Exploring the Connection between Neuron Coverage and Adversarial Robustness in DNN Classifiers

William Piat, Jalal Fadili, Frédéric Jurie

EXPLORING THE CONNECTION BETWEEN NEURON COVERAGE AND ADVERSARIAL ROBUSTNESS IN DNN CLASSIFIERS

William PIAT *

Safran Tech
DST Departement
Châteaufort, France.

Jalal FADILI, Frédéric JURIE

Normandie Univ, ENSICAEN, CNRS
GREYC
Caen, France

ABSTRACT

The lack of robustness in neural network classifiers, especially when facing adversarial attacks, is a significant limitation for critical applications. While some researchers have suggested a connection between neuron coverage during training and vulnerability to adversarial perturbations, concrete experimental evidence supporting this claim is lacking. This paper empirically investigates the impact of maximizing neuron coverage during training and assess the effectiveness of adversarial attacks on under-covered neurons. Additionally, we explore the potential of leveraging coverage for designing more efficient attacks. Our experiments reveal no clear correlation between neuron coverage, adversarial robustness, or attack effectiveness.

Keywords: Deep learning, Coverage, Robustness, Adversarial Training

1. INTRODUCTION AND RELATED WORK

Code coverage [1] is a widely used metric that assesses the robustness of source code by measuring the portion of code executed during a test suite. It plays a crucial role in identifying potential undetected bugs and is considered a critical evaluation metric for code quality before deployment.

Drawing inspiration from code coverage, researchers [2, 3, 4] have explored the concept of neuron coverage to detect faulty behavior in deep learning classifiers. Due to the prevalence of the ReLU activation function in modern neural networks, certain parts of the network remain unused during training, creating vulnerabilities that can lead to incorrect predictions when the model is deployed.

The vulnerability of deep neural networks (DNNs) to adversarial attacks has raised significant concerns. Adversarial attacks pose a serious threat to the reliability of neural networks. In the context of a neural network $f : \Theta \times \mathcal{X} \to \mathcal{Y}$, with a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, parameters $\theta \in \theta$, and a positive value ε, adversarial attacks aim to find $\tilde{x} \in \mathrm{Argmax}_{z:|x-z|_\infty \leq \varepsilon} \ell(f(\theta, z), y)$, where $(x, y) \in \mathcal{X} \times \mathcal{Y}$ represents an input/output tuple. Commonly used algorithms for generating adversarial examples include Projected Gradient Descent (PGD) [5], Fast Gradient Sign Method (FGSM) [6], and Carlini and Wagner attack (C&W) [7].

Existing approaches for countering adversarial attacks generally follow a dual strategy: generating adversarial examples and retraining the classifier by incorporating these examples into the training set (as seen in [8], for example).

As mentioned earlier, several authors have highlighted the relationship between coverage and robustness. The pioneering work of DeepXplore [2] drew a parallel between coverage and adversarial attacks, suggesting that adversarial attacks exploit parts of the deep neural network (DNN) that were not utilized during training. Pei et al. [2] defined neuron coverage in the context of a test set $X = x_1, ..., x_m \in \mathcal{X}^m, m \in \mathbb{N}$. Neuron coverage is calculated as the ratio of activated neurons in the test set to the total number of neurons, i.e. $\mathrm{NCov}(X, \theta) = \frac{|\{n_i | \forall x \in X, n_i(\theta, x) > 0\}|}{|N|}$, where $N = \{n_1, n_2, ...\}$ is the set of neurons, and $n_i : \Theta \times \mathcal{X} \to \mathbb{R}$ is a function that returns the activation value of neuron i before applying the nonlinearity for a given input $x \in \mathcal{X}$. However, this metric has been experimentally shown to be insufficient in capturing the diversity of activation distributions [3]. It lacks sensitivity to the distribution of activations across the test set and does not account for the specific patterns of activation. To address these limitations, DeepGauge [3] introduced the concept of k-multisection coverage. For a neuron $n_i \in N$, k-multisection coverage involves dividing the activation in-

*The author performed the work while still in Normandie Université

terval $[l_i, u_i]$ of neuron n_i into k subdivisions $(S_j^{n_i})_{j\in[k]}$, such that $\bigcup_{j\in[k]} S_j^{n_i} = [l_i, u_i]$. Ideally, the subdivisions should be of equal length. The coverage metric is then computed by counting the number of sections that contain activation values when tested on the test set and dividing it by the total number of sections: $\text{KMNCov}(X, \theta, k) = \frac{1}{|N|}\sum_{i\in[|N|]} \frac{|\{S_j^{n_i} | \exists x\in X, n_i(\theta,x)\in S_j^{n_i}\}|}{k}$. This metric exhibits high non-smoothness and non-convexity, making it challenging to optimize. Furthermore, the original paper did not investigate the impact of the parameter k on the metric, leaving it as an open question for further exploration.

In the software engineering community, there is a growing concern that current coverage-based testing methods may not be suitable for effectively testing neural networks, as they do not seem to generate robust tests. Several studies, such as [9, 10, 11], have discussed this issue. While some works, like [12, 13], focus on comparing the correlations between coverage and robustness, little attention has been given to their joint effect. This paper aims to bridge this gap and investigates the potential of utilizing adversarial attacks and coverage together to disrupt predictions.

In this context, we propose a novel sub-differentiable coverage metric that can be employed in conjunction with robust training methods. The primary objective is to explore the relationship between coverage and adversarial training, specifically how adversarial attacks impact the model's accuracy when both coverage and robustness are taken into account. Our research focuses on the defender's perspective, aiming to leverage coverage during training to enhance the model's robustness.

2. DEFINITION OF A NOVEL COVERAGE METRIC

Let $a_i \in \mathcal{P}(\mathbb{R})$ represent the activation distribution of a single neuron i, where $i \in [|N|]$ and $\mathcal{P}(\mathbb{R})$ denotes the space of probability measures on the real line. The activation distribution a_i is computed before applying the non-linearity, meaning that a_i represents the distribution of $n_i(\theta, \cdot) : \mathcal{X} \to \mathbb{R}$ evaluated on the input distribution. to lighten notation, we have omitted the dependence of a_i on θ. The activation distribution a_i is supported on a compact interval $[l_i, u_i]$, where $(l_i \leq u_i)$ and $(l_i, u_i) \in \mathbb{R}^2$, as we assume bounded input and parameter spaces throughout our analysis. Let $\mathcal{U}(l_i, u_i)$ denote the uniform distribution over the interval $[l_i, u_i]$. In our context, maximizing the coverage metric entails making the activation distribution lean towards the uniform distribution on the activation interval. Therefore, we define coverage as the negative Kullback-Leibler (KL) divergence between $\mathcal{U}(l_i, u_i)$ and a_i:

$$\text{Cov}(a_i, \theta) = -\text{KL}(\mathcal{U}(l_i, u_i), a_i). \tag{1}$$

One advantage of this metric is that it does not resort to any binning strategy influencing the result. In practice, we approximate this metric using m samples $(n_i(\theta, x_1), ..., n_i(\theta, x_m)) = (a_i^1, ..., a_i^m)$ and a kernel density estimator [14, 15],

$$a_i(t) \approx \frac{1}{mh}\sum_{j=1}^{m} K\left(\frac{t - a_i^j}{h}\right),$$

where K is a kernel function, and $h > 0$ is the bandwidth. This approximation is called the activation trace [16, 17]: it is used as a way to quantify surprise in a new input. We use it to approximate (1) as

$$\widehat{\text{Cov}}(a_i, \theta) := -\text{KL}\left(U(l_i, u_i), \frac{1}{mh}\sum_{j=1}^{m} K\left(\frac{\cdot - a_i^j}{h}\right)\right). \tag{2}$$

By extension, we define the coverage of a network $\widehat{\text{Cov}}(f, \theta)(X)$ as the mean coverage across all neurons in the network. The dependence on θ is crucial since we will subsequently aim to optimize the model parameters based on this coverage metric. In this paper, we choose K as the Gaussian kernel which then ensures that the KL divergence is well-defined regardless of the activation values. Moreover, $\mathscr{C}^1$ smoothness of the Gaussian kernel enables us to employ gradient-based optimization methods. The bandwidth parameter h is determined based on a standard bias-variance trade-off [14]: $h = \frac{4\hat{\sigma}}{3m^{\frac{1}{5}}}$, where $\hat{\sigma}$ represents the standard deviation of the data points used for estimating the distribution. The values of l_i and u_i are determined after randomly initializing the weights.

2.1. Attacking the uncovered neurons of DNNs

In this section we present different ways of perturbing the prediction of a neural network by changing the input values within a given perturbation set. These attacks, like the code coverage tests, target the neurons that are not activated. Throughout this section, we will set $\mathcal{X} = \mathbb{R}^p$ and $\mathcal{Y} = \mathbb{R}^q$.

Coverage attack. This "attack" is simply a search of a bounded local perturbation $\delta = \{\delta_1, ..., \delta_m\} \in (\mathbb{R}^p)^m$ of the input values of the test set $X = \{x_1, ..., x_m\} \in (\mathbb{R}^p)^m$ in order to increase the coverage of the model. It corresponds to maximizing $\widehat{\text{Cov}}(f, \theta)(\{x_i + \delta_i, i \in [m]\})$ with respect to δ. In practice, the $(\delta_i)_{i\in[m]}$ are found by running a PGD-like algorithm on a batch of data, but using the coverage loss.

FGSM and PGD attack with added coverage. By incorporating a coverage penalty into adversarial attacks during batch processing, it is possible to introduce a notion of coverage to the attacks themselves:

$$\max_{\|\delta_i\|_\infty \leq \epsilon, \forall i \in [m]} \Bigg[\frac{1}{m} \sum_{i=1}^{m} \ell(f(\theta, x_i + \delta_i), y_i) + \nu \, \widehat{\mathrm{Cov}}(f, \theta)(\{x_i + \delta_i, i \in [m]\}) \Bigg]. \tag{3}$$

ν allows balances between pure adversarial attacks (for $\nu = 0$) and pure coverage attacks (as ν approaches infinity).

Carlini and Wagner attack with added coverage. The C&W attack can also be processed in batch mode, so we can define a coverage version of this attack in the same way as we did in (3):

$$\min_{\|\delta_i\|_\infty \leq \epsilon, y_i' \neq y_i, y_i' \in \mathcal{Y}, \forall i \in [m]} \Bigg[\frac{1}{m} \sum_{i=1}^{m} \ell(f(\theta, x_i + \delta_i), y_i') - \nu \, \widehat{\mathrm{Cov}}(f, \theta)(\{x_i + \delta_i, i \in [m]\} \Bigg]. \tag{4}$$

The sign before the coverage has changed since the C&W attack is solution to a minimization problem, not a maximization like FGSM or PGD.

2.2. Coverage as a defense for DNNs

The idea is to ensure that the maximum number of neurons in the network are activated during training, so that no weights in the network remain untrained.

Vanilla training with added coverage. A first logical approach when considering a coverage metric is basically to increase the coverage in the training phase to ensure that activations are well covered. More specifically, we act on the parameters $\theta \in \theta$ of the neural network to simultaneously minimise an empirical loss and maximise coverage. Similarly to adversarial attacks, we will trade off between training loss and coverage via a parameter $\nu > 0$

$$\min_{\theta \in \Theta} \sum_{i=1}^{m} \ell(f(\theta, x_i), y_i) - \nu \, \widehat{\mathrm{Cov}}(f, \theta)(\{x_1, ..., x_m\}). \tag{5}$$

This problem can be solved using classical stochastic (sub)gradient-based algorithms.

Robust training against attacks. This approach leverages the attacks described in Section 2.1 to conduct adversarial training. As a result, it enables the development of multiple defenses against different attack types. These defenses encompass protection against the coverage attack, adversarial defenses against FGSM, PGD, and C&W attacks, as well as their variants augmented with coverage. Furthermore, this approach also facilitates the creation of defenses against random attacks.

3. EXPERIMENTS

The objective of these experiments is to investigate the impact of incorporating coverage into training procedures on the robustness of models and the effectiveness of adversarial attacks. We use the attacks and defenses described in the previous sections for adversarial training and subsequently evaluate the adversarial accuracy using both reference attacks and coverage attacks. The following list provides an overview of the training and attack methods we examine:

Unperturbed Coverage (UC): This method involves performing training as described in Section 2.2. It serves as a defense and is not considered within the range of attacks. **No Perturbation (NP):** This method uses an unperturbed test set as an attack and employs vanilla training as a defense. **Random (R):** This method perturbs the inputs randomly with uniform noise as an attack and includes training against white noise as a defense. **FGSM (F):** This method sets $\nu = 0$ and performs a single-step approximation of the argument in (3) as an attack, while adversarial training against this attack is used as a defense. **FGSM + Coverage (F+C):** Similar to FGSM, this method involves a single-step approximation of the argument in (3) as an attack. Adversarial training against this attack is employed as a defense. **PGD (P):** This method sets $\nu = 0$ and performs a multistep approximation of the argument in (3) as an attack. Adversarial training against this attack is used as a defense. **PGD + Coverage (P+C):** Similar to PGD, this method involves a multistep approximation of the argument in (3) as an attack. Adversarial training against this attack is employed as a defense. **C&W (CW):** This method aims to maximize the likelihood of the second most probable class (for classification only) as an attack. Adversarial training using these samples is used as a defense. **C&W + Coverage (CW+C):** This method adds a coverage penalty to the C&W attack (see (4)). Adversarial training against this attack is employed as a defense.

Tables 1, 2, and 3 present the evolution of adversarial robustness (i.e., the accuracy against the corresponding attack) for $\nu = 100$ on the SVHN, FMINIST, and CIFAR10 datasets. The value of $\nu = 100$ was chosen empirically as it represents a good compromise between coverage training and adversarial training. From the defender's perspective, a higher accuracy indicates better resilience, and thus, higher values are desirable.

In all the tables, we observe similar performance between the coverage used for adversarial training and the coverage-maximizing training procedure (row 4 and row 1, respectively), except for the coverage attacks (columns 3, 5, 7, and 9). In those cases, the coverage training (row

Defenses \ Attacks	NP	R	C	F	F+C	P	P+C	CW	CW+C
UC	0.95	0.95	0.76	0.62	0.80	0.47	0.81	0.35	0.46
NP	0.95	0.95	0.92	0.34	0.89	0.24	0.90	0.27	0.71
R	0.95	0.95	0.92	0.35	0.91	0.28	0.90	0.27	0.73
C	0.95	0.94	0.94	0.49	0.93	0.42	0.91	0.59	0.85
F	0.92	0.93	0.93	0.85	0.93	0.87	0.93	0.89	0.90
F + C	0.95	0.94	0.94	0.46	0.93	0.34	0.91	0.49	0.82
P	0.93	0.93	0.93	0.82	0.93	0.88	0.92	0.90	0.92
P + C	0.94	0.94	0.94	0.57	0.94	0.65	0.92	0.70	0.88
CW	0.93	0.94	0.94	0.79	0.94	0.83	0.93	0.88	0.91
CW + C	0.94	0.95	0.95	0.67	0.94	0.74	0.94	0.82	0.91

Table 1: Adversarial performance of multiple adversarial trainings on SVHN, trainings row-wise and attacks column-wise.

Defenses \ Attacks	NP	R	C	F	F+C	P	P+C	CW	CW+C
UC	0.92	0.82	0.77	0.46	0.74	0.18	0.74	0.09	0.29
NP	0.93	0.88	0.91	0.27	0.90	0.07	0.88	0.09	0.59
R	0.93	0.92	0.92	0.35	0.91	0.11	0.89	0.19	0.72
C	0.92	0.92	0.92	0.42	0.91	0.22	0.89	0.33	0.77
F	0.88	0.90	0.90	0.87	0.76	0.70	0.89	0.70	0.83
F + C	0.91	0.92	0.92	0.36	0.91	0.09	0.89	0.20	0.72
P	0.86	0.83	0.83	0.85	0.74	0.91	0.84	0.90	0.86
P + C	0.92	0.92	0.92	0.62	0.91	0.54	0.90	0.69	0.86
CW	0.91	0.88	0.88	0.86	0.79	0.88	0.89	0.91	0.88
CW + C	0.92	0.93	0.93	0.69	0.91	0.65	0.91	0.78	0.88

Table 2: Adversarial performance of multiple adversarial trainings on FMNIST, trainings row-wise and attacks column-wise.

4) exhibits a slight advantage since it has been specifically trained against coverage attacks. However, the increase in robustness is not as significant as that achieved through pure adversarial training (rows 3, 5, 7, and 9). Both coverage methods (rows 1 and 4) demonstrate a slight improvement over vanilla training (row 2) in all measures of adversarial robustness. The non-adversarial coverage training (row 1) proves to be as effective as regularization and produces better results than unconstrained training (row 2). Adversarial training is still the best in terms of adversarial robustness, as it performs best against the full range of adversarial attacks.

The main observations we derived from these experiments are as follows: i) The different approaches to coverage training perform similarly: there seems to be little difference between employing coverage-guided training or coverage-guided adversarial training. Both approaches yield better training results than vanilla training by enhancing robustness against adversarial noise. ii) Coverage approaches outperform the addition of white noise

Defenses \ Attacks	NP	R	C	F	F+C	P	P+C	CW	CW+C
UC	0.81	0.80	0.58	0.40	0.80	0.28	0.73	0.14	0.24
NP	0.82	0.80	0.81	0.20	0.81	0.13	0.79	0.14	0.57
R	0.83	0.82	0.82	0.21	0.80	0.14	0.79	0.15	0.58
C	0.82	0.81	0.84	0.27	0.83	0.20	0.80	0.23	0.64
F	0.78	0.79	0.81	0.69	0.80	0.72	0.79	0.73	0.76
F + C	0.82	0.82	0.81	0.24	0.83	0.18	0.80	0.20	0.64
P	0.79	0.79	0.82	0.69	0.81	0.71	0.81	0.74	0.78
P + C	0.77	0.81	0.84	0.34	0.83	0.34	0.80	0.41	0.72
CW	0.81	0.81	0.83	0.64	0.82	0.69	0.81	0.73	0.78
CW + C	0.82	0.82	0.84	0.48	0.83	0.53	0.82	0.60	0.77

Table 3: Adversarial performance of multiple adversarial trainings on CIFAR10, trainings row-wise and attacks column-wise.

to input data: they provide a more meaningful sense of robustness compared to random noise. iii) Coverage approaches are surpassed by adversarial training in terms of adversarial robustness: when it comes to defending against adversarial attacks, pure adversarial training outperforms coverage-based approaches. iv) There is no optimal balance between coverage and adversarial approaches: combining coverage and adversarial training never achieves better performance, regardless of the value of ν, compared to pure adversarial training.

These experiments demonstrate that coverage-based approaches can serve as effective regularization techniques, yielding parameter sets that are more suitable than those obtained through vanilla training. However, when it comes to adversarial robustness, the benefits of incorporating coverage into adversarial training appear limited.

4. CONCLUSIONS

The connections between robustness and coverage, as suggested in the literature, do not hold as expected. From the defender's perspective, training with coverage acts as a regularizer and is preferable to vanilla training when considering robustness metrics. However, it falls short of outperforming adversarial training against targeted attacks. Adversarial training remains the preferred choice in white box settings, where the attacker has access to the learned function and can customize the attack accordingly. In light of these findings, it appears that neuron coverage is not a relevant metric for quantifying or enhancing the robustness of neural networks, contrary to common beliefs.

Acknowledgments. Research reported in this paper was supported by the Agence Nationale pour la Recherche (ANR) under award number ANR-19-CHIA-0017.

5. REFERENCES

[1] Joan C. Miller and Clifford J. Maloney, "Systematic mistake analysis of digital computer programs," *Communications of the ACM*, vol. 6, no. 2, pp. 58–63, Feb. 1963.

[2] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana, "DeepXplore: Automated Whitebox Testing of Deep Learning Systems," *Proceedings of the 26th Symposium on Operating Systems Principles*, pp. 1–18, Oct. 2017, arXiv: 1705.06640.

[3] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang, "DeepGauge: multi-granularity testing criteria for deep learning systems," in *ASE18*. 2018, ACM.

[4] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See, "DeepHunter: a coverage-guided fuzz testing framework for deep neural networks," in *ISSTA19*, 2019.

[5] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," in *ICLR'18*, 2018.

[6] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and Harnessing Adversarial Examples," in *ICLR'15*, 2015.

[7] Nicholas Carlini and David A. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *SP'17*, 2017, pp. 39–57.

[8] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang, "Recent Advances in Adversarial Training for Adversarial Robustness," in *IJCAI'21*, Montreal, Canada, 2021.

[9] Zenan Li, Xiaoxing Ma, Chang Xu, and Chun Cao, "Structural coverage criteria for neural networks could be misleading," in *ICSE/NIER'19*, 2019.

[10] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Jianye Hao, Xinyu Wang, Li Wang, Jin Song Dong, and Dai Ting, "There is Limited Correlation between Coverage and Robustness for Deep Neural Networks," *CoRR*, vol. abs/1911.05904, 2019, arXiv: 1911.05904.

[11] Jasmine Sekhon and Cody H. Fleming, "Towards improved testing for deep learning," in *ICSE/NIER'19*, 2019.

[12] Shenao Yan, Guanhong Tao, Xuwei Liu, Juan Zhai, Shiqing Ma, Lei Xu, and Xiangyu Zhang, "Correlations between deep neural network model coverage criteria and model quality," in *ESEC/FSE'20*, 2020.

[13] Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, Quanquan Gu, and Miryung Kim, "Is neuron coverage a meaningful measure for testing deep neural networks?," in *ESEC/FSE'20*, 2020.

[14] B.W. Silverman, *Density Estimation for Statistics and Data Analysis*, Routledge, 1 edition, Feb. 2018.

[15] M. Chris Jones, J. S. Marron, and Simon J. Sheather, "A brief survey of bandwidth selection for density estimation," *Journal of the American Statistical Association*, vol. 91, pp. 401–407, 1996.

[16] Reuben Feinman, Ryan R. Curtin, Saurabh Shintre, and Andrew B. Gardner, "Detecting adversarial samples from artifacts," *CoRR*, vol. abs/1703.00410, 2017.

[17] Jinhan Kim, Robert Feldt, and Shin Yoo, "Guiding deep learning system testing using surprise adequacy," in *ICSE19*, 2019.