CIA evasion attacks transferability between machine learning models

Boussad ADDAD, Jerome Kodjabachian, and Christophe Meyer

Theresis, Thales Group, Palaiseau, FRANCE
surname.name@thalesgroup.com

Abstract. In the aftermath of the outstanding success of machine learning during the recent years, many investigations have shown that this technique is prone to evasion attacks, also known as adversarial examples, at test time. An adversarial example is a modified copy of the original data that aims to fool a machine learning model. However, this modification is to be not only efficient but also as minimal as possible to avoid detection. For this purpose, we developed a method called CIA (Centered Initial Attack) in [23] that guarantees the perturbation to be smaller than a given threshold. The first demonstration was made on image classification. In this paper, we show its applicability in the context of cyber security . More precisely, on malware detection in PDF files. Since transferability between machine learning models is an important property of adversarial examples, we studied the impact of crafting them using CIA. It turns out after our experiments that this method displays higher transferability rates compared to one of the most well-known attacks called iter-FGSM. This is demonstrated on both artificial neural networks and traditional ML techniques like SVM and random forests.

Keywords: Machine learning · Model evasion · Adversarial examples · PDF Malware · Transferability.

1 Introduction

Since the success of machine learning (ML) in different domains, the trend is towards its application to cyber security as a countermeasure to zero day attacks that are not detected by the traditional signature-based defense techniques [1], [2]. Obviously many security threats can be addressed, depending on the viewpoint. When dealing with networks security, we talk about NIDS (Network Intrusion Detection Systems). Many investigations have been carried out with regard to the possibility of application of machine learning [3], [4], [5] and [6]. On the other hand, when dealing with end points security, we talk about HIDS (Host-based Intrusion Detection Systems). Some published works in this context are [7], [8], and [9]. While the previous investigations address large scopes, by analyzing the global behaviour of a machine, other works target specific threats like malware detection in files. PDF being one of the most used files formats and given the dramatic increase in related attacks, many papers have been published to address the issue using machine learning [10], [11]

and [12]. Other files formats like executable files [13], [14] and Microsoft office documents [15], [16] have been investigated for the same reasons. Thanks to the use of machine learning, all the previous mentioned publications often display satisfactory results with intrusion detection rates approaching one hundred percent and low false alarm rates. Unfortunately, these solutions inherit also the current drawbacks of this technique. Indeed, as pointed out by many investigations, machine learning is vulnerable to evasion attacks a.k.a adversarial examples which are data manipulation to mislead ML models. Many approaches to forge adversarial examples have been proposed for that purpose; fast gradient sign method (FGSM) [17], iter-FGSM [18], saliency map [19] deep fool method [20], and using genetic algorithms [21]. In order to evade the models while minimizing the perturbation added to the original data, a recent technique known as C&W (After Carlini and Wagner) has been proposed in [22]. However, this perturbation is not guaranteed to be smaller than a fixed threshold. A clipping is therefore applied to respect strictly this constraint. Unfortunately, this process may degrade the quality of the adversarial example. So, another approach called CIA (Centered Initial Attack) has been developed in [23] to avoid the clipping process and respect the threshold constraint by design. It was demonstrated in [23] how this method improves the success rate of attacks in image classification task on ImageNet dataset against five classifiers and even against a voting ensemble classifier constituted of all of them.

In this paper, we present the use of CIA in the context of cyber security and evaluate its impact on the transferability of adversarial examples between ML models, a very important feature with regard to model evasion attacks. The sequel is organized as follows: Section 2 gives some recalls about the theory behind adversarial examples and how they are crafted using iter-FGSM and CIA methods. The transferability principle is also explained before depicting some experiments and results in Section 3. Finally, a conclusion is given with some perspectives for future work.

2 Adversarial examples principle

Models evasion can be applied to most machine learning algorithms [24], from classical ones like SVM, KNN or random forests to the newest deep neural networks. While we show how to attack both kinds of models via transferability, we focus in this paper on forging attacks using neural networks as a proxy model. A neural network can be seen as a function $F(x) = y$ that accepts an input x and produces an output y. The function F depends actually on some model parameters often called weights and biases. These are the variables that are adjusted during the learning process to fit the training data on one hand and generalize well to unseen data on the other hand. Since they do not change in our models, we omit them in our notations. The input x can be a vector or an array of any dimension. So, without loss of generality, we consider $x \in \Re^n$ as it can be flattened in any case. So, the i^{th} component of x is noted x_i with $i \in [1, n]$.

When considering m-class classifiers, the output is calculated using the $softmax$ function. The sigmoid function is used for binary classification. The output $y = F(x)$ can be seen as a vector of m probabilities p_j with $j \in [1, m]$. The component with the biggest value gives the predicted class $C(x)$ as:

$$C(x) = \arg\max_{j \in [1,m]} \{p_j\}$$

We note the output corresponding to the correct class as $C_c(x)$. An adversarial example noted $\hat{x}$ can be targeted or non targeted. It is crafted as a non targeted attack so as to get $C(\hat{x}) \neq C_c(x)$ or a targeted one to get $C(\hat{x}) = t$ where t is the target class. Crafting an example can then be formulated using a loss function $L(F, x)$ to maximize the probability of getting a class different from the correct one. Cross entropy is used in the current study. The adversarial example $\hat{x}$ can be written as $\hat{x} = x + \delta$ where δ is the added perturbation. We constrain δ to be within domain $[-\Delta, \Delta]$, Δ being the maximum perturbation. For room reasons, we recall very briefly only three methods, the ones which have a clear relation with the current paper. For interested readers, some defense methods are also developed in many papers as in [25], [26], [27] but unfortunately not always that successful as pointed out in [28] and demonstrated using CIA in [23].

Some attacks theory

Fast Gradient Sign Method (FGSM): The method FGSM is introduced in [17]. Given an input x, the fast gradient sign method sets an adversarial example as:

$$\hat{x} = x - \epsilon \cdot sign[\frac{\partial L(F, x)}{\partial x}] \tag{1}$$

, where ϵ is chosen to be smaller than Δ. Intuitively, the fast gradient sign method uses the gradient of the loss function to determine in which direction the original data should be changed (whether it should be increased or decreased) to minimize the loss function i.e. to maximize the probability to misclassify $\hat{x}$.

Iterative Fast Gradient Sign Method (iter-FGSM): This method [26] is a simple refinement of the fast gradient sign method. Instead of taking a single step of size ϵ in the direction of the gradient sign, multiple smaller steps are taken, and the result is clipped to respect the maximum threshold Δ. Iterative gradient sign was found to produce superior results to fast gradient sign [26].

Centered Initial Attack (CIA): With the approach above, adversarial examples are generated through some iterations then clipped in the end to respect the threshold constraint. With C&W attacks proposed in [22] for instance, the loss function includes a norm term $\|\delta\|$ to minimize perturbation δ but the clipping is still needed. The CIA method has been developed to avoid this. We give here only briefy the main concepts of this method. More details can be found in [23].

The main idea behind Centered Initial Attack is to find for each component i the center x_i^* of the domain in which $\hat{x}_i$ is allowed to vary. This is not trivial

since we have to insure at the same time the component $\hat{x_i}$ to be within another domain $[\alpha_i, \beta_i]$ to be valid, α_i and β_i are respectively the minimum and maximum values that can be taken by the i^{th} feature variable i.e i^{th} component of x and $\hat{x}$. For instance, all components must be in domain [0, 255] for RGB images. Such constraint is needed almost all the time as data are normalized to [0, 1] or [-1, 1] before feeding neural networks. Now if we consider a continuous differentiable function g such that: $g : \Re \rightarrow [-1, +1]$, then we can write component $\hat{x_i}$ as:

$$\hat{x_i} = x_i^* + \Delta_i^* \cdot g(r_i) \tag{2}$$

This equation can be rewritten using arrays as:

$$\hat{x} = x^* + \Delta^* \odot g(r) \tag{3}$$

where operator $\odot$ is the elementwise product and r is a new variable on which we optimize the loss function. Finally, the loss function can be written as:

$$L_r = L(F, x^* + \Delta^* \odot g(r)) \tag{4}$$

No constraint is to be considered on the variable r since it is well defined in domain $(-\infty, +\infty)$. Any initialization of r is possible but we consider it as zero for simplicity. The initial attack $\hat{x}$ is therefore different from x whereas it is centered in its domain of definition. This explains the CIA attack name. It is interesting to note that any gradient descent optimizer can be used to craft attacks using CIA. Adam [29] turns out to be the fastest in our experiments and therefore we adopted it.

Transferability of adversarial examples

This is probably the most important property of adversarial examples. It consists in the fact that an example forged to mislead a machine learning model A is likely to fool another model B. Moreover, the more knowledge we get about the victim, the higher is the probability of transferability [30]. We therefore define three categories of attacks, depending on this knowledge:

White box attacks: in this case, we know the exact model used by the victim and have all its parameters.

Black box attacks: in this case, we don't know the model but we can query it and get the output corresponding to any input. In this case, we can build a proxy substitute model and train it using the collected data. This improves highly the rate of transferability [30].

No-box attacks: in this case, we cannot collect any data from the victim. Obviously, the attacks are more difficult to transfer but not impossible as we will see in the next section.

It is worth noting that the knowledge of the model is not the only parameter to consider when regarding transferability. The data and features used for training the victim model for instance are important and if we get them, our chance to succeed in attacks increases [30]. As a rule of thumb, the more knowledge we get the more the attacks are likely to be transferred. In our experiments, we considered no-box attacks to harden the task and increase the objectivity of our results since this case is the most met one in practice.

3 Experiments and results

In this study, we focus on the task of discriminating between benign and malicious PDF files, an important medium for spreading malware [31]. As a matter of fact, PDF files are excellent vectors for malicious-code, due to their flexible logical structure, which can be described by a hierarchy of interconnected objects. As a result, an attack can be easily hidden in a PDF to circumvent file-type filtering. The PDF format further allows a wide variety of resources to be embedded in the document, including JavaScript, Flash, and even binary programs. The type of the embedded object is specified by keywords, and its content is in a data stream. As already mentioned in the introduction, several recent works proposed machine-learning techniques for detecting malicious PDFs using the files logical structure to accurately identify the malware. In the current study, we use the feature representation of Maiorca et al. [32] in which each feature corresponds to the number of occurrences of a given keyword. To extract these features, we used a Python script developed in [33]. In this study, we limit the features to the 21 most important keywords defined by the same author. One should be aware that since this features extraction is applied the same way for both attacker and victim models in our study, it can legitimately be considered as an a priori knowledge about the victim. The relaxation of this hypothesis is in our roadmap but left for future investigations.

For experiments, we used a PDF corpus with around 10,000 malicious samples and 10,000 benign samples from the Contagio dataset [34]. To play the role of victim model, we considered three different models; SVM (Support Vector Machine), Random Forests and an Artificial Neural Network (ANN). As a proxy attack model, we built an ANN. To limit the correlation between attacker A and victim V, we split the dataset into three sub-datasets: the first to train model A, the second to train V, and the third to perform transferability tests from A to V. In other words we craft an adversarial example, a malicious file modified to pass the detection, using model A then send it to V to check the transferability. If model V classifies a malicious file as benign, then we consider that the attack is successfully transferred.

To compare the performance of CIA and iter-FGSM, we carried out many experiments by varying the maximum perturbation threshold delta defined in section 2. The results are depicted on Fig. 1. It is worth noting that due to the nature of features used in our classifiers which are positive integers (number of occurrence of keywords), the adversarial examples are rounded to the nearest

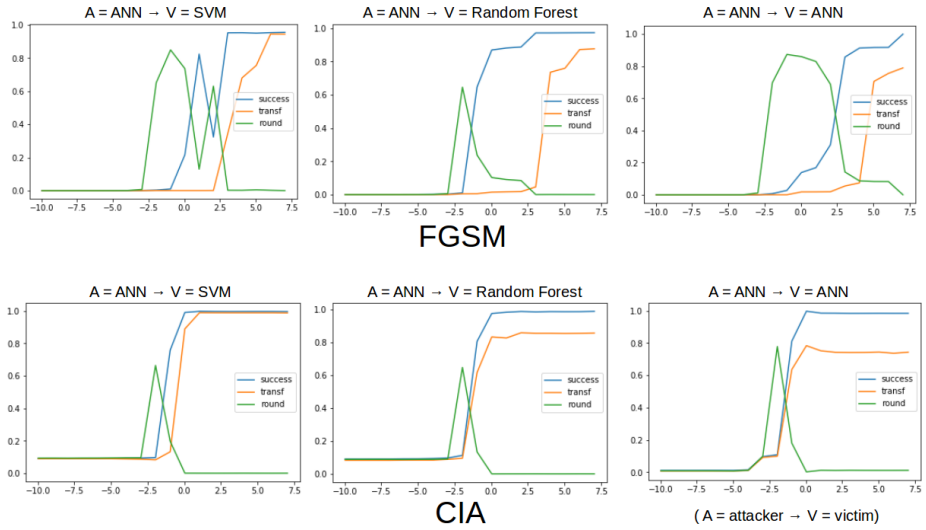

Fig. 1: attacks success rate as a function of the maximal perturbation threshold (for visual purpose, we used a logarithmic scale). Blue curve: success rate of attacks on proxy model A. Orange curve: transferability success rate from A to V. Green curve: attacks failure rate on A due to rounding features.

integers. So, in addition to the degradation due to clipping, solved using CIA, the attacks undergo another one due to this features rounding as can be seen on Fig. 1. This phenomenon concerns all ML-based algorithms using this kind of features. It should be thoroughly investigated but this is out of the scope of this study.

Discussion of the results: As expected, when the perturbation threshold is very small, no attack is successful, even on the proxy model A. However, beyond some value, depending on the case, the attacks start to fool the models. Obviously, this occurs earlier on the proxy model than the victim model. This demonstrates clearly that if we perform white box attacks, which can be seen as the case here with regard to the proxy model, then the attacks are more successful. Full knowledge of the victim means high sucess rates of attacks. When looking at the transferability curves (orange), we note that whatever the target algorithm is, the attacks are highly transferred if the threshold is enough big. All these results are not original and surprising. However, when we compare the performance of CIA w.r.t iter-FGSM, we can see clearly that the success rate of attacks is higher using CIA. This is the case on the proxy model but more striking when we look at the transferability curves. The orange curve follows closely the blue curve, which means that almost all attacks are transferred from the proxy to the victim. This can be explained by the fact that CIA exploits more optimally the domain of variation of data given the threshold that has to be respected. This is not the case using iter-FGSM which faces a dilemma to respect this constraint: either

it uses a high epsilon step to update the data or more iteration to cover better the domain but with the risk of overshoots and therefore clipping the added perturbation. This degrades the attack as already highlighted in [23]. If we try to avoid clipping using very small epsilon, then the domain is not well exploited since the gradient descent would be very slow.

4 Conclusion and perspectives

In this paper we demonstrated that the CIA method is more effective to forge adversarial examples, that are likely to be transferred to different types of target models, than other approaches like iter-FGSM. It was also proven to resist better to feature rounding, a weakness of CIA and all ML-based classifiers handling integers. This is an important result showing that even with a very limited knowledge about the attacked machine learning model and the rounding constraint, it can sometimes be fooled with a high probability. This naturally raises questions about the readiness of applicability of machine learning for security tasks. The good news is that we can take profit from CIA as an aggressive method to develop better strategies of defense. Isn't it known that the best defense is attack? This is one of our current investigations.

References

1. Alazab et al, Zero-day Malware Detection based on Supervised Learning Algorithms of API call Signatures. Proceedings of the 9-th Australasian Data Mining Conference, 171-181 (2011).
2. Gavrilut et al, Malware Detection Using Machine Learning. Proceedings of the International Multiconference on Computer Science and Information Technology, 735-741 (2009).
3. Quamar Niyaz et al, A Deep Learning Approach for Network Intrusion Detection System, International Conference on Wireless Networks and Mobile Communications (WINCOM) (2016).
4. M. Garuba, C. Liu, D. Fraites, "Intrusion Techniques: Comparative Study of Network Intrusion Detection Systems", Fifth International Conference on Information Technology: New Generations, pp. 592-598, 2008
5. O. Depren, M. Topallar, E. Anarim, M. K. Ciliz, "An Intelligent Intrusion Detection System (IDS) for Anomaly and Misuse Detection in Computer Networks", Expert Systems with Applications, pp. 713-722, 2005
6. W. Gang, H. Jinxing, M. Jian, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering", Expert systems with applications, vol. 376, pp. 6225-6232, 2011
7. D. Yeung and Y. Ding, Host-based intrusion detection using dynamic and static behavioural models, Pattern Recognition, vol. 36, no. 1, pp. 229243, 2003.
8. G. Creech and J. Hu, "A Semantic Approach to Host-based Intrusion Detection Systems Using Contiguous and Discontiguous System Call Patterns," in IEEE Transactions on Computers (2013).
9. Baskaran et al, Balaji, A Study of Android Malware Detection Techniques and Machine Learning. Proceedings of the 27th Modern Artificial Intelligence and Cognitive Science Conference, 15-23 (2016).

10. Igino Corona et al, Detection of Malicious PDF-embedded JavaScript code through Discriminant Analysis of API References, In the proceedings of Aritificial Intelligence and Security Workshop, 2014.
11. Jose Torres and Sergio De Los Santos, Malicious PDF Documents Detection using Machine Learning, A Practical Approach with Cloud Computing Applications Techniques, ICISSP 2018: 337-344.
12. Dragos Gavrilut et al, Malware Detection Using Machine Learning, Proceedings of the International Multiconference on Computer Science and Information Technology pp. 735741 (2009).
13. E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas. Malware detection by eating a whole exe. arXiv preprint arXiv:1710.09435, 2017.
14. J. Z. Kolter ve M. A. Maloof, "Learning to Detect Malicious Executables in the Wild", Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004.
15. Sergio De los Santos and Jos Torres, Macro Malware Detection using Machine Learning Techniques A New Approach, In Proc of the 3rd International Conference on Information Systems Security and Privacy (ICISSP 2017), pages 295-302.
16. R. Bearden and D-Chai-Tien Lo, Automated Microsoft Office Macro Malware Detection Using Machine Learning, IEEE International Conference on Big Data (2017).
17. Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In International Conference on Learning Representations (ICLR), 2015.
18. Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Examples in the Physical World. In International Conference on Learning Representations (ICLR) Workshop, 2017.
19. Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. In IEEE European Symposium on Security and Privacy (EuroSP), 2016.
20. Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: a simple and accurate method to fool deep neural networks. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
21. Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015
22. Nicholas Carlini and David Wagner. Defensive Distillation is not Robust to Adversarial Examples. arXiv preprint arXiv:1607.04311, 2016.
23. Boussad Addad, Jerome Kodjabachian, Christophe Meyer, clipping free attacks against artificial neural networks, https://arxiv.org/abs/1803.09468 (2018)
24. Nicolas Papernot et al, Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples, https://arxiv.org/abs/1605.07277, 2016.
25. Shixiang Gu and Luca Rigazio. Towards Deep Neural Network Architectures Robust to Adversarial Examples. arXiv preprint arXiv:1412.5068, 2014.
26. Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes and Patrick McDaniel. On the (Statistical) Detection of Adversarial Examples. arXiv preprint arXiv:1702.06280, 2017
27. Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On Detecting Adversarial Perturbations. arXiv preprint arXiv:1702.04267, 2017
28. Nicholas Carlini and David Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods, https://arxiv.org/abs/1705.07263, 2017.

29. KINGMA , D., AND B A , J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
30. Nedim rndi and Pavel Laskov, Practical Evasion of a Learning-Based Classifier: A Case Study, IEEE Symposium on Security and Privacy (2014).
31. Young, R.: 2010 IBM X-force mid-year trend and risk report. Tech. rep., IBM (2010)
32. Maiorca, D., Giacinto, G., Corona, I.: A pattern recognition system for malicious pdf files detection. In: MLDM. pp. 510524 (2012)
33. https://blog.didierstevens.com/programs/pdf-tools/
34. http://contagiodump.blogspot.it