SwarmNet: Towards Imitation Learning of Multi-Robot Behavior with Graph Neural Networks

Siyu Zhou[*]
Arizona State University
Tempe, Arizona, USA
`siyu.zhou@asu.edu`

Mariano J. Phielipp
Intel AI Lab
Chandler, Arizona, USA
`mariano.j.phielipp@intel.com`

Jorge Sefair
Arizona State University
Tempe, Arizona, USA
`jorge.sefair@asu.edu`

Sara I. Walker
Arizona State University
Tempe, Arizona, USA
`sara.i.walker@asu.edu`

Heni Ben Amor
Arizona State University
Tempe, Arizona, USA
`hbamor@asu.edu`

Abstract

In this paper, we propose SwarmNet – a neural network architecture that can learn to predict and imitate the behavior of an observed swarm of agents in a centralized manner. Tested on artificially generated swarm motion data, the network achieves high levels of prediction accuracy and imitation authenticity. We compare our model to previous approaches for modelling interaction systems and finally discuss an extension of SwarmNet that can deal with nondeterministic, noisy, and uncertain environments, as often found in robotics applications.

1 Introduction

Multi-Robot Systems (MRS) [1] describe groups of robotic agents that collectively perform complex tasks in a distributed and parallel manner through repeated interactions among each other and the environment. Such systems have attracted considerable attention in recent years with remarkable successes in a number of application domains, including defense, agriculture, logistics, disaster management, and entertainment.

In this paper, we propose an alternative approach based on learning-from-demonstration (LfD) [2] for specifying group behavior in multi-robot systems. The LfD methodology has a rich history in single-robot systems, with a number of successful applications in real-world tasks such as table-tennis, manipulation, locomotion, and helicopter flying [3]. We extend this methodology to the MRS case by introducing a novel graph neural network architecture that can be trained from execution traces of a swarm. The introduced graph neural network, called *SwarmNet*, extracts all rules governing the group behavior from data alone, i.e., sequences of agent positions and velocities. Once a SwarmNet is extracted, it can be used to perform complex inferences including prediction, imitation or replacement of an existing swarm from a centralized view point.

2 Methodology

Learning is achieved by recording execution traces of an observed swarm. Execution traces are discretely sampled trajectories specifying the position and velocity of each agent at time step t. Each trace represents a demonstration of the group behavior as observed in the swarm. The set of traces is,

[*]

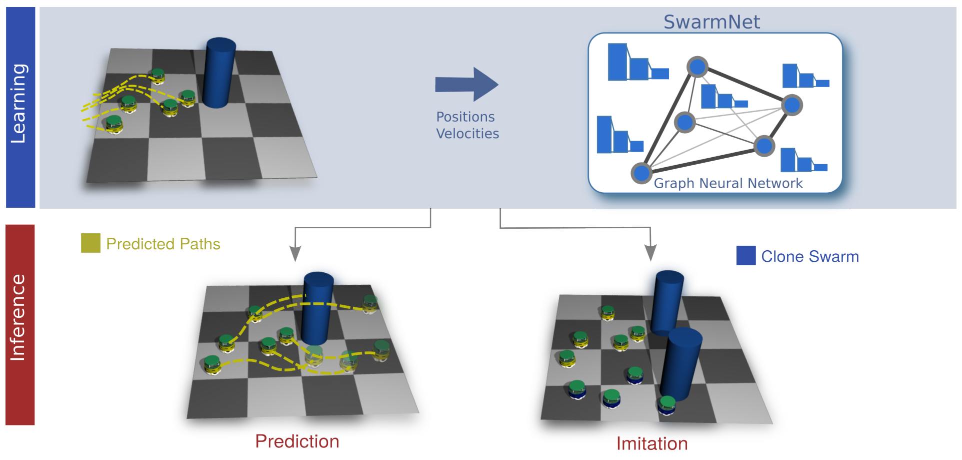

Figure 1: Overview of the proposed methodology: we collect training data from an observed swarm to learn a compact graph neural network representation called SwarmNet. In turn, it can be used to predict future behavior, augment an existing swarm with more agents, or create a clone of the swarm with similar behavior.

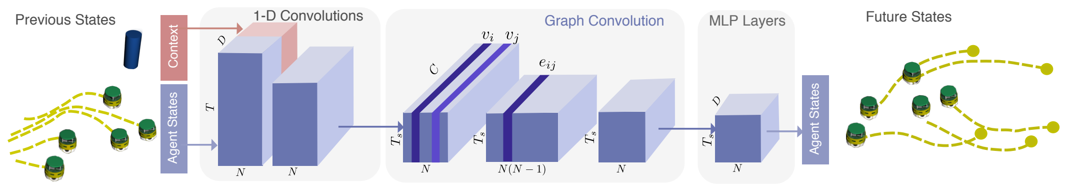

Figure 2: Diagram of the network architecture for SwarmNet: Letters along sides indicate dimensions. T is the number of time steps, N is the number of agents in the system, and D is the length of the state vector. H is the size of the encoded state vector, which is the size of the last layer of an MLP. Node states v_i and v_j are passed through Graph Convolution to produce interactions e_{ij}

in turn, used to train *SwarmNet*. Note that SwarmNet needs to account for both inter-agent interactions (e.g. how the members influence each other's behavior), as well as agent-environment interactions with environment context modeled as compliment information to agent states. For example, the obstacle needs to be taken into account to generate reasonable behaviors that avoid collisions with the environment.

Fig. 2 describes the full architecture of SwarmNet. Input to the network is a sequence of previous robot states, as well as contextual information regarding the environment. Robot states are specified by a time window of positions and velocities of either (1) observed agents, or (2) controlled agents, depending on the mode of operation. Assuming an N-agent swarm, the dynamical state of agent $i \in \mathbb{N}$ can be written as $\boldsymbol{s}_i(t) = [\mathbf{x}_i(t), \dot{\mathbf{x}}_i(t)]$, where $\mathbf{x}_i(t)$ is the position and $\dot{\mathbf{x}}_i(t)$ is the velocity of agent i at timestep t. The output of the network are the future state vectors $\mathbf{s}_i(t+1)$, given the state history of the swarm system and context. Multistep prediction or imitation is achieved by reattaching the output of next step back to the end of input sequence.

The states $\boldsymbol{s}_i(t)$ and context $\mathbf{c}(t)$ are concatenated and first passed through a series of one-dimensional convolution layers along the time axis. These convolutions allow the network to extract information regarding short-term dynamics of a single trajectory. The evolution of the agents' dynamic states is a result of pair-wise interactions between them. The dynamics of the system and the underlying elementary interactions can be formulated as a graph, with nodes being the agents, their dynamical states being node states, the interaction relations being edges, and the interaction effects being edge states. To avoid any limiting assumptions or constraints, the states are embedded in a fully-connected directed graph which models all relationships, i.e., the learnable functions along the graph edges define whether agents are interacting or not. To process the interactions and update the node states for predicting the next step, a graph convolution over our representation is employed. The final set of operations in SwarmNet is a set of traditional fully-connected neural network layers. These final

Method	Boids	Helbing	Chaser
Kipf's GNN	17.45 ± 1.00	19.61 ± 0.43	14.23 ± 0.05
LSTM	41.20 ± 0.24	42.88 ± 0.75	126.3 ± 3.5
SwarmNet	10.47 ± 0.91	3.855 ± 0.305	$\mathbf{3.691 \pm 0.042}$
SwarmNet (Context)	$\mathbf{2.778 \pm 0.066}$	$\mathbf{0.484 \pm 0.023}$	$\mathbf{3.691 \pm 0.042}$

Table 1: Normalized MSE loss for long term prediction (40 steps) of models on data sets of different swarms

layers take the result of the graph convolution as input and generate a window of predictions over the future states of all agents.

Training of the network is performed in a supervised fashion by comparing the predicted agent states (containing positions and velocities) with the ground-truth states in the time series. The mean squared-error (MSE) is used as the loss function L for training and as the metric for evaluation. The loss is normalized over the "natural skip", i.e., the MSE of state vectors between two consecutive steps in the ground truth trajectories. We scaffold the training by gradually increasing the prediction horizon. The curriculum learning has the benefit of starting with a simpler version of the task (i.e. using limited prediction horizon) and slowly exposing the network to more complex, long-term predictions and encourages the model to learn the true dynamics of the system rather than potentially overfitting to fixed-term predictions.

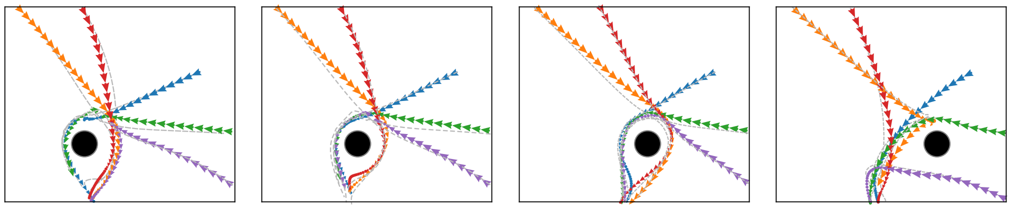

Figure 3: From left to right: four different predictions of the swarm behavior using SwamNet. In each one of the four experiments, the obstacle is place slightly more to the right. The colored arrows show the movement of 5 boids, while the gray dashed lines are the ground truth trajectories. The black circle represents the obstacle.

3 Experiments and Results

We generated training data using popular techniques for synthesizing swarm behavior, e.g., flocking and swarming models. In turn, we compared different versions of SwarmNet to the graph neural network (GNN) in [4] and the LSTM network method in [5].

Our data set consists of motion data produced using the Boids model [6], the Helbing model [7], and a simple chaser model. In the Boids model, $N = 5$ agents demonstrate flocking behaviors while approaching a common goal location and avoiding obstacles. The Helbing model [7] generates swarm behavior via repulsive forces only. The final model, called chaser, places a set of agents on a circle. Each of these agents generates steering actions that make it chase another agent of the swarm. For each of the above models we generate data sets by running the simulation for about 50 time steps while recording the agent positions and velocities, as well as the environmental variables. In both the Boids and Helbing model, the environmental variable is the positions of obstacles.

The first experiment focuses on the prediction accuracy when compared to other methods, in particular the method in [4] and LSTMs [5]. Table 1 summarizes the loss of the methods across the three data sets. We performed the experiment for prediction horizon of 40 steps. The best performance is achieved by SwarmNet with contextual inputs. Removing the context information has a significant impact on the SwarmNet performance, in particular in the long-term prediction case (right side of the table). In general, SwarmNet outperforms all other methods on all data sets and in both conditions. An interesting aspect of these results is that SwarmNet was only trained on data for predictions of up to 10 steps. Despite that, it correctly learned to make accurate predictions for 40 steps and beyond indicating that it focused on the true dynamics of the task.

Fig. 3 shows the prediction results for different locations of the obstacle (black circle). We see that the SwarmNet predictions are qualitatively implementing the correct swarming and avoidance behavior, even if small deviations from the ground truth occur. The rightmost example in Fig. 3 shows an interesting behavior in which the orange agent is first trying to circumvent the object on the right side and then turns to head back to goal location. While unintuitive, this behavior is also existent in the ground truth data. Fig. 4 shows the prediction on the chaser swarm by SwarmNet and GNN in [4].

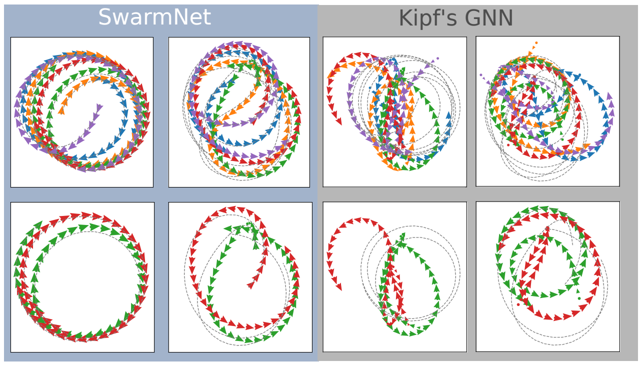

Figure 4: Qualitative comparison of SwarmNet and the approach in [4]. Two samples of predicted trajectories (scattered arrows) against the ground truth trajectories (grey dashed lines) for a 5 agent system performing chasing motions. Top row: traces of all 5 agents. Bottom: traces of only two agents for visibility.

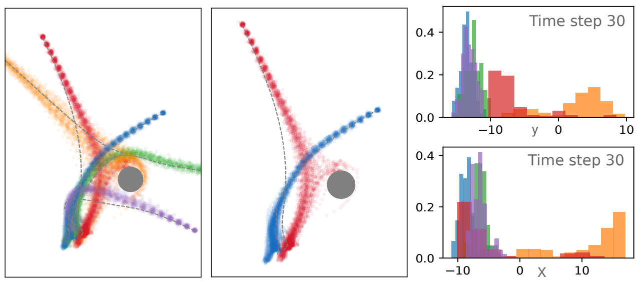

Figure 5: Predictions for the *nondeterministic* behavior of a swarm with 5 boids in the presence of uncertainty. Left graph: the stochastic output predictions of the SwarmNet network for all agents. Middle: the predictions for only the blue and the red agent, for visual clarity. Right: The probability distributions over x and y coordinates of all agents at time step 30. We can see that the predictions for the red agent bifurcate to the left and right side of the obstacle.

We also investigated how well the SwarmNet can deal with uncertainty, noise and nondeterminism. In particular, we simulated both perceptual noise, as well as actuation noise. Perceptual noise was incorporated by adding a value sampled from a univariate normal distribution $\mathcal{N}(0, 1)$ to input for each dimension separately. Actuation noise is simulated by randomly dropping out neurons in test time. Fig. 5 depicts the stochastic outputs of our SwarmNet model for the Boids data. We can see in Fig. 5 (left) that the predictions now form envelopes according to the uncertainty at different time steps in the future. In general, the uncertainty appears to grow with larger prediction horizons. In the case of the red agent, the predictions slightly bifurcate around the obstacle. This clearly shows that the model is able to predict multiple potential futures of an agent (and the swarm) based on the inherent uncertainty of the task and environment. On the right, the probability distributions for discretized x- and y-coordinates of red and orange agents are multimodal, which reinforces the insight that our predictions can produce multiple, diverse, and potentially conflicting future states.

4 Conclusion

In this paper, we presented a neural network architecture, called SwarmNet, which learns to predict and imitate behavior of an observed swarm. The network uses a combination of one-dimensional convolutions, graph convolutions, contextual inputs, along with a curriculum learning scheme to efficiently extract the swarm dynamics from positions and velocities of a set of agents. We showed that SwarmNet achieves high levels of prediction accuracy and that it can even be applied to nondeterministic and uncertain environments. For future work, we want to investigate how different application domains and tasks affect the sample complexity of the method, and the effectiveness of a decentralized implementation of the network.

References

[1] T. Arai, E. Pagello, L. E. Parker *et al.*, "Advances in multi-robot systems," *IEEE Transactions on robotics and automation*, vol. 18, no. 5, pp. 655–661, 2002.

[2] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[3] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[4] T. Kipf, E. Fetaya, K.-c. W. Max, and W. Richard, "Neural Relational Inference for Interacting Systems," 2018.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] C. W. Reynolds, "Steering behaviors for autonomous characters," *Game Developers Conference*, pp. 763–782, 1999.

[7] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature*, vol. 407, no. 6803, pp. 487–490, 2000.