Generating Adversarial Examples for Static PE Malware Detector Based on Deep Reinforcement Learning

Jun Chen, Jingfei Jiang, Rongchun Li and Yong Dou

National University of Defense Technology, Hunan Changsha 410000, China
Email: chenjun3530@163.com

Abstract. Machine learning technology has been applied in filed of malware detection; it can improve efficiency of malware detection to deal with more and more increasing malware variants. However, malware detection model based on machine learning also has weakness which can be cheated by adversarial examples. Researching on method of generating adversarial examples could be benefit to exposing vulnerability of malware detection model and designing better malware detector. In this paper, we propose a reinforcement learning environment named gym-malware-mini based on gym-malware through which we generate adversarial examples using DQN and A2C deep reinforcement learning algorithm. As a result, DQN agent learned better policy of generating adversarial examples in gym-malware-mini, Success rate is increased by 18% compared with gym-malware. Success rate of DQN agent and A2C agent is increased by 20% and 15% compared with random agent in gym-malware-mini environment.

1. Introduction

In recent years, computer network is rapidly developing, so as the malware. Traditional method isn't enough to detect large number of malwares and new variants. Machine learning becomes important technology to build malware detection model. Static PE malware detection is one of research fields using machine learning. PE (Portable Executable) is an executable file format in Windows system. Most malware in Windows system is PE malware. Static detection is a kind of malware detection method which only uses static feature of application program without executing it.

Malware detector based on machine learning depends on feature of program which can be changed easily without affecting function of the program. Adversarial examples can be generated autonomously by modifying static feature of program and keep normal execution. Deep reinforcement learning(DRL) [1] could learn policy of generating adversarial examples for malware detector. DRL is a type of machine learning which consists of agent and environment. Agent constantly interacts with environment and learns policy via feedback of environment.

Anderson H S et al. [2] proposed a gym-malware model which is a DRL environment and it can be used to generate malware adversarial examples. A static PE malware detector is contained in gym-malware, it can judge whether a PE program is malware or not by it’s static feature, and malware detector is attack target of adversarial examples. Gym-malware has 11 actions which could be used to modify malware to cheat malware detector. Agent can learn best action series to attack the malware detector via it's feedback. Wu C et al. [3] proposed gym-plus model which makes improvement of gym-malware by adding new actions in action space. DQN, SARSA and Double DQN algorithm have been used in gym-plus and DQN learned better policy than other algorithms. Accuracy of malware detectors is improved through retraining on the adversarial examples generated by DQN agent. Fang Z et al. [4] proposed DQEAF framework which only has 4 actions. They check whether modified
malware can execute normally or not by executing them in Cuckoo sandbox. Environment state is shorter in dimension than gym-malware. Chen B et al. [5] studied adversarial examples of MalConv malware detector, including black-box attack and white-box attack. They try to enhance the MalConv using adversarial examples. Choi J et al. [6] generates adversarial examples by inserting code to source-code of malware.

In our paper, we proposed a DRL environment named gym-malware-mini based on gym-malware. Gym-malware-mini has smaller state space and action space compared with gym-malware, and a new reward function has been designed. DQN and A2C are two different kinds of DRL algorithm and have been used to train agent to learn policy of generating adversarial examples. DQN has learned better policy in gym-malware-mini than in gym-malware. Both DQN agent and A2C agent learned better policy than random policy in gym-malware-mini environment.

2. Method
The paper which proposed gym-malware [2] also trained an agent by DQN algorithm. Result is not very good that success rate is increased by 1% compared with random policy. We propose gym-malware-mini to improve gym-malware so that DRL algorithm could learn better policy. DQN[7] and A2C[8] algorithm are used to train in gym-malware-mini. The same DQN algorithm also has been used to train in gym-malware so that we can compare gym-malware-mini and gym-malware.

2.1. Propose Gym-malware-mini DRL Environment
Gym-malware [2] has 11 actions in action space, actually the number of action is more than 11 because some actions have random number and will modify malware randomly, the same action will lead to different state if generated random number is different. Huge action space makes the scale of search space large; it is difficult to learn good policy for DRL algorithm.

Gym-malware-mini modified action space of gym-malware. There are only 10 deterministic actions in action space of gym-malware-mini. There are 6 actions have been changed from randomness to deterministic in gym-malware, they are overlay_append, imports_append, section_rename, section_add, section_append and upx_pack. We get another 4 actions from gym-malware directly, they are remove_signature, remove_debug, upx_unpack and break_optional_header_checksum. Action space of gym-malware-mini is smaller compared with gym-malware, but these actions have almost same ability of evading malware detector as actions in gym-malware because they contain all kinds of PE modifying skills in gym-malware. Deterministic and less actions in action space make DRL algorithm easier to learn better policy.

Reward function of gym-malware is that reward will be 10 if evading malware detector successfully, and it will be 0 if failed or one episode is over. New reward function has been designed in gym-malware-mini, environment will return 10 when action evaded malware detector successfully, otherwise it will return -1. Maximum step in one episode is 10, which is the same as gym-malware. This reward function could make agent learn faster policy compared with reward function in gym-malware because actions that failed to evade malware detector will be punished.

Gym-malware-mini contains malware detector and malware modifying operations. DQN and A2C algorithm can train agent by interacting with gym-malware-mini. Relationship between agent and gym-malware-mini is shown in figure 1.
2.2. Design DQN Algorithm for Gym-malware-mini

DQN algorithm [7] has an action-value network that is a deep neural network and outputs value of each action at current state. Action which gets maximum value is the best action. The algorithm makes use of action-value network to fit true action value of each state. Policy is derived from action-value network. For gym-malware-mini, we design a deep neural network which has three fully-connected layers as action-value network. Batch Normalization is added before ReLu activation layer. L2_normalize layer is added after output layer. These tricks are good for network convergence at training stage. Detailed information of network structure is shown in Table 1.

| Layers | Type                  | Number of units |
|--------|-----------------------|-----------------|
| 0-1    | Dense                 | 256             |
| 1-2    | Batch Normalization   |                 |
| 2-3    | ReLU                  |                 |
| 3-4    | Dense                 | 128             |
| 4-5    | Batch Normalization   |                 |
| 5-6    | ReLU                  |                 |
| 6-7    | Dense                 | 10              |
| 7-8    | L2_normalize          |                 |

Figure 1. Relation graph of gym-malware-mini and agent

For balance of exploit and exploration, epsilon-greedy method is used to choose action when training the network; agent randomly chooses action from action space at probability of epsilon. True action value is calculated by TD algorithm which adds current reward and maximum action value of next state multiplying gamma factor [7]. The complete procedure of DQN training is as follows:

(1) Initialize DQN network parameter, gym-malware-mini environment, hyper-parameter of DQN training algorithm.

(2) Loop traversal:

(2.1) Gym-malware-mini gets the state which is a feature vector of malware.

(2.2) DQN network calculates action value at current state, epsilon-greedy algorithm chooses action to be executed.

(2.3) Gym-malware-mini executes the action to modify the malware, and returns reward depend on detection result of malware detector.

(2.4) If one episode is over (evading successfully or reaching maximum step), a new malware will be sampled randomly as new state. Otherwise, state will be transferred to malware modified.

(2.5) Store the state transition into replay buffer.
2.6) Sample mini-batch randomly from replay buffer, update action value using TD method, and update DQN network parameter with gradient descent optimization algorithm.

2.3. Design A2C Algorithm for Gym-malware-mini

A2C algorithm [8] contains actor network and critic network. Actor network outputs probability of each action. Action with maximum probability will be chosen. Critic network calculates state value. For gym-malware-mini, critic network and actor network are made up of three fully-connected layer neural networks. Batch Normalization is added before ReLu activation layer. The output of critic network is clipped to [-5, 5] for faster convergence. The layer information is shown in table2.

Table 2. A2C actor network and critic network structure. Critic network 6-7 layer has 1 unit.

| Layers | Type          | Number of units |
|--------|---------------|-----------------|
| 0-1    | Dense         | 256             |
| 1-2    | Batch Normalization |                |
| 2-3    | ReLU          | 128             |
| 3-4    | Dense         |                 |
| 4-5    | Batch Normalization |            |
| 5-6    | ReLU          |                 |
| 6-7    | Dense         | 10 (1)          |

Epsilon-greedy method is used to balance relation between exploit and exploration at training stage. New state value that is updated by TD method is used to train the critic network at training stage. Actor network is trained by policy gradient optimization method which uses advantage value calculated by critic network [8]. The complete procedure of A2C training is as follows:

1) Initialize actor network and critic network parameter, gym-malware-mini environment, hyper-parameter of A2C training algorithm.

2) Loop traversal:
   1.1) Gym-malware-mini gets the current state which is a feature vector of malware.
   1.2) actor network calculates each action probability at current state, epsilon-greedy algorithm chooses action.
   1.3) Gym-malware-mini executes the action to modify the malware, and returns reward depend on detection result of malware detector.
   1.4) If one episode is over (evading successfully or reaching maximum step), a new malware will be sampled randomly as new state. Otherwise, state will be transferred to malware modified.
   1.5) Store the state transition into replay buffer
   1.6) Sample mini-batch randomly from replay buffer, update state value by TD method, calculate advantage value. Update critic network parameter with gradient descent algorithm, update actor network by policy gradient method with advantage value.

   In our work, evaluation method of DRL algorithm is different from method used by original gym-malware which evaluate DQN agent in reserved 200 malware samples which have no intersection with training data-set. We evaluate DRL agent in all samples of gym-malware-mini. Generalization ability of model depends on data-set distribution and network structure. DRL algorithm has no direct correlation with generalization ability. Evaluation method this paper used is more accurate to evaluate DRL algorithm.

3. Evaluation

We downloaded 835 malware samples from Malshare website [9]. 299 malware samples have been chosen by random policy through gym-malware-mini. These 299 malware samples can be correctly detected as malware by malware detector in gym-malware-mini and can cheat malware detector using 10 actions in gym-malware-mini. These 299 malware samples are put into gym-malware-mini and gym-malware for following training and evaluation.
3.1. Evaluate DQN, A2C Algorithm

Hyper-parameter of DQN is chosen through experiment. DQN agent is evaluated at intervals of every specific update step, the final result is average of 10 times training results. Evaluation method of DQN algorithm in gym-malware is the same as in gym-malware-mini. Both training curves are shown in figure 2. Result shows that DQN algorithm reaches convergence after 1500 step update for gym-malware-mini and after 2000 step update for gym-malware.

A2C algorithm also has many hyper-parameters, they are adjusted by experiment. A2C agent is tested in 299 malware samples after specific update step. We train A2C agent with 10 times, the success rate has been recorded for each training, final result is average of 10 training results. Training curve of A2C in gym-malware-mini can be seen in figure 2. A2C algorithm reaches convergence after 500 step update.

![Figure 2](image)

**Figure 2.** Training curves of DQN algorithm for gym-malware-mini and gym-malware. Training curve of A2C algorithm for gym-malware-mini

3.2. Result Analysis

Random policy is used in gym-malware-mini and gym-malware for comparison. Random policy will randomly choose action from action space. Final result of random policy is average of 10 times training results. Success rate of random policy is 63% and 59% corresponding to gym-malware-mini and gym-malware respectively. All results are shown in table 3.

|                     | Gym-malware-mini | Gym-malware |
|---------------------|------------------|-------------|
| Random policy       | 63%              | 59%         |
| DQN                 | 83%              | 65%         |
| A2C                 | 78%              |             |

Table 3. All of the experiment result.

DQN algorithm gets highest success rate both in gym-malware-mini and gym-malware, and it has higher success rate in gym-malware-mini than in gym-malware. Difference between gym-malware-mini and gym-malware is action space and reward function. Experiment result proves that gym-malware-mini environment is better for DRL algorithm than gym-malware. A2C algorithm also learned a policy that is better than random policy in gym-malware-mini, A2C reaches convergence faster than DQN although it has lower success rate than DQN agent.
4. Conclusion
In this work, we mainly use deep reinforcement learning to generate malware adversarial examples. Gym-malware-mini environment is designed which is better than gym-malware for DRL algorithm. DQN and A2C algorithm have learned policy of generating adversarial examples. Purpose of our work is to make malware detector better through researching on adversarial examples. Accuracy of malware detector could be improved by retraining model on adversarial examples or building model to detect adversarial examples in advance.

5. Acknowledgments
This work was supported in part by National Major Special Projects of Nuclear High Base (2018ZX01028101), National Key Research & Development Program of China (2018YFB1003400), the National Key Research and Development Program of China under No. 2018YFB1003405, the Key Program of National Natural Science Foundation of China under No. 61732018, the National Natural Science Foundation of China (Grant No. 61802419).

6. References
[1] Sutton R S, Barto A G. Reinforcement learning: An introduction[J]. 2011.
[2] Anderson H S, Kharkar A, Filar B, et al. Learning to evade static PE machine learning malware models via reinforcement learning[J]. arXiv preprint arXiv:1801.08917, 2018.
[3] Wu C, Shi J, Yang Y, et al. Enhancing machine learning based malware detection model by reinforcement learning[C]//Proceedings of the 8th International Conference on Communication and Network Security. 2018: 74-78.
[4] Fang Z, Wang J, Li B, et al. Evading anti-malware engines with deep reinforcement learning [J]. IEEE Access, 2019, 7: 48867-48879.
[5] Chen B, Ren Z, Yu C, et al. Adversarial examples for CNN-based malware detectors [J]. IEEE Access, 2019, 7: 54360-54371.
[6] Choi J, Shin D, Kim H, et al. AMVG: Adaptive Malware Variant Generation Framework Using Machine Learning[C]//2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC). IEEE, 2019: 246-24609.
[7] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv:1312.5602, 2013.
[8] Mnih V, Badia A P, Mirza M, et al. Asynchronous methods for deep reinforcement learning[C]//International conference on machine learning. 2016: 1928-1937.
[9] A free Malware repository providing researchers access to samples, malicious feeds, and Yara results.https://malshare.com/. Accessed: 2020-04-06.