Malicious Agent Classification In Multi-Agent Formation Control

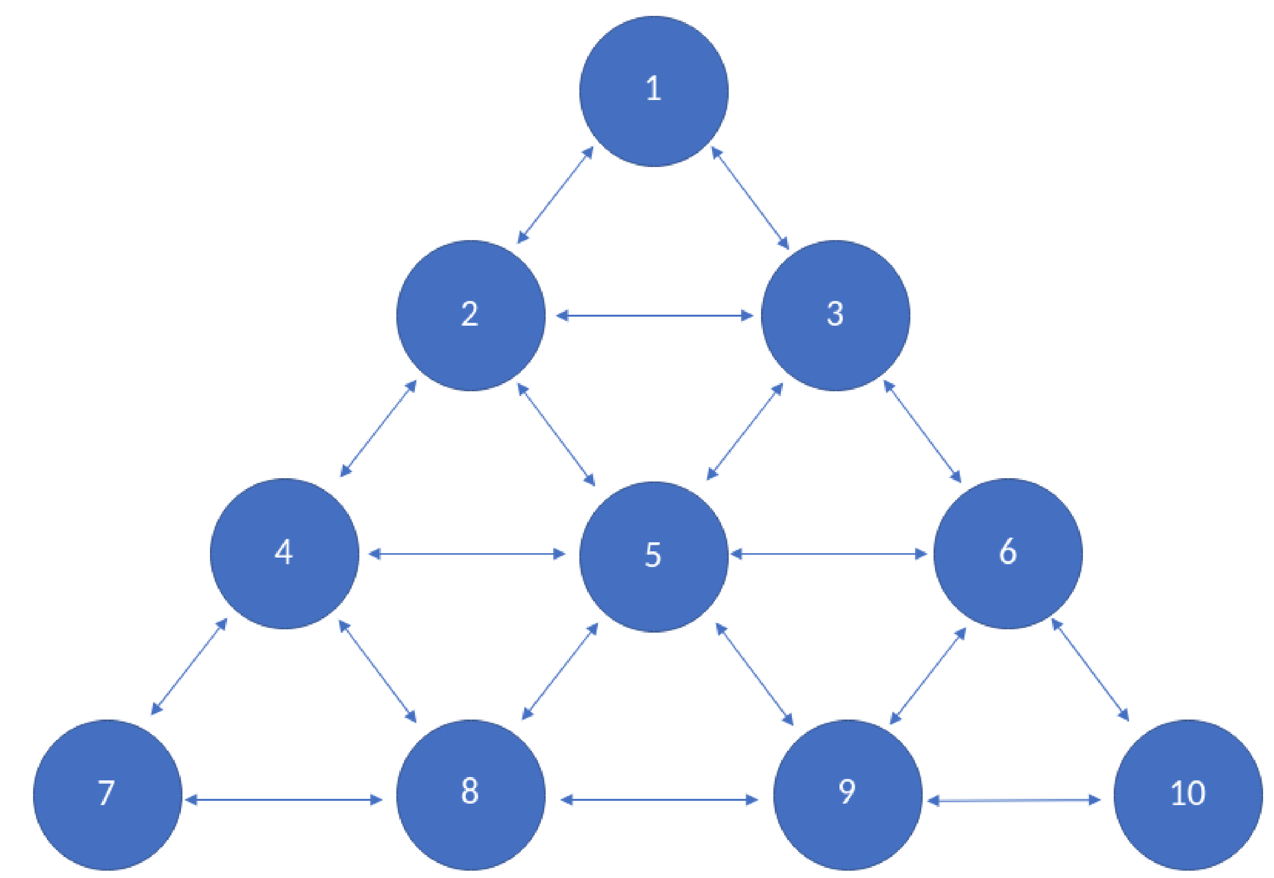
William Chong: wmchong@stanford.edu
Department of Mechanical Engineering: Stanford, CA. USA

Introduction

Multi-agent formation control laws typically depend on the graph state of the robot formation to calculate control inputs for each robot in a decentralized method to reach a goal formation [1]. If malicious agents are present, then those actors can impede the formation control. This work attempts to train a malicious agent classifier that has no knowledge of the formation control law as well as the connectivity of the graph. Feature inputs to the network consist of observed robot positions, velocities, and position FFT data along the trajectories. The network is designed with two layers where the first layer is a parallel structure of 2 LSTMs (position, velocity) and 1 MLP (position FFT), which output is then concatenated to an MLP for the second layer. The output is a 10x1 binary vector indicating whether each robot is a good or bad agent. The training and testing data contain 0-5 malicious agents for each individual simulation. Results show high precision and recall, as well as a high balanced accuracy given no knowledge of the graph connectivity or the control law.

Figure 1 (next column): Desired robot formation shape. Robot positions are randomly initialized at the start of every simulation and assigned random indices, and malicious agents are randomly chosen. Arrows indicate edge connections for the control law.

System Dynamics and Control



$\dot{p}_i = u_i : p \in \mathbf{R}^{20x1}$ (robot (x,y) positions). Malicious agents: $u_i \sim \mathcal{N}(0.1,\, Q)$.

Solving $(L(G) \otimes I_d)P = (L(G) \otimes I_d)Z^*$ yields control law u_i (Jacobi over relaxation method)

$L(G)$ is the Laplacian matrix, $P = [p_1^T, p_2^T ...]^T$ is the coordinate matrix

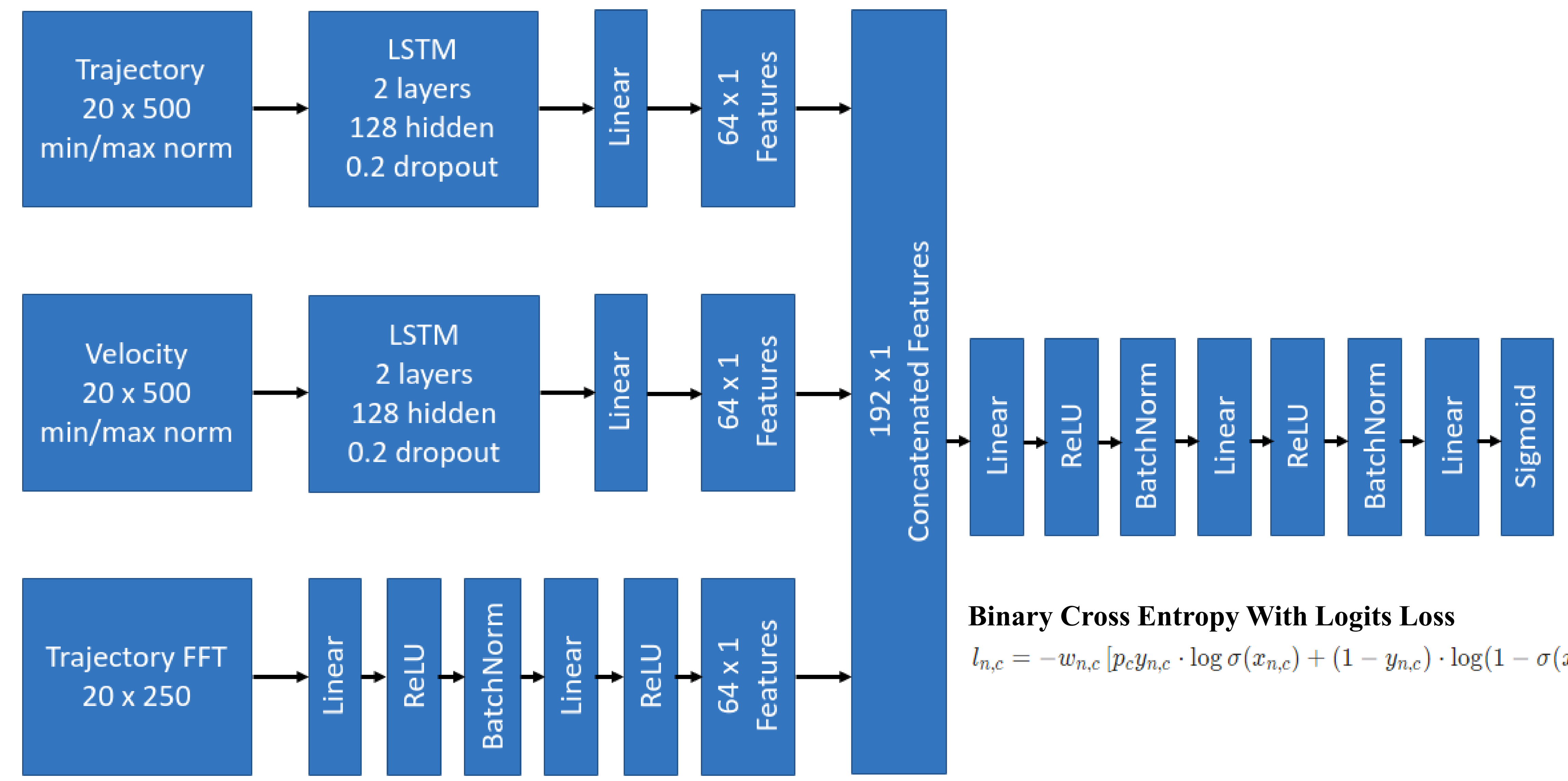
$Z^* = [(z_1^*)^T ... (z_n^*)^T]^T; (z_i^*)^T = (x_i, y_i)$ is the desired formation shape coordinate

$Y = [b_1^T, b_2^T ...]^T = (L(G) \otimes I_d)Z^*$

Adjacency notation for neighbors N_i: $a_{ij} = \begin{cases} 1, j \in N_i \\ 0, j \notin N_i \end{cases}$

Discrete dynamics: $p_i(l+1) = (1-h)p_i(l) + \frac{h}{d_i}(\sum_{j \in N_i} a_{ij} p_j(l) + b_i) + wh; w \sim \mathcal{N}(0,\, Q)$

Network Architecture



Binary Cross Entropy With Logits Loss

$l_{n,c} = -w_{n,c}\left[p_c y_{n,c} \cdot \log \sigma(x_{n,c}) + (1 - y_{n,c}) \cdot \log(1 - \sigma(x_{n,c}))\right]$

Figure 2: LSTMs were selected to learn temporal relationships through the time series data (i.e. the control law). Position and velocity features provide dynamical behavior description, and position FFT provides dynamics frequency information for correlation. BCEWithLogitsLoss used with positive_weight (p_c) = 0.333 for class imbalance. Batch size of 100 with 15 epochs of training with Adam. Training with 3000 trajectories with equal split among different numbers of malicious agents. Testing with 1500 trajectories, and validation with 1500 trajectories (equal split as well). Sigmoid layer is removed during training since the loss function includes it, and is added during evaluation.

Data and Simulation Plots

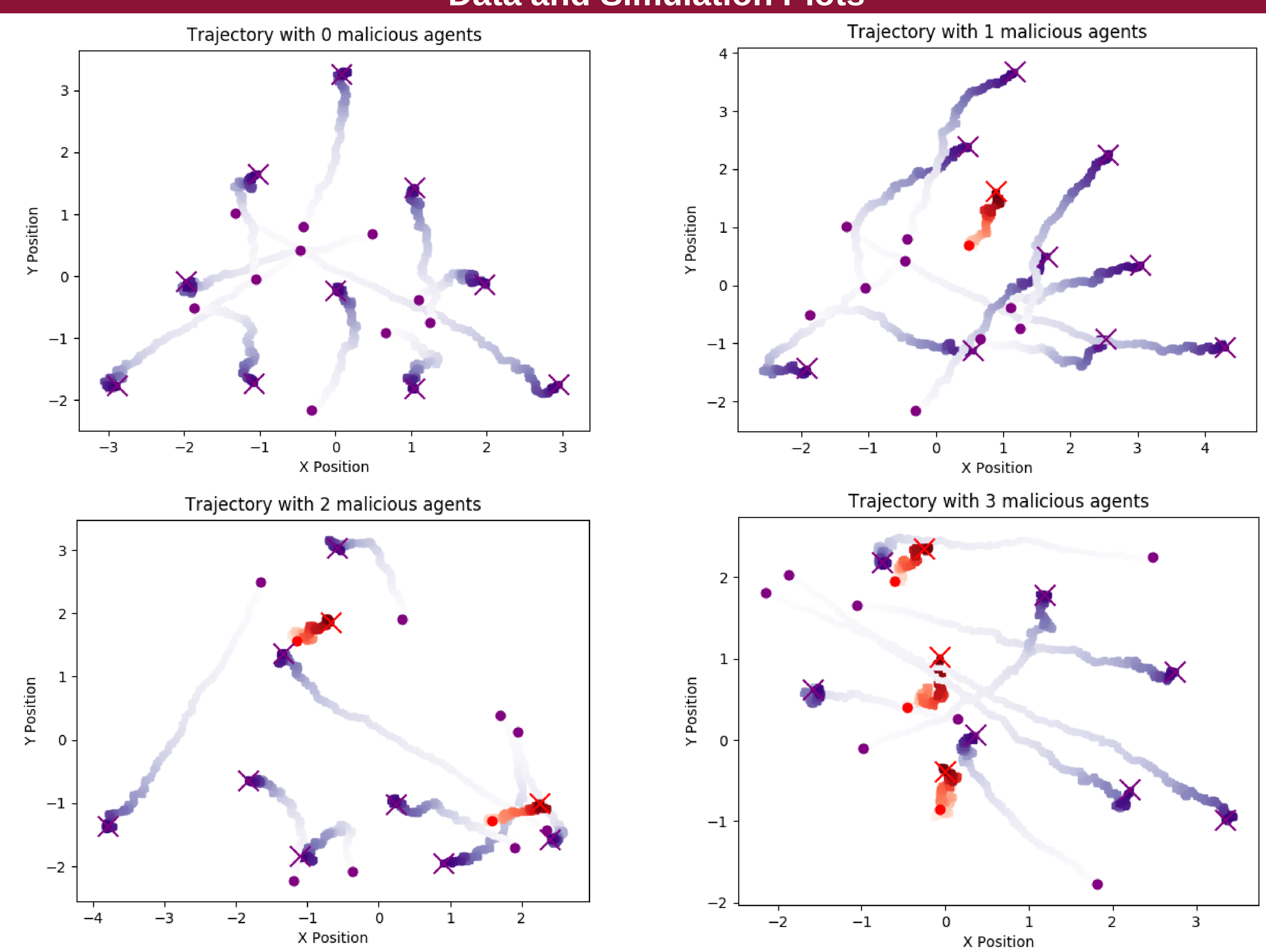
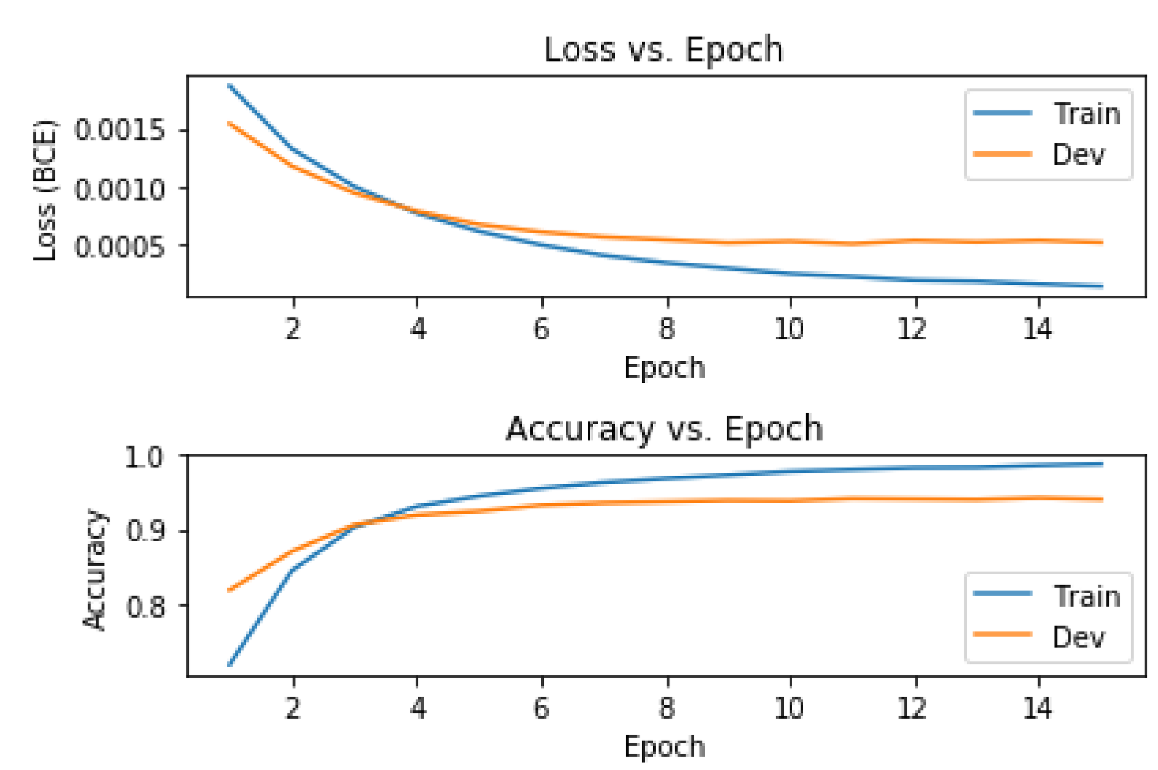
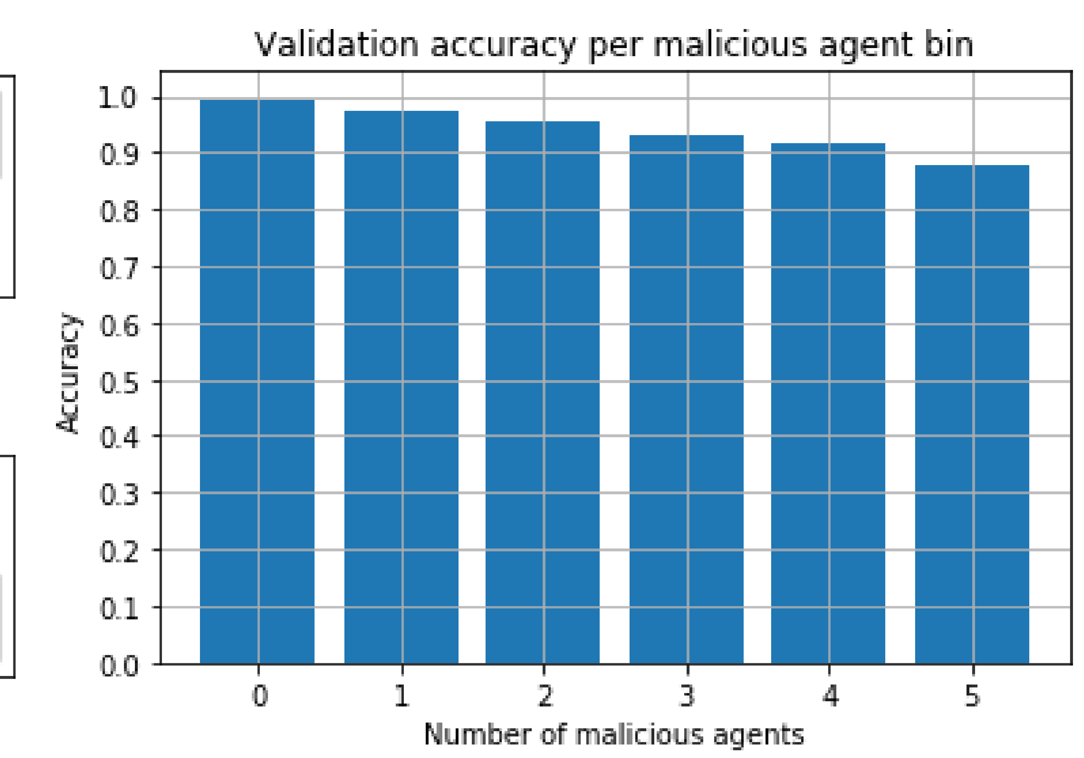


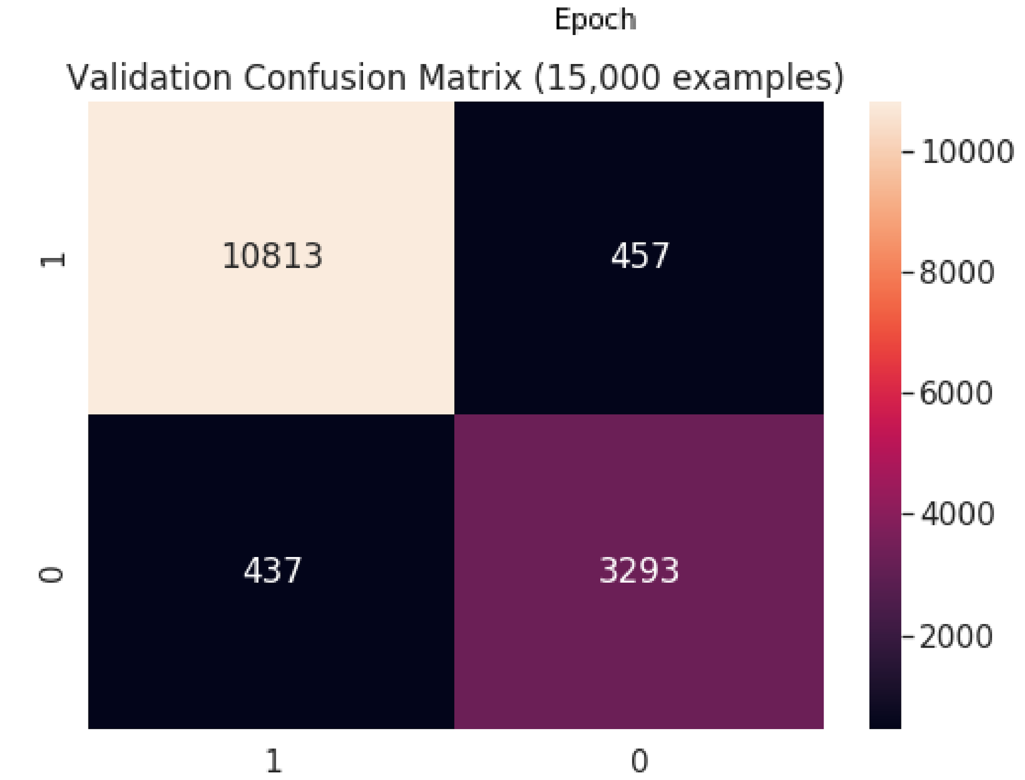
Figure 3-6: Simulation plots of robot trajectories for 0-3 malicious agents (5 sec, 100 Hz). Malicious agent strategy is a Brownian motion with velocity that is similar to the average of good agent velocities with the same noise covariance.

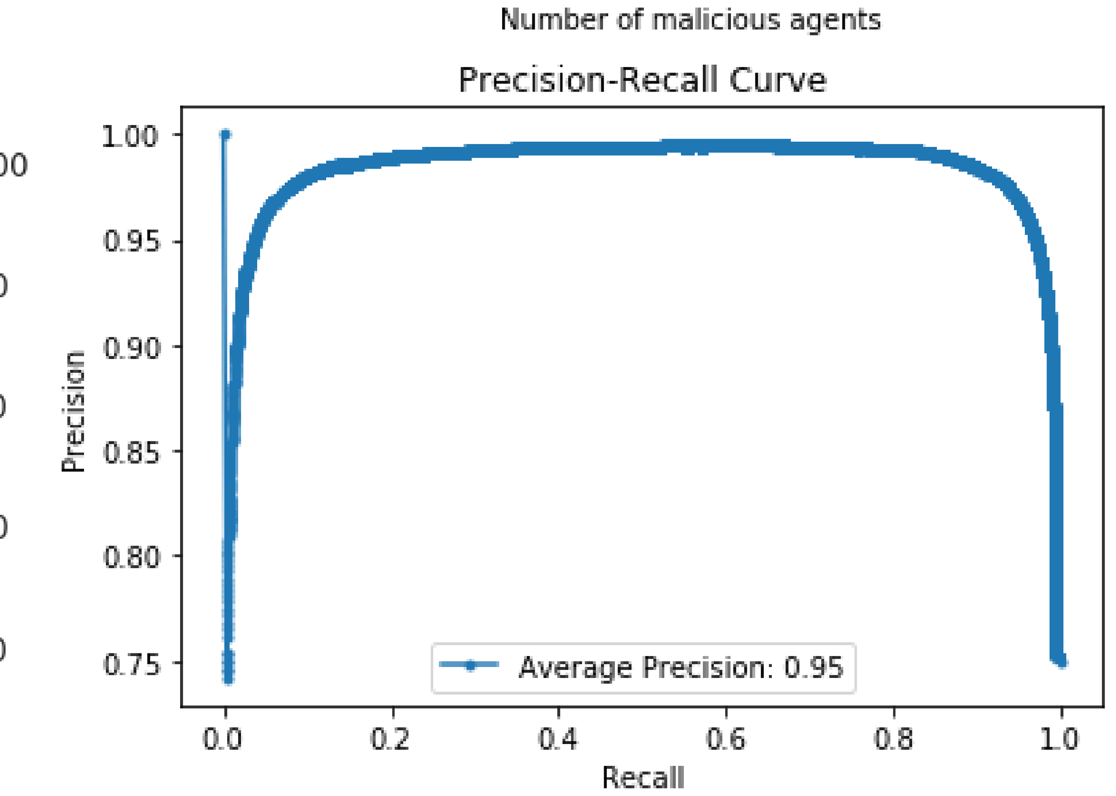
Results





Classifier Performance	
Precision	0.959
Recall	0.961
Specificity	0.878
Balanced Accuracy	0.920
Average Precision	0.949





Conclusion and Future Work

The malicious agent classifier yielded fairly good results in terms of precision and recall with the imbalanced dataset. This is expected since LSTMs are designed for learning temporal relationships, and the FFT position information was extremely useful in showing correlation among good agents and among bad agents. Future work involves extending the malicious agent strategies and classifying malicious agents with their respective strategies.

Reference: [1] Mondal, Arindam et al. "A novel multi-agent formation control law with collision avoidance." IEEE/CAA Journal of Automatica Sinica 4 (2017): 558-568.