Hardware/Algorithm Codesign for Adversarially Robust Deep Learning

Mojan Javaheripi and Mohammad Samragh
University of California San Diego,
La Jolla, CA 92093 USA

Tara Javidi and Farinaz Koushanfar
University of California San Diego,
La Jolla, CA 92093 USA

Bita Darvish Rouhani
Microsoft, Redmond, WA 98052 USA

Editor's notes:
This article describes DeepFense, a framework to make deep learning models automatically and efficiently realizable on constrained devices.
—Rosario Cammarota, Intel Labs
—Francesco Regazzoni, University of Amsterdam and Università della Svizzera Italiana

Deep learning (DL) has provided a paradigmatic shift in devising automated systems that can surpass human performance in controlled environments. Although advanced learning techniques are essential for enabling coordination between autonomous agents and the environment, a careful analysis of their vulnerabilities and their reliability in face of malicious attacks is still in its infancy. To date, reliability and safety considerations pose major obstacles to the wide-scale adoption of emerging learning algorithms in sensitive scenarios, such as intelligent transportation, healthcare, and video surveillance.

Recent research showcases the vulnerability of DL models in face of adversarial attacks, i.e., carefully crafted input instances that lead machine learning algorithms into misclassifying while the input changes are imperceptible to the human eye. Figure 1 shows one such adversarial attack against DL-based object classifiers employed in self-driving cars. The adversary jeopardizes the vehicle's safety by adding a specific perturbation to a "stop" sign which misleads the DL model into classifying it as "yield." Thereby, it is highly important to equip autonomous systems with an accelerated defense that identifies and rejects potential adversaries in real-time.

This article features our research paper titled "DeepFense: Online Accelerated Defense Against Adversarial Deep Learning" [1] which is selected for the 2019 top picks in hardware and embedded security. DeepFense provides the first end-to-end hardware-accelerated framework that enables robust defense against adversarial attacks on DL models.

Our solution addresses three main challenges:

- Understanding the root cause of DL vulnerabilities to adversarial samples.
- Characterizing and thwarting the adversarial space for effective model assurance and defense against adversaries.
- Devising automated hardware tools for real-time accelerated defense.

We propose a novel method called modular robust redundancy (MRR). MRR is robust against the state-of-the-art adaptive white-box attacks in which the adversary knows everything about the victim model and the defense mechanism. The MRRs characterize the

Digital Object Identifier 10.1109/MDAT.2021.3063344
Date of publication: 2 March 2021; date of current version: 20 May 2021.

Copublished by the IEEE CEDA, IEEE CASS, IEEE SSCS, and TTTC

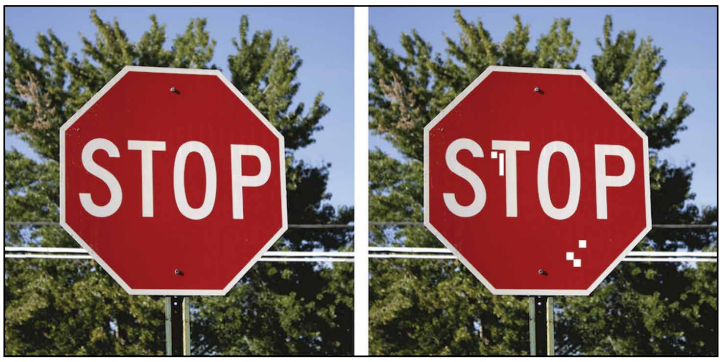

Figure 1. Left image is a legitimate "stop" sign that is correctly classified by the DL model. Right image is a perturbed adversarial input which is misclassified as "yield."

distribution of legitimate samples in the high-dimensional representations (feature-maps) of the DL model. The outlier samples are then marked as potential adversaries. Our MRR training is *unsupervised*, i.e., the training data set is merely composed of benign samples. This, in turn, ensures resiliency against potential new attacks.

DeepFense leverages hardware/algorithm codesign to enable safe DL while customizing system performance in terms of latency, energy consumption, and/or memory footprint with respect to the underlying resource provisioning. There is an inherent tradeoff between system performance and robustness against adversarial attacks that is determined by the number of MRRs. DeepFense provides an automated tool to adaptively maximize the defense robustness while adhering to the user-defined and/or hardware-specific constraints. We chose FPGAs to provide fine-grained parallelism and real-time response by our defender modules.

In summary, our contributions are as follows:

- Proposing DeepFense, the first hardware, software, and algorithm codesign that empowers online defense against adversarial samples for DNNs. DeepFense methodology is unsupervised and robust against the most challenging attack scenarios (white-box attacks).
- Devising an automated customization tool to adaptively maximize DL robustness against adversarial samples while complying with the underlying hardware constraints in terms of runtime, energy, and memory.
- Providing the first implementation of custom DL adversarial defense using FPGAs. DeepFense leverages dictionary learning and probability density function (PDF) estimation to statistically detect abnormalities in the input samples.
- Performing extensive proof-of-concept evaluations on modern DL benchmarks against a wide range of adversarial attacks. Thorough performance comparison on various hardware platforms including embedded CPUs, GPUs, and FPGAs corroborates DeepFense's efficiency.

Preliminaries and background

Adversarial attack algorithms

In an adversarial setting, the attacker aims to find a perturbed adversarial sample (x^a) such that it incurs minimal distance from the source sample (x^s) while its corresponding output is different enough to mislead the victim model. Several attack mechanisms have been proposed in the literature to craft adversarial samples. We evaluate DeepFense performance against a variety of attacks to empirically confirm the generalizability of our unsupervised MRR methodology across a wide range of attacks. In particular, we study: 1) fast gradient sign (FGS) [2]; 2) Jacobian saliency map attack (JSMA) [3]; 3) Deepfool [4]; 4) basic iterative method (BIM) [5]; and 5) Carlini&WagnerL2 adaptive attacks [6]. In the following, we provide a brief explanation of the attack algorithms evaluated in this article.

- *Fast gradient sign*: FGS attack [2] crafts an adversarial sample as $x^a = x^s + \epsilon \cdot \mathrm{Sign}\left(\frac{\partial C}{\partial x^s}\right)$ where C is the cost function of the DL model, and $\mathrm{Sign}(\cdot)$ outputs the sign of its operand. The attack is parameterized by ϵ, which determines the amount of additive perturbation.
- *Basic iterative* method: BIM [5] is an iterative version of FGS characterized by the number of iterative updates, n_{iters}, and the per-iteration perturbation coefficient, ϵ.
- *Deepfool*: This algorithm iteratively modifies the input image based on a specific update rule to obtain an adversarial sample [4]. In each iteration, the perturbation vector $\frac{\partial C}{\partial x^s}$ is normalized and added to the sample. Deepfool is parameterized by the number of iterative updates n_{iters}.
- *Carlini&WagnerL2*: This attack is formalized as a minimization problem where the objective is the

L2 norm of the perturbation vector. Carlini&WagnerL2[1] proposes an iterative method for solving this minimization objective. The detailed set of parameters for the attack is provided in [6].

Adversary threat models

Depending on the attacker's knowledge, the threat model can be categorized into three classes:

- *White-box attacks*: The attacker knows everything about the victim model including the learning algorithm, model topology, defense mechanism, and model/defender parameters.
- *Gray-box attacks*: The attacker only knows the underlying learning algorithm, model topology, and defense mechanism but has no access to the model/defender parameters.
- *Black-box attacks*: The attacker does not have any knowledge about the pertinent machine learning algorithm, DL model, or defense mechanism. This attacker can only obtain the outputs of the victim model corresponding to input samples. In this setting, the adversary can perform a differential attack by observing the output changes with input variations.

Present defenses

Several research attempts have been made to design DL strategies that are more robust in the face of adversarial examples. A line of work in *certifiable robustness* proposes to modify the training objective of DL models to guarantee robustness against norm-bounded adversarial attacks [7]. The majority of these methods are demonstrated for small data sets and simple DL models, under limited perturbation budget. Cohen et al. [8] is the first work to provide robustness guarantees for a large-scale visual data set (ImageNet). Nevertheless, their guarantee is still relatively limited in the amount of perturbation allowed. Several recent works rely on *adversarial retraining*, where the defender insets adversarial samples inside training data to enhance robustness [9]. Adversarial training is time-consuming and computationally prohibitive for large-scale data sets. Although recent work [10] significantly improves the training overhead, the victim model's accuracy degradation from adversarial training remains a standing challenge. DeepFense unsupervised countermeasure impacts neither the training complexity nor the final accuracy of the victim model. In another track of research, post-training defense methodologies have been proposed, where the victim model and its training routine remain unchanged. One example is the statistical characterization of model behavior in face of benign versus adversarial samples [11], [12]. Nevertheless, all aforementioned methods study the statistics of the victim model's features which cannot optimally identify adversarial subspaces. This is because the victim model's main objective is the accurate classification of benign samples. DeepFense alternatively changes the training objective for the defender to enable accurate probabilistic modeling of the adversarial samples. Our proposed MRR methodology does not assume any particular attack strategy and/or perturbation pattern. This is specifically important as it enables the generalizability of the proposed approach in the face of adversarial attacks.

Methods based on generative adversarial networks (GANs) aim at finding a distribution for adversarial perturbations (supervised defense) or clean data samples (unsupervised defense). At inference time, input images are processed through the GAN and potential adversarial perturbations are removed [13]. Another set of unsupervised defense strategies inject random operations into the inference phase of the neural network. The randomization can take various forms such as noise injection [14], resizing, and zero-padding inputs [15], or

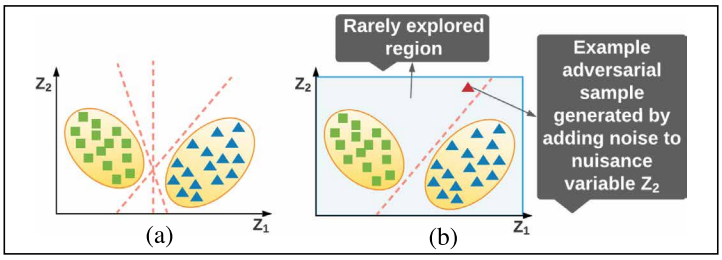

Figure 2. (a) Data points (green and blue squares) can be easily separated in one-dimensional space. Extra dimensions add ambiguity in choosing the decision boundaries: all shown boundaries (dashed lines) result in the same classification accuracy but are not equally robust to noise. (b) Rarely explored space in a learning model leaves room for adversaries to manipulate the noncritical dimensions (Z2 in this figure) and mislead the model by crossing the decision boundaries.

[1]For brevity we denote this attack as CarliniL2 in this article.

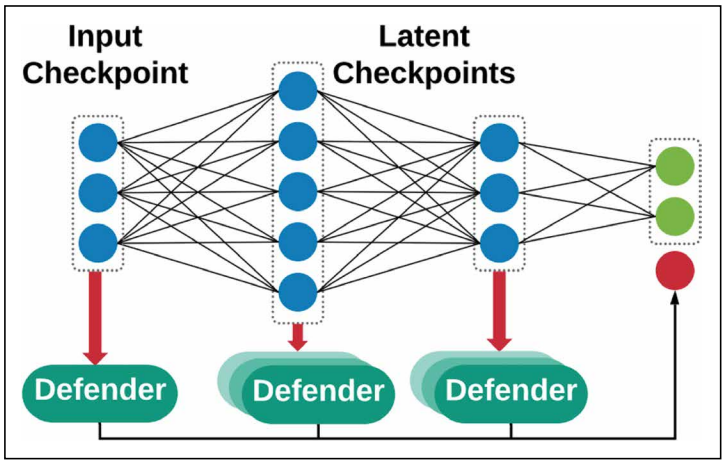

Figure 3. High-level block diagram of MRR methodology. Multiple defenders checkpoint the input and intermediate activation maps in parallel. The output of the victim neural network (green neurons) is augmented with a confidence measure (red neuron) determining the prediction legitimacy.

randomized compression [16]. The randomized operation may drop the accuracy of the network [17] and is not able to withstand white-box attacks that are aware of the randomness-injection being in place [18]. In contrast, DeepFense is able to defend against white-box attackers by employing multiple defender modules.

DeepFense overview

Understanding the root cause of DL vulnerability

Our hypothesis is that the vulnerability of deep neural networks (DNNs) to adversarial samples originates from the existence of rarely explored subspaces in each feature map. This phenomenon is particularly caused by the limited access to labeled data and/or inefficiency of regularization algorithms. Figure 2 provides a simple illustration of the partially exploredspace in a two-dimensional setup.

Let us denote the output of the ith layer of a DL model given an input sample x by $f_i(x)$. One can construct a probabilistic density function $P_X(f_i(x))$ for each layer where X is a random variable drawn from the model input space. Our conjecture is that adversarial samples cannot lie in high-probability regions of the PDF function $P_X(\cdot)$, which is learned using the samples drawn from legitimate training data. More formally, the expected value of the probability corresponding to legitimate samples is higher than that of adversarial samples:

$$E\left(P_X\left(f_i\left(x^s\right)\right)\right) >> E\left(P_X\left(f_i\left(x^a\right)\right)\right) \quad (1)$$

where $E(\cdot)$ is the expectation and x^s and x^a are safe and adversarial input samples, respectively. DeepFense leverages this difference between the expected values of the probability distribution to characterize and thwart adversarial attacks.

DeepFense MRR strategy for adversarial robustness

Building upon our hypothesis, we characterize the explored subspace in the input and hidden layers of the victim DL model by learning the PDF of legitimate data points and marking the complement subspaces as rarely observed regions. These PDF estimators are used by the MRRs to identify adversarial samples, i.e., samples that deviate from the learned PDFs at each layer. Figure 3 illustrates the main DL model along with the MRRs (defender modules). We refer to MRR modules that checkpoint the intermediate DL layers as "latent defenders." Whereas, the redundancy modules operating on the input space are referred to as the "input defenders."

The MRRs evaluate each incoming input sample in parallel with the victim model. The outputs of the defender modules are then aggregated into a single output node (the red neuron in Figure 3) that quantitatively measures the reliability of the original prediction. For any input sample, the new neuron outputs a risk measure in the unit interval [0, 1], with 0 and 1 indicating safe and highly risky samples, respectively. The extra neuron incorporates a "don't know" class into the model: samples with a risk factor higher than a certain threshold (also known as security parameter) are treated as adversarial inputs. The threshold is determined based on the safety-sensitivity of the application for which the ML model is employed. This approach is beneficial in the sense that it allows dynamic reconfiguration of the detection policy with minimal required recomputing overhead.

DeepFense hardware stack

DeepFense hardware acceleration stack enables online detection of adversarial samples. We devise a parameterized hardware library for implementation of DeepFense input and latent defenders on FPGA. We design customized kernels for sparse recovery

via orthogonal matching pursuit to execute the input defenders and perform input anomaly detection [1]. Additionally, all incoming samples are passed through the latent defenders which are executed via DeepFense DNN kernels on FPGA. To increase inference throughput, we convert the DNN layer computations into multiple parallel operations performed by individual processing units as suggested in [19]. DeepFense tunes the number of processing units (N_{PU}) using an automated customization unit to meet task-specific runtime constraints as will be explained in the "DeepFense global flow" section. We refer the reader to [1] for in-depth details of DeepFense hardware.

DeepFense global flow

The first step in deployment of DeepFense is the unsupervised training of MRRs. We devise a custom training objective for the MRRs that perform careful realignment of benign data manifolds to increase interclass data separation. The separated manifolds are then characterized with a unique PDF corresponding to each class. Once the MRRs are trained, DeepFense automatically generates the hardware implementation for the modules by performing two main phases as illustrated in Figure 4: 1) offline preprocessing phase to obtain the MRR configurations and 2) online execution phase in which the legitimacy of each incoming input data is validated on the fly.

Preprocessing phase: This phase consists of one main task, i.e., resource profiling and design customization. There is a tradeoff between the computational complexity (e.g., runtime overhead) of the modular redundancies and the overall system reliability in terms of successful adversarial detection rate. DeepFense uses physical profiling to estimate resource utilization for the victim model as well as the defender modules. These resources include storage blocks (i.e., block RAMs) and arithmetic units (i.e., DSP slices). The output of physical profiling along with a set of user-defined constraints (i.e., real-time requirements and security level) is then fed into the design customization unit. The customization unit analyzes the tradeoff between model reliability (robustness), resource limitation, and throughput to decide the best number of defenders suitable to the task and target hardware by solving the following optimization:

$$\max_{\mathcal{X}}(\mathrm{Robustness}\mathcal{X}) s.t. \begin{cases} T_{\max}(\mathcal{X}) \le T_u \\ \mathrm{Mem}(\mathcal{X}) \le M_u \\ \mathrm{DSP}(\mathcal{X}) \le R_u \end{cases} \quad (2)$$

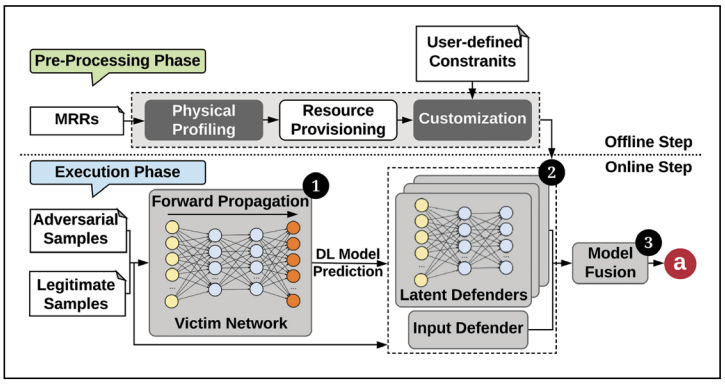

Figure 4. High-level flow of DeepFense. Based on the user-provided constraints, DeepFense outputs the best defense layout that ensures maximum robustness and throughput.

where T_u, M_u, and R_u are user-defined constraints for system latency, block random access memory (BRAM) budget, and available DSP resources, respectively. Here, represents all design parameters, including the number of defenders N_{def}, and the configuration of DeepFense hardware cores, i.e., the number of parallel processing units (N_{PU}). DeepFense performs an exhaustive search over the design parameter N_{PU} and solves (2) using the Karush–Kuhn–Tucker (KKT) method to calculate N_{def}. The above customization stage is performed only once and incurs a negligible overhead. The reader is referred to [20] for additional details.

Execution phase: Once the redundancy modules are customized per hardware and user-defined physical constraints, the DL model is ready to be deployed for online execution. DeepFense performs three tasks in the execution phase.

Forward propagation: The predicted class for each incoming sample is acquired through forward propagation in the victim DL model. The predicted output is then fed to defenders for validation.

Validation: DeepFense executes the learned MRRs on FPGA to validate the legitimacy of the input data and its associated label. We devise a parameterized hardware library for implementation of DeepFense input and latent defenders on FPGA. We design customized kernels for sparse recovery via orthogonal matching pursuit to execute the input defenders and perform input anomaly detection. Additionally, all incoming samples are passed through the latent defenders which are executed via DeepFense DNN kernels on FPGA. The output of the DNN kernel undergoes dimensionality reduction via customized

Table 1. AUC score obtained by 16 latent defenders that checkpoint the second-to-last layer of the victim model for MNIST and CIFAR-10 benchmarks. For ImageNet bench mark, we only used one defender due to the high computational complexity of the pertinent neural network and attacks.

	MNIST	SVHN	CIFAR-10	CIFAR-100	ImageNet
FGS	0.997	0.969	0.911	0.885	0.881
JSMA	0.995	0.995	0.966	0.961	-
Deepfool	0.996	0.974	0.960	0.850	0.908
CarliniL2	0.987	0.963	0.929	0.944	0.907
BIM	0.994	0.931	0.907	0.821	0.820

Table 2. Evaluation of MRR methodology against adaptive white-box attack. For each evaluation, the *L2* distortion is normalized to that of the attack without any defense present.

	DeepFense					Prior-Art Defenses		
						[22]	[23]	[13]
N_{def}	N=1	N=2	N=4	N=8	N=16	N=16	-	-
TP Rate	46%	63%	69%	81%	84%	1%	0%	0%
L_2 Distortion	1.09	1.28	1.28	1.63	1.57	1.37	1.30	1.06

PCA kernels. The distance calculation kernel then determines the legitimacy of the sample. In particular, outlier samples that are far from the learned PDF of benign samples are marked as adversarial.

Model fusion: The output of redundancy modules are finally aggregated to determine the legitimacy of the inputdata and its associated inference label. Specifically, we usethe noisy-OR model for decision aggregation and perform expectation maximization to fine-tune the underlying parametersof the noisy-OR.

Evaluations

Black-box attack resiliency

We perform state-of-the-art attacks on several visual classification benchmarks. Each attack is performed with various attack parameters to ensure that the evaluation data includes a diverse set of adversarial examples (refer to [1] for attack parameters). Below, we summarize the results obtained from the black-box threat model where the attacker has complete access to the parameters and the architecture of the victim model but is not aware of the defense mechanism. To assess DeepFense defense against the attack algorithms, we measure the area under curves (AUCs) which quantitatively assesses the tradeoff between DeepFense ability in detecting adversarial samples and false interpretation of benign samples as malicious data. Table 1 summarizes the detection performance of DeepFense.

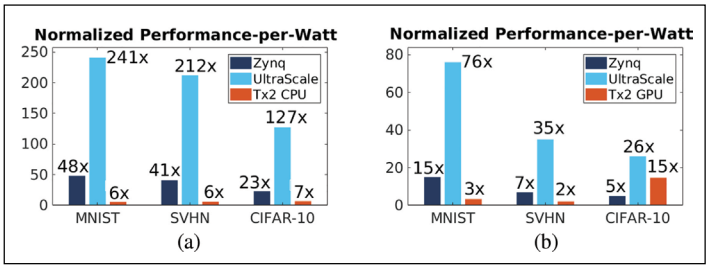

Figure 5. Performance-per-Watt comparison with (a) embedded CPU and (b) CPU-GPU platforms. Reported numbers are normalized by the performance-per-Watt of *TK1*.

White-box attack resiliency

We further evaluate DeepFense against the most powerful attack scenario, i.e., white box threat model, where the attacker has complete access to the parameters and the architecture of the victim model as well as the defenders. We apply the state-of-the-art Carlini&WagnerL2 attack in a white-box setting [21], and compare our defense with the state-of-the-art countermeasures including MagNet [22], APE-GAN [13], and other recently proposed efficient defences methods (e.g., [23]).

Table 2 presents the success rate of the attack algorithm for different number of defender modules and security parameters for the MNIST benchmark. As shown in Table 2, compared to the state-of-the-art defenses of, DeepFense achieves a better TP rate, which can be further improved by increasing the number of deployed MRRs. In addition, the attacker is required to inject a higher amount of perturbation (in terms of L_2 norm) to mislead DeepFense defenders in a white-box setting which can be detected via DeepFense input defenders.[2]

Hardware performance

We implement the customized defender modules on *Xilinx Zynq-ZC702* and *Xilinx UltraScale-VCU108* FPGA platforms. DeepFense performance is optimized via the customization unit on both FPGA boards. To corroborate the efficiency of DeepFense, we compare the customized DeepFense FPGA-accelerator with optimized baselines executed implementation on two low-power embedded boards: 1) the *Jetson TK1* development kit which contains an *NVIDIA Kepler* GPU with 192 CUDA cores as well

[2]We did not include input defenders in this evaluation for comparison fairness.

as an *ARM Cortex-A15* 4-core CPU and 2) a more powerful Jetson TX2 board with an NVIDIA Pascal GPU with 256 cores and a 6-core ARM v8 CPU.

All evaluations are performed with only one input and latent defender. Figure 5 (left) illustrates the performance-per-Watt for different hardware platforms. We define performance-per-Watt as the throughput over the total power consumed by the system. This metric is an effective representation of system performance since it integrates two influential factors for real-world embedded applications. Numbers are normalized by the performance-per-Watt of the *TK1* platform in CPU mode. As shown, DeepFense implementation on *Zynq* shows an average of 38× improvement over *TK1* and 6.2× improvement over *TX2* in CPU mode. The more expensive UltraScale FPGA performs relatively better with an average improvement of 193× and 31.7× over *TK1* and *TX2*, respectively.

DEEP FENSE IS A holistic DL execution system that is both robust and efficient in the face of adversarial attacks. To the best of our knowledge, there is no prior countermeasure that took such a comprehensive approach to simultaneously address both DL resource efficiency and security requirements. DeepFense performance efficiency is particularly important to enable real-time response in autonomous systems. DeepFense vision is to make DL models automatically and efficiently realizable on constrained devices in a safe and reliable way using a unique automated holistic codesign of data subspaces, algorithm, hardware, and software. DeepFense is developed based on an *unsupervised* learning approach, meaning that no adversarial sample is leveraged to train defender modules. Adopting an unsupervised methodology ensures generalization to a broad range of adversarial attacks. To the best of our knowledge, DeepFense is the first framework that provides unsupervised defense and hardware-accelerated execution. Furthermore, the proposed methodology does not require the victim model to change its architecture/parameters, making it amenable for accuracy-sensitive applications. ■

Acknowledgment

This work was done while Bita Darvish Rouhani was with the University of California San Diego, La Jolla, CA, USA.

References

[1] B. D. Rouhani et al., "DeepFense: Online accelerated defense against adversarial deep learning," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2018, pp. 1–8.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*. [Online]. Available: http://arxiv.org/abs/1412.6572

[3] N. Papernot et al., "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Mar. 2016, pp. 372–387.

[4] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2574–2582.

[5] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2016, *arXiv:1607.02533*. [Online]. Available: http://arxiv.org/abs/1607.02533

[6] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.

[7] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[8] J. M. Cohen, E. Rosenfeld, and J. Zico Kolter, "Certified adversarial robustness via randomized smoothing," 2019, *arXiv:1902.02918*. [Online]. Available: http://arxiv.org/abs/1902.02918

[9] A. Madry et al., "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

[10] A. Shafahi et al., "Adversarial training for free!" in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3353–3364.

[11] X. Ma et al., "Characterizing adversarial subspaces using local intrinsic dimensionality," 2018, *arXiv:1801.02613*. [Online]. Available: https://arxiv.org/abs/1801.02613

[12] K. Roth, Y. Kilcher, and T. Hofmann, "The odds are odd: A statistical test for detecting adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5498–5507.

[13] S. Shen et al., "APE-GAN: Adversarial perturbation elimination with GAN," 2017, *arXiv:1707.05474*. [Online]. Available: http://arxiv.org/abs/1707.05474

[14] Z. You et al., "Adversarial noise layer: Regularize neural network by adding noise," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 909–913.

[15] C. Xie et al., "Mitigating adversarial effects through randomization," 2017, *arXiv:1711.01991*. [Online]. Available: http://arxiv.org/abs/1711.01991

[16] N. Das et al., "Shield: Fast, practical defense and vaccination for deep learning using JPEG compression," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 196–204.

[17] O. Taran et al., "Defending against adversarial attacks by randomized diversification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11226–11233.

[18] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," 2018, *arXiv:1802.00420*. [Online]. Available: http://arxiv.org/abs/1802.00420

[19] H. Sharma et al., "From high-level deep neural models to FPGAs," in *Proc. 49th Annu. IEEE/ACM Int. Symp. Microarchit. (MICRO)*, Oct. 2016, p. 17.

[20] M. Javaheripi et al., "CuRTAIL: ChaRacterizing and thwarting AdversarIal deep learning," *IEEE Trans. Dependable Secure Comput.*, early access, Sep. 15, 2020, doi: 10.1109/TDSC.2020.3024191.

[21] N. Carlini and D. Wagner, "MagNet and 'Efficient defenses against adversarial Attacks' are not robust to adversarial examples," 2017, *arXiv:1711.08478*. [Online]. Available: http://arxiv.org/abs/1711.08478

[22] D. Meng and H. Chen, "MagNet: A two-pronged defense against adversarial examples," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 135–147.

[23] V. Zantedeschi, M.-I. Nicolae, and A. Rawat, "Efficient defenses against adversarial attacks," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, Nov. 2017, pp. 39–49.

Mojan Javaheripi is currently pursuing a PhD with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA, USA. Her research interest includes codeveloping machine learning algorithms and their corresponding specialized hardware with the goal of maximizing efficiency and performance. Javaheripi has a BSc in electrical engineering from Sharif University of Technology, Tehran, Iran (2017). She is a Student Member of IEEE.

Mohammad Samragh is currently pursuing a PhD with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA, USA. His research interests include deep learning, safety and robustness of machine learning algorithms, hardware acceleration of learning algorithms, and low-power computing. Samragh has an MSc in electrical and computer engineering from the University of California at San Diego. He is a Student Member of IEEE.

Bita Darvish Rouhani is currently a Senior Researcher at Microsoft, Redmond, WA, USA. Her research interests include deep learning, safety of machine learning models, and low-power computing. Rouhani has a PhD in electrical and computer engineering from the University of California at San Diego, La Jolla, CA, USA.

Tara Javidi is currently a Professor of Electrical and Computer Engineering with the University of California at San Diego, La Jolla, CA, USA. Her research interests include communication networks, stochastic resource allocation, and wireless communications. Javidi has a PhD in electrical engineering and computer science from the University of Michigan, Ann Arbor, MI, USA. She is a Senior Member of IEEE.

Farinaz Koushanfar is currently a Professor and a Henry Booker Faculty Scholar of Electrical and Computer Engineering with the University of California at San Diego, La Jolla, CA, USA. Her research interests include embedded and cyber–physical systems design, embedded systems security, and design automation of domain-specific/mobile computing. Koushanfar has a PhD in electrical engineering and computer science from the University of California Berkeley, Berkeley, CA, USA. She is a Fellow of IEEE.

■ Direct questions and comments about this article to Mojan Javaheripi, University of California San Diego, La Jolla, CA 92092 USA; mojan@ucsd.edu.