} \left[ \sum_i \nabla_{\theta} \pi_i(\tau_i) \nabla_{\theta} k_i(\tau) Q_i(\tau, a_i) | a_i = \pi_i(\tau_i) \right]$$
DOP Deterministic Policy Gradient

- **Sufficient Representational Capability**

  **Theorem 5.** For $\forall \tau, a \in \{a \mid ||a - \pi(\tau)|| \leq \delta\}$, there are infinite tuples of feasible $Q_{i}^{\phi_{i}}(\tau, a_{i})$, s.t.

  $$|Q_{\text{tot}}^{\phi}(\tau, a) - Q_{\text{tot}}^{\pi}(\tau, a)| \leq 2Ln\delta = O(n\delta),$$

  where $Q_{\text{tot}}^{\phi}(\tau, a) = \sum_{i=1}^{n} k_{i}(\tau)Q_{i}^{\phi_{i}}(\tau, a_{i}) + b(\tau)$.

- **Attenuating Centralized-Decentralized Mismatch**

  **Theorem 6.** Denote r.v.s $g_{1} = \nabla_{\theta_{i}} \pi_{i}(\tau_{i}; \theta_{i}) \nabla_{a_{i}} Q_{\text{tot}}^{\phi}(\tau, a)$, $g_{2} = \nabla_{\theta_{i}} \pi_{i}(\tau_{i}; \theta_{i})k_{i}(\tau) \nabla_{a_{i}} Q_{i}^{\phi_{i}}(\tau, a_{i})$. Use $\mu_{i}$ to denote the distribution of $\alpha_{i}'$, which is the action of agent $i$ accompanied by an exploration noise $\epsilon \sim P_{\epsilon}$, and use $\mu$ to denote the joint distribution of all $\alpha_{i}'$. Under any $\tau$ we have:

  $$\frac{\text{Var}_{\mu_{i}}(g_{2})}{\text{Var}_{\mu}(g_{1})} = O\left(\frac{1}{n}\right).$$

(15)
State of the art on SC2 Benchmark

- Stochastic DOP (Ours)
- COMA
- QMIX
- DOP with Undecomposed Tree Backup
- On-Policy DOP

Graphs show the test win percentage over time for different scenarios:
- MMM
- 2s3z
- 2m_vs_1z
- 10m_vs_11m
- so_many_banelings
- 3s_vs_3z
Continuous Action Spaces

Multi-Agent Particle Environment (MPE)
Summary

- **Value-Based Methods**
  - Paradigm: centralized training with decentralized execution
  - Methods: VDN, QMIX, QPLEX, NDQ, ROMA

- **Policy Gradient Methods**
  - Paradigm: centralized critic and decentralized actors
  - DOP: off-policy decomposed multi-agent policy gradient

- **Take-away**
  - Value factorization or decomposition is very useful
  - Dynamic shared learning + communication for complex tasks
  - MARL plays a critical role for AI, but is at the early stage
Challenges in MARL

- Exploration (e.g., sparse interaction)
- Scalability (e.g., number of agents and large action spaces)
- Hierarchical learning (e.g., long horizon problems)
- Decentralized or semi-centralized training
- Non-stationary environments
- Adversarial environments
- Mixed environments (e.g., social dilemma)
- Communication emergence
- Theoretical analysis
- …
References

- Sunehag, et. al.. Value-decomposition networks for cooperative multi-agent learning. *(AAMAS 2018)*
- Rashid, et. al. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. *(ICML 2018)*
References