A Review of Security Attacks and Defenses in Reinforcement Learning

Cam Lischke, Frank Liu, Joe McCalmon

Abstract

We present existing attacks against single and multi-agent reinforcement learning (RL). We define an attack as any adversarial interference with an RL system, with the intention to decrease performance of targeted agents within the system. We survey the existing literature to find which aspects of the RL problem have been exploited for these attacks. We also record the existing defenses and detection models used to improve the robustness of RL systems against the existing attacks. We finally suggest that since detection models can also be fooled by attacks such as FGSM and JSMA, real-time verification techniques will be required to ensure robustness against adversarial attacks.

1 Reinforcement Learning Background

1.1 Single Agent RL

Artificial intelligence has provided major breakthroughs in longstanding fields such as natural language processing, image recognition, and navigation. Reinforcement learning (RL) is a subset of artificial intelligence which uses deep neural networks to train an autonomous agent to perform a task optimally. RL has been applied to problems such as autonomous navigation, robotics, and Atari games. The RL problem involves an agent within an environment. RL provides a framework for this agent to interact with its surrounding environment and receive feedback signals. RL algorithms use these feedback signals to improve the decision-making policy of the agent.

More specifically, the agent interacts with the environment in a series of discrete timesteps, indexed $t = 0, 1, 2, 3, \ldots, T$, where T is the end of a single environment trajectory, called an episode. At each timestep within an episode, the agent observes a representation of the environment, called the state, $S_t \in S$, where S is the set of all possible states. The agent then chooses an action $A_t \in A$, where A is the set of all possible actions, using its policy, π. $\pi(a|S_t)$ is the probability action a will be chosen by the agent, given state S_t. Then, the environment's transition function decides the resulting next state in timestep $t + 1$, where $p(s_{t+1}|a, S_t)$ is the probability of state s resulting from an agent taking action a in state S_t. After each action, the agent receives a reward based on the outcome of its action in S_t, $R_{t+1} = R(S_t, A_t)$. The goal of an RL agent is to maximize the expected sum of these rewards over the course of the episode, or the return denoted $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T$ (Sutton and Barto, 2018). While many problems can be formulated using this framework, not all are mathematically proven to be solvable by current RL methods. An environment has the Markov property if the reward and state at timestep t+1 are functions only of the state and action at timestep t, or $p(s', r|s, a) = Pr\{Rt+1 = r, St+1 = s'|St, At\}\ \forall r, s', S_t$, and A_t (Sutton and Barto, 2018). If the environment has this property, and the state and action spaces are finite, then we call the problem a Finite Markov Decision Process (MDP), and the RL problem is proven to be solvable.

Even if the environment is not a Finite Markov Decision Process, methods exist to approximate an optimal policy. Deep Reinforcement Learning methods such as A2C (Sutton and Barto, 2018) and DQN (Mnih et al., 2013) can deal with high-dimensional state spaces and continuous action spaces. Also, the introduction of deep learning into RL allows the use of long-short term memory

Preprint. Under review.

layers (LSTM) to remember previous timesteps in environments whose transition functions rely on information not present at every timestep.

1.2 Multi Agent RL

Multi-agent Reinforcement Learning (MARL) is an extension of single-agent RL, which employs $N > 1$ agents in a single environment. MARL environments are not inherently Markov since the transition probabilities $Pr\{R_{t+1} = r, S_{t+1} = s' | S_t, A_t^1, \ldots, A_t^N\}$ depends on the actions chosen by each agent, instead of just one. To each agent, the function by which the environment selects the state at $t + 1$ appears non-stationary (Lowe et al., 2020). In addition, each agent receives their own observations, o_t^i, where $S_t = \{o_t^1, o_t^2, \ldots, o_t^N\}$. So, parts of the environment's state can be hidden from each agent.

There are different classifications of MARL problems: cooperative, adversarial, and mixed. In cooperative MARL (c-MARL), each agent works together to accomplish a common goal, often sharing the same reward function. In adversarial MARL, each agent has their own objective, and are often rewarded based on both their own successes and their opponents' failures. In mixed MARL, some agents are cooperative, and some are adversarial. Adversarial and mixed MARL require separate reward functions for each agent, and therefore need special learning paradigms, as each deep neural network will try to minimize a unique loss function.

Much focus has been placed on c-MARL, with algorithms such as COMA (Foerster et al., 2017), QMix (Rashid et al., 2018), and multi-agent soft-Q (Wei et al., 2018) all improving performance of cooperative agents in environments such as StarCraft. MADDPG is the current state-of-the-art algorithm for solving c-MARL, as well as adversarial and mixed settings. MADDPG's main contribution is the introduction of the centralized critic-decentralized actor structure. The actor-critic structure was introduced by A2C for single-agent RL. This method uses two deep neural networks, one to learn a value representation for the expected return (the critic), and one to decide the correct action for the agent (the actor). MADDPG allows the critic network to learn a value function based on the observations of each agent combined, but the actor network must learn a policy using only the corresponding agent's own observations. The result is an agent which has a unique decision-learning policy from the others in the environment, allowing success for multiple agents with different goals. In addition, MADDPG concatenates the actions of each agent onto the state at each timestep during training. Doing so eliminates the non-stationarity of the environment, since the actions of other agents are considered part of the state. This is only done for the centralized critic network, and allows for the convergence of that network.

Multi-agent RL shows promise for a new wave of RL applications, including in robotics and self-driving cars (Peake et al., 2020). Therefore, like single agent RL, the security and robustness of multi-agent systems is important to study. We present attacks and defenses to both single agent and multi agent RL from the literature.

2 Security attacks against RL systems

The prevalence of Deep Reinforcement Learning (DRL) brought significant technological advancements in the AI community and different state-of-the-art algorithms allow more effective training and more promising results. However, researchers have proved that most DL models and algorithms have security and privacy issues (Szegedy et al., 2014). Recently, researchers showed that adversarial examples and policies can effectively attack the DRL system as well. If those models and algorithms are implemented into reality with the nature of their vulnerability and if subjected to carefully designed attack, the consequence is extremely terrible, even may bring harm to human life. Attacking strategy in DRL involves more complicated thought since 1) RL agents interact with the environment for a sequence of actions, which further change the observation of the environment. Therefore, there are more opportunities to craft the attack. 2) The goal of the DRL agent is to maximize the expected sum of reward rather than successfully classify an input. Given the nature of DRL attacks, in the following literature review, we are going to present recent research work on attacking strategy and defense mechanisms in single-agent DRL and multi-agent DRL.

In 2017, Huang et al. (2017) introduced the idea of adversarial attacks on a DRL agent by attacking every single timestep in a given episode. They analyze three types of DRL algorithms,

including DQN (Mnih et al., 2013), TRPO (Schulman et al., 2017), and A3C (Mnih et al., 2016). To include some background information, DQN approximately computes the Q-value for the available actions that could be taken in that state via a neural network trained to minimize the squared Bellman error instead of modeling the policy directly. DQN introduces the idea of experience replay to reduce the variance of Q-learning updates and a replay buffer to sample from so the past experience tuple is not correlated due to time. TRPO is an on-policy batch learning algorithm the entire trajectory of a policy is used to update the policy parameters, and KL divergence between old and new policy controls the change of the policy. A3C uses asynchronous gradient descent to speed up and stabilize the learning of a policy based on the actor-critic approach. In a white-box setting, Huang et al. (2017) crafted adversarial examples to the network of the agent trained using the above three algorithms when playing four different Atari games. They use the FGSM method to craft their adversarial examples that will be inputted into the DNN. In the RL setting, the output y is a weighting over possible actions. They claimed that regardless of which Atari game is played and which algorithm is used to train, the policy's performance significantly decreased by only introducing relatively small perturbations in the inputs to the DNN. And then they apply the adversarial examples crafted using FGSM to the DNN in the black-box setting, assuming the adversarial examples in RL are transferable as well, a concept from Szegedy et al. (2014). For most games, exploiting the transferability across algorithms is still able to significantly decrease the agent's performance. The result of this paper brings out the vulnerability of DNN applied in RL settings. If people deploy such attacks in a real-world setting, for example, an adversarial example crafted to confuse an autonomous car's camera, the consequence may be very severe. It's then important to consider the defense mechanism against adversarial attacks, for example, to include adversarial examples during the training process, namely adversarial training. This attack happens in 2017, where Huang et al. (2017) started to focus on the security issue in DRL settings. Their attack on every single timestep proves the vulnerable nature of the DL model still exists in the RL setting but still it's very easy for a model with a defense mechanism to detect the attack from the adversary. In addition to that, they ignore a fact that the observation is correlated. The adversarial example crafted at timestep t is dependent on the observation of the previous timestep, which also was attacked with adversarial examples. To further improve the security of the DRL model, a more carefully designed attack is needed in order for a more elaborate defense mechanism. This paper studies adversarial example attacks on DRL agents trained using A3C and DQN.

In 2019, Lin et al. (2019) introduced two novel types of attack on a single DRL agent system — the strategically-timed attack and the enchanting attack. The goal of the strategically-timed attack is to reduce the agent's reward by only feed adversarial examples to the deep learning network of the agent for a selected small subset of timesteps in a certain episode. Limiting the attacking behavior to only a portion of the episode further reduces the risks of being detected by the model and system. The idea is to only craft adversarial examples to the DNN only when it's worthy to do so. In other words, we input the adversarial examples to output an action that can reduce the most reward. To solve the when-to-attack problem, researchers propose relative action preference function c for attacking the agents trained by the A3C and DQN algorithms. The c function is defined as:

$$c(S_t) = \max_{A_t} \pi(S_t, A_t) - \min_{A_t} \pi(S_t, A_t) \tag{1}$$

A large value for $c(S_t)$ represents that the agent strongly prefers one action over the other, which means this particular action is crucial to increase the accumulated reward. We implement the attack in timestep t if and only if $c(S_t) > \beta$, where β is a pre-defined threshold corresponding to attack rate. It's then the task to manipulate different β to control the frequency of attacking to achieve the most balanced rate. The how-to-attack problem then becomes a targeted attack (Carlini and Wagner, 2017) aiming to output the action from the most preferable one to the least preferable one. In the enchanting attack, the researcher utilizes the fact that each action taken by the agent influenced its future observation. Therefore, the adversary could plan a sequence of adversarial examples to maliciously lure the agent toward a dangerous state. The researcher used a generative model to predict the sequence of action that could lead the agents into the dangerous state, and then craft adversarial examples that could output each action in the sequence using the same method above. They evaluated both tactics on 5 Atari games, and each DRL agent is pre-trained using A3C and DQN. For strategically-timed attacks, researchers tried different levels of attacking rate β and conclude that strategically-timed attacks can reach the same level of uniform attack but only use 25% of timesteps to attack. They also claim that A3C is more robust against adversarial attacks than DQN, which echos the conclusion that a stronger DNN based algorithm is more robust to adversarial attacks

in the previous DL research paper. For enchanting attacks, both A3C and DQN are subjected to this type of attack, and the success rate is over 70% when $H < 40$, where H represents the average length of an episode. This work introduced two novel tactics of attacking DRL agents, and these attacks are successful in their interest of measurement. This further proves the vulnerability of the DNN network, even in an RL setting. However, this paper still fails to show a defense mechanism against such sophisticated attacks. Basically, there are two ways of making the model more robust, first is to do adversarial training by incorporating adversarial examples while training the mode. However, this has been proven to not work since the black-box attack by using adversarial examples generated from another model would still successfully attack the model's classification even it's been trained with adversarial examples (Tramer et al., 2020). In other words, the transferability property still holds for adversarially trained models. Therefore, it is important for the AI community to come up with a more reliable adversarial training strategy. Another defense mechanism is to train a subnetwork used to detect adversarial input at run time, similar to the GANs.

Following Huang et al. (2017), Russo and Proutiere (2019) model the selection of attacking time steps as a Markov decision process. By solving this Markov decision process, an attacker could identify the optimal time steps to launch attacks and thus minimize his effort on environment manipulation. Their attack outperforms traditional gradient-based attack and significantly influence the performance of agent, both in discrete space and continuous space environment.

Recently, researchers have begun exploring vulnerability to adversarial attacks in MARL. The decision-making process of MARL entails more factors, for instance, other agents' actions are also part of the observation of one of the agents, according to the previous background information. Thus the security issue still remains. While the previous works mentioned involve directly modifying the observation of an agent who interacts with the environment, this is not always possible in the multi-agent RL setting, especially in complicated high-dimensional environments. Gleave et al. (2021) introduced the idea of an adversarial policy in 2020, which produced an adversarial DRL agent who successfully beat other RL agents in a game. They came up with a physically realistic threat model and found that their adversarial policy actually creates natural observations that are adversarial. In other words, the agent learns to produce observations which the victim agent has not encountered. This policy harms the performance of other agents. Researchers conclude that victim policies in higher-dimensional environments are significantly more vulnerable to adversarial policies due to their nature of inducing different policy activations than normal opponents. Adversarial policies work by pushing the activation of the victim's policy network off-distribution. Researchers first pre-trained the victim's policy as normal RL agents training. They then freeze the victim's optimal policy and train the adversarial agent as a single-player MDP game by maximizing the sum of discounted reward from an attacker's perspective (positive reward when the adversarial wins the game). The adversarial policies reliably win against most victim's policies. However, after carefully observing the animation, researchers find out that adversarial agent wins not by training a stronger policy, but by exploiting the weaknesses in the victim's policy. They further provide two defense mechanisms — single training and dual training, by either retrain the victim's policy against an adversarial policy or randomly picking either an adversary or victim policy to train at the start of episodes. But the attack can simply repeat the single training defense, suggesting that adversarial policies are difficult to eliminate, but the actions of new adversarial models no longer make the victim's policy off-distribution. Therefore it's highly possible that this is the future direction to scale-up the training cycle for a more robust model.

In 2020, Weng et al. (2020) introduced an attack against DRL agents using a model-based technique. They observed a DRL agent within it's system enough to build a model of the environment transition function using supervised learning. Then they propose two threat models, one where the adversary can perturb the agent's observations, and one where the adversary can perturb the agent's actions. For both these threat models, the corresponding target vector must be continuous, to allow for ϵ perturbations. In both cases, the adversary identifies a target state beforehand, and perturbs the observations or actions of the agent to lead it towards that target state. Their attacks outperform three versions of a random attack.

In most multi-agent cases, reinforcement learning agents are in a cooperative setting, termed cooperative Multi-Agent Reinforcement Learning (c-MARL). Lin et al. (2020), introduced an attack on the c-MARL situation by manipulating one of the agent's observations in order to decrease the total team reward. They proposed a two-step attack. They first train an adversarial policy where the outputted action can minimize the combined team reward, and then use that action as a goal to

perturb the agent's observation with an adversarial example. The objective is to alter the state so that the victim agent performs the outputted worst action. Their experiment claims that c-MARL systems are highly vulnerable to perturbations applied to one of the team member's observation models. Their attack is as efficient as simply modifying one agent's behavior. Specifically, their attack on the StarCraft II environment brings down the team winning rate from 98.9% to 0% by only perturbing a single agent's observations. However, it is difficult to identify how a single agent's behavior contributed to the decrease in the total team reward. The drop could be due to the failure of the adversary's task, or the behavior of the other cooperative agents, reacting to the adversary's actions. If the latter is the case, to improve the robustness of the c-MARL system, the ability to eliminate the impact of the malicious agents is essential. We identify this direction as one for future work.

Communication among MARL agents is an important part of the c-MARL system. A modern autonomous system can perform better if information can be shared and workloads can be distributed. In 2021, Tu et al. (2021) explore an attack on the MARL communication channel by providing an indistinguishable adversarial message which severely degrades the performance of the system. Current communication protocols between agents have potential security threats since the shared information may be malicious. The recipients of those messages are essentially DNN, which has a significant security issue. Their experiment focuses primarily on a cooperative object detection task where each agent can observe the object from different perspectives and then share their observation together for a final decision. Their experiment on two practical multi-view perception situations shows that communication is subjected to adversarial attacks, but the robustness of the system increases when benign agents take up more percentages of the total agents. They also find out that adversarial training is very effective in defending the attacks.

In 2021, Alqahtani et al. (2021) adapt the strategically-timed attack from Lin et al. (2019) to the multi-agent particle environments (Lowe et al., 2020), as well as develop a new model-based attack similar to Weng et al. (2020) for this setting. They propose a novel threat model where a cooperative agent is taken over by an adversary. They also adapt these attacks to a white-box scenario, where the adversary has full access to the agents' policies and the environment dynamics, as well as a black-box scenario, where the adversary can only observe the behavior of the multi-agent system. They found that c-MARL systems are as susceptible to adversarial attacks as single-agent RL systems.

Adversarial attacks and defenses in multi-agent settings are still an underdeveloped area of research.

3 Detection of adversarial agents

The literature has shown that many MARL policies are especially vulnerable to adversarial attacks (Chen et al., 2019). With reinforcement learning's major implications in artificial intelligence applications ranging from computer-driven games to connected autonomous vehicles, the reliability and security of these algorithms is of upmost importance. Because of the common goal of cooperation or competition (or both) between agents within many MARL systems (Barton et al., 2018), agents react based on the actions of others within the same environment. For example, connected autonomous vehicles must interact with other vehicles, pedestrians, and infrastructure, considering the actions and reactions of others before acting on its own (Loke, 2019). For this reason, it only takes one corrupted agent to cause major security issues. Adversaries that intentionally aim to defect targeted networks can confuse agents and lead them to make mistakes that can result in poor performance and even harm to humans that rely on these systems (Huang et al., 2017). Often, the noise added by corrupted agents to fool MARL systems are invisible to humans (Chen et al., 2019; Huang et al., 2017), creating the critical need for reliable machine learning algorithms to detect and mitigate these attacks.

To detect and mitigate such attacks that are occasionally invisible to the human eye (Chen et al., 2019; Huang et al., 2017), machine learning techniques are especially valuable. While much research involves investigations into how to prevent adversarial examples from fooling MARL policies with methods like adversarial training (Gleave et al., 2021; Goodfellow et al., 2015; Chen et al., 2019), data augmentation and randomization (Xie et al., 2018), and detector subnetworks (Chen et al., 2019; Metzen et al., 2017), anomaly detection tools have been applied using a variety of machine learning techniques. Whether it is basic classifiers for normal and abnormal data or predictive networks that

can validate actions and observations, it is becoming increasingly more common to employ various techniques within MARL systems.

Due to some limitations of a normal feed-forward neural network's ability to remember information from previous frames of data, a Long-Short Term Memory Neural Network (LSTM) was developed using real-time recurrent learning (Verner, 2019). LSTMs mitigate this vanishing-gradient problem through their implementation of various recurrent cycles from subsequent neurons to preceding ones, creating hidden layers–especially conducive to time-series data to act like memory (Verner, 2019; Malhotra et al., 2015).

LSTMs demonstrate a viable technique to predict normal time-series behavior that consequently classifies anomalous behavior without real knowledge of the domain of the data (Verner, 2019; Staudemeyer and Omlin, 2013). Compared to other deep learning techniques, LSTM based prediction models may give better results and performance (Malhotra et al., 2015). Much work has been done to prove that LSTMs are extremely effective in classifying anomalous behavior (Metzen et al., 2017) in various domains ranging from medicine (Verner, 2019) to computer network traffic (Staudemeyer and Omlin, 2013). In addition, LSTM networks can go even further, differentiating different categories of anomalous behavior from normal behavior (Verner, 2019). Using prediction error distributions as a baseline (Malhotra et al., 2015), LSTM networks are perfectly suited to analyze sequential data with temporal dynamics (Verner, 2019). Stacked LSTM neural networks have been found in many works to show exceptional performance in detecting anomalies (Verner, 2019; Malhotra et al., 2015; Naseer et al., 2018), demonstrating the promise of deep learning technologies in security applications.

In recent work by Alqahtani et al. (2021), an ensemble detection model using a binary LSTM and predictive LSTM networks achieved the highest precision and recall across other anomaly detection classification techniques. This is shown in the table below. Though this is a potential mitigation for c-MARL environments plagued by an adversary, deviations in agent actions can be due to multiple factors. So naively marking an agent as an adversary could itself become a potential threat if the reason for an agent's deviation is due to a benevolent mishap.

Cooperative Navigation												
	White Box						**Black Box**					
Attack Rate	Random		Timed		Counterfactual		Random		Timed		Counterfactual	
25%	60%	96%	64%	98%	54%	92.5%	61%	97%	62%	99%	55%	92%
50%	85%	98.5%	85.5%	98%	82%	92.5%	84.5%	98%	83.6%	98%	82.6%	92%
75%	94.5%	99%	99%	95%	95%	94%	94%	99%	94.4%	97%	95%	94%
100%	100%	98%	100%	99%	100%	96%	100%	99%	100%	95.5%	100%	96%
Physical Deception												
	White Box						**Black Box**					
Attack Rate	Random		Timed		Counterfactual		Random		Timed		Counterfactual	
25%	51%	96%	54%	97%	45%	91%	52%	97%	55%	99%	40%	97%
50%	77%	97%	77%	96%	74%	92%	78%	97%	77%	97%	72%	94%
75%	90%	97%	90%	95.5%	90%	93%	91%	98%	89%	96%	91%	88%
100%	100%	96%	100%	94.5%	100%	92%	100%	97%	100%	94%	100%	84%

Figure 1: The detection results of the attacks using the ensemble LSTM model (precision | recall) in two different environments across different attack rates in both 3 white and 3 black box settings (Alqahtani et al., 2021).

Unsupervised classifier techniques have been applied for anomaly detection. Typically, a sparse region correlates to outliers–points that do not belong to a specific group (Mazarbhuiya et al., 2018; Flanagan et al., 2017). Previously, signature-based methods of anomaly detection stored normal and anomalous examples in a database and a supervised method checks certain time intervals with the database to detect intrusions. However, due to increasingly large network traffic volumes and the infeasibility of maintaining a signature-logging system (Hasan et al., 2019; Flanagan et al., 2017), using a typical clustering algorithm has been found to be useful in detecting anomalies in network intrusion detection (Mazarbhuiya et al., 2018; Flanagan et al., 2017; Eskin et al., 2002). New studies show modified clustering algorithms finding outliers that are present across multiple time windows (Flanagan et al., 2017) while using hybrid techniques and new similarity measures to analyze a mix of numerical and categorical data (Mazarbhuiya et al., 2018). While most research approaches this issue by combining model-based predictors with clustering in a semi-supervised setting (Das et al., 2008; Shukla and Chandel, 2013), there have been few efforts to apply strictly unsupervised clustering methods to anomaly detection (Flanagan et al., 2017; Münz et al., 2007; Eskin et al., 2002).

In other application settings, random forests have been proven useful. In many classification problem domains such as credit card fraud detection (Hasan et al., 2019) and medical anomaly detection (Verner, 2019), random forests can serve as beneficial due to their simplification in algorithms and flexibility in handling data of different types (Xuan et al., 2018). Some works propose that random forest approaches perform similarly to deep learning techniques in network intrusion detection on IoT devices (Hasan et al., 2019; Naseer et al., 2018; Zhang and Zulkernine, 2006a; Primartha and Tama, 2017). Hybrid approaches that use anomaly detection followed by misuse detection (Zhang and Zulkernine, 2006a) and random forest algorithms can break the dependency on training sets with attack-free data (Zhang and Zulkernine, 2006b) by classifying outliers found by the system as intrusions (Zhang and Zulkernine, 2005). Some limitations have been uncovered, including problems imposed by imbalanced data (Xuan et al., 2018), a lack of strong guidelines for hyperparameters (Primartha and Tama, 2017; Naseer et al., 2018), and the fact that intrusions with a high degree of similarity cannot be detected as outliers in most cases (Zhang and Zulkernine, 2005).

Though these techniques may provide impressive results, using general classification techniques to identify adversaries in MARL environments have many drawbacks. For instance, high traffic volumes and advanced adversaries may make detection subnetworks obsolete on a larger-scale project (Hasan et al., 2019; Flanagan et al., 2017; Alqahtani et al., 2021). In addition, new novel attacks can easily fool the detector subnetworks using modified techniques of FGSM and JSMA attacks (Chen et al., 2019; Huang et al., 2017; Gleave et al., 2021; Hu and Tan, 2017; Wiyatno and Xu, 2018; Grosse et al., 2016; Papernot et al., 2015). Even without white-box knowledge of the victim detection network, these attacks can be implemented using a replica model trained on querying the victim subnetwork. Using the gradient of this replica model and a modified FGSM or JSMA attack, the transferability property of deep neural networks can be leveraged (Papernot et al., 2017). Lin et al. (2020) modifies these attacks for c-MARL settings, however, their approaches can be adapted to perturb detector subnetworks, potentially marking anomaly detection as just a temporary solution.

4 Defense without Detection Subnetworks

4.1 Defense strategies in machine learning

Clearly, detection subnetworks have extreme security concerns. To combat this, Metzen et al. (2017) proposed dynamic adversarial training by introducing a novel adversarial agent that could fool both the c-MARL policy, as well as the detector subnetwork. This is extremely prevalent when the attacker has access to both the reinforcement learning policy and the detector subnetwork (whether replicated or white-box). They found that if an attacker were to replace the cost $J_{cls}(x, y_{true}(x))$ with a new formula $(1-\sigma)J_{marl}(x, y_{true}(x))+\sigma J_{det}(x, 1)$, where $\sigma \in [0, 1]$ is a set coefficient hyperparameter and $J_{det}(x, 1)$ is the cost (cross-entropy) of the detector for the generated input x and the label 1, i.e., being adversarial. An adversary maximizing this cost would thus aim at letting the policy mis-label the input x and making the detectors output padv as small as possible. The most concerning part is that this method can be implemented into an iterative fast sign attack (an extension of FGSM) to dynamically find the most effective σ.

Although this attack is threatening to both the model and the detector subnetwork, it can be utilized for securing both targets through adversarial training. The basics of adversarial training is exposing the model to adversarial attacks during the training phase. The designer can then teach the model the expected way to handle adversarial examples. Extending this, Metzen et al. (2017) use their aforementioned novel attack to provide adversarial training examples, deemed dynamic adversarial training. Based on the approach proposed by Goodfellow et al. (2015), instead of precomputing a dataset of adversarial examples, they compute the adversarial examples on-the-fly for each mini-batch and let the adversary modify half of the data points. Note that a dynamic adversary will modify a data point differently every time it encounters the data point since it depends on the detector's gradient and the detector changes over time. They extend this approach to dynamic adversaries by employing a dynamic adversary, whose parameter σ is selected uniform randomly from $[0, 1]$, for generating the adversarial data points during training. By training the detector in this way, the model will resist dynamic adversaries for various values of σ. As you can see in the figure below, the dynamically-trained model outperformed the static-trained model significantly, especially with values of σ within the range of 0.25 through 0.9.

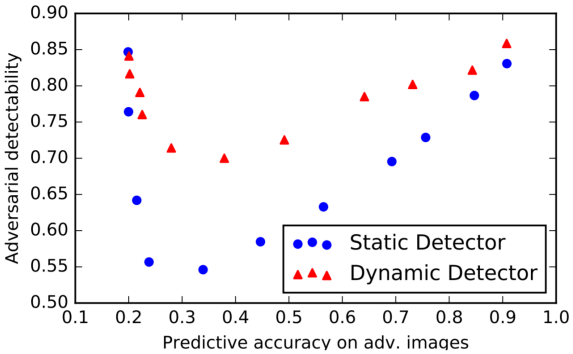

Figure 2: Illustration of detectability versus classification accuracy of a dynamic adversary for different values of σ against a static and dynamic detector. The parameter σ has been chosen as $\sigma \in \{0.0, 0.1, \ldots, 1.0\}$, with smaller values of σ corresponding to lower predictive accuracy, i.e., being further on the left (Metzen et al., 2017).

In attempts to further train models against adversarial attacks, researchers have extended normal adversarial training. For example, Tramer et al. (2020) introduced the ensemble adversarial training, which enhances training data with perturbations transferred from other static pre-trained models, this approach separates the generation of adversarial examples from the model being trained, simultaneously drawing an explicit connection with robustness to black-box adversaries. Sinha et al. (2020) suggested principled adversarial training, which, by utilizing worst-case perturbations and the Lagrange penalty forms of the training data to reinforce model parameter updates, guaranteed the performance of neural networks under adversarial data perturbation. In addition, Chen et al. (2018) proposed a generalized attack-immune model based on gradient band, which mainly consists of Generation Modules, Validation Modules, and Adversarial Training Modules. The three components work together to generate, validate, and finally train other models.

Apart from training with adversarial examples, Srisakaokul et al. (2019)) explored a novel defense approach, MULDEF, based on the principle of diversity. The MULDEF approach firstly constructs a family of models by combining the target model with additional models constructed from querying the seed model. Then the method randomly selects one model in these models to be applied on a given input example. The randomness of selection can reduce the success rate of the attack. The evaluation results demonstrate that MULDEF augmented the adversarial accuracy of the target model by about 35-50% and 2-10% in the white-box and black-box attack scenarios, respectively.

However, Moosavi-Dezfooli et al. (2017) pointed out that no matter how many adversarial examples are added, there are new adversarial examples that can cheat the trained networks. For permanent robustness against adversarial attacks, MARL algorithms must incorporate real-time formal verification to ensure reliability and security.

4.2 Defense strategies in RL systems

While much work has been done in defense for machine learning models, little has been transported to RL settings. As mentioned, Gleave et al. (2021) provide a framework for adversarial training in RL, but Tramer et al. (2020) point out that these adversarially trained models are still susceptible to adversarial examples outside the distribution of examples they were presented with during training.

Oikarinen et al. (2020) propose a new framework for training RL agents robust to adversarial behaviors, labeled RADIAL-RL, which trains agents based on a worst-case reward, in contrast to the environment reward. This worst-case reward is computed using existing methods to identify the worst-case perturbations for a given ϵ budget for attacking. RADIAL-RL has mixed results when compared to RL baselines, but generally outperforms when presented with adversarial examples. This work draws ideas from the pre-existing field of RL, Safe RL. Safe RL focuses on maximizing an objective function, while accounting for possible worst-case scenarios or other optimization criterion. Safe RL primarily has applications in robotics, where systems risk critical failures, but

could potentially be adapted for protection against adversarial attacks. We identify this as an avenue for potential future work.

5 Formal Verification in RL

Formal Verification in ML is the process of showing mathematically that an ML model is robust against adversarial examples, given some ϵ budget constraint. This work generally strives to calculate the worst-case perturbation for an input for specified ϵ budgets. Weng et al. (2018) provide a derivation for computing this value quickly in ReLu networks. Lütjens et al. (2020) adapt this framework to RL, by computing the certified lower bound of the output of the Q-function. $Q_L(S_{adv}, A_j) := \min_{S \in B_p(S_{adv}, \epsilon)} Q_l(S, A_j)$ They then define the optimal action under his perturbation as the argmax of the Q function given the epsilon-ball around the worst-case perturbation. We can evaluate robustness against attacks by comparing the chosen actions with this derived optimal action.

References

Sarra Alqahtani, Joseph McCalmon, Cam Lischke, and Frank Liu. Adversarial policies and defense in cooperative multi-agent reinforcement learning. 2021.

Sean L. Barton, Nicholas R. Waytowich, and Derrik E. Asher. Coordination-driven learning in multi-agent problem spaces. *CoRR*, abs/1809.04918, 2018. URL `http://arxiv.org/abs/1809.04918`.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks, 2017.

Tong Chen, Wenjia Niu, Yingxiao Xiang, Xiaoxuan Bai, Jiqiang Liu, Zhen Han, and Gang Li. Gradient band-based adversarial training for generalized attack immunity of a3c path finding, 2018.

Tong Chen, Jiqiang Liu, Yingxiao Xiang, Wenjia Niu, Endong Tong, and Zhen Han. Adversarial attack and defense in reinforcement learning-from ai security view. *Cybersecurity*, 2, 12 2019. doi: 10.1186/s42400-019-0027-x.

Kaustav Das, J. Schneider, and D. Neill. Anomaly pattern detection in categorical datasets. In *KDD*, 2008.

Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Salvatore Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. *Applications of Data Mining in Computer Security*, 6, 02 2002. doi: 10.1007/978-1-4615-0953-0_4.

Kieran J Flanagan, E. Fallon, P. Connolly, and A. Awad. Network anomaly detection in time series using distance based outlier detection with cluster density analysis. *2017 Internet Technologies and Applications (ITA)*, pages 116–121, 2017.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients, 2017.

Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning, 2021.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. 2015. doi: arXiv:1412.6572.

Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick D. McDaniel. Adversarial perturbations against deep neural networks for malware classification. *CoRR*, abs/1606.04435, 2016. URL `http://arxiv.org/abs/1606.04435`.

Mahmudul Hasan, Md. Milon Islam, Md Ishrak Islam Zarif, and M.M.A. Hashem. Attack and anomaly detection in iot sensors in iot sites using machine learning approaches. *Internet of Things*, 7:100059, 2019. ISSN 2542-6605. doi: https://doi.org/10.1016/j.iot.2019.100059. URL `https://www.sciencedirect.com/science/article/pii/S2542660519300241`.

Weiwei Hu and Ying Tan. Generating adversarial malware examples for black-box attacks based on gan, 2017.

Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies, 2017.

Jieyu Lin, Kristina Dzeparoska, Sai Qian Zhang, Alberto Leon-Garcia, and Nicolas Papernot. On the robustness of cooperative multi-agent reinforcement learning, 2020.

Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. Tactics of adversarial attack on deep reinforcement learning agents, 2019.

S. W. Loke. Cooperative automated vehicles: A review of opportunities and challenges in socially intelligent vehicles beyond networking. *IEEE Transactions on Intelligent Vehicles*, 4(4):509–518, 2019.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020.

Björn Lütjens, Michael Everett, and Jonathan P. How. Certified adversarial robustness for deep reinforcement learning, 2020.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. 04 2015.

Fokrul Alom Mazarbhuiya, Mohammed Y. Alzahrani, and Lilia Georgieva. Anomaly detection using agglomerative hierarchical clustering algorithm. In *ICISA*, 2018.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations, 2017.

Gerhard Münz, Sa Li, and Georg Carle. Traffic anomaly detection using kmeans clustering. In *In GI/ITG Workshop MMBnet*, 2007.

S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han. Enhanced network anomaly detection based on deep neural networks. *IEEE Access*, 6:48231–48246, 2018. doi: 10.1109/ACCESS.2018.2863036.

Tuomas Oikarinen, Tsui-Wei Weng, and Luca Daniel. Robust deep reinforcement learning through adversarial loss, 2020.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings, 2015.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning, 2017.

A. Peake, J. McCalmon, B. Raiford, T. Liu, and S. Alqahtani. Multi-agent reinforcement learning for cooperative adaptive cruise control. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 15–22, 2020. doi: 10.1109/ICTAI50040.2020.00013.

R. Primartha and B. A. Tama. Anomaly detection using random forest: A performance revisited. In *2017 International Conference on Data and Software Engineering (ICoDSE)*, pages 1–6, 2017. doi: 10.1109/ICODSE.2017.8285847.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning, 2018.

Alessio Russo and Alexandre Proutiere. Optimal attacks on reinforcement learning policies, 2019.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017.

Dheeraj Basant Shukla and Gajendra Singh Chandel. An approach for classification of network traffic on semi-supervised data using clustering techniques. *2013 Nirma University International Conference on Engineering (NUiCONE)*, pages 1–6, 2013.

Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training, 2020.

Siwakorn Srisakaokul, Yuhao Zhang, Zexuan Zhong, Wei Yang, Tao Xie, and Bo Li. Muldef: Multi-model-based defense against adversarial examples for neural networks, 2019.

Ralf C. Staudemeyer and Christian W. Omlin. Evaluating performance of long short-term memory recurrent neural networks on intrusion detection data. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, SAICSIT '13, page 218–224, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450321129. doi: 10.1145/2513456.2513490. URL `https://doi.org/10.1145/2513456.2513490`.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses, 2020.

James Tu, Tsunhsuan Wang, Jingkang Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Adversarial attacks on multi-agent communication, 2021.

Alexander Verner. *LSTM Networks for Detection and Classification of Anomalies in Raw Sensor Data*. PhD thesis, 04 2019.

Ermo Wei, Drew Wicke, David Freelan, and Sean Luke. Multiagent soft q-learning, 2018.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S. Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks, 2018.

Tsui-Wei Weng, Krishnamurthy (Dj) Dvijotham*, Jonathan Uesato*, Kai Xiao*, Sven Gowal*, Robert Stanforth*, and Pushmeet Kohli. Toward evaluating robustness of deep reinforcement learning with continuous control. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SylL0krYPS`.

Rey Wiyatno and Anqi Xu. Maximal jacobian-based saliency map attack, 2018.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Sk9yuql0Z`.

S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang. Random forest for credit card fraud detection. In *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pages 1–6, 2018. doi: 10.1109/ICNSC.2018.8361343.

J. Zhang and M. Zulkernine. A hybrid network intrusion detection technique using random forests. In *First International Conference on Availability, Reliability and Security (ARES'06)*, pages 8 pp.–269, 2006a. doi: 10.1109/ARES.2006.7.

J. Zhang and M. Zulkernine. Anomaly based network intrusion detection with unsupervised outlier detection. In *2006 IEEE International Conference on Communications*, volume 5, pages 2388–2393, 2006b. doi: 10.1109/ICC.2006.255127.

Jiong Zhang and Mohammad Zulkernine. Network intrusion detection using random forests. In *PST*, 2005.