

Black Box Attacks Using Adversarial Samples Against Machine Learning Malware Classification To Improve Detection

Raphael Labaca Castro and Gabi Dreo Rodosek

Department of Computer Science, CODE
Universität der Bundeswehr München, Germany
{`raphael.labaca`, `gabi.dreo`}@unibw.de

Abstract. Machine learning algorithms are every day more accurate and are getting ready to be deployed ubiquitously. In fact, many technologies have been using them for years such as credit card fraud predictions and spam detection whilst others are starting to evaluate implementations like autonomous vehicles and different applications in IT security. However, the accuracy of these algorithms has prevailed over its security implication and hence the latter was not a priority when improving this technology. Therefore, adversarial learning is important in general to ensure AI safety because it forces the model to act in unexpected ways, which allows to observe its behavior under worst case input. The goal of researching adversarial machine learning on malware classifiers is not only to predict how a bad actor could cause security issues through malware misclassification but also to generate adversarial samples that can help retrain these models to improve its detection.

Keywords: Adversarial learning · Malware classifier · Machine learning

1 Introduction

Machine learning *per se* is nothing new in malware detection. Early research has been published since 1991 [1] and some security companies have been working with neural networks for over a decade. The technology has been improving ever since and nowadays is a popular approach to detect a big part of unseen malware as these algorithms, when properly trained, are able to generalize and classify a portable executable (PE) sample between benign or malicious. Implementations to detect malicious executables also date back since early 2000's, when Schultz et al. created a dataset and its labels with the help of a McAfee antivirus [2] and extracted the list of DLLs, DLL function calls and strings as features. Later in 2004, Kolter et al. implemented an approach [3] taking two datasets of clean and malicious executables of similar size and encoded them using millions of byte level n -grams as features. Further approaches focused on extracting only seven format specific features to predict malicious samples [4] and afterwards to categorize PE samples based on functions of their payloads. However, most of the

efforts focused on improving performance by measuring different metrics such as accuracy or area under the ROC curve as well as keeping false positive and false negative rates low. That means, most classifiers were only measured by its ability to tell apart malicious from benign without considering that attackers might be able to target the classifier forcing it, for instance, to misclassify samples. In other words to increase its false-positive rate using adversarial techniques. The rest of this paper is structured as follows: section 2 is the background and current state-of-the-art, section 3 a problem description followed by section 4 where we present our approach and section 5 the preliminary results. Ultimately, final considerations are given in section 6.

2 Background

Early research on adversarial learning has been conducted analyzing different attacks on an IDS learning algorithm that could cause it to misclassify events [5]. Additional contributions by Szegedy et al. focused on deep neural networks for image classification. They proposed to inject perturbations to images from some datasets, such as ImageNet [6], that were imperceptible for the human eye but mislead the model to classify the images incorrectly [7]. A gradient-based approach was proposed by Goodfellow et al. and focused on neural networks' linearity [8]. Further research by Papernot et al., on adversarial image classification in deep neural networks introduced a Jacobian Matrix that provided high evasion results with a low rate of modification of the input features [9]. Though, in order to identify which features have a stronger impact on classification, the attacker needs knowledge of the model's architecture. Similarly, adversarial examples with full knowledge of the model and access to the parameters were implemented on trained feed forward neural networks in order to deceive an Android malware classifier [10].

Malware is by nature a different problem than image classification as the input is rather binary than continuous and perturbations, which are invisible to the human eye, are no longer relevant. Xu et al., proposed a method to search for PDF adversarial samples by manipulating API calls from PE files and probing them against two PDF malware classifiers [11] by leveraging its score. A more recent approach from Hu et al., proposed a generative adversarial network (GAN) [12]. The idea is to create a substitute model that fits the black box classifier and a second model called generative network, which is trained to create adversarial samples and hence minimize detection rates on the substitute model [13]. Then it performs cross-evasion from the model to the actual black box classifier and reports perfect evasion for the samples analyzed. Similar to previous approaches, it requires to have deep knowledge from the target model. Recently, Anderson et al. proposed to target a more realistic scenario, where the attacker does not have such a knowledge and hence it is only able to probe the classifier in order to receive a binary output whether a sample is malicious or benign [14]. On the other hand, given the complexity of the scenario the success rates reported are modest, around 16%.

3 Problem description

The aim of the research is to use reinforcement learning (RL) to inject perturbations on malicious samples in order to deceive a malware classifier. The intention is to probe a static PE antivirus with perturbed adversarial samples and analyze its behavior under worst case input until it misclassifies a malicious sample as benign or a defined discrete number of stages is reached. Afterwards, these adversarial samples will be used to retrain the classifier in order to improve detection against adversarial techniques. The problem takes the following steps:

- Agent observes state s at discrete time step t : $s_t \in S$
- Learning agent selects action a at step t : $a_t \in A(s_t)$
- Agent receives reward $r_t = r(s_t, a_t)$ and next state $s_{t+1} = \delta(s_t, a_t)$
- Updates policy π at step t : $\pi_t(s, a)$

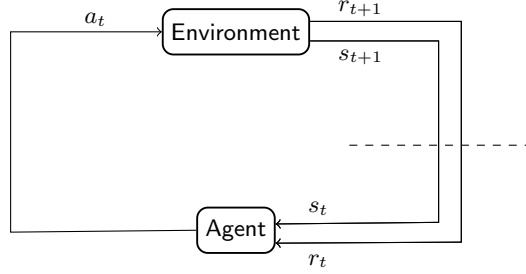


Fig. 1. Representation of the reinforcement learning problem.

To each policy a value can be assigned $V^\pi(s_t) = \sum_{i=0}^{\infty} \gamma^i r_{t+1}$, where $V^\pi(s_t)$ is the cumulative value that an agent receives by following a policy π and γ is a discount factor for future rewards. Therefore, the agent's optimal policy is defined as:

$$\pi^* \equiv \underset{\pi}{\operatorname{argmax}} V^\pi(s), (\forall s) \quad (1)$$

The states will be mapped using static features that include, for example, PE headers and sections, imports and exports, resources. The actions need to be defined carefully focusing on preserving the sample's functionality. Thus, we will only add features, and not remove, mainly because we should not get higher misclassification rates at the expense of losing functionality.

In summary, we have identified the following research questions, which outline the main aspects of the research:

1. At which extent can an attacker create automatic perturbations on malware samples to avoid detection without affecting the malware's functionality?

2. How successful can be an attacker in transferring results from one classifier to another?
3. How to protect classifiers against adversarial attacks?
4. How much adversarial input does a classifier need to improve its robustness against adversarial samples?

4 Proposed approach

The goal of this research is to extensively analyze static features and actions to perform on adversarial samples in order to misclassify malicious PEs as benign by using reinforcement learning without knowledge of the target model. Then, feed the classifier back with these new malware mutations to analyze how detection techniques can be improved. Current research usually approaches attacks with some knowledge of the model, which is more a theoretical case than a real scenario. On the other hand, research that indeed targets black box models are on its early stages and hence success rates are still low.

Research question 1 aims to understand modifications on a PE sample without affecting its structure. Unlike image classification, keeping integrity is one of the biggest challenges in adversarial research on malware samples. Therefore, changes need to be subtle to avoid affecting internal functionality but strong enough to pass the classifier.

Research question 2 focuses on assess how cross-evasion from target model to further classifiers or products would work. Given the fact that an attack will adjust to bypass a target model, how likely it is to use the same malicious mutations across different classifiers successfully.

Research question 3 is about remediation. Adversarial samples will prove that these models are not reliable under worst case input but how to make them safer still remains an open issue. We expect to address whether retraining using adversarial samples is effective against these threats or hardening machine learning antimalware engines might be a better option.

Finally, research question 4 aims to evaluate how much would the agent need to reinforce its policy until the model is effectively deceived or a certain number of iterations without modifying the PE structure is achieved.

5 Preliminary results

We started by collecting data and testing perturbations that can be successfully integrated into specific samples and do not affect its functionality whilst it gets misclassified by scanners. One way to apply the modifications is to use a library that provides representations of PE files into raw data like LIEF (Library to Instrument Executable Formats). It consists of a cross platform library, which allows to parse and modify PE files. Once the file is rendered into bytes, manipulations can be performed and later on the new sample compiled. Initial testing showed that the library might return errors for some samples, which may lead to apply a pre-filtering step on the input data as well as to analyze further libraries

to parse PE files. Additionally, we are testing different RL frameworks trying to find the best solution to implement perturbations efficiently.

6 Final considerations

Adversarial examples are important to help us be able to trust machine learning predictions. Even without the presence of a bad actor adversarial samples can provide a better understanding about a classifier when unexpected input is given. Although our approach is to understand behavior under pressure despite of whether an adversary is able to replicate an attack today, we will focus on keeping the scenario as realistic as possible as this could be an issue in the future.

Unlike adversarial attacks, defending a model is much more complex and therefore it will probably be an additional challenge to our approach. However, we expect a better understanding by researching both offensive and defensive techniques combined.

References

1. Guinier, Daniel. "Computer 'virus' identification by neural networks." ACM SIGSAC Review 9.4 (1991): 49-59.
2. Schultz, Matthew G., et al. Data mining methods for detection of new malicious executables. Security and Privacy, 2001. S&P. Journal Proceedings. IEEE Symposium (pp. 38-49). IEEE (2001).
3. Kolter, J. Z., & Maloof, M. A. Learning to detect malicious executables in the wild. Journal Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 470-478). ACM (2004).
4. Shafiq, M. Zubair, et al. A framework for efficient mining of structural information to detect zero-day malicious portable executables. Technical Report, TR-nexGINRC-2009-21, available at <http://www.nexginrc.org/papers/tr21-zubair.pdf> (2009).
5. Barreno, Marco, et al. Can machine learning be secure? Proceedings of the ACM Symposium on Information, computer and communications security. ACM (2006).
6. Deng, Jia, et al. Imagenet: A large-scale hierarchical image database. Computer Vision and Pattern Recognition. CVPR. IEEE Conference on. IEEE (2009).
7. Szegedy, Christian, et al. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013).
8. Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014).
9. Papernot, Nicolas, et al. The limitations of deep learning in adversarial settings. Security and Privacy (EuroS&P), IEEE European Symposium on. IEEE (2016).
10. Grosse, Kathrin, et al. Adversarial perturbations against deep neural networks for malware classification. arXiv preprint arXiv:1606.04435 (2016).
11. Xu, Weilin, Yanjun Qi, and David Evans. Automatically evading classifiers. Proceedings of the Network and Distributed Systems Symposium (2016).
12. Goodfellow, Ian, et al. Generative adversarial nets. Advances in neural information processing systems. (2014).
13. Hu, Weiwei, and Ying Tan. Generating adversarial malware examples for black-box attacks based on GAN. arXiv preprint arXiv:1702.05983 (2017).
14. Anderson, Hyrum S., et al. Learning to Evade Static PE Machine Learning Malware Models via Reinforcement Learning. arXiv preprint arXiv:1801.08917 (2018).