MULTI AGENT REINFORCEMENT LEARNING

Peter M. VanNostrand, 05/07/2020
CSE 410 Reinforcement Learning
Table of Contents

1. What is MARL?
2. MDPs and Stochastic Games
3. Environment Implementation
4. Experimentation
5. Results
Table of Contents

1. What is MARL?
2. MDPs and Stochastic Games
3. Environment Implementation
4. Experimentation
5. Results
Multi-Agent Reinforcement Learning (MARL)

- Form of Reinforcement Learning (RL)
  - Agent(s) learn to take actions that maximize a reward derived from the environment
- Includes multiple independent actors (agents)
  - Each agent's actions may change the environment
  - Changes to the environment could affect reward for all agents
- Agents may interact to maximize their reward
  - Intentional changes to the environment
  - Direct agent-to-agent communication
  - Cooperation vs competition
Table of Contents

1. What is MARL?
2. MDPs and Stochastic Games
3. Environment Implementation
4. Experimentation
5. Results
Markov Decision Processes

- Typical RL problem
- Characterized by a Markov Decision Process (MDP)
- MDP Parameters
  - $S$ – set of possible states in the environment
  - $A$ – set of possible actions the agent can take
  - $R$ – reward function
  - $P$ – state to state transition probability based on action
Stochastic Games

- Combination of MDP with a repeated game
  - Agents must account for actions of other agents
- Stochastic Game Parameters
  - $\mathcal{N}$ agents
  - $A_i$ - action set for agent $i$
  - $A$ – combined action
    - $a_1 \times a_2 \times \cdots \times a_n$
  - $R$ – rewards function for $A$
MARL Challenges

• Moving Target
  • Reward for each agent can be affected by actions of other agents
  • Agent’s action change over time
  • Action-reaction loops can cause rewards to fluctuate

• Reward attribution
  • Which action(s) from which agent(s) led to states with large rewards
  • Impossible to maximize each agent independently

• Curse of dimensionality
  • More agents can increase observation and action spaces
  • Complexity grows exponentially with space and state dimensions
Table of Contents

1. What is MARL?
2. MDPs and Stochastic Games
3. Environment Implementation
4. Experimentation
5. Results
Multi Agent Grid-World

- Square grid-world of size $s \geq 6$
- Support $n$ agents, $n \in \{2, 3, 4\}$
  - Process $n$ actions per timestep
  - Return $n$ unique rewards
- Observation space
  - Current agent position $[row, col]$
- Action Space
  - 5 possible actions
  - Move up/right/down/left, don’t move
- Enable agent-to-agent communication
  - Agents should be able to share their location with other agents
Task: Enemy Containment

- $n$ agents coordinate to “contain” a static enemy
  - View environment as city streets
  - Location of enemy is neighborhood infected with a virus
  - Agents learn the best location to place testing stations
- Enemy is contained when surrounded on all sides, game terminates
- Reward Function
  - $-2$ if $d_{t+1} \geq d_t$
  - $-1$ if $d_{t+1} < d_t$
  - $0$ if $d_{t+1} = 1$
Table of Contents

1. What is MARL?
2. MDPs and Stochastic Games
3. Environment Implementation
4. Experimentation
5. Results
Agents

• Compare performance of two types of agents
  • Tabular Q-Learning (TQ)
  • Deep Q-Network (DQN)

• Agent Inputs
  • Observation: Current location
  • Communication: \( n - 1 \) other agent locations

• Agent Outputs
  • Next action

• For size \( s \) with \( n \) agents and \( a \) actions
  • Q-table size: \( s^2 \times n^2 \times a \)
  • DQN input layer of size \( 2n \)
    • 2 Dense ReLU layers of size 48
Training

- All agents trained for 1000 epochs
  - $\gamma = 0.95$
  - $\alpha = 0.05$
- Exponential $\epsilon$ decay
  - $\epsilon_0 = 1.0$
  - $\epsilon_{min} = 0.03$
  - $\delta = 0.005$
- At each iteration we recorded the score and learned movements for each agent
  - Score = total cumulative reward for one epoch
  - Learned movements recorded with $\epsilon = 0$, greedy
Results

• Results for $n = 2$
• Plots of score with averaging over a sliding window of size 10
• Final learned path
  • Agents shown in yellow and green
  • Enemy shown in blue
• Both TQ and DQN agents converge to the same optimal path
• DQN agents learn much more quickly, but with slightly more noise
Results $n = 3$

Tabular

DQN

Results $n = 4$

Tabular

DQN
Analysis

• In all cases $n = 2, 3, 4$ the tabular and DQN agents learned a path from their starting locations to a containment position
  • DQN achieved the same paths, but with less training time
    • Scores reach optimal values at earlier epoch
    • Difference of ~400 epochs for $n = 3, 4$
  • Improvement likely due to curse of dimensionality
    • Tabular agents must visit and learn each of $s^2 \times n^2 \times a$ values independently
    • DQN agents update weights on every iteration
      • Use same weights to predict Q-Values for every state-action pair
      • Allows DQN agents to generalize to unseen states
      • Greatly improves training sample efficiency

TQ an DQN scores, $n = 4$
THANK YOU! QUESTIONS?
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris vehicula a dui in neque dignissim, in aliquet nisl varius. Sed a erat ut magna vulputate feugiat. Quisque varius libero placerat erat and lobortis congue. Integer a arcu vel ante bibendum et scelerisque.
Graphic elements

Copy and paste these graphic elements to give your presentation a touch of color. Only use the official UB brand color palette. For more information, please visit [www.buffalo.edu/brand/creative/color/color-palette](http://www.buffalo.edu/brand/creative/color/color-palette).

neque dignissim, and in aliquet nisl et umis varius.

Note: neque in dignissim, and quet nis et umis varius.