Detecting Adversarial Examples - A Lesson from Multimedia Security

Pascal Schöttle, Alexander Schlögl, Cecilia Pasquini, and Rainer Böhme
Department of Computer Science, University of Innsbruck, Innsbruck, Austria
{pascal.schoettle; alexander.schloegl; cecilia.pasquini; rainer.boehme}@uibk.ac.at

Abstract—**Adversarial classification is the task of performing robust classification in the presence of a strategic attacker. Originating from information hiding and multimedia forensics, adversarial classification recently received a lot of attention in a broader security context. In the domain of machine learning-based image classification, adversarial classification can be interpreted as detecting so-called adversarial examples, which are slightly altered versions of benign images. They are specifically crafted to be misclassified with a very high probability by the classifier under attack. Neural networks, which dominate among modern image classifiers, have been shown to be especially vulnerable to these adversarial examples.**

However, detecting subtle changes in digital images has always been the goal of multimedia forensics and steganalysis, two major subfields of multimedia security. We highlight the conceptual similarities between these fields and secure machine learning.

Furthermore, we adapt a linear filter, similar to early steganalysis methods, to detect adversarial examples that are generated with the projected gradient descent (PGD) method, the state-of-the-art algorithm for this task. We test our method on the MNIST database and show for several parameter combinations of PGD that our method can reliably detect adversarial examples.

Additionally, the combination of adversarial re-training and our detection method effectively reduces the attack surface of attacks against neural networks. Thus, we conclude that adversarial examples for image classification possibly do not withstand detection methods from steganalysis, and future work should explore the effectiveness of known techniques from multimedia security in other adversarial settings.

Index Terms—**Adversarial Classification, Adversarial Examples, Multimedia Forensics, Steganalysis**

I. Introduction

The task of *adversarial classification* is to perform robust and reliable classification in the presence of strategic attackers [1]. The nature of a strategic attacker is that she will not disregard knowledge about possible defense mechanisms. Rather, she adapts her attack strategy to circumvent the most probable defense mechanisms [2].

In machine learning-based classification, state-of-the-art attacks are so-called *adversarial examples* [3]. Adversarial examples are benign inputs that have been strategically modified by an attacker such that they are misclassified with a very high probability and confidence. Initially, adversarial examples were generated against classifiers based on *convolutional neural networks* (CNNs), but soon it was shown that they generalize to other machine learning algorithms as well [4].

Adversarial classification against adversarial examples can be achieved in two different ways: either the designers of the CNNs try to detect adversarial examples as adversarial (*adversarial detection*) or they try to increase the robustness of the CNN in such a way that adversarial examples are classified in the class of the underlying benign example (*robust classification*). But, to this day, no method to detect adversarial examples effectively exists and earlier work from the area of *adversarial machine learning* [5], [6] did not prove to be useful against adversarial examples, either.

Although adversarial examples do not only exist for image classifiers (e. g., they also exist for malware classifiers [7]), the main body of work is performed for the area of digital images. Thus, we restrict ourselves to this domain.

Every method for generating adversarial examples from benign images calculates which pixels should be modified by how much (restricted by a distortion constraint) to maximize the probability of a misclassification, e. g., [3], [8]–[10].

Detecting subtle malicious changes in digital images has always been the goal of multimedia forensics and steganalysis[1]. Without explicitly using the term adversarial classification, both domains perform adversarial detection since the very beginning of scientific research in either of the fields. The strategic nature of the attackers here is defined by research in counter-forensics and steganography [11]. Both are implicitly aware of possible detection methods and try to evade them.

In the field of digital image forensics [12], a forensic investigator has to decide if a given image was manipulated by an image forger or not. Oftentimes, the image forger manipulates large connected parts of the image with the aim to change its semantic [11]. Thereby, she might use *tamper hiding* techniques [13] to conceal traces she expects the forensic investigator to identify.

In steganalysis, a steganalyst has to decide if a given image has a message embedded by a steganographer [14]. While embedding her message, the steganographer tries to modify individual pixel values in such a way that the steganalyst gets the least information about the fact that a message is embedded. To achieve this, every modern steganographic algorithm defines an adaptivity criterion that identifies which pixels are most suitable for embedding.

This research was funded by Archimedes Privatstiftung, Innsbruck, Austria and Deutsche Forschungsgemeinschaft (DFG) under grant "Informationstheoretische Schranken digitaler Bildforensik".

[1] Note that in steganography/steganalysis jargon usually the steganalyst is the attacker and the steganographer is the defender. We refrain from statements about who is good or bad, but reverse their roles in this paper.

ISBN 978-90-827970-1-5 © EURASIP 2018

Machine learning-based approaches, and especially detectors built on CNNs, are by now very common in image forensics and steganalysis. But, to the best of our knowledge, nobody has tried to go the other way around, i. e., to use established methods from multimedia security to detect adversarial examples against CNNs. However, adversarial examples are generated by changing individual pixels of a benign image, thus particularly resembling the embedding process in steganography.

We address this gap and give an intuition on why, how, and what the area of secure machine learning can learn from the field of steganalysis, by formalizing further parallels between them and providing evidence of its effectiveness in a practical scenario. We develop a steganalysis-inspired linear prediction method to detect adversarial examples that are generated with the *projected gradient descent* (PGD) method [9].

The remainder of the paper is organized as follows: Section II gives the background about secure machine learning and steganalysis, and highlights the parallels and differences of these fields. As a proof of concept, we develop our method to detect adversarial examples in Section III and show its effectiveness in Section IV. Section V concludes.

II. Background & Parallels

The publication closest to our work is [15]. The authors show the parallels of attacks against and defenses for secure machine learning and digital watermarking, another subfield of multimedia security. We argue that the detection of adversarial examples rather falls into the domain of steganalysis and encourage researchers to make use of established steganalysis methods before they start out to reinvent the wheel.

A. Secure Machine Learning

The underlying assumption of every machine learning-based classifier is that the training data follows the same, possibly unknown distribution as the test data. For example, a supervised CNN-based classifier that has to distinguish n different classes is trained with many samples x_i and their corresponding labels $i \in \{1, 2, \dots, n\}$. So, the CNN learns an approximation of the classification function $F(x_i) = i$, given a specific loss function $\ell_F(x_i, i)$, and predicts a label $i \in \{1, 2, \dots, n\}$ for every sample encountered during testing.

1) Creating Adversarial Examples: Intuitively, every (untargeted) adversarial examples starts from a benign example x_i. The attacker tries to find r, subject to a distortion constraint, such that $x_i + r$ gets misclassified by $F(\cdot)$. This can be achieved by solving the following optimization problem:

$$\begin{aligned} &\arg\min_r \; d(x_i, x_i + r), \\ &\text{s. t. } F(x_i + r) = i' \neq i, \end{aligned} \tag{1}$$

for a given distance metric d.

For a given x_i, we define the i-th class as the *benign class*, whose samples follow a distribution $\mathcal{P}_1$. Accordingly, we denote as $\mathcal{P}_0$ the distribution of samples belonging to every other class except i. This allows to transform every multi-class problem to the binary case. With a slight abuse of notation, the goal of an attacker is to modify an image $x_1 \sim \mathcal{P}_1$ such that it gets classified as drawn from $\mathcal{P}_0$.

Among all the methods proposed for the generation of adversarial examples, e. g., [3], [8], [10], the so-called *projected gradient descent* (PGD) method [9] constitutes the state-of-the-art at the time of writing.

The PGD method basically is an iterated variant of the *Fast Gradient Method* (FGM) [16], which takes a single step of value α in the direction of the gradient of the loss function $\nabla \ell_F$. Additionally, all pixel values are clipped to the range of 0 and 1 to ensure a valid image in the end. PGD introduces a second variable ε and sets $x^{[0]} = x_1$ to iteratively calculate

$$x^{[k+1]} = \text{clip}_{[x_1 - \varepsilon, x_1 + \varepsilon]}\big(\text{FGM}(x^{[k]})\big), \tag{2}$$

for a given number of iterations K. So, $x^{[K]}$ serves as an approximation of the optimal adversarial example $x_i + r$ in Eq. (1). The outer clipping ensures that $||x_1 - x^{[K]}||_\infty \leq \varepsilon$ to model an attacker that is restricted by the infinity norm. Note that depending on the value of the gradient of the loss function the PGD method changes individual pixel values up to a maximum of ε.

2) Proposed Countermeasures: Recent research on the countermeasures against adversarial examples mainly concentrates on increasing the robustness of the underlying neural network so that it classifies adversarial examples to the class of the benign object used for creating the example. One recent approach is to cut off the lower bit layers and only classifying the remaining image [17], in the hope that the adversarial modifications are concentrated in the lower bit layers. Another one is to re-train the neural network with adversarial examples [9], so-called *adversarial re-training*.

Although the nature of adversarial examples is still not fully known [18], it is accepted that they generalize over different networks: adversarial examples generated against one network are also likely to be misclassified by other networks [19].

B. Steganalysis

In steganalysis, the distribution $\mathcal{P}_0$ defines the distribution of all possible benign images (cover images), while $\mathcal{P}_1$ is the distribution of stego images. Every modern steganographic embedding function defines a so-called adaptivity criterion which measures the distortion when changing single pixels in a specific way. During the embedding of the message, the steganographer tries to minimize the overall distortion with the goal that the stego image $x_1 \sim \mathcal{P}_1$ gets classified by the steganalyst as drawn from $\mathcal{P}_0$. The steganographer decides on a per-pixel basis about the changes she introduces. The goal of the steganalyst is to decide for a given image x_i, from which distribution it was drawn. As this is a classic task for a neural network classifier, it comes as no surprise that CNN-based steganalysis attracted a lot of attention recently.

C. Parallels and Differences

We summarize the main parallels and differences of secure machine learning and steganalysis in Table I.

TABLE I
PARALLELS AND DIFFERENCES OF SECURE MACHINE LEARNING AND STEGANALYSIS

	Secure Machine Learning	Steganalysis
Attack point	adversarial example	stego image
Decision by	network (designer)	steganalyst
Attack algorithm	modification	embedding
Attack parameters	(internal) parameters	adaptivity criterion
Attack surface	individual pixels	individual pixels
$\mathcal{P}_0$: distribution of	other class(es)	cover images
$\mathcal{P}_1$: distribution of	benign class	stego images
Attacker's goal	get $x_1 \sim \mathcal{P}_1$ classified as drawn from $\mathcal{P}_0$	get $x_1 \sim \mathcal{P}_1$ classified as drawn from $\mathcal{P}_0$
Nature of $\mathcal{P}_1$	exogenously given	influenced by attacker

The main difference is the nature of the distribution $\mathcal{P}_1$. In secure machine learning, both distributions $\mathcal{P}_0$ and $\mathcal{P}_1$ are exogenously given from real-world examples, and the goal of the attacker is to modify images drawn from $\mathcal{P}_1$ so that they are classified as being drawn from $\mathcal{P}_0$. This can be regarded as moving x_1 across the decision boundary as far as possible under the given distortion constraint, e. g., ε for PGD.

Contrary to that, in steganalysis the distribution $\mathcal{P}_1$ is given by the images created by the steganographer, and can thus be influenced by her attack algorithm. So, the goal of a steganographer is to create a distribution $\mathcal{P}_1$ similar enough to $\mathcal{P}_0$, such that a steganalyst cannot reliably differentiate between objects drawn from $\mathcal{P}_1$ and objects drawn from $\mathcal{P}_0$.

A higher value of ε in PGD enables an attacker to move her adversarial examples farther into the space of $\mathcal{P}_0$ by changing individual pixels more. By doing so, the attacker makes it harder even for adversarially re-trained networks to correctly classify adversarial examples [9].

But, detecting objects that deviate from an expected distribution is exactly what established methods from steganalysis are designed to do. So, the higher ε for PGD is, and the less robust the CNN classifiers get, the better the performance of methods adapted from steganalysis should be.

III. Proof-of-Concept: Adapting Steganalysis

A. Method

One of the simplest and earliest steganalysis methods is based on the intuition that pixels that were changed during the embedding behave different than pixels that were not changed [14]. For example, in a (unmodified) cover image, the original pixel values should be estimable from values of the surrounding pixels. If pixel i was changed during embedding, its observed value x^i will deviate from the estimated value $\hat{x}^i$. This estimation can be achieved by a simple linear filter of the following form [20]:

$$\hat{x}^i = x^i * \begin{pmatrix} -1/4 & 1/2 & -1/4 \\ 1/2 & 0 & 1/2 \\ -1/4 & 1/2 & -1/4 \end{pmatrix}.$$

Taking the average of the weighted differences over all n pixels in an observed image can serve as an indicator if a message was embedded or not.

$$\hat{p} = \frac{1}{n} \sum_{i=0}^{n-1} w_i \left(x^i - \hat{x}^i \right) \tag{3}$$

If $\hat{p}$ is relatively small, it can be expected that the image is a cover image. The weights w_i in Eq. (3) account for local predictability, and one successful initialization [20] suggest that $w_i^{-1} \propto 5 + \sigma_i^2$ give accurate estimates, where σ_i^2 denotes the local variance in the neighborhood of pixel i (but excluding the center pixel). It was shown that such an estimator, adapted to a specific way of changing the pixel values during embedding [20], coincides with an asymptotically uniformly most powerful (AUMP) hypothesis test [21].

B. Experimental Setup

We test our method on the MNIST dataset [22] against adversarial examples generated by the PGD method [9]. The MNIST dataset contains 60 000 grayscale images of handwritten digits, which are split up into 50 000 images in the *training set* and 10 000 images in the *test set*.

The detection of adversarial examples is performed by calculating $\hat{p}$ as in Eq. (3) for every image and classifying it as adversarial if $\hat{p}$ is above a certain threshold $\bar{p}$. We chose a conservative approach and set $\bar{p}$ to the maximum value obtained from the 50 000 images from the training set, so that $\bar{p}$ ensures no false positives on the training set.

To ensure reproducibility, we did not train the CNNs ourselves but fetch the *natural* and *secret* models from [9][2]. Here, the secret model was re-trained with adversarial examples generated by PGD against the natural model with $\varepsilon = 0.3$.

Furthermore, we use the provided *attack* script to generate adversarial examples for every image in the test set, for every $\varepsilon \in \{0.01, 0.02, \ldots, 0.5\}$ for both models. This brings the total number of adversarial images used for testing to 100 000[3]. All adversarial examples are generated with full knowledge of the CNN they are targeting, making this a white-box attack, and

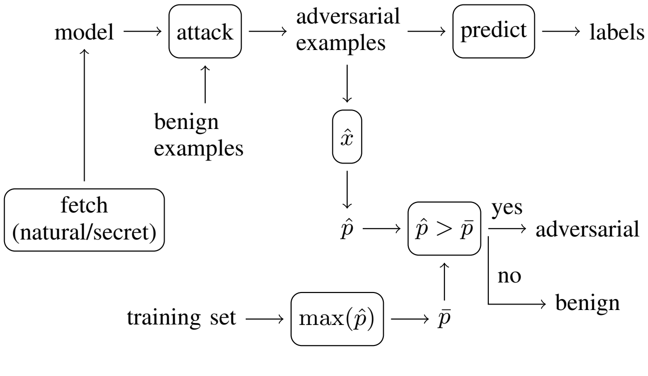

Fig. 1. Block diagram of our experiments' workflow

[2] Available at: https://github.com/MadryLab/mnist_challenge

[3] 50 (values for ε) × 2 (models) × 10 000 (images in the test set) = 100 000

TABLE II
TRUE POSITIVE RATE OF OUR DETECTOR (OVER ALL 10 000 IMAGES OF THE MNIST TEST SET)

Model	ε														
	0.010	0.050	0.100	0.150	0.200	0.250	0.300	0.330	0.350	0.370	0.400	0.420	0.450	0.470	0.500
natural	0.000	0.000	0.529	0.982	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
secret	0.000	0.000	0.000	0.001	0.009	0.025	0.047	0.137	0.320	0.574	0.848	0.937	0.985	0.995	0.999

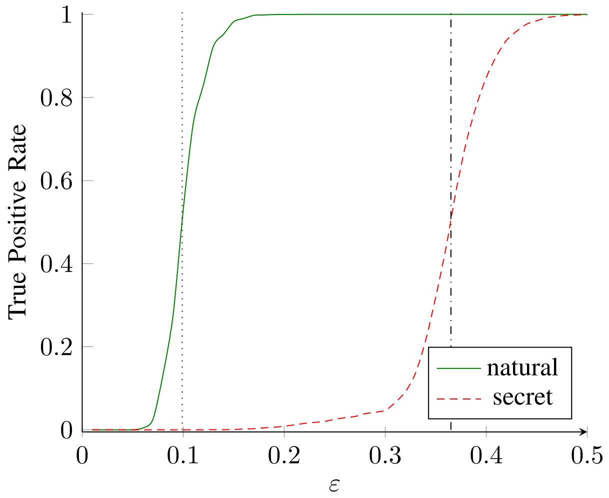

Fig. 2. True positive rate of our method for different ε

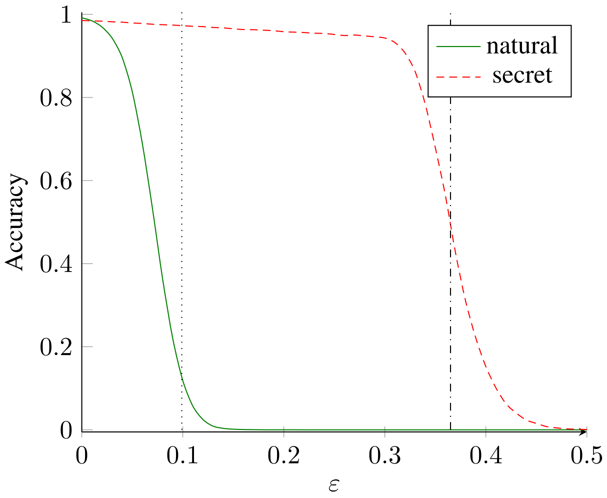

Fig. 3. Accuracy of both classifiers from [9] for different ε

thus the worst-case scenario for a defender [23]. The overall setup of our experiments[4] is depicted in Fig. 1.

IV. RESULTS

A. Detecting Adversarial Examples

As explained in Sec. III-B, our method is constructed in such a way that false positives are extremely unlikely and indeed, in none of our tests we encountered a single benign image that was classified as adversarial by our method.

Thus, to fully assess the performance of our method, it is enough to report the true positive rate (TPR), i. e., the amount of adversarial examples that were correctly identified by our method. Table II lists the TPR for the test set for different values of ε. As we can see, our method improves for higher values of ε for both models. This is to be expected, as with increasing value of ε, the adversarial examples will deviate farther from $\mathcal{P}_1$ and thus are better detectable by our method.

It is interesting to observe that attacks against the secret model are harder to detect for our method. First tests about the difference of the adversarial examples created against the re-trained model (omitted here due to space constraints) show that these adversarial examples change more pixel values in a homogeneous way, thus lying closer to our pixel prediction.

We plot the TPR for both models and all tested values of ε in Fig. 2. In comparison to the accuracy of the models from [9] (cf. Fig. 3), it is observable, that our method indeed improves approximately at the point where the CNN classifiers lose robustness (the dotted (dash-dotted) vertical line indicates where our method achieves $\approx 50\%$ TPR for the natural (secret) model). This orthogonal behavior of our adversarial detection and the robust classification obtained by adversarial re-training motivates a combination of both approaches.

B. Combination of our method with adversarial re-training

To test the combination of our method and the adversarially re-trained CNNs from [9], we first create adversarial examples for every value of ε for every image from the test set. Then, we apply our method to decide if the example is adversarial or not. All images that are not detected as adversarial by our method are handed to the CNN classifiers which predict their labels, see Fig. 1. We plot the accuracies of the combined approach for both models in Fig. 4. As we can see, for the natural model the accuracy never falls below 50% and for the secret model the accuracy in never less then 96%. This confirms that our method detects a majority of the adversarial examples that would have been misclassified by the CNNs.

It is notable that for the natural model we achieve perfect separation of adversarial and benign samples for $\varepsilon \geq 0.17$ (dotted vertical line in Fig. 4), which is also the exact value where the undefended network's accuracy is zero (cf. Fig. 3).

For the secret model we achieve perfect separation for $\varepsilon \geq 0.31$ (dash-dotted vertical line in Fig. 4), whereas an attacker would need $\varepsilon \geq 0.47$ to achieve zero accuracy (cf. Fig. 3).

[4] Our code is available at: https://github.com/alxshine/stego4secureML

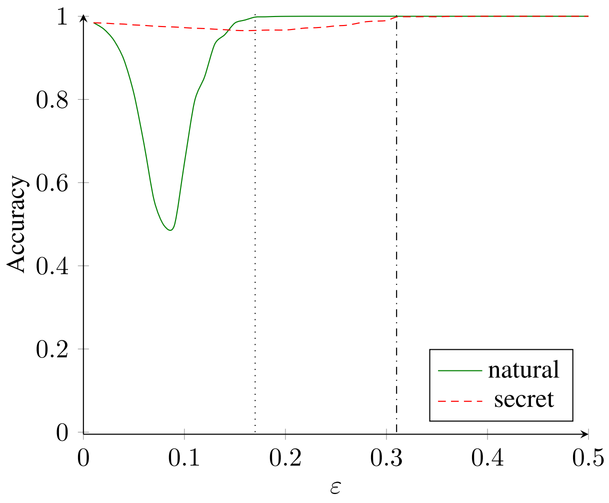

Fig. 4. Accuracy of the combination of both methods for different ε

V. Conclusion

Adversarial classification in the area of secure machine learning can roughly be divided into adversarial detection and robust classification. While the latter approach gained more attention recently, we argue in this paper that the detection of adversarial examples crafted against CNN-based classifiers can draw on long established methods from steganalysis.

We highlight the conceptual parallels between the creation of adversarial examples and the generation of stego images and develop a very simple method to reliably detect adversarial examples generated by the PGD method. Furthermore, we give theoretical insights on why methods adapted from steganalysis can successfully complement robust classification: they are designed to perform well against exactly the adversarial examples that are hard to classify robustly.

An even better performance can be achieved by combining our method with adversarial re-trained CNNs. The minimum accuracy of the combined approach over all parameters is 96%, almost at par with the accuracy of the tested CNNs for benign images. An additional benefit of the combined approach is that it efficiently reduces the freedom of an attacker, as it is very hard to defeat our method and adversarially re-trained CNNs with the same adversarial examples.

For our proof-of-concept, we restrict ourselves to a very simple method from the domain of steganalysis. Future work should identify, which methods from the field of multimedia forensics can be leveraged to further improve the performance of adversarial detection in secure machine learning. For example, methods from the area of copy-move forgery detection could be adapted to identify adversarial examples that modify large connected parts of an image and thus cannot be reliably detected by our method.

The bigger picture suggests not only that machine learning is useful in steganalysis and multimedia forensics, but also that secure machine learning should learn from these fields.

References

[1] L. Dritsoula, P. Loiseau, and J. Musacchio, “A game-theoretic analysis of adversarial classification,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 3094–3109, 2017.

[2] M. Barni and F. Pérez-González, “Coping with the enemy: Advances in adversary-aware signal processing,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8682–8686.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *International Conference on Learning Representations (ICLR)*, 2014.

[4] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” *arXiv preprint arXiv:1605.07277*, 2016.

[5] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, “Can machine learning be secure?” in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*. ACM, 2006, pp. 16–25.

[6] B. Biggio, B. Nelson, and P. Laskov, “Poisoning attacks against support vector machines,” in *29th International Conference on Machine Learning*, J. Langford and J. Pineau, Eds. Omnipress, 2012, p. 1807–1814.

[7] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, “Adversarial examples for malware detection,” in *European Symposium on Research in Computer Security*. Springer, 2017, pp. 62–79.

[8] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in *IEEE European Symposium on Security and Privacy*. IEEE, 2016, pp. 372–387.

[9] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *International Conference on Learning Representations*, 2018.

[10] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *IEEE Symposium on Security and Privacy*. IEEE, 2017, pp. 39–57.

[11] M. Kirchner and R. Böhme, “Hiding traces of resampling in digital images,” *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 4, pp. 582–592, Dec 2008.

[12] H. T. Sencar and N. Memon, *Digital Image Forensics: There is More to a Picture than Meets the Eye*. Springer Science & Business Media, 2013.

[13] M. Kirchner and R. Böhme, “Tamper hiding: Defeating image forensics,” in *Information Hiding*, T. Furon, F. Cayre, G. Doërr, and P. Bas, Eds. Springer Berlin Heidelberg, 2007, pp. 326–341.

[14] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. New York, NY, USA: Cambridge University Press, 2009.

[15] E. Quiring, D. Arp, and K. Rieck, “Forgotten siblings: Unifying attacks on machine learning and digital watermarking,” in *IEEE European Symposium on Security and Privacy*, 2018.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *arXiv preprint arXiv:1412.6572*, 2014.

[17] W. Xu, D. Evans, and Y. Qi, “Feature squeezing: Detecting adversarial examples in deep neural networks,” in *Network and Distributed System Security Symposium*, 2018.

[18] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, “The space of transferable adversarial examples,” *arXiv preprint arXiv:1704.03453*, 2017.

[19] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 86–94.

[20] A. D. Ker and R. Böhme, “Revisiting weighted stego-image steganalysis,” in *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, E. J. Delp III, P. W. Wong, J. Dittmann, and N. D. Memon, Eds., vol. 6819. SPIE, 2008, p. 681905.

[21] L. Fillatre, “Adaptive steganalysis of least significant bit replacement in grayscale natural images,” *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 556 –569, 2012.

[22] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database,” *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010.

[23] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, “SoK: Security and privacy in machine learning,” in *IEEE European Symposium on Security and Privacy*, 2018.

ISBN 978-90-827970-1-5 © EURASIP 2018