Multi-Agent Learning

Scaling up to High-Dimensional States.

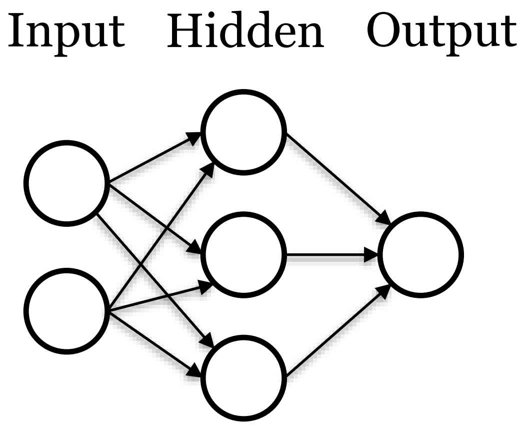
Multi-Agent Deep Reinforcement Learning

Deep Reinforcement Learning

- The tabular approach to reinforcement learning does not scale well to real-world problems
- An alternative is to approximate the policy or value function
  - Hopefully the agent can generalize and handle unseen states
  - Guarantees of optimality will no longer hold
- The most common function approximator is a neural network

Neural Networks

- Fully Connected Net
  - ”Basic” neural network
- Convolutional Neural Network (CNN)
  - Capture spatial relations
  - E.g., image analysis
- Recurrent Neural Network (RNN)
  - Capture temporal relations
  - E.g., for handling partial observability
- See, e.g., https://www.deeplearningbook.org/



Types of Reinforcement Learning Algorithms

- Value-based
  - Learn value function and use for action selection
  - E.g., Deep Q Networks (DQN)
- Policy-based
  - Learn policy directly, without learning a value function
  - E.g., REINFORCE
- Actor-Critic
  - Learn value function (critic) and use it to guide updates of policy (actor)
  - E.g., Asynchronous Advantage Actor-Critic (A3C)

Deep Q Networks (DQN)

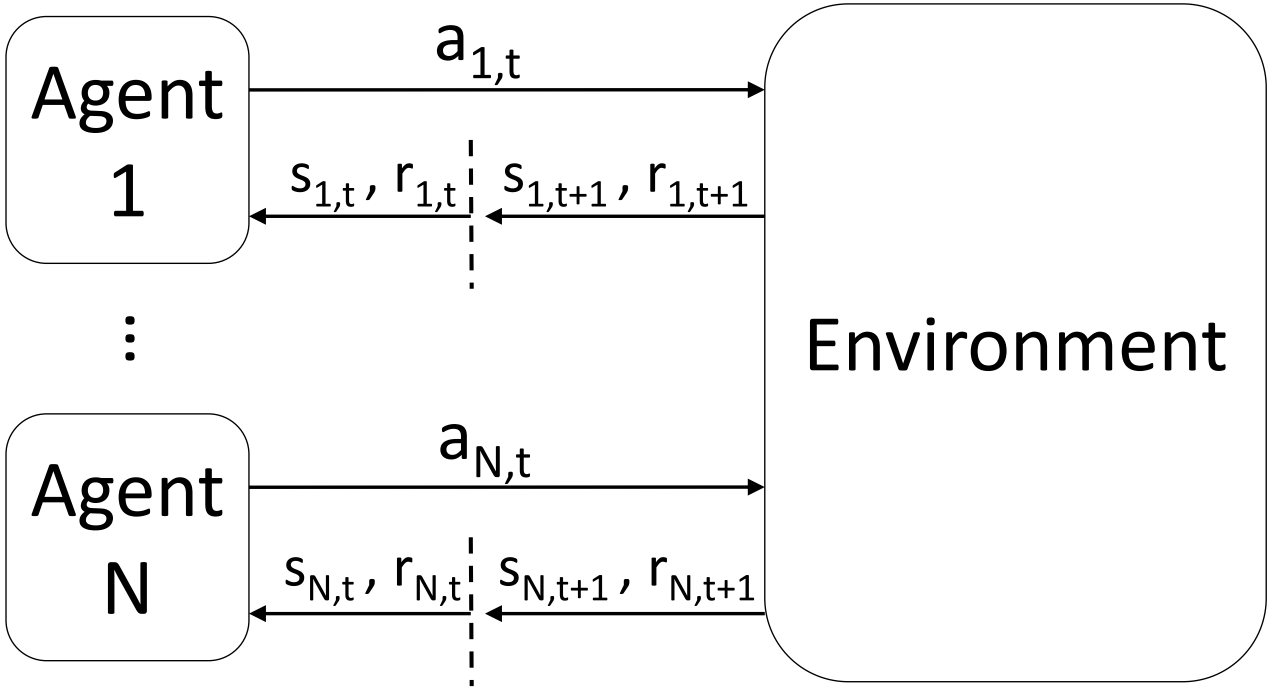
- Deep RL version of Q-learning, evaluated on Atari
  - Uses neural network to approximate Q function
- Instead of updating the (approximate) Q function in every step
  - Store data from experiences with the environment in replay buffer
  - Sample (replay) batch of experiences periodically to train neural network
- Avoids overfitting to the most recently seen interactions with the environment

Other Examples of Deep RL Algorithms

- DDPG: Actor-critic method for learning deterministic policies with continous actions
- A3C: Asynchronous actor-critic method for parallell learning in multiple environments, for improved performance
- UNREAL: Extension of A3C with auxiliary tasks to stabilize learning

Multi-Agent Deep Reinforcement Learning

- Modification of single-agent algorithms
- Approaches
  - Centralized learning and execution with factored action space
  - Fully decentralized learning
  - Centralized learning, decentralized execution

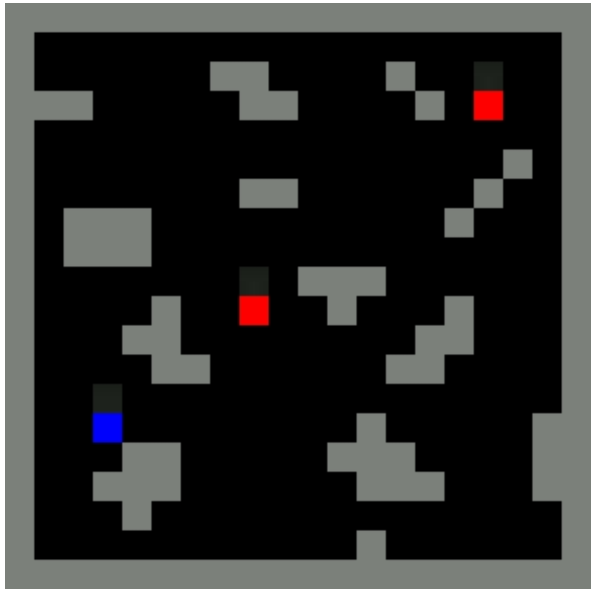


Decentralized Multi-Agent Deep Reinforcement Learning

- Though no theoretical guarantees exist, single-agent algorithms may produce interesting results in multi-agent systems
- Ways to stabilize the learning process
  - Clever design of reward systems
  - Training populations of agents
    - Can allow agents to generalize

Example: RL in Sequential Social Dilemas

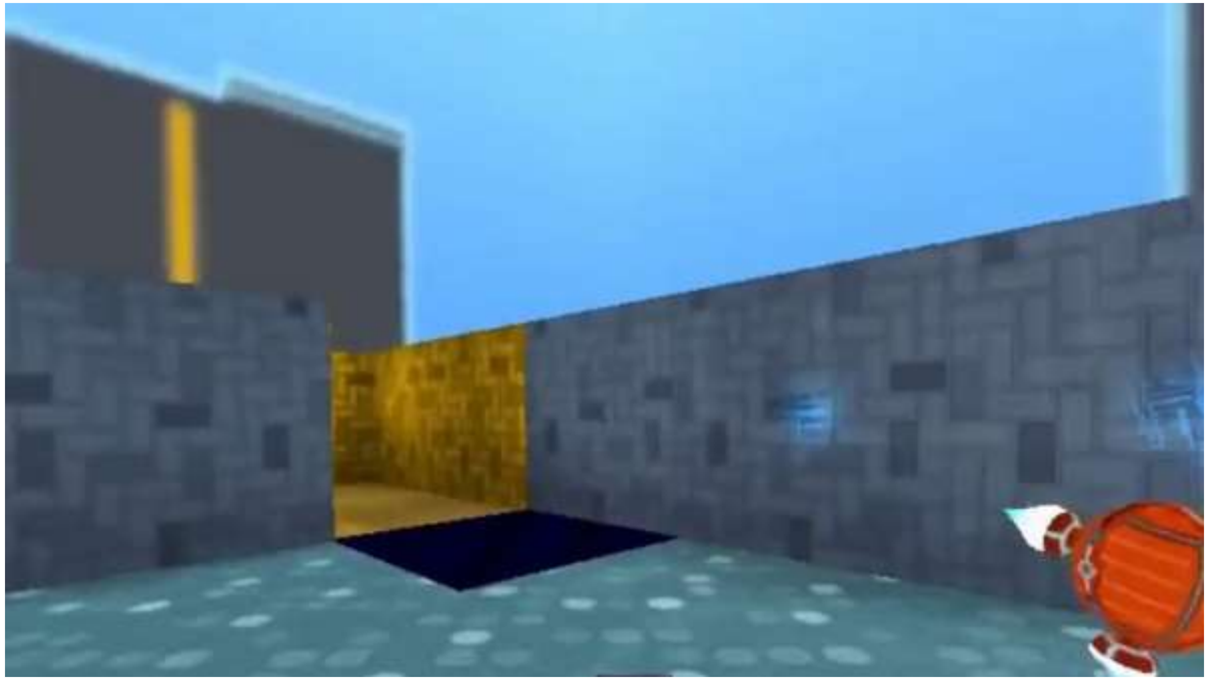
- Agents based on DQN
- Reward proportional to number of hunters in proximity of prey when captured
- Learn to hunt in pack or wait for other hunter to arrive before capturing prey
- Authors: Leibo et al. (2017)



Source: DeepMind

Example: Capture the Flag

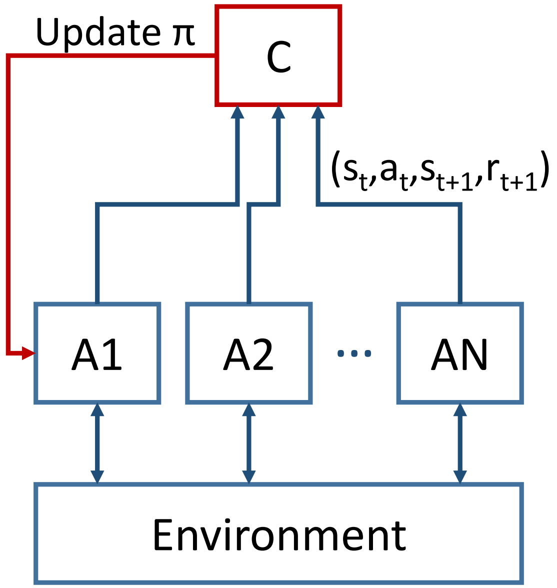
- Agent based on UNREAL architecture
- Population-based training with random teams playing random maps
  - Agents learn to cooperate with human-like strategies
    - Move in teams
    - ”Base camping”
- Authors: Jaderberg et al. (2018)



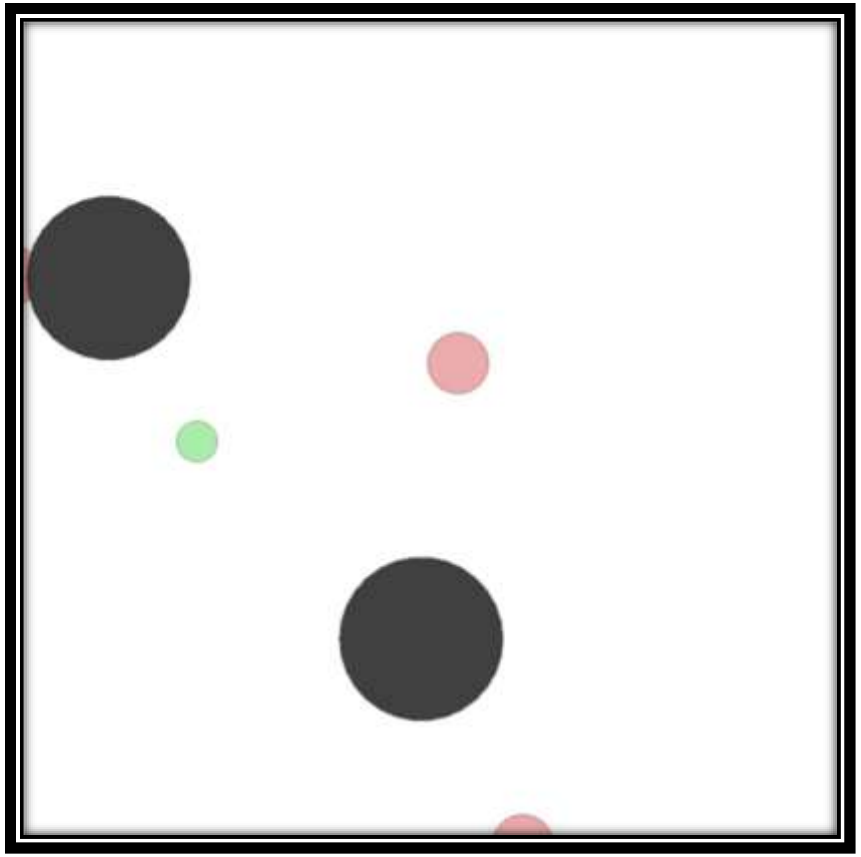
Source: DeepMind

Centralized Learning, Decentralized Execution

- Extra information is used for guidance during learning, e.g., actor-critic setup or value function decomposition
- At execution time agents act based on local observations
- Examples of algorithms
  - QMIX
  - COMA
  - MADDPG



Example - MADDPG

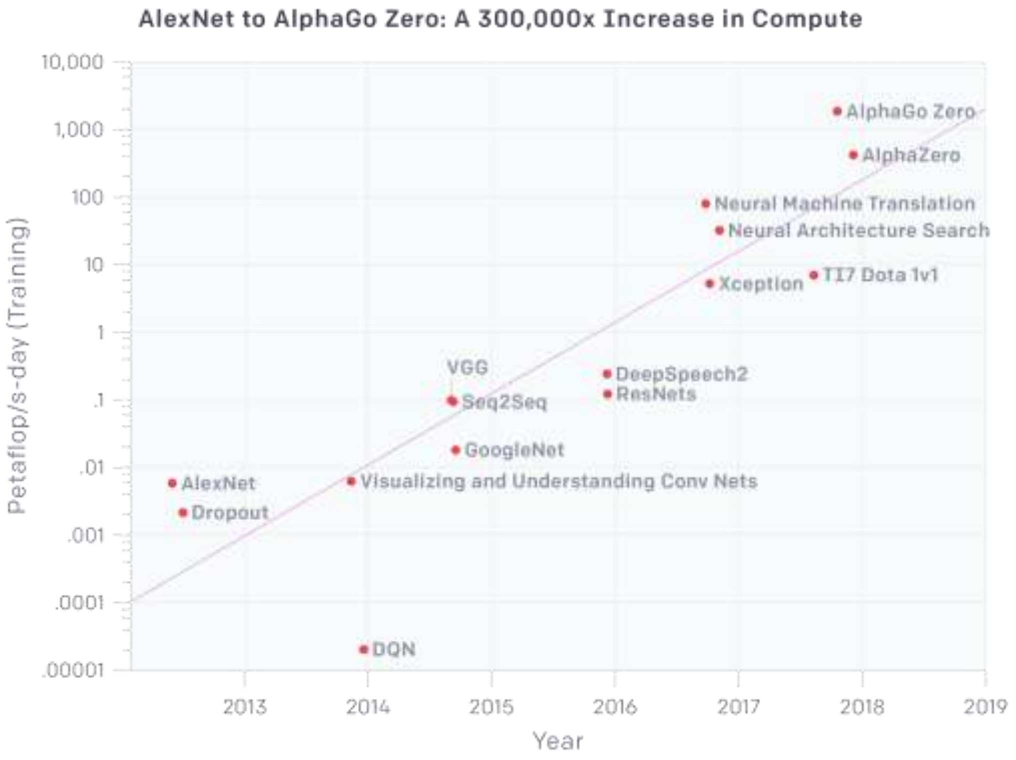


Moving to Increasingly Challenging Environments

- Increase in duration and number of moves (source: OpenAI)
  - Chess ~40 moves per game
  - Go: ~150 moves per game
  - Dota-RTS: ~20000 moves per game (45 min)
- Difficult to define frequent rewards, and therefore difficult to explore the state and action spaces to find an efficient policy

Challenges in Multi-Agent Learning

- Computational complexity
  - AlphaGo Zero (per agent):
    - 64 GPUs & 19 CPUs
  - OpenAI Dota Five
    - 256 GPUs & 128000 CPUs
- Lack of good benchmarks
- Reproducability



Source: OpenAI

Deep Learning for Modeling other Agents

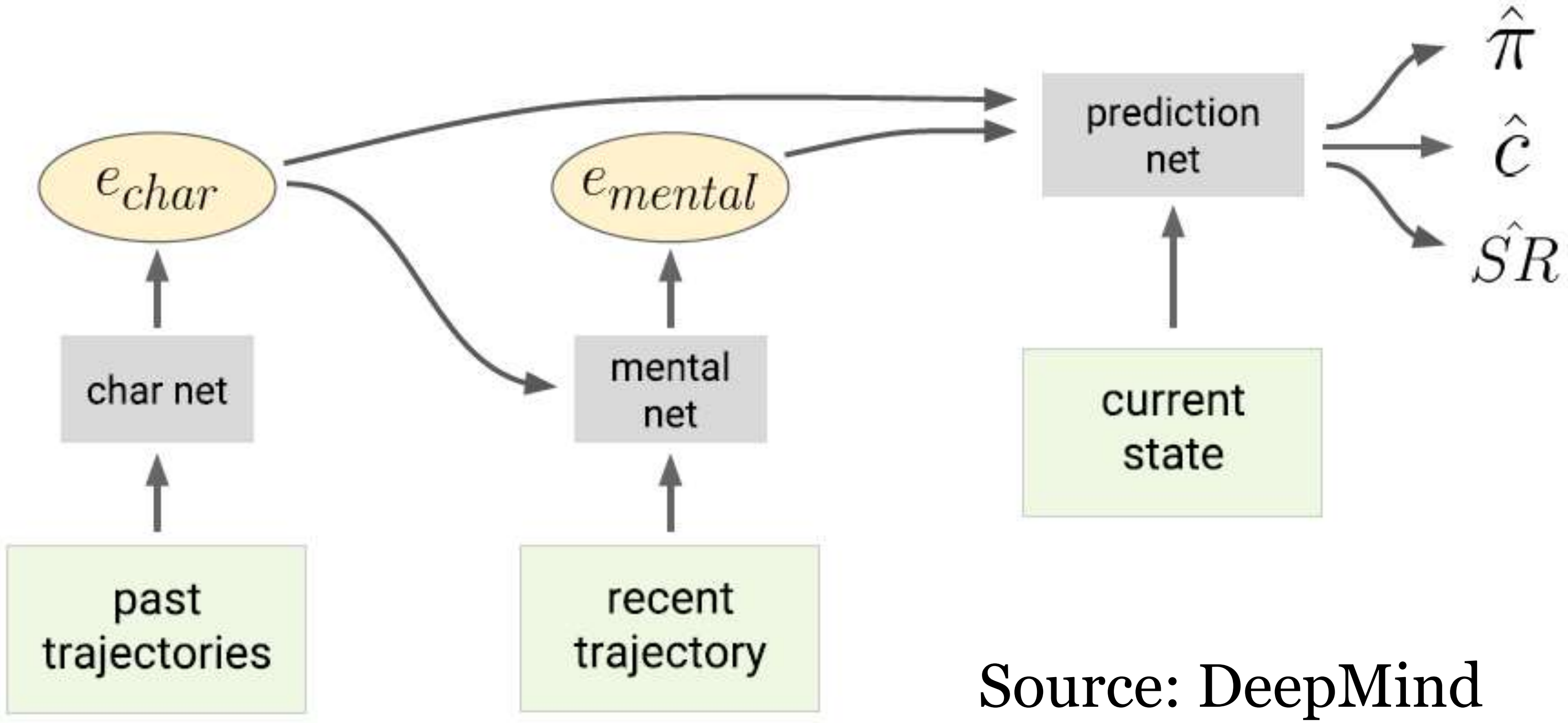
Overview

- Explicit models of other agents can support decision making in MAS
  - Provide more abstract input to learning algorithms
  - Guide planning algorithms, e.g. MCTS
- Models can be built based on recorded data or online
- Example: AlphaGo, AlphaStar

Example: Machine Theory of Mind

- *Theory of mind (ToM) broadly refers to humans' ability to represent the mental states of others, including their desires, beliefs, and intentions.*
- Machine Theory of Mind
  - Seeks to build a system which learns to model other agents, a Machine Theory of Mind, focusing on the problem of how an observer could learn autonomously how to model other agents using limited data
- Authors: Neil C. Rabinowitz, Frank Perbet, H. Francis Song, Chiyuan Zhang, S. M. Ali Eslami, and Matthew Botvinick (DeepMind & Google Brain)

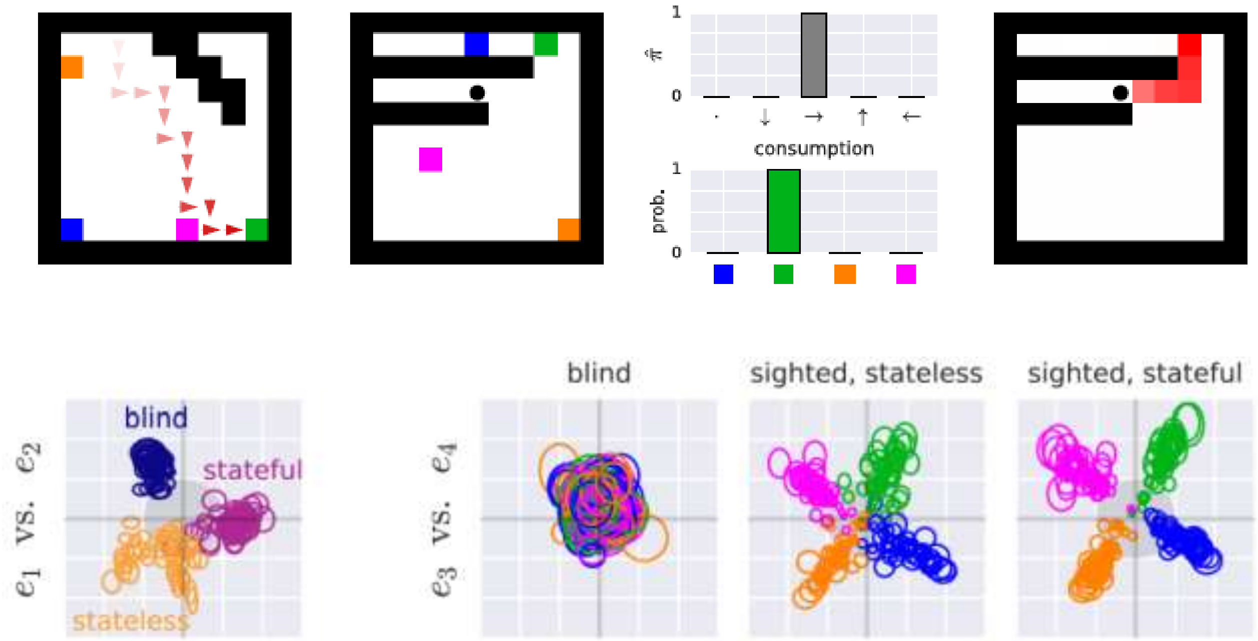
Example: Machine Theory of Mind



Source: DeepMind

Example: Machine Theory of Mind

- Evaluted on deep reinforcement learning agents with diverse characteristics
- Learns to predict
  - Goals/Actions
  - Beliefs
  - Successor states
- Embedding space clusters agents with different chars.



Source: DeepMind

www.liu.se