Graph Convolutional Imitation and Reinforcement Learning in Multi-Robot Systems

Parth Khopkar, Computer Science
Mentor: Heni Ben Amor, Assistant Professor
School of Computing, Informatics, and Decision Systems Engineering

Introduction

The process of creating control strategies for a group of agents collaborating on tasks is a complex challenge. This work presents an approach for learning agent behavior from observation of swarm systems and improving upon this behavior by using Reinforcement Learning. We use a Graph Neural Network (GNN) policy and show its potential in uncovering low-level interaction dynamics of the system. We show that it is possible to investigate the individual representation of the low-level dynamics and benefit from their fine control to learn desired behaviors.

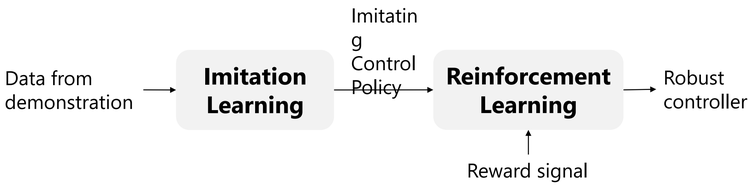

Fig. 1 Visualization of our approach

Network Architecture

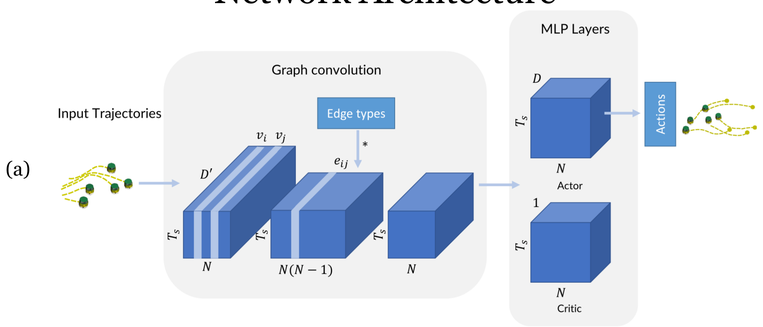

Fig. 2 (a) Network architecture (b) Example of the Graph Convolution operation

$$e_{ji}^t = \phi^e(v_j^t, v_i^t)$$
$$\bar{e}_i^t = \sum_{j \in \mathcal{N}_i} e_{ji}^t$$
$$h_i^t = \phi^v(x_i^t, \bar{e}_i^t)$$
$$y_i^t = \psi(h_i^t)$$

Methodology

Our approach is to train a centralized Graph Neural Network (GNN) based controller for a system with N agents with state $S = \{x_1^t, x_2^t, \dots x_N^t\}$ at timestep t that are present in a 2D environment along with obstacles O and goal location G. The different types of interactions, namely from agents, obstacles and the goal to agents are assigned separate function approximators: $\phi_{o\to a}^e, \phi_{g\to a}^e, \phi_{a\to a}^e$ and are modeled as MLPs.

Imitation Learning: Using the observed demonstrations of a multi-agent system based on the Boid model[3], we train the controller to clone this behavior by minimizing the mean squared error (MSE) between the output of the controller and the observations. This allows the controller to learn behaviors such as cohesion, collision avoidance, obstacle avoidance and goal seeking.

Reinforcement Learning: Since the behavior learned from Imitation Learning cannot be guaranteed to be collision-free in a real-world environment, we use RL to further tune the controller based on reward signals that the agents receive from the environment. These reward signals include inter-agent, obstacle-agent and goal-agent distances with a large negative reward in case of a collision.

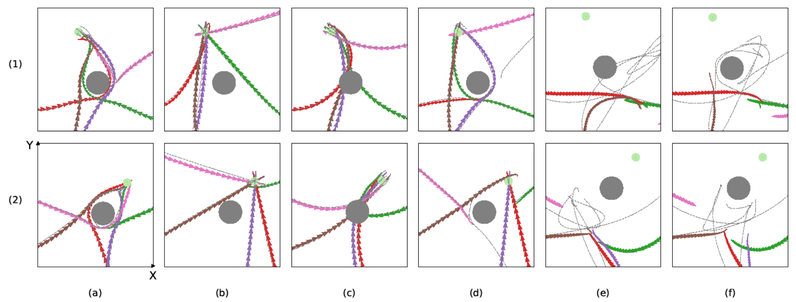

Fig. 3 Two sample trajectories showing function separation in the GNN. The colored arrows show the predicted trajectories of agents which try to move toward the goal represented by a green dot while trying to avoid the gray obstacle. (a) Trajectories generated by the Boid model (b) $\phi_{g\to a}^e$ present (c) $\phi_{g\to a}^e$, $\phi_{a\to a}^e$ present (d) $\phi_{g\to a}^e \phi_{o\to a}^e$ present (e) $\phi_{a\to a}^e$ present (f) $\phi_{a\to a}^e, \phi_{o\to a}^e$ present

Results

For all IL + RL modes, we first train the controller on 10,000 demonstrations obtained from the Boid model and then train the controller using RL in different modes:
Mode 0 – All parameters free to be trained
Mode 1 – $\phi_{g\to a}^e$ function fixed
Mode 2 – All functions except $\phi_{a\to a}^e$ and $\phi_{o\to a}^e$ fixed

All IL + RL modes outperform the case when only RL is used and have a head start when RL training begins. By turning off training for specific functions, we are taking advantage of the separation of functions which control interactions.

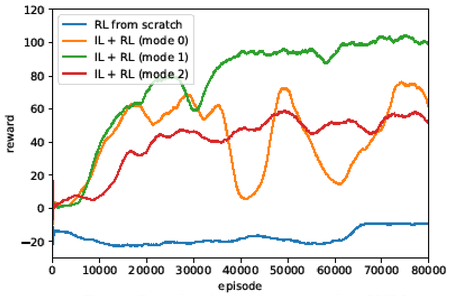

Fig. 4 Results from training the GNN based controller on IL + RL

Conclusion

We have shown that a GNN based controller is useful in uncovering group dynamics as it exploits relational inductive biases present in the problem structure. After training the controller with Imitation Learning, we can use Reinforcement Learning to fine tune specific behaviors while keeping other behaviors fixed. In the future we aim to implement our controller on real-world physics-based simulators in order to validate our results.

References

[1] Zhou, Siyu, et al. "Clone Swarms: Learning to Predict and Control Multi-Robot Systems by Imitation." *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019*. Institute of Electrical and Electronics Engineers Inc., 2019.
[2] Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." arXiv preprint arXiv:1806.01261 (2018).
[3] Craig W. Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. SIGGRAPH Comput. Graph. 21, 4 (July 1987), 25–34.

MORE
Masters Opportunity for Research in Engineering

Ira A. Fulton Schools of Engineering
Arizona State University

$$e_{ij}^t = \phi^e\left(x_i^t, x_j^t\right)$$
$$\bar{e_i^t} = \sum_{j \in \mathcal{N}_i} e_{ij}^t$$
$$h_i^t = \phi^v\left(x_i^t, \bar{e_i^t}\right)$$
$$y_i^t = \psi\left(h_i^t\right)$$