





Learning Activation Functions for Adversarial Attack Resilience in CNNs

Maghsoud Salimi^(✉), Mohammad Loni^{}, and Marjan Sirjani^{}

Malardalen University, Vasteras, Sweden

{maghsoud.salimi,mohammad.loni,mmarjan.sirjani}@mdu.se

Abstract. Adversarial attacks on convolutional neural networks (CNNs) have been a serious concern in recent years, as they can cause CNNs to produce inaccurate predictions. Through our analysis of training CNNs with adversarial examples, we discovered that this was primarily caused by naively selecting ReLU as the default choice for activation functions. In contrast to the focus of recent works on proposing adversarial training methods, we study the feasibility of an innovative alternative: learning novel activation functions to make CNNs more resilient to adversarial attacks. In this paper, we propose a search framework that combines simulated annealing and late acceptance hill-climbing to find activation functions that are more robust against adversarial attacks in CNN architectures. The proposed search method has superior search convergence compared to commonly used baselines. The proposed method improves the resilience to adversarial attacks by achieving up to 17.1%, 22.8%, and 16.6% higher accuracy against BIM, FGSM, and PGD attacks, respectively, over ResNet-18 trained on the CIFAR-10 dataset.

Keywords: Convolutional Neural Network · Robustness · Adversarial Attack · Activation Function

1 Introduction

In an adversarial attack, malicious inputs are deliberately introduced into a machine learning model to make incorrect predictions [32]. Recent studies demonstrate that adversarial attacks can present a significant threat to various applications, such as computer vision [24], cyber-physical systems [16, 39], medical machine learning models [9], and wireless communication [1]. Convolutional Neural Networks (CNNs) have shown their great ability to solve problems in various artificial intelligence fields. However, the vulnerability of CNNs to adversarial attacks has been shown in many studies [16, 33, 41]. This can be attributed to the activation functions (AFs) of a CNN with adversarial examples as they are never optimized, with the ReLU [34] being the default choice due to its simplicity and mitigating the vanishing gradient problem.

There have been efforts to improve the robustness of CNNs, either by using architecture search [8, 13] or adversarial training [7, 38]. Nevertheless, no systematic study has been conducted on the impact of learning novel network AFs

over the robustness of CNNs against perturbed examples. Note that, despite the success of adversarial training methods, they require extensive input datasets, which necessitates resource-intensive data augmentation processes [47].

A natural step, thus, is to ask how the AFs impact the learning process for CNNs against adversarial examples. Our analysis of training CNNs with adversarial examples demonstrated that ReLU reduces trainability due to blocking the gradient flow (Sect. 2). A promising research direction in the field of Automated Machine Learning (AutoML) [17] is to optimize network AFs [2, 3, 6, 10, 18, 27, 37]. However, most of the proposed AF tweaking methods have huge computing demands (up to 2000 GPU hours [2]), resulting in a lack of interest in searching for AFs for various deep learning problems.

In this paper, we introduce an AutoML method that discovers robust AFs against adversarial examples by considering robustness accuracy as the search objective. We leverage a novel search algorithm that employs an ordered sequence of Simulated Annealing (SA) [19] and Late Acceptance Hill-Climbing (LAHC) [5] as the optimization stages. The intuition behind the efficiency of the proposed search method is that SA, as a global search method, practically traps in a local optimum after some search iterations. The LAHC method, on the other hand, starts with a solution augmented by SA and exploits the search space to find the global optimum as quickly as possible.

Unlike [7], our proposed method is a generic optimization approach that does not require data augmentation or adversarial training. Inspired by [27, 28], we rely on lower fidelity estimations by training each candidate during the search iterations with fewer epochs, leading to expediting the search procedure.

We demonstrate the effectiveness of our proposed method by achieving up to 17.1%, 22.8%, and 16.6% higher accuracy against BIM, FGSM, and PGD adversarial attacks, respectively, over ResNet-18 trained on CIFAR-10 [20] with ReLU AFs. Additionally, our proposed method improves search efficiency by requiring up to 8.3 GPU days for learning new AFs, which is $9.3\times$ faster than [2]. The proposed method generates similar results with a 4.8% standard deviation, demonstrating the reproducibility of our results.

2 Research Motivation

In this section, we validate the need for new AFs for a CNN under attack, and therefore, propose a robust activation function learning regime. Figure 1 shows the gradient flow for training ResNet-18 with ReLU AFs and new AFs learned by our method. In training ResNet-18 with ReLU, gradient flow is poor, indicating that ReLU potentially increases the information loss during forward propagation when adversarial attacks are present. Additionally, AF optimization is more appropriate for the early layers of the network since the gradient flow for optimized AFs is higher in those layers.

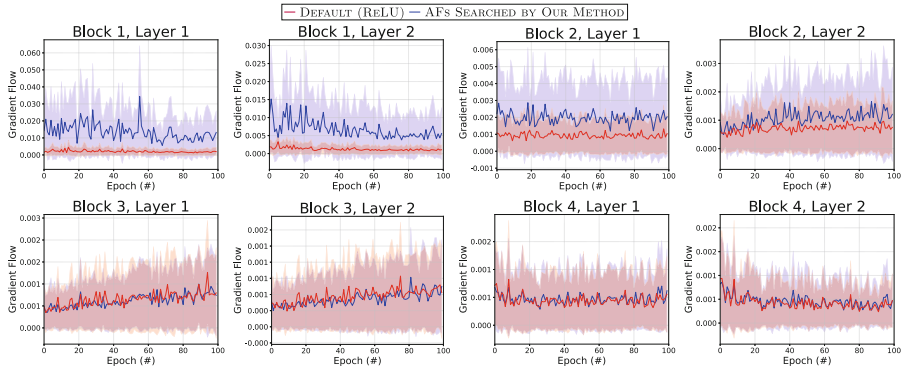


Fig. 1. Showing the gradient flow of the output of the Residual blocks in ResNet-18 with ReLU AF (red) and optimized AFs (blue). (Color figure online)

3 Proposed Method

In general, searching for new AFs is an NP-hard problem with exponential time complexity [27]. Thus, in a reasonable time, polynomial optimization cannot find the optimal solution. In addition, using exhaustive search methods is infeasible in practice, e.g., to exhaustively search an 8000-solution design space, [30] needs 334 GPU days. To this end, we utilize a meta-heuristic search method to deal with the exponential complexity of the AF search problem.

3.1 Search Space

Let us assume we have a CNN model with l hidden layers. The search space is represented by vectors which we call chromosomes. Chromosomes are divided into three parts, each with a length of l . Figure 2 shows an example of a chromosome for a CNN with four hidden layers. The first part of each chromosome is *Switch* which selects the corresponding operation between two AFs. For the sake of simplicity, every possible option for *Switch* is coded into the numbers, as listed in Table 1. As the second and third part of the chromosome, a set of potential candidate AFs is selected where different operations could be applied to them. We consider ReLU, LeakyReLU, Sigmoid, SELU, CELU, Mish, and GELU as the candidate AFs.

The size of the search space depends on various parameters, including the number of layers in the CNN and the number of candidate AFs being considered. The search space size is calculated by the following formula:

$$\text{Search Space Size} = \alpha \times (1 + \beta + (\alpha \times \theta))^l \quad (1)$$

where l is the number of layers, α is the number of candidate AFs, β is the number of possible values for the constant coefficient ($\{0.25, 0.5, 0.75\}$), and θ is the number of possible mathematical operations. According to Table 1, the size of the search space is equal to 7×32^7 for AlexNet with seven hidden layers.

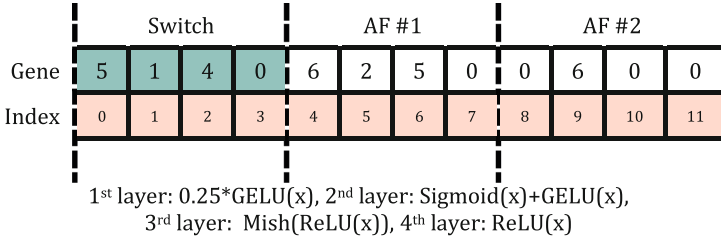


Fig. 2. A chromosome example for a CNN with four hidden layers.

Table 1. Possible values for the *Switch* part of the chromosome and corresponding operations.

Code	<i>Switch</i>	Description
0	$f(x) = g(x)$	Replacing the current AF with another AF selected from the list of candidates
1	$f(x) = g(x) + h(x)$	Accumulating selected AFs, where $g(x)$ comes from <i>AF #1</i> and $h(x)$ comes from <i>AF #2</i>
2	$f(x) = g(x) - h(x)$	A minus operation is performed on the selected AFs, $g(x)$ and $h(x)$
3	$f(x) = g(x) \times h(x)$	Multiplication of selected AFs, $g(x)$ and $h(x)$
4	$f(x) = g(h(x))$	Composition of selected AFs
5	$f(x) = 0.25 \times g(x)$	A constant value of 0.25 is multiplied by the selected AF $g(x)$ from <i>AF #1</i>
6	$f(x) = 0.5 \times g(x)$	A constant value of 0.5 is multiplied by the selected AF $g(x)$ from <i>AF #1</i>
7	$f(x) = 0.75 \times g(x)$	A constant value of 0.75 is multiplied by the selected AF $g(x)$ from <i>AF #1</i>

Results of applying different operations on two examples AFs, Sigmoid and Tanh, are shown in Fig. 3. Results demonstrate that by applying different operations, newly generated AFs significantly differ from the original ones, indicating the proposed search space is flexible to generate very different outputs.

3.2 Search Strategy

In order to solve different AutoML problems, several studies examined various meta-heuristic search methods, e.g., genetic algorithm [27], Late Acceptance Hill-Climbing (LAHC) and Simulated Annealing (SA) [28], and Particle Swarm Optimization [15]. In this paper, we leverage a multi-stage optimization method comprised of SA [19] and LAHC [5] algorithms.

Simulated Annealing. SA is a meta-heuristic search method that provides LAHC with initial solutions. SA iteratively explores solutions with better *Energy* function values. If a solution with a better *Energy* function is found, the current solution is replaced with the newly generated neighbor, otherwise, the current solution remains unchanged. To avoid becoming trapped in

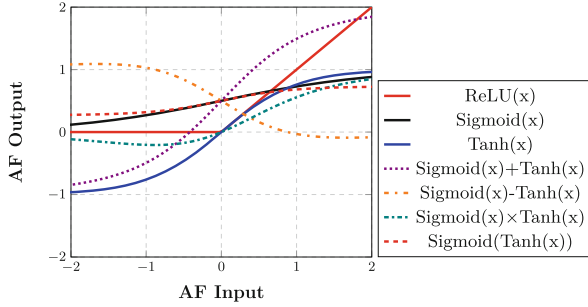


Fig. 3. Generating different mathematical operations using Sigmoid and Tanh.

a local optimum, SA sometimes accepts a bad solution with a probability of $\exp(-\Delta/(k \times T))$. k is the Boltzmann’s constant and T is the cooling parameter which is decreased with a logarithmic shape based on the predefined maximum (T_{Max}) and minimum temperatures (T_{Min}). SA starts with a high T_{Max} for preventing being prematurely trapped in a local optimum. By approaching T toward T_{Min} , most uphill moves will be rejected. The SA process continues until no further improvements can be made or it will be terminated after a specified number of iterations. Finally, it is worth mentioning that the convergence of SA to global results is guaranteed [12].

Late Acceptance Hill-Climbing. This is a heuristic search method that starts with a near-optimal solution provided by the SA algorithm. LAHC is an extension of the simple hill-climbing algorithm [36], in which a limited number of worse solutions are accepted in hopes of finding a better one later. The *Energy* function of both LAHC and SA algorithms is defined by Eq. 2. In this paper, the *Energy* function and the objective function are used interchangeably.

$$Energy = 100 - Robustness Accuracy(\%) \quad (2)$$

Our proposed method has a fast convergence which is due to the single-solution nature of LAHC and SA, while for example, the genetic algorithms are relatively slow due to a population-based optimization [29]. Our experimental results show that our proposed method requires ≈ 8.3 GPU days on a single NVIDIA $\text{\textcircled{R}}$ RTX A4000 for finding the best-performing AF for ResNet-18 trained on CIFAR-10.

4 Experiments

4.1 Experimental Setup

To verify the effectiveness of our proposed method, we use MNIST [23] and CIFAR-10 [20] classification datasets. Our evaluations have been performed on AlexNet and ResNet-18, a variation of ResNet [14], network architectures. To test

Table 2. Summarizing experimental setup.

Search Configuration			
Parameter	MNIST (AlexNet)	CIFAR-10 (AlexNet)	CIFAR-10 (ResNet-18)
Search Epoch (#)	30	100	50
Optimizer	Adam	Adam	Adam
Learning Rate	0.001	0.001	0.001
Train Batch Size	500	512	200
Test Batch Size	250	256	100
Hardware Specification			
GPU	NVIDIA® RTX A4000		
GPU Compiler	NVIDIA® NVCC v. 10.1		
CO ₂ Emission/Day [†]	1.45 Kg		
Training System Memory	64 GB		
CPU	Intel® Xeon® W-2245 CPU @ 3.90GHz		
† Calculated using the ML CO ₂ impact framework: https://mlco2.github.io/impact/ [22]			

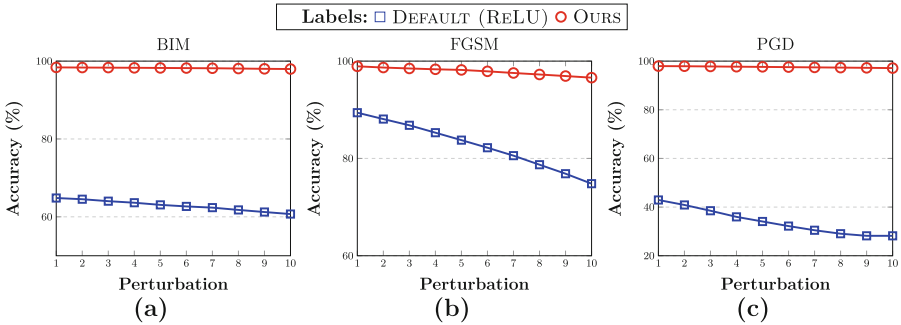


Fig. 4. AlexNet accuracy trained on MNIST with \square ReLU AFs and AFs searched by \circ our proposed method against (a) BIM, (b) FGSM, and (c) PGD. (Color figure online)

the robustness, we consider three popular adversarial attacks, including FGSM [11], PGD [32] and BIM [21]. Table 2 presents the search configuration. Inspired by [27], we trained each candidate with fewer epochs to expedite the search process. The search step takes up to ≈ 8.3 GPU days on a single NVIDIA® RTX A4000 for ResNet-18 trained on the CIFAR-10 dataset.

4.2 Results on MNIST

Figure 4 shows the results of learning AlexNet AFs on MNIST using our proposed method against three different adversarial attacks including BIM, FGSM, and PGD. AlexNet with ReLU AF is selected as the compression baseline (\square). Our proposed method (\circ) significantly outperforms the default configuration by providing up to 37.3%, 21.8%, and 69.0% higher accuracy over BIM, FGSM, and PGD attacks, respectively.

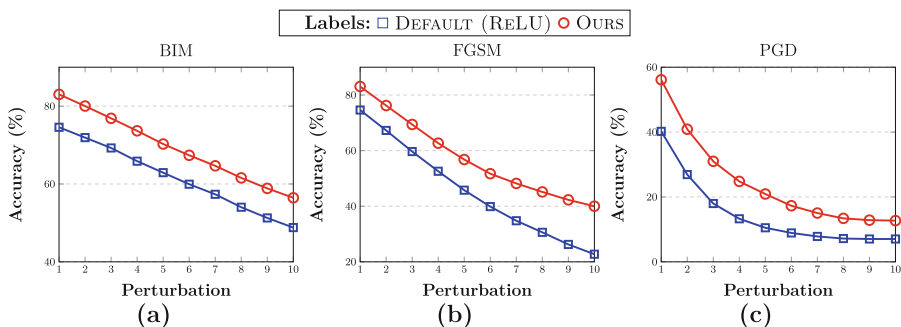


Fig. 5. AlexNet accuracy trained on CIFAR-10 with \square ReLU AFs and AFs searched by \circ our proposed method against (a) BIM, (b) FGSM, and (c) PGD. (Color figure online)

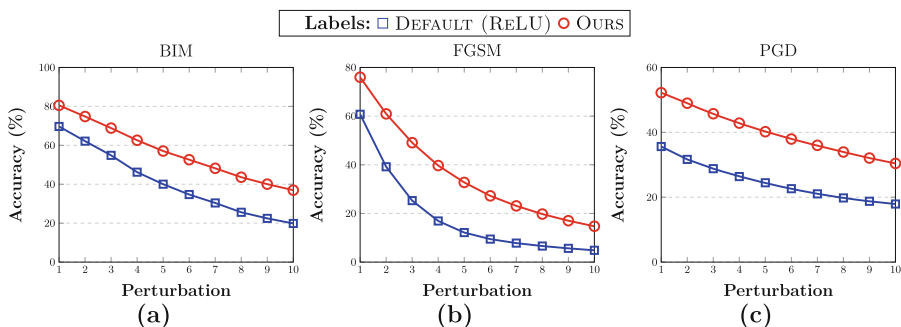


Fig. 6. ResNet-18 accuracy trained on CIFAR-10 with \square ReLU AFs and AFs searched by \circ our proposed method against (a) BIM, (b) FGSM, and (c) PGD. (Color figure online)

4.3 Results on CIFAR-10

Figure 5 shows the results of learning AlexNet AFs trained on CIFAR-10 using our proposed method against three different adversarial attacks including BIM, FGSM, and PGD. AlexNet with ReLU AF is selected as the compression baseline (\square). Our proposed method (\circ) remarkably outperforms the default configuration by providing up to 7.4%, 10.1%, and 15.9% higher accuracy over BIM, FGSM, and PGD attacks, respectively.

Figure 6 shows the results of learning ResNet-18 AFs on CIFAR-10 using our proposed method against three different adversarial attacks including BIM, FGSM, and PGD. ResNet-18 with ReLU AF is selected as the compression baseline (\square). Our proposed method (\circ) significantly outperforms the default configuration by providing up to 18.0%, 23.8%, and 17.3% higher accuracy over BIM, FGSM, and PGD attacks, respectively.

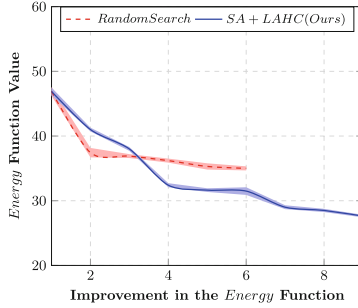


Fig. 7. Comparing the convergence of the proposed method with random search.

4.4 Results of Search Convergence

Figure 7 depicts the *Energy* function (Eq. 2) across search iterations for ResNet-18 trained on CIFAR-10 against the FGSM attack. Our proposed search method finds AFs with a monotonic increase in *Energy*, indicating our proposed method leads to a higher accuracy with fewer search iterations. We also present an empirical evaluation of our method, compared to a random search to show its superior performance. Random search is able to find the optimal architecture in many applications [25, 45]. However, as shown in Fig. 7, our method reached the highest values compared to the random search for the *Energy* function (Eq. 2). Thus, the approach succeeds to find a feasible solution in a reasonable time.

4.5 Analyzing the Discrimination Power of Our Proposed Method

We use the t-distributed stochastic neighbor embedding (t-SNE) method [31] for visualizing the decision boundaries of the original ResNet-18, ResNet-18 with perturbation, and our proposed method for the FGSM attack ($\epsilon = 10/255$) on the CIFAR-10 dataset. Figure 8 illustrates the decision boundaries of classification for each scenario. According to the results, our proposed method has a higher discrimination power than ResNet-18 with perturbation, and our proposed method behaves similarly to the original ResNet-18.

4.6 Reproducibility Statement

Several AutoML papers have problems reproducing their results [26]. We re-ran our proposed method search procedure three more times with different random seeds to verify the reproducibility of our method. Results show that the average of multiple runs converges to AFs with similar results with a standard deviation (STD) of 4.8%. The open-source code is available on GitHub through: <https://github.com/RobustInsight/AdversarialAttackResilience>.

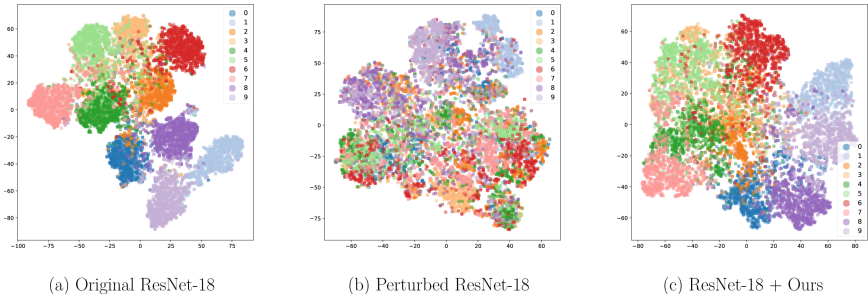


Fig. 8. Visualizing the decision boundary with t-SNE embedding method for (a) original ResNet-18 without perturbation, (b) perturbed ResNet-18, and (c) ResNet-18 with our proposed method optimization.

5 Related Work

To the best of our knowledge, our proposed method is the first automated framework that rapidly learns robust AFs using a multi-stage optimization method. In the past, extensive research has been conducted on improving CNN accuracy. Prior studies are mainly categorized as (i) adversarial training [7, 38], (ii) robust optimization [4, 40], (iii) architecture modification [8, 13], and (iv) AF optimization for adversarial attack resilience in CNNs [42]. In the rest of this section, we briefly discuss state-of-the-art research on AF optimization and compare them with our proposed method.

Studies indicate that AFs are a significant contributor to the vulnerability of neural networks to adversarial examples [46]. Since ReLU is a non-smooth AF, [44] proposed replacing ReLU with its smooth approximations to find harder adversarial examples. A new AF is suggested by [43] that relies on the data and demonstrated its resistance to adversarial attacks. Since ReLU is not smooth and inputs close to zero cause its gradient to abruptly change, the Softplus AF is proposed by [44] whose derivative is continuous and n -times differentiable. Another approach examined the resilience of various layer types within CNNs against adversarial attacks, by considering each layer as a separate nonlinear system and assessing its robustness, utilizing Lyapunov theory. Instead of using non-linear AFs, SPLASH uses piece-wise linear AFs which boosts the robustness of CNNs against adversarial attacks and accuracy as well [42]. The authors in [7] investigated the influence of the shape of AFs on the accuracy and robustness of CNNs by parameterizing various AFs. The approach achieved this by introducing an α parameter to different AFs and examining the effects of altering the α parameter. However, the performance improvement observed is limited as the study only used a restricted set of values and the effect of the α parameter is linear.

These methods have been quite effective, but they suffer from huge computational costs due to the use of reinforcement learning or evolutionary algorithms [27, 35, 37]. To expedite the learning process of AFs, this work proposes using

simulated annealing and late acceptance hill climbing, which can lead to up to $9.3\times$ faster search than [2]. Finally, we make no assumptions about the input dataset, which makes it a more generalized method.

6 Conclusion

The purpose of this study was to investigate how learning activation functions impact the robustness of CNNs against adversarial attacks. Experimental results demonstrate that ReLU is not robust against adversarial attacks, whereas learning network AFs greatly enhances robustness. Overall, our work contributes to the growing body of research on adversarial attack resilience in CNNs and provides a promising approach for designing more robust CNNs.

References

1. Bahramali, A., Nasr, M., Houmansadr, A., Goeckel, D., Towsley, D.: Robust adversarial attacks against DNN-based wireless communication systems. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp. 126–140 (2021)
2. Bingham, G., Macke, W., Miikkulainen, R.: Evolutionary optimization of deep learning activation functions. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference, pp. 289–296 (2020)
3. Bingham, G., Miikkulainen, R.: Discovering parametric activation functions. arXiv preprint [arXiv:2006.03179](https://arxiv.org/abs/2006.03179) (2020)
4. Bradshaw, J., Matthews, A.G.d.G., Ghahramani, Z.: Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. arXiv preprint [arXiv:1707.02476](https://arxiv.org/abs/1707.02476) (2017)
5. Burke, E.K., Bykov, Y., et al.: A late acceptance strategy in hill-climbing for exam timetabling problems. In: PATAT 2008 Conference, Montreal, Canada, pp. 1–7 (2008)
6. Cui, P., Shabash, B., Wiese, K.C.: EvoDNN—an evolutionary deep neural network with heterogeneous activation functions. In: 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 2362–2369. IEEE (2019)
7. Dai, S., Mahloujifar, S., Mittal, P.: Parameterizing activation functions for adversarial robustness. In: 2022 IEEE Security and Privacy Workshops (SPW), pp. 80–87. IEEE (2022)
8. Devaguptapu, C., Agarwal, D., Mittal, G., Gopalani, P., Balasubramanian, V.N.: On adversarial robustness: a neural architecture search perspective. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 152–161 (2021)
9. Finlayson, S.G., Bowers, J.D., Ito, J., Zittrain, J.L., Beam, A.L., Kohane, I.S.: Adversarial attacks on medical machine learning. *Science* **363**(6433), 1287–1289 (2019)
10. Godfrey, L.B., Gashler, M.S.: A continuum among logarithmic, linear, and exponential functions, and its potential to improve generalization in neural networks. In: 2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K). vol. 1, pp. 481–486. IEEE (2015)

11. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint [arXiv:1412.6572](https://arxiv.org/abs/1412.6572) (2014)
12. Granville, V., Krivánek, M., Rasson, J.P.: Simulated annealing: a proof of convergence. *IEEE Trans. Pattern Anal. Mach. Intell.* **16**(6), 652–656 (1994)
13. Guo, M., Yang, Y., Xu, R., Liu, Z., Lin, D.: When NAS meets robustness: in search of robust architectures against adversarial attacks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 631–640 (2020)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016)
15. Huang, J., Xue, B., Sun, Y., Zhang, M., Yen, G.G.: Particle swarm optimization for compact neural architecture search for image classification. *IEEE Trans. Evol. Comput.* 1–1 (2022)
16. Huang, S., Papernot, N., Goodfellow, I., Duan, Y., Abbeel, P.: Adversarial attacks on neural network policies. arXiv preprint [arXiv:1702.02284](https://arxiv.org/abs/1702.02284) (2017)
17. Hutter, F., Kothhoff, L., Vanschoren, J. (eds.): *Automated Machine Learning*. TSS-CML, Springer, Cham (2019). <https://doi.org/10.1007/978-3-030-05318-5>
18. Jin, X., Xu, C., Feng, J., Wei, Y., Xiong, J., Yan, S.: Deep learning with S-shaped rectified linear activation units. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 30 (2016)
19. Kirkpatrick, S., Gelatt, C.D., Jr., Vecchi, M.P.: Optimization by simulated annealing. *Science* **220**(4598), 671–680 (1983)
20. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-10 and CIFAR-100 datasets. **6**(1), 1 (2009)
21. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: *Artificial Intelligence Safety and Security*, pp. 99–112. Chapman and Hall/CRC (2018)
22. Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T.: Quantifying the carbon emissions of machine learning. arXiv preprint [arXiv:1910.09700](https://arxiv.org/abs/1910.09700) (2019)
23. LeCun, Y.: The MNIST database of handwritten digits (1998)
24. Lee, M., Kolter, Z.: On physical adversarial patches for object detection. arXiv preprint [arXiv:1906.11897](https://arxiv.org/abs/1906.11897) (2019)
25. Li, L., Talwalkar, A.: Random search and reproducibility for neural architecture search. In: *Uncertainty in Artificial Intelligence*, pp. 367–377. PMLR (2020)
26. Lindauer, M., Hutter, F.: Best practices for scientific research on neural architecture search. *J. Mach. Learn. Res.* **21**(243), 1–18 (2020)
27. Loni, M., Sinaei, S., Zoljodi, A., Daneshtalab, M., Sjödin, M.: DeepMaker: a multi-objective optimization framework for deep neural networks in embedded systems. *Microprocess. Microsyst.* **73**, 102989 (2020)
28. Loni, M., et al.: DenseDisp: resource-aware disparity map estimation by compressing siamese neural architecture. In: *2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8. IEEE (2020)
29. Loni, M., et al.: FastStereoNet: a fast neural architecture search for improving the inference of disparity estimation on resource-limited platforms. *IEEE Trans. Syst. Man Cybern. Syst.* **52**(8), 5222–5234 (2021)
30. Loni, M., Zoljodi, A., Sinaei, S., Daneshtalab, M., Sjödin, M.: NeuroPower: Designing Energy Efficient Convolutional Neural Network Architecture for Embedded Systems. In: Tetko, I.V., Kurková, V., Karpov, P., Theis, F. (eds.) *ICANN 2019*. LNCS, vol. 11727, pp. 208–222. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30487-4_17

31. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(11), 2579–2605 (2008)
32. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint [arXiv:1706.06083](https://arxiv.org/abs/1706.06083) (2017)
33. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: DeepFool: a simple and accurate method to fool deep neural networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582 (2016)
34. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: *Proceedings of the 27th International Conference on Machine Learning (ICML)* (2010)
35. Nogami, W., Ikegami, T., Takano, R., Kudoh, T., et al.: Optimizing weight value quantization for CNN inference. In: *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE (2019)
36. Norvig, P.R., Intelligence, S.A.: *A modern approach*. Prentice Hall Upper Saddle River, NJ, USA: Rani, M., Nayak, R., & Vyas, OP (2015). An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. *Knowledge-Based Systems* **90**, 33–48 (2002)
37. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv preprint [arXiv:1710.05941](https://arxiv.org/abs/1710.05941) (2017)
38. Rebuffi, S.A., Gowal, S., Calian, D.A., Stimberg, F., Wiles, O., Mann, T.: Fixing data augmentation to improve adversarial robustness. arXiv preprint [arXiv:2103.01946](https://arxiv.org/abs/2103.01946) (2021)
39. Rosenberg, I., Shabtai, A., Elovici, Y., Rokach, L.: Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Comput. Surv. (CSUR)* **54**(5), 1–36 (2021)
40. Strauss, T., Hanselmann, M., Junginger, A., Ulmer, H.: Ensemble methods as a defense to adversarial perturbations against deep neural networks. arXiv preprint [arXiv:1709.03423](https://arxiv.org/abs/1709.03423) (2017)
41. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint [arXiv:1312.6199](https://arxiv.org/abs/1312.6199) (2013)
42. Tavakoli, M., Agostinelli, F., Baldi, P.: Splash: learnable activation functions for improving accuracy and adversarial robustness. *Neural Netw.* **140**, 1–12 (2021)
43. Wang, B., Lin, A.T., Zhu, W., Yin, P., Bertozzi, A.L., Osher, S.J.: Adversarial defense via data dependent activation function and total variation minimization. arXiv preprint [arXiv:1809.08516](https://arxiv.org/abs/1809.08516) (2018)
44. Xie, C., Tan, M., Gong, B., Yuille, A., Le, Q.V.: Smooth adversarial training. arXiv preprint [arXiv:2006.14536](https://arxiv.org/abs/2006.14536) (2020)
45. Yang, A., Esperança, P.M., Carlucci, F.M.: NAS evaluation is frustratingly hard. arXiv preprint [arXiv:1912.12522](https://arxiv.org/abs/1912.12522) (2019)
46. Zantedeschi, V., Nicolae, M.I., Rawat, A.: Efficient defenses against adversarial attacks. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 39–49 (2017)
47. Zhang, J., Li, C.: Adversarial examples: opportunities and challenges. *IEEE Trans. Neural Netw. Learn. Syst.* **31**(7), 2578–2593 (2019)