The City College of New York

CSC 59866-E: Senior Project I
AI Agents for Decision Making in the Real World

By Saptarashmi Bandyopadhyay
Email: sbandyopadhyay@ccny.cuny.edu, sbandyopadhyay@gc.cuny.edu
Assistant Professor of Computer Science
City College of New York and Graduate Center at the City University of New York

Deep Learning, Reinforcement Learning (RL) & Multi-Agent Deep Reinforcement Learning (MARL): REINFORCE, Independent Q Learning and Convergence

Today's Agenda

Sample Complexity, Generalizability, IQL, Actor-Critic, and MADDPG

Imitation Learning

- Behavioral Cloning
- DAgger

Sample Complexity, Generalizability, IQL, Actor-Critic, and MADDPG

Sample Complexity and Generalizability

Sample Complexity: How much data (environment interactions) does an algorithm need to learn a good policy?

- *Pure RL (DQN / REINFORCE):* High sample complexity. Requires millions of trial-and-error interactions to learn from scratch.
- *Imitation Learning (Covered later today):* Low sample complexity. Learns very fast *if* you have a good expert to demonstrate.

Generalizability: Can the agent handle Out-Of-Distribution (OOD) states?

- *Deep RL:* High generalizability. Because the agent naturally explores the state space during training through trial and error, it learns how to recover from mistakes.

From Exponential to Polynomial Complexity

Tabular (Exponential):

- In standard Q-Learning, the number of states and actions grows exponentially with the environment's complexity.
- For N agents, the table size is $O(|S|^N \times |A|^N)$.
- To fill this table, the **Sample Complexity is Exponential**. The agent must visit almost every possible combination to learn effectively.

Deep RL (Polynomial):

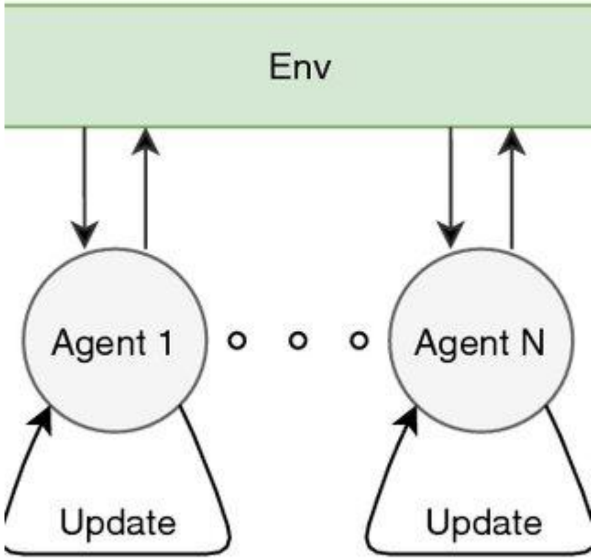
- By replacing the table with a Neural Network, we no longer need to visit every state.
- The network learns underlying features (e.g., "being near a wall is bad") and generalizes that knowledge to unseen states.

Recall: Independent Q Learning

Mechanism:

- Each agent treats all other agents as part of the background environment.
- Agent *i* observes its local state *o_i*, takes action *a_i* , and updates its own network *θ_i* using standard DQN.

The Appeal: Highly scalable, fully decentralized, and incredibly easy to implement.
The Reality: It mathematically violates the core assumption of Q-Learning (The Markov Property).



Convergence Problems

Non-Stationarity: Because Agent A is updating its policy at the exact same time Agent B is updating its policy, the environment's transition dynamics are constantly shifting from the perspective of either agent.

Failure to Converge:

- In single-agent RL, Q-learning is mathematically proven to converge to an optimal policy.
- In MARL, IQL often results in **Limit Cycles** (endless oscillation of strategies) or gets stuck in highly sub-optimal Nash Equilibria.

The "Moving Target": An agent cannot accurately calculate the Expected Return (*G_t*) of an action if the other agents keep changing how they react.

Actor-Critic

Instead of *only* learning a policy or *only* learning a value function, can we combine the best of both worlds? Yes we can!

In Actor-Critic Methods we train two networks:

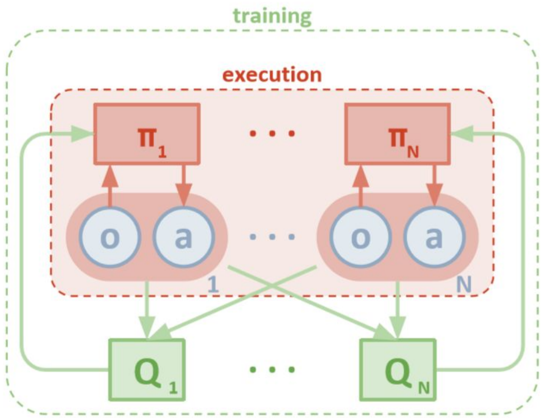
1. The **Actor** network chooses the action. This is similar to the policy network we used in REINFORCE
2. The **Critic** network evaluates the action chosen by the actor with a value, akin to the Q-Networks we learned in Deep Q Learning

Multi-Agent Deep Deterministic Policy Gradients

The Algorithm: The gold standard for continuous-action MARL. It is an extension of the Actor-Critic framework utilizing **CTDE**.

- **Decentralized Actors:** Each agent has its own Actor network that only sees local observations (used during execution).
- **Centralized Critic:** During training, the Critic is given access to the observations and actions of *all* agents simultaneously.

The Result: Because the Central Critic sees what everyone is doing, the environment is no longer non-stationary! It can provide accurate, stable gradients to train the local Actors to converge.



Imitation Learning

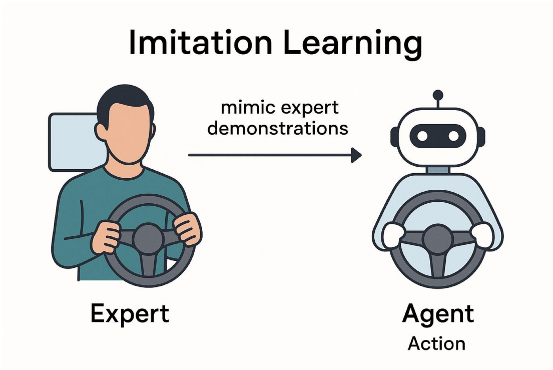
Imitation Learning

We saw that ALOHA uses Imitation Learning, not Reinforcement Learning. But what is Imitation Learning?

It is often extremely difficult or sometimes even impossible to train in the environment purely using traditional Reinforcement Learning.

Imitation Learning gives us a way to learn from *expert demonstrations*:

- Collect data from rollouts of an *expert policy*
- *Imitate* the expert as best you can!



Behavioral Cloning

The simplest algorithm for Imitation learning is **Behavioral Cloning:**

$\xi \in \Xi$	trajectory in expert demonstrations
$x \in \xi$	state along a trajectory
L	loss function (e.g., Euclidean norm, KL divergence, Cross-Entropy)

$$\hat{\pi}^* = \arg\min_{\pi} \sum_{\xi \in \Xi} \sum_{\boldsymbol{x} \in \xi} L(\pi(\boldsymbol{x}), \pi^*(\boldsymbol{x})).$$

Behavioral Cloning

The simplest algorithm for Imitation learning is **Behavioral Cloning:**

With this setup, we can think of Imitation Learning as a *supervised learning* problem!

- Input: x state encountered along expert trajectories
- Output: a action chosen by our policy network
- Target: a* action chosen by the expert for x

$$\hat{\pi}^* = \arg\min_{\pi} \sum_{\xi \in \Xi} \sum_{x \in \xi} L(\pi(x), \pi^*(x)).$$

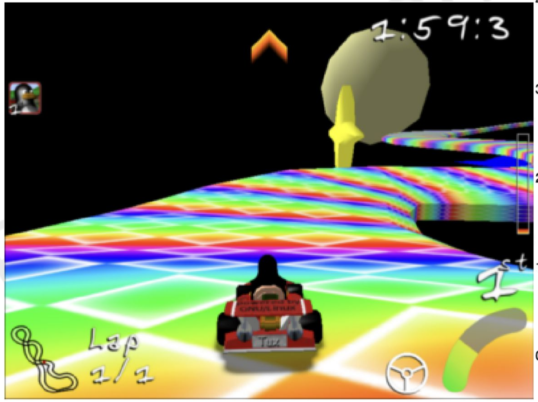
Limitation: Out-of-distribution generalization

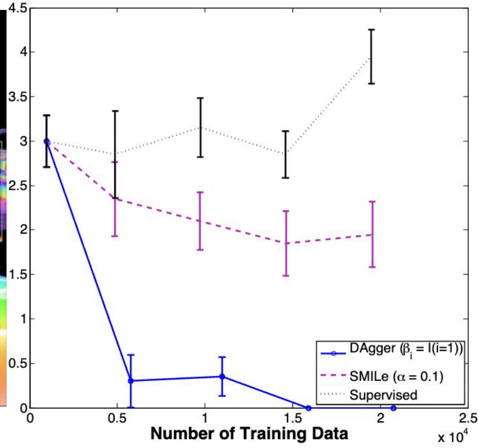
- If the expert doesn't explore the entire state space, our learned policy will have blind spots!

Dataset Aggregation

DAgger is an algorithm for training ***deterministic stationary policies*** in an imitation learning setting.

1. **Initialize the policy with the expert's decisions and do a rollout**
2. **Add encountered states never seen by the expert to the training dataset**
3. **Train the policy on the new, aggregated dataset**





DAgger performs very well against Supervised Learning on Super Tux Kart

This results in a policy robust to deviations from the expert's choices.

Online no-regret reduction to fix covariate shift in IL

Strengths

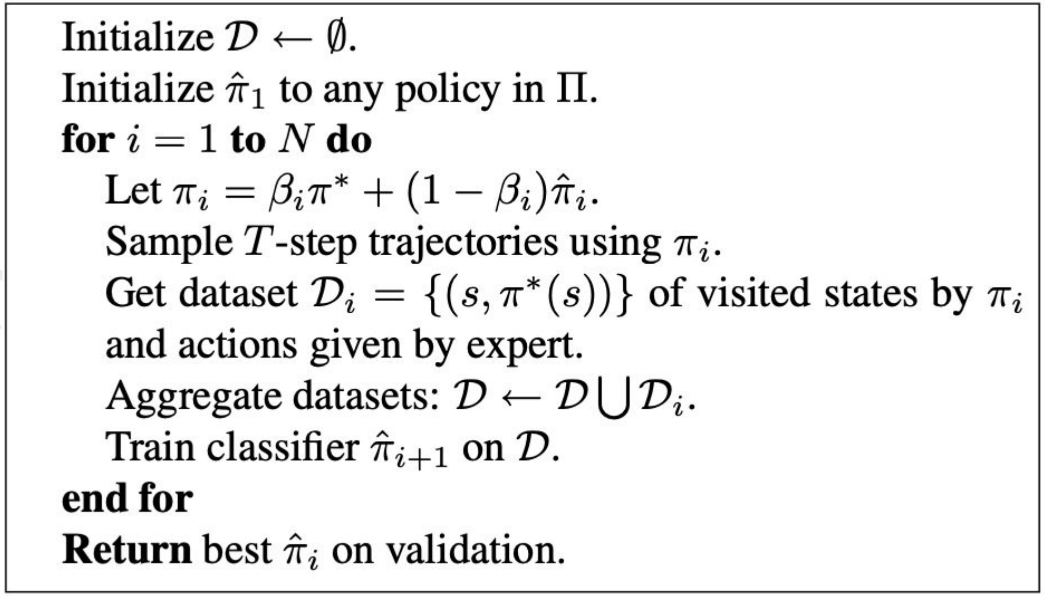
1. Strong theoretical guarantees for efficiency
2. Augmenting the data means you can still use any training procedure you want, and it will still improve your policy
3. Simple to understand and implement

Weaknesses

1. Over-reliance on querying the expert, which for many problems is impractical
2. In order to learn, the policy needs to interact with the environment directly, which could be unsafe for real-world problems (e.g. autonomous vehicles)

Algorithm

- Dataset D
- Policy pi
- State s
- Hyperparameter Beta - Controls how much the expert policy is imitated



```
Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in $\Pi$.
for $i = 1$ to $N$ do
    Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
    Sample $T$-step trajectories using $\pi_i$.
    Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by $\pi_i$
    and actions given by expert.
    Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
    Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.
```

Algorithm 3.1: DAGGER Algorithm.

Project Assignments

AI Conference Workshop Targets

ACL (Assoc. for Computational Linguistics): Perfect for the groups working on LLM Resource Allocation and NLP Agents.

CVPR & WACV (Vision): The ideal targets for our Vision-Language Model (VLM) and AR groups.

CHI (Human-Computer Interaction): Great for projects focusing on Imitation Learning, Human-AI teaming, or Behavioral Cloning safely.

IEEE S&P (Security & Privacy): The gold standard for groups working on Fault-Tolerant or Byzantine-resilient consensus agents.

DYSPAN (Dynamic Spectrum Access): Highly relevant for the edge computing and distributed wireless network projects.

Project Groups & Topics

Protocol:

1. I will display the 10 Groups.
2. Find your partner immediately after class.
3. **Task 1:** Exchange emails/Discord handles/Your preferred way to communicate.
4. **Task 2:** Set up a weekly meeting time outside of class.

Groups 1-5

- **Adaptive Resource Allocation for LLM Agents in Edge Networks**
  - Arsh Anand & Farida Balogun
- **Benchmarking Multi-Agent Reinforcement Learning (MARL) in Augmented Reality**
  - Karthikeya Reddy Basavanagoudgari & Hamim Choudhury
- **Resilient Data Routing Algorithms for NextGen Multi-Agent Connected Autonomous Vehicle Networks**
  - Marcley Colin & Alhassana Diallo
- **AI Agents for Distributed Chip Design**
  - Johir Hossain & Linwei Zheng
- **Standardized Benchmarking of Multi-Agent Distributed Machine Learning in Augmented Reality**
  - Parthkumar Joshi & Alexis Juarez Gomez

Groups 6-10

- **Fault-Tolerant Multi-Agent Systems for Vision-Language Models**
    - Fardin Khandaker & Xin Fan Li
- **Multi-Agent MIMO Strategies for Robust Communication in Distributed Wireless Networks**
    - Yuki Li & Minning Liu
- **Robust Multi-Agent Communication in Wireless MIMO Networks**
    - Mahdi Mahin & Amir Nabiyev
- **Interpretable Multi-Agent Coordination for Urban Mobility**
    - Sajid Patwary & Kenneth Romero Linares
- **Energy-Efficient Protocols for Distributed AI on Edge Devices**
    - Sri Suvetha Meenaa Subramanian & Atif Tausif

Questions?