Advanced Persistent Threat in Cooperative Multi-Agent Reinforcement Learning

Supplementary Material

A Anomalous Behavior Detection

A.1 Detection Models

To combat the adversarial attacks on MARL systems, we develop several detection schemes and compare them by performance. Namely, a four-layer Dense deep learning, a three-layer binary LSTM, Random Forest, SVM classifier, K-nearest Neighbors classifier, and a novel stacked-LSTM ensemble architecture. The proposed ensemble outperformed the baselines in all three environments.

To form the ensemble model, we combine the binary LSTM with separate predictive LSTM networks for each of the agents. Instead of predicting normal or anomalous, these models predicted one of the possible k actions. In our experiments, the particle environments have an action space of 5, whereas Starcraft II agents have a possible action space of 9. These models corresponded with specific agents within the environments. They were trained on clean datasets with $220,000$ samples of normal datapoints generated from each environment with fully cooperative agents. We found that the ensemble model (Figure 1) achieved the highest recall across all models when tested on datasets generated by our attacks.

Figure 2 shows the comparison of the proposed ensemble model against other baselines. The ensemble model continued to have success in the other attack strategies across all environments based on its recall and precision metrics. The ensemble consistently showed the best recall (percentage of anomalies detected) with some attacks only going undetected 0.1% of the time. Figure 3 emphasizes the performance of the ensemble model, particularly in the Starcraft II environment where the average recall across all attack rates and strategies in the proposed ensemble model improved by approximately seven percent. This large improvement in recall between the ensemble and other baselines in the Starcraft II environment is likely due to the high-complexity of this environment. Because of the environment's high-dimensionality, basic machine learning algorithms struggle to distinguish between normal and anomalous datapoints.

The detection results for the 4 attacks in the black-box settings are shown in Table 1.

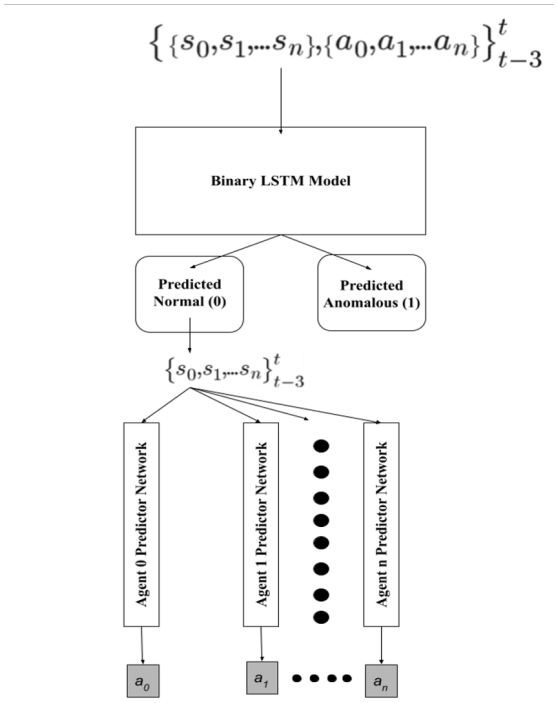


Figure 1: The ensemble model for anomalous behavior detection

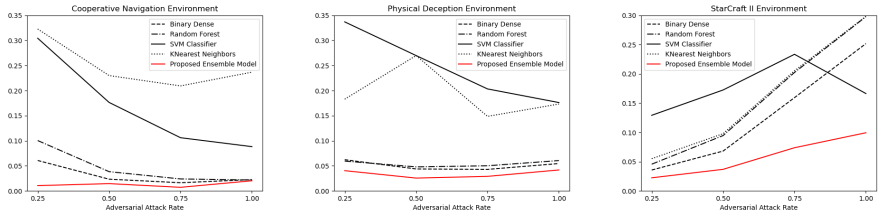


Figure 2: The percentage of undetected anomalies for each model in the whitebox randomly-timed attack strategy by attack rate

A.2 APTs via The Evasion Technique: More Results

Although the proposed ensemble detection method has promising results, there are several drawbacks. Most notably are the susceptibilities of neural networks to adversarial attacks like FGSM [1] and JSMA . To further argue the need for c-MARL robustness to adversarial attacks, this paper proves the infeasibility of using detection subnetworks in detecting anomalies in reinforcement learning by applying the fast gradient sign method attack (FGSM) to fool the proposed ensemble network. The authors assume no knowledge of the interior workings of

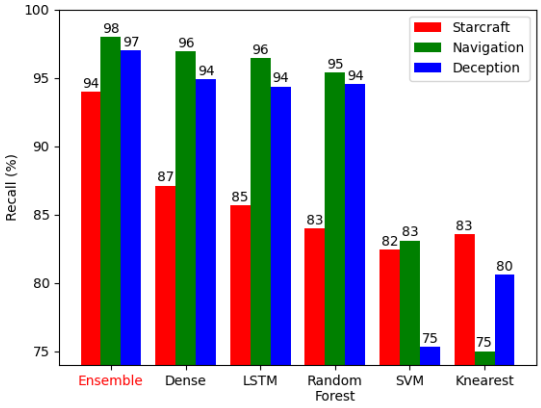


Figure 3: Average recall across all six attack strategies and attack rates for each of the detection models.

this ensemble model in an attempt to show the true implications of this attack. Recall that the FGSM equation is $x_{\text{adversarial}} = \epsilon * sign(\Delta_x J(\Theta, x, y))$.

Several steps are executed to implement this attack at a large scale. First, FGSM needs access to a replica model in order to approximate the gradient of the network used in the attack. Because we assume black-box knowledge of the ensemble, the attacker must generate samples that can query the model. By using the model's output to the attacker's queries as the true label, we can then train an approximated replica model. In other words, the attacker issues queries to the target model and labels a carefully selected dataset. For any arbitrarily chosen x the adversary obtains its label y by querying the target model f. The adversary then chooses a procedure $train'$ and model architecture f' to train a surrogate model over tuples (x, y) obtained from querying the target model. The surrogate model is used in the FGSM attack.

Next, for each environment and type of attack, the most effective ϵ was found. To do this, we tested ϵ values between [0,1) at an increment of .05. For each of the tested ϵ, every anomalous datapoint that was misclassified by the surrogate model was then passed through the true ensemble model. The most effective ϵ was chosen by selecting the ϵ with the highest percentage of misclassified anomalies by the ensemble. Though only the most-effective epsilons were found for only the whitebox randomly-timed attacks due to time consumption, it can easily be extended to all others. For the cooperative navigation, physical deception, and Starcraft II environments, the most effective ϵ were .35, .9, .7, respectively (Figure.4).

To strengthen the attack of our detection subnetwork and to prove the infeasibility of deep detection subnetworks, the exact same datasets were tested for all six attacks on all environments. Except now, not only did the compromised agent attack the other MARL agents, the compromised agent also perturbed

Table 1: The detection results of the attacks using the ensemble LSTM model (*precision%—recall%*) in the cooperative navigation and physical deception environments across the different attack rates in black-box settings

MADDPG:Cooperative Navigation Black Box						
Attack Rate	Random		Timed		Counterfactual	
25%	61%	97%	62%	99%	30%	76%
50%	85%	98%	94%	97%	56%	82%
75%	94%	99%	99%	95%	75%	77%
100%	100%	99%	100%	96%	100%	83%
MADDPG: Physical Deception Black Box						
Attack Rate	Random		Timed		Counterfactual	
25%	52%	97%	55%	99%	30%	82%
50%	78%	97%	77%	97%	54%	86%
75%	91%	98%	89%	96%	75%	80%
100%	100%	97%	100%	94%	100%	83%
QMIX: StarCraft II Black Box						
Attack Rate	Random		Timed		Counterfactual	
25%	44%	99%	49%	99%	28%	93%
50%	56%	99%	69%	99%	54%	98%
75%	69%	99%	70%	99%	72%	94%
100%	100%	99%	100%	97%	100%	93%

its own observations. The only perturbed observations belonged to the compromised agent, assuming it can control the observation it reports to its target agents. Keeping all else constant, the FGSM attacks severely discounted the proposed ensemble model's performance in detecting anomalies in all attacks and all environments. With [2] proof that new adversarial perturbations can always be generated, these supplemental perturbations will render detection subnetworks obsolete.

After applying FGSM to the compromised agent's observations, we tested the same attacks again in all three environments across all 4 attacks in white and black box settings.

Figure 5 shows the drastic reduction in performance of the ensemble model across all attack rates and strategies. The greatest detriment was seen in the Starcraft II environment, where a 36% drop in recall was observed after the FGSM re-attack. Again, this is likely due to the high-complexity of the Starcraft II environment. In this specific setting, the adversarial agent's observations accounted for one-third of the global state. Due to the perturbation of a larger proportion of the global state, the detection model suffered greatly. Out of the different attacks, the ensemble suffered most with the re-perturbation of the whitebox randomly-timed attack. Figure 5 shows the stark difference

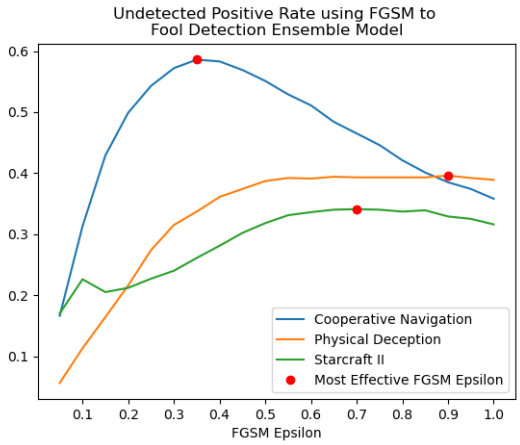


Figure 4: Finding the most effective ϵ for the attacks.

between recall before and after the FGSM re-attack in the whitebox randomly-timed setting. Clearly, the attack rate of .25 was most successful after the FGSM re-attack. This was consistent across the other attack strategies. With a lower attack rate, the difference between model recall before and after the re-perturbation was largest. In some attack settings, the recall was halved, resulting in the majority of positives going undetected.

The idea that slightly perturbing one compromised agent's observations can reduce performance at such a large scale is alarming. With the seemingly heavy dependence on small fractions of the global observations, attackers can implement similar attacks to go virtually unnoticed by detection subnetworks. By only compromising one agent, the entire system will be affected. And without reliable detection, it is clear that these attacks can be devastating.

B Deep Learning Models for Behavioral Cloning and State Transition Estimating

The deep learning based behavioral cloning is trained to approximate the policies of the compromised agent m and other agents $-m$ using the collected state-action $(s_t, at)_i \ \forall i \in M$ pairs from observing the c-MARL agents during the reconnaissance phase. This model is a multi-headed model (Figure 6) which reads input from the three previous timesteps and the current timestep for each agent's state, as well as the compromised agent's action. When the input data is processed by the first 2 LSTM layers, the model branches off into separate heads such that each head is responsible for predicting the corresponding agents' actions. Each head then adds an additional LSTM layer and ends with a Dense layer with a softmax activation function that predicts the probability confidence of all of the possible actions performed by that specific agent $a \in A$. We train

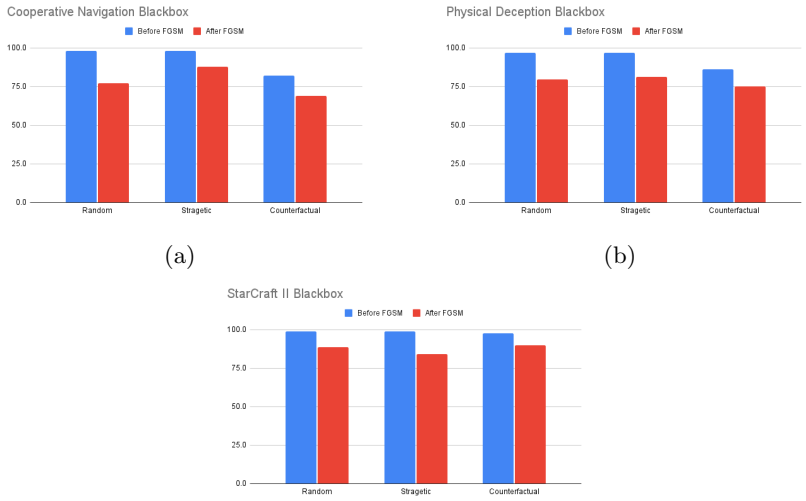


Figure 5: The detection results of the c-MARL attacks in all environments before and after applying the evasion technique in the black-box settings

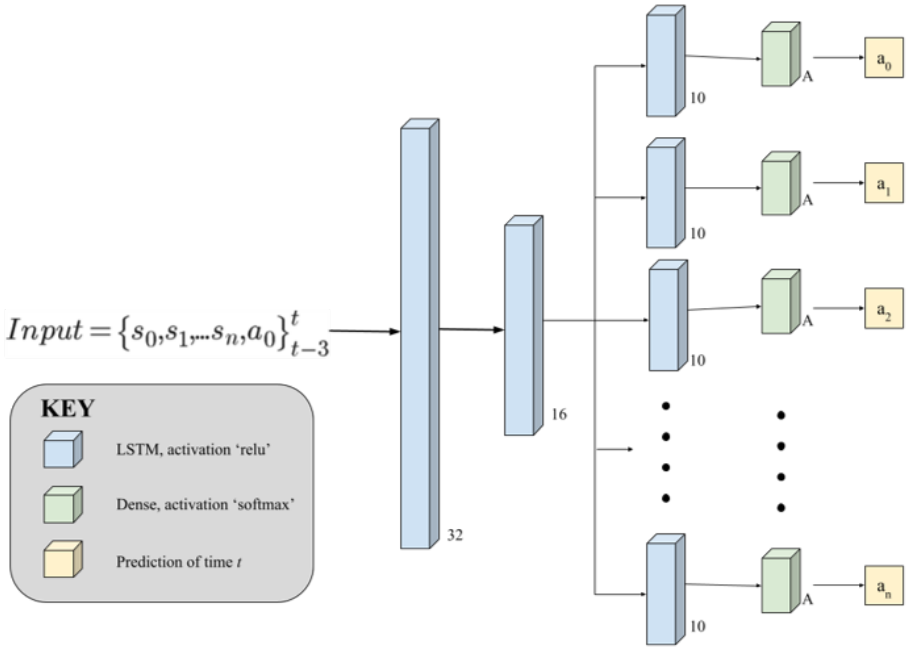
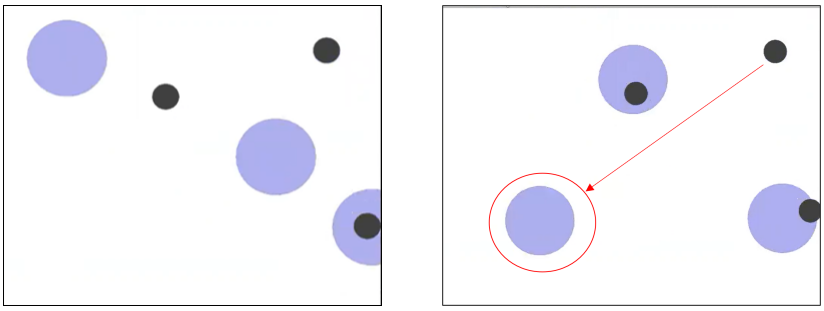


Figure 6: The architecture of the deep learning based behavioral cloning.



(a) (b)

Figure 7: Screenshots from replayed episodes of the cooperative navigation environment from MADDPG under the strategically-timed attack

this model by minimizing the loss function $L(a, \pi_\theta)$ for each agent's policy.

The supervised learning model for the state transition function is similar to the behavioral cloning model with multiple heads but with a different learnable parameter:

$$T_\Phi = p(s_{t+1}|s_t, a^1, ..., a^n, a*^m) \qquad \forall * m \in A \tag{1}$$

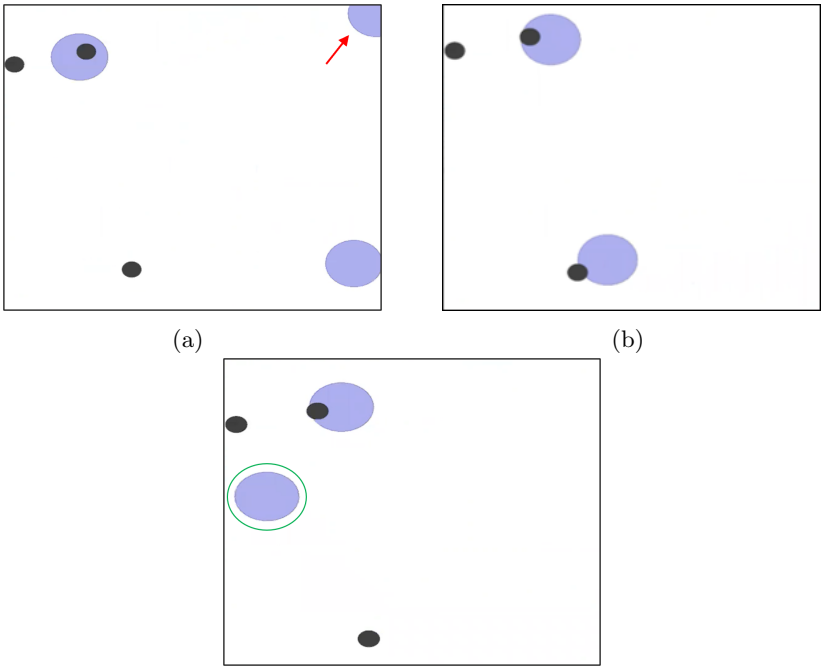
C Demonstrations of The Attacks

C.1 Cooperative Navigation Environment

We show here screenshots from the cooperative navigation environment under the strategically-timed and counterfactual reasoning based attacks. We choose those attacks to represent the attacks in each category. In the strategically-timed attack, the compromised agent immediately moves away from the landmarks to prevent its teammates from covering all landmarks. In this environment, there is not much room for the agents to recover from the compromised agent's adversarial actions. Figure 7a shows the beginning of the episode where the agents attempting to cover the 3 landmarks (black circles). Then, in Figure 7b, the compromised agent (the one with red circle around it) is moving away from its supposedly assigned landmark.

In the counterfactual reasoning based attack, the compromised agent displays interesting behaviors. The first behavior is shown in Figure 8. The compromised agent (pointed to with a red arrow) is moving away from the landmark just before its teammate(s) gets to a landmark which makes the affected agent (within a green circle) confused about which landmark to cover (stands in between the 2 empty landmarks for the rest of the episode). This behavior suggests that the compromised agent succeeds by manipulating its teammate's observations through its actions.

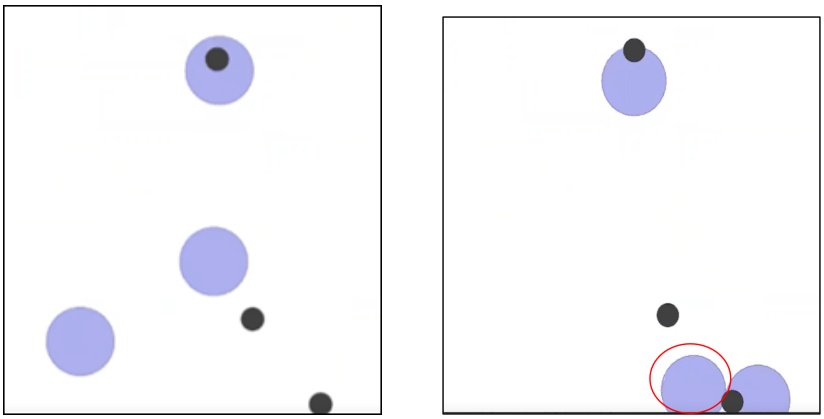
We notice another behavior under the counterfactual reasoning based attack (Figure 9). In this episode, the compromised agent (with the red circle around



(a) (b)

(c)

Figure 8: Screenshots from replayed episodes of the cooperative navigation environment from MADDPG under the counterfactual reasoning based attack



(a) (b)

Figure 9: Screenshots from replayed episodes of the cooperative navigation environment from MADDPG under the counterfactual reasoning based attack

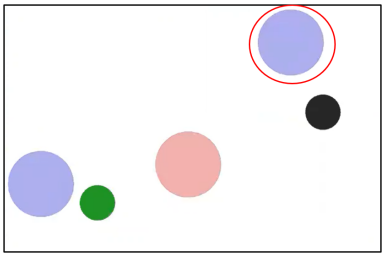
it) moves to the same landmark that has been already covered by one of its teammate causing the team reward to decrease.

C.2 Physical Deception Environment

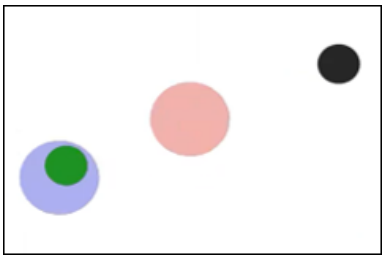
In the physical deception environment,during the strategically-timed attack, we notice that when the compromised agent moves away from the non-target landmark (Figure 10), the other cooperative agent does not alter its behavior to move away from the target. This is a robustness issue in MADDPG algorithm. Ideally, the agents should be robust enough to not have to rely on a teammate who is not doing its job. Similarly, we see that when the compromised agent leaves the non-target landmark uncovered and moves towards the target landmark, the adversary does not always take advantage of this clue (Figure 10). The optimal behavior of the adversary would be to move towards the only landmark which is being covered by a cooperative agent, instead it stays still. This behavior once again shows a lack of robustness in the agents under our attack.

C.3 StarCraft II

In the StarCraftII environment, during all attack except the counterfactual reasoning based attack, we notice that the compromised agent moves away from its team until its teammates gets defeated. Then, it moves towards the enemies to get killed. In strategically-timed attack specifically, the compromised agent runs away only when its team is getting closer to the enemy (Figure 12a). In the counterfactual reasoning based attack, we notice sometimes the compromised agent luring one of its teammates to start attacking the enemy team then it itself runs away (Figure 12b) causing its team to get defeated.

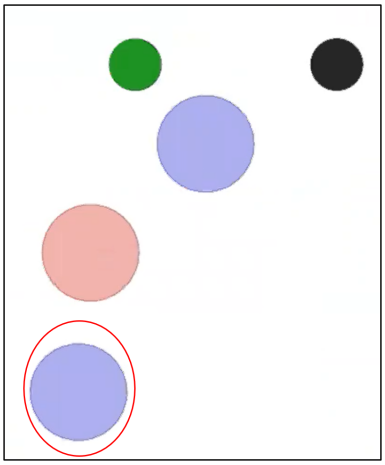


(a) The beginning of the episode where the cooperative agents are splitting over the target and non-target landmarks

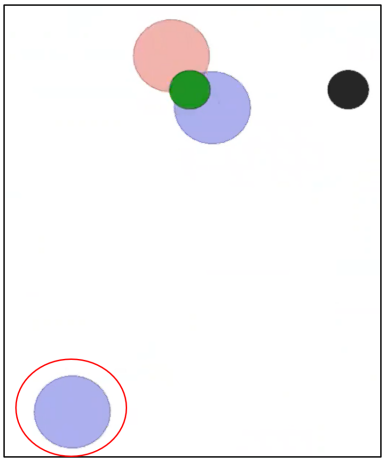


(b) The compromised agent is controlled by an adversarial policy trained with strategically-timed attack is moving away from the non-target landmark leaking the information to the adversary

Figure 10: Screenshots from replayed episodes of the physical deception environment from MADDPG under the strategically-timed attack



(a) The beginning of the episode where the cooperative agents are splitting over the target and non-target landmarks



(b) The compromised agent is controlled by an adversarial policy trained with counterfactual reasoning based attack is moving away from the non-target landmark in a perfect timing to leak the information to the adversary which got the clue and moved towards the target

Figure 11: Screenshots from replayed episodes of the physical deception environment from MADDPG under the counterfactual reasoning based attack

(a)

(b)

Figure 12: Screenshots from replayed episodes of StarCraft II.The compromised agent is controlled by an adversarial policy trained with (a) strategically-timed attack, and (b) counterfactual reasoning based attack

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[2] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," 2017.