Is Neuron Coverage a Meaningful Measure for Testing Deep Neural Networks?

Fabrice Harel-Canada
fabricehc@cs.ucla.edu
UCLA, USA

Lingxiao Wang
lingxw@cs.ucla.edu
UCLA, USA

Muhammad Ali Gulzar
gulzar@cs.vt.edu
Virgina Tech, USA

Quanquan Gu
qgu@cs.ucla.edu
UCLA, USA

Miryung Kim
miryung@cs.ucla.edu
UCLA, USA

ABSTRACT

Recent effort to test deep learning systems has produced an intuitive and compelling test criterion called neuron coverage (NC), which resembles the notion of traditional code coverage. NC measures the proportion of neurons activated in a neural network and it is implicitly assumed that increasing NC improves the quality of a test suite. In an attempt to automatically generate a test suite that increases NC, we design a novel diversity promoting regularizer that can be plugged into existing adversarial attack algorithms. We then assess whether such attempts to increase NC could generate a test suite that (1) *detects adversarial attacks* successfully, (2) produces *natural* inputs, and (3) is *unbiased* to particular class predictions. Contrary to expectation, our extensive evaluation finds that increasing NC actually makes it harder to generate an effective test suite: higher neuron coverage leads to fewer defects detected, less natural inputs, and more biased prediction preferences. Our results invoke skepticism that increasing neuron coverage may not be a meaningful objective for generating tests for deep neural networks and call for a new test generation technique that considers defect detection, naturalness, and output impartiality in tandem.

CCS CONCEPTS

• **Software and its engineering → Software testing and debugging**; *Software reliability*; • **Computing methodologies → Neural networks**.

KEYWORDS

Testing, Software Engineering, Machine Learning, Neuron Coverage, Adversarial Attack

ACM Reference Format:
Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, Quanquan Gu, and Miryung Kim. 2020. Is Neuron Coverage a Meaningful Measure for Testing Deep Neural Networks?. In *Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20), November 8–13, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3368089.3409754

ESEC/FSE '20, November 8–13, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7043-1/20/11.
https://doi.org/10.1145/3368089.3409754

1 INTRODUCTION

Extensive progress in machine learning has enabled computers to model expected behavior with minimal human guidance and has led to its integration into many safety-critical systems [5, 24]. Since all software is prone to unanticipated and undesirable defects, creating test suites and assessing their quality is an important part of building confidence during the software lifecycle.

To assess the test adequacy of neural networks, prior work proposed neuron coverage (NC) [47] and its variants [37, 58]. This notion of NC builds on the intuition of code coverage, whilst recognizing the unique challenges and structures of neural networks. NC describes the proportion of neurons activated beyond a given threshold. The intuition here is that NC captures the magnitude of individual neuron activations independently and thus serves as a proxy for observing model behavior. Based on the implicit assumption that increasing NC can improve test suite quality, NC was used to guide test generation [47, 58]. Prior work found preliminary evidence that NC is correlated with defect detection capability [58].

To systematically increase NC during test generation, we develop a novel *diversity-promoting regularizer* that can be plugged into existing adversarial attack algorithms such as PGD [39] and CW [8]. This regularizer penalizes skewed layer-wise activations to promote more diverse neuron activation distributions. As a result, our regularizer can be added to augment existing adversarial attack methods so that these methods can induce previously inactive neurons to fire and thereby increase NC. While prior work [47, 58] has attempted to improve a few neurons' activation magnitudes at each optimization step, our diversity-promoting regularizer makes this process more systematic by incorporating NC increase and diversification into the optimization objective.

We then assess the generated test suites using three criteria. The first is *defect detection* capability, i.e., the ability to detect adversarial attacks. The second is the *naturalness* of the generated test inputs and we use the Inception Score (IS) [4, 51] and the Frèchet Inception Distance (FID) [17, 42] to assess how realistic the generated test inputs are. The third criterion is *output impartiality*, the degree to which model predictions are biased (or unbiased) towards particular class labels. Assessing impartiality is inspired by the output-uniqueness test selection criteria [2], as the test suite must exercise diverse output behavior and should not prefer only a few output values. We quantify output impartiality via Pielou's evenness [49], an *entropy-based* measure [54] from the field of ecology.

* This research was done while the third author was a graduate student at UCLA.

Equipped with the above evaluation metrics and the novel diversity promoting input generation method, we investigate the trade-offs between neuron coverage, defect detection, naturalness and output impartiality. We study two image classification datasets (MNIST and CIFAR10), one autonomous vehicle dataset (Udacity Self-Driving Car), six classification-based DNN models, two regression-based DNN models, and two attack algorithms (CW and PGD). In total, 2095 test suites, over 200,000 images, are generated. Each test suite represents a different configuration of models, datasets, attack algorithms, and hyperparameter combinations used for targeting certain layers and promoting diversity in neuron activations. This extensive analysis finds that increasing NC actually makes it harder to generate an effective test suite.

(1) **Defect Detection**: Only 2 out of 64 experimental results supported the hypothesis that NC is both strongly and positively correlated with defect detection (i.e., adversarial attack success), whereas 33 were negatively correlated, implying that increasing NC is likely to harm defect detection.
(2) **Naturalness**: Only 1 out of 64 results supported the hypothesis that NC is both strongly and positively correlated with the realism and naturalness of the inputs, whereas 44 were negatively correlated, implying that increasing NC is likely to make the generated inputs more unnatural.
(3) **Output Impartiality**: Only 3 out of 64 results supported the hypothesis that NC is both strongly and positively correlated with impartiality in output predictions, whereas 21 were negatively correlated. Certain class labels have higher NC by default and the process of increasing NC in fact biases perturbations towards those output class labels.

Our key contributions are summarized as follows:

- We develop a novel regularization technique that can be seamlessly integrated into existing adversarial attack methods to promote neural activation diversity and increase neuron coverage during test suite generation.
- We adopt the Inception Score (IS) [51] and Frèchet Inception Distance (FID) [17] as generic, scalable, and automatic means of evaluating naturalness. We are the first to apply Pielou's evenness [49] to examine the previously under-investigated issue of output impartiality in test suites.
- We conduct extensive evaluations to show that NC is neither positively nor strongly correlated with attack success, input realism, and output impartiality, which we argue are important properties to consider when testing DL systems.
- We put forward the complete code and artifacts to automatically generate test suites and replicate our empirical analysis at `https://doi.org/10.5281/zenodo.4021473`

Overall, our findings invoke skepticism that neuron coverage may not be a meaningful measure for testing deep neural networks. This result is aligned with recent skepticism that, while code coverage remains a widely used test adequacy criterion [6, 23], code coverage may not be correlated with defect detection [22] and thus may not be a meaningful metric by itself. Similar to how Inozemseva *et al.* [22] highlight an empirical lack of correlation between traditional code coverage and defect defection, our result is about a lack of *correlation*, not causation. We do not claim that NC is useless; rather, we warn researchers about the potential misuse of NC as *the objective* for test generation because a naive attempt to increase NC could sacrifice other desired properties.

These findings call for a new test generation method that not only improves defect detection, but also promotes naturalness and output impartiality to create *realistic* inputs and to exercise *diverse* output behavior. This argument to incorporate additional objectives is aligned with a recent survey of testing ML-based systems [62] that lists multiple desired testing properties, including correctness, model relevance, robustness, security, efficiency, fairness, interpretability, privacy, and surprise adequacy. Satisfying such multiple objectives may necessitate the use of multi-objective search techniques [31] or enable users to easily add domain-specific constraints to guide meaningful input transformation and oracle checking in metamorphic testing [52].

2 RELATED WORK

This section reviews related work on DL systems, DNN testing, and adversarial attacks. Work relevant to our methodology is described in greater detail in Section 3.

Deep Learning Systems. DNNs have achieved many breakthroughs in the field of artificial intelligence, such as speech recognition [18], image processing [28], statistical machine translation [3], and game playing [55]. Each DNN contains basic computational units called neurons, which are connected with one another via edges of varying importance or weight. Neurons apply a nonlinear activation function to the inner product of their inputs and weights to output a value, which becomes the input to a subsequent neuron. Layers are used to organize the directed connections between neurons and there is always one or more hidden layers between one input and one output layer. Overall, a DNN can be viewed as a meta-function that aggregates the weighted contributions from its neural subfunctions to map some input into some target output. Suboptimally set weights make the DL system vulnerable to erroneous behaviors and the opacity of these numerically-derived rules make them difficult to understand and debug.

DNN Testing. With the success of deep learning, there emerged a line of research into testing DNNs by leveraging the ideas in traditional software testing methods [15, 40]. We discuss several of the most relevant DNN testing methods that utilize the NC-based criteria as follows.

DeepXplore [47] is a white-box differential testing algorithm that leverages NC to guide systematic exploration of DNN's internal logic. Input images are modified by several domain-specific transformations, and a transformed image is selected for inclusion into a test suite if it fools at least one of several similarly trained DNNs. Their study finds that NC is a better metric than code coverage and increasing NC tends to increase ℓ_1-distance among inputs.

DeepTest [58] is a gray-box, NC-guided test suite generation approach using metamorphic relations. This effort introduced a wider range of affine transformations to predict the steering angle of an autonomous vehicle. DeepRoad [63] is a GAN-based metamorphic testing approach that utilizes a shared latent space representation to perform a sophisticated style transfer of some target road condition, i.e., rain, snow, etc., to a given source image. DeepRoad makes no attempt to systematically explore the possible input space via a metric like NC but finds that GAN-based transformations could expose new faulty behaviors.

DeepGauge expands on the idea of NC [37] by introducing three new neuron-level coverage criteria and two layer-level coverage criteria to produce a multi-granular set of DNN coverage metrics. To argue for the utility of these metrics, DeepGauge uses standard adversarial attack techniques [8, 14, 30, 46] to generate test suites. It then compares the NC of the original test suite against that of the new, augmented test suite, boosted by the generated adversarial examples. By doing so, it finds some evidence that adding adversarial examples tends to increase NC in terms of most of the proposed criteria. In Section 5, we report our results that explicit effort to increase NC actually does not improve defect detection and is often harmful in terms of naturalness and output impartiality.

Recent on-going work [11, 33, 53] found preliminary evidence that the correlation between NC and DNN robustness is rather limited and that similar structural coverage metrics for DNNs could be misleading. Specifically, their test suites are generated using the standard adversarial attack methods, and their evaluation is limited to defect detection only. Our study scope is more *comprehensive*: we use automated, quantitative measures of naturalness and output impartiality in addition to defect detection and systematically investigate the trade-offs; we design a novel diversity promoting regularizer to extend existing adversarial attack algorithms; and we include both classification models and regression models (8 models in total), as opposed to classification models only.

While our evaluation focuses on *generating* test suites, others focus on selecting existing tests based on model uncertainty [38] or surprise adequacy (*i.e., significantly different and adversarial*) [25].

Finally, it is worth noting that our proposed output impartiality criteria discussed in Section 4.3 is different from the concept of fairness in machine learning [9]. Fairness in ML is concerned with the bias of an ML model with respect to sensitive attributes, such as gender or race. Along a similar vein, Themis, a software fairness testing tool by Galhotra *et al.* [12], automatically detects causal discrimination between input-output pairs for user-specified attributes. In sharp contrast with these notions of fairness, our output impartiality is a measure of the bias on how a *test suite* exercises diverse output behaviors in an ML model.

Adversarial Attacks. Recent studies show that DNNs are vulnerable to adversarial examples [14, 57], i.e., by adding a very small, often visually imperceptible, perturbation to an input, a well-trained DNN may produce misclassifications. While adversarial attacks employ a variety of methods to induce erroneous behavior, their effectiveness is largely measured by the attack success rate of the perturbed inputs and its distortion from the original inputs. Most optimization-based adversarial attacks [8, 39] are based on ℓ_2 or ℓ_∞ norm-based perturbation. Some work [47, 58] has attempted to improve or side step the norm constraint with domain specific transformations. In our evaluation of neuron coverage, we use the standard attack methods with ℓ_∞ norm constraint, because these methods are efficient and can generate natural examples.

Adversarial attack algorithms offer both targeted and untargeted attacks for perturbing inputs to be predicted as some other class. Untargeted attacks aim to turn the prediction into any incorrect class, while targeted attacks aim to turn the prediction into a specific class. We use untargeted attacks to give them more freedom to perturb the input in whichever way NC maximization incentivizes.

Table 1: DNN Architectural Details

DNNs	Dataset	Primary Layer Type	# Layers	# Neurons
FCNet5	MNIST	Fully Connected	5	478
FCNet10	MNIST	Fully Connected	10	3,206
Conv1DNet	MNIST	Conv1D	4	35,410
Conv2DNet	MNIST	Conv2D	4	15,230
ResNet56 [16]	CIFAR10	Conv2D	56	532,490
DenseNet121 [19]	CIFAR10	Conv2D	121	563,210
DAVE2 [5]	Driving	Conv2D	10	82,669
DAVE2-N [47] ‘	Driving	Conv2D	10	82,669

3 STUDY METHODS

This section describes the datasets, DNN models, and adversarial attack algorithms used for our empirical study and describes our diversity promoting regularizer to increase neuron coverage.

3.1 Datasets and DNNs

Table 1 summarizes architectural details of all the DNNs under test.

CIFAR10 [27] is a dataset containing 32x32x3 RGB pixel images representing ten mutually exclusive classes of naturally occurring entities that are suitable for IS and FID realism measurement. We use two well-known pre-trained DNNs: a 56-layer ResNet [16, 20] and a 121-layer DenseNet [19, 48], both of which achieve competitive performance on this dataset.

MNIST [32] is a large, well-studied dataset containing 28x28x1 gray-scale pixel images representing handwritten digits from 0 to 9. For this dataset, we consider two fully connected neural networks: FCNet5 with 5 hidden layers and FCNet10 with 10 hidden layers, and two convolutional neural networks: Conv1DNet and Conv2DNet. Both convolutional neural networks have 2 convolutional layers followed by 2 fully connected layers, but vary the primary convolutional layer type from 1D to 2D. All MNIST DNNs were trained for 10 epochs using an Adam optimizer [26].

The two realism metrics we employ—IS [51] and FID [17]—are tuned on the internal structures of natural images which generally have both foregrounds and backgrounds. Because such naturalism is not applicable to a digit recognition task, we exclude MNIST when studying the relationship between NC and naturalness.

Udacity Self-Driving Car [1] is a dataset containing 480×640×3 RGB pixel images extracted from video footage shot by a camera mounted to the front of a moving vehicle and the corresponding angle of the steering wheel (±25°) for each frame. We use two pre-trained DNNs: DAVE2 and DAVE2-Norminit (abbreviated DAVE2-N), used by DeepXplore [47] and originally from NVIDIA [5].

3.2 Measuring Neuron Coverage

Pei *et al.* [47] formally define neuron coverage by the following:

$$neuron\_cov(T, \mathbf{x}, t) = \frac{|\{n|\forall \mathbf{x} \in T, out(n, \mathbf{x}) > t\}|}{|N|}$$

where $N = \{n_1, n_2, \ldots\}$ represents all the neurons in the DNN; $T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ represents all test inputs (i.e., those to be perturbed); $out(n, \mathbf{x})$ is a function that returns the output value of neuron n for a given test input $\mathbf{x}$ *scaled* to be between 0 and 1 based on the minimum and maximum neuron activations for the layer; and

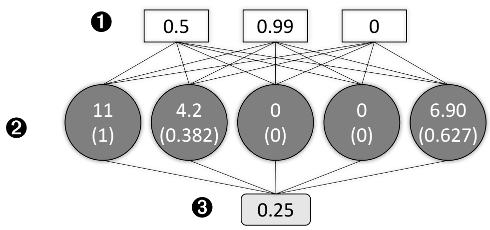

Figure 1: Single Layer DNN. ❶ represents inputs (i.e., pixels, features, etc.). ❷ represents a hidden layer of 5 neurons, where parentheses denote activations scaled between 0 and 1 for comparison against a NC threshold. ❸ represents an output layer of 1 neuron (i.e., class logits, probabilities, etc.).

t is the user-set threshold for determining whether a neuron is sufficiently activated.

Figure 1 depicts an example neural network with a single hidden layer. Each circular node corresponds to a neuron organized and color-coded by layer. The hidden layer neurons also contain their layer-wise scaled activations in parentheses for comparison against a chosen threshold t. If $t = 0$, then $NC_{t=0} = 4/6 = 0.67$, or if $t = 0.75$, then $NC_{t=0.75} = 1/6 = 0.17$. Selecting an appropriate threshold t was an open issue in early NC research. When measuring NC, we vary a threshold t for the range used by prior work [37, 47, 58], $t \in \{ 0, 0.2, 0.5, 0.75 \}$.

3.3 Adversarial Attack Algorithms

Using adversarial attacks for test generation is analogous to fuzzing in software testing and acts as a means of introducing targeted perturbations. We select the following two adversarial attack algorithms [8, 39] due to their widespread usage in the ML literature.

Carlini-Wagner (CW) [8] constructs the adversarial example $\mathbf{x} + \boldsymbol{\delta}$, where $\mathbf{x}$ is the original input to attack, $\boldsymbol{\delta}$ is the adversarial perturbation, by solving the following optimization problem:

$$\min_{\boldsymbol{\delta}} \alpha \cdot L\big(h(\mathbf{x} + \boldsymbol{\delta}), y\big) + \|\boldsymbol{\delta}\|_p \quad \text{subject to} \quad \mathbf{x} + \boldsymbol{\delta} \in [0, 1]^n,$$

where y is the label of $\mathbf{x}$, L is a suitable loss function, h is the target model, $\|\cdot\|_p$ denotes the ℓ_p-norm such as ℓ_∞, ℓ_0, ℓ_2 norms, and α is a scaling constant to balance the the loss L and the ℓ_p-norm. The intuition behind the CW attack is to find some small perturbation $\boldsymbol{\delta}$ that we can add to the original input $\mathbf{x}$ such that it will lead the target model to change its classification. To achieve this, the CW attack exploits the loss function L to guide the generation of $\boldsymbol{\delta}$ that will make the target model's classification on $\mathbf{x} + \boldsymbol{\delta}$ different from $\mathbf{x}$. By minimizing the ℓ_p-norm of $\boldsymbol{\delta}$, the CW attack can ensure that such perturbation is small. In this effort, we use the ℓ_∞ norm, where distance is measured by the pixel with the greatest magnitude change from its original value. As for the loss function L, we use the loss function provided by Carlini and Wagner [8] for our classification tasks. For our regression models, we substitute the standard CW loss function for a custom loss designed for regression tasks by Meng *et al.* [41].

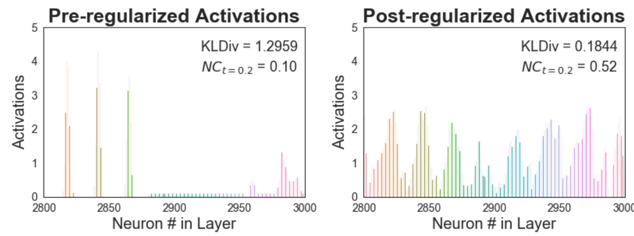

Figure 2: Neural activation before and after regularization: our regularization significantly promotes NC at $t = 0.2$.

Projected Gradient Descent (PGD) [39] finds the adversarial example $\mathbf{x} + \boldsymbol{\delta}$ by solving the following maximization problem:

$$\max_{\boldsymbol{\delta}} L\big(h(\mathbf{x} + \boldsymbol{\delta}), y\big) \quad \text{subject to} \quad \|\boldsymbol{\delta}\|_p \leq \epsilon,$$

where y is the label of $\mathbf{x}$, h represents the target model, L is the loss function for training h, ϵ is the perturbation limit. The maximization step will guide us to find the adversarial example and the ℓ_p norm constraint will make the perturbation small. For the PGD attack, projected gradient descent is performed to solve the above constrained optimization problem. We consider the ℓ_∞ norm constraint as in the CW attack, and use the sign of the gradient [14] to efficiently solve the maximization problem. For the loss function L, we choose the cross-entropy loss for classification tasks and mean square error for regression tasks. We vary a different perturbation limit $\epsilon \in \{0.1, 0.2, 0.3\}$ for the norm bounds to explore its possible effects on NC.

3.4 Extending Attacks to Increase NC

Adversarial attacks aim at creating perturbed inputs to achieve two primary objectives—maximizing loss while keeping ℓ_p-norm distance from the original inputs small. Previous research [37, 47] found that these algorithms do not produce any significant variation in NC. To increase NC while leveraging the skeleton of existing adversarial attacks, we design a novel adversarial attack regularizer to incorporate the maximization of NC as an additional objective. Our regularizer works by penalizing skewed layer-wise activations and thus promotes more diverse neural activation distributions. Diversity promotion has the effect gravitating all neurons toward the average magnitude of activation. Here we show the extended CW attack, augmented with our new diversity-promoting regularizer:

$$\min_{\boldsymbol{\delta}} \alpha \cdot L\big(h(\mathbf{x} + \boldsymbol{\delta}), y\big) + \|\boldsymbol{\delta}\|_p + \lambda \cdot \sum_{l} div(out_l(\mathbf{x} + \boldsymbol{\delta}), U)$$
$$\text{subject to} \quad \mathbf{x} + \boldsymbol{\delta} \in [0, 1]^n,$$

where $\lambda > 0$ is a user-set diversity weight to control how strongly we wish to induce higher NC; $div(\cdot)$ is a divergence function; $out_l(\cdot)$ is a function that returns the neural activations from the l^{th} layer of the DNN for the perturbed inputs $\mathbf{x} + \boldsymbol{\delta}$; U represents a uniform distribution; and we consider ℓ_∞ norm in our method (i.e., choosing $p = \infty$). We use the Kullback-Leibler (KL) divergence [29] to implement our $div(\cdot)$ function, but any other measure of the distance between two probability distributions could be suitable. KL divergence measures how much information is lost by approximating the neural activations as if they were perfectly uniform—the higher the loss, the less diverse the activations. With a sufficiently high

Table 2: Original NC, Average % Increase from Original NC, and Maximum % Increase from Original NC

	$NC_{t=0}$ (%)			$NC_{t=0.2}$ (%)			$NC_{t=0.5}$ (%)			$NC_{t=0.75}$ (%)		
DNNs	Orig	Avg ↑	Max ↑	Orig	Avg ↑	Max ↑	Orig	Avg ↑	Max ↑	Orig	Avg ↑	Max ↑
FCNet5	96.09	0.56	0.66	61.21	22.59	61.72	15.01	45.75	173.98	4.33	38.72	171.88
FCNet10	78.52	7.65	18.63	16.79	54.68	98.90	3.70	38.27	71.63	0.89	63.77	123.91
Conv1DNet	68.08	4.82	14.50	12.67	8.77	45.33	1.26	12.85	139.58	0.48	15.16	63.38
Conv2DNet	94.96	1.77	4.18	23.89	9.11	36.47	6.66	25.48	55.69	1.23	67.88	122.04
ResNet56	95.07	0.22	0.40	26.87	4.31	11.77	5.42	6.17	21.76	1.29	6.20	15.71
DenseNet121	96.46	0.04	0.06	12.88	7.00	14.49	1.20	6.59	15.85	0.16	11.77	31.87
DAVE2	78.11	9.99	15.09	13.32	4.39	30.62	2.45	-5.78	9.28	0.72	-16.17	29.23
DAVE2-N	77.57	11.90	17.26	14.69	26.77	59.26	2.54	2.63	28.91	0.46	-1.26	37.14
Average	85.61	4.62	8.85	22.79	17.20	44.82	4.78	16.50	64.59	1.20	23.26	74.40

Table 3: Experimental Variables

Variable	Values
Adversarial Attacks	CW, PGD
DNNs	FCNet5, FCNet10, Conv1DNet Conv2DNet, ResNet56, DenseNet121
Datasets	MNIST, CIFAR10
Target Layers	Varies
λ **Diversity Weights**	0, 10^0, 10^1, 10^2, 10^3, 10^4, 10^5
c **Confidence (CW)**[1]	0, 20, 40
ϵ **Limit (PGD)**	0.1, 0.2, 0.3

regularization weight placed on this objective, diversity promotion can induce previously inactive neurons to fire and increase NC. It is important to note that adding the regularizer does not necessarily harm the attack success rate as approximately 23% of our generated suites have 100% attack success. However, there tends to be an inverse relationship between the regularization weight (λ) and the attack success rate. For example, the average attack success rate is 65% when λ is 0, and with increasing λ to 1, 10^1, 10^2, 10^3, 10^4, and 10^5, the average attack success rate is 53%, 51%, 48%, 43%, 38%, and 35%, demonstrating some decrease. Figure 2 shows how our regularization promotes higher NC by having more neurons activated by visualizing neuron activation at a given layer in Conv2DNet.

Table 2 shows our regularizer's effectiveness in terms of the average and maximum percent increases in NC over the baseline NC of the original test suite images for all models. Naturally, already highly activated DNNs are more difficult to activate further, making $NC_{t=0}$ undesirable for comparison purposes. On the other hand, $NC_{t=0.5}$ and $NC_{t=0.75}$ activate significantly smaller portions of the network. We report primarily on $NC_{t=0.2}$ for visual figures.

As an implementation note, our diversity-promoting regularizer can target a specific layer, contiguous and non-contiguous layer subsets, or all layers simultaneously. In our experiments, we vary the target layer one at a time, primarily to evaluate the sensitivity of NC to this regularization. For the MNIST models, we target each layer in turn. However, for larger models, we target k layers (default $k = 6$) evenly spaced in the model, starting from the first hidden layer and ending at the output layer.

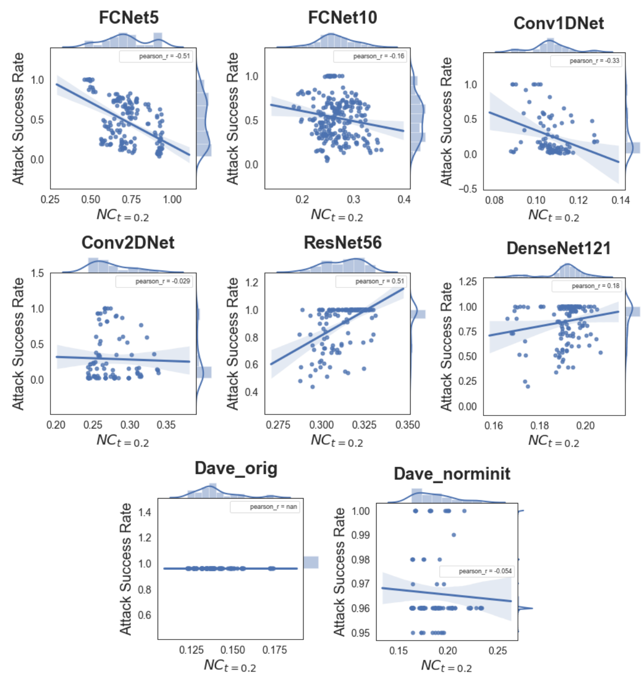

Figure 3: $NC_{t=0.2}$ vs ASR: the results show that NC does not consistently correlate with defect detection.

4 FINDINGS

For each configuration, we construct a test suite of 100 randomly selected images such that each class is equally represented. This is to ensure that the suite has complete output impartiality before perturbation. We then use the NC-augmented adversarial attack algorithm to perturb the original tests before computing NC at threshold $t \in \{0, 0.2, 0.5, 0.75\}$, defect detection, IS, FID, and output impartiality. Finally, we perform an analysis of 2,095 test suites to measure the strength, direction, and significance of correlation. The experimental conditions are listed in Table 3.

[1]The parameter c encourages the solver to find an adversarial instance that is classified as a specific class with high confidence, see Carlini and Wagner [8] for detail.

Table 4: Correlation between NC & ASR: Gray indicates a p-value > 0.05

	CW - ASR Correlations				PGD - ASR Correlations			
DNNs	**NC**$_{t=0}$	**NC**$_{t=0.2}$	**NC**$_{t=0.5}$	**NC**$_{t=0.75}$	**NC**$_{t=0}$	**NC**$_{t=0.2}$	**NC**$_{t=0.5}$	**NC**$_{t=0.75}$
FCNet5	-0.20	-0.23	-0.18	0.07	-0.10	-0.52	-0.52	-0.32
FCNet10	-0.67	**0.76**	**0.75**	0.04	-0.18	-0.16	-0.10	0.14
Conv1DNet	NA	NA	NA	NA	0.58	-0.37	0.10	0.05
Conv2DNet	-0.16	-0.20	-0.29	-0.23	0.08	-0.04	-0.16	-0.36
ResNet56	-0.46	0.59	0.58	0.57	-0.11	0.52	0.53	0.21
DenseNet121	-0.83	-0.21	-0.06	0.13	0.19	0.18	0.20	0.11
Dave2	0.02	-0.17	-0.27	-0.21	0.30	-0.16	-0.45	-0.34
Dave2-N	NA	NA	NA	NA	0.00	-0.10	0.00	-0.08
Average	-0.38	0.09	0.09	0.06	0.10	-0.08	-0.05	-0.07

All correlations are presented in a tabular form and we visualize a sample of the NC$_{t=0.2}$ results for PGD for presentation purposes. We adopt a standardized delineation of correlative significance laid out by Ratner [50] to characterize values between 0 and ±0.3 as weak, ±0.3 to ±0.7 as moderate, and ±0.7 to ±1.0 as strong. Correlation coefficients are also color-coded according to whether or not they are statistically significant. Gray indicates a p-value > 0.05 and such values are discounted in our subsequent analysis. **Emboldened** values indicate that the results support the associated hypothesis and all others do not.

4.1 Defect Detection

4.1.1 Study Method. Since our approach relies on adversarial attacks to generate test suites, we equate the attack success rate (ASR) with defect detection rate (DDR) and use both measures interchangeably. Let *pert_acc* represent the classification accuracy on the adversarially perturbed suite of test inputs (T), then DDR is simply $ASR(T) = 1 - pert\_acc$. In order to use the same metric for the regression driving models, we discretize their continuous outputs into 25 equal-width intervals [59], each representing a 2° difference in steering angle.

4.1.2 Results. Figure 3 visualizes the relationship between NC and ASR, broken down by DNN for the PGD attack, which shows that NC is volatile and NC does not consistently correlate with defect detection. Even for models that share a large degree of architectural similarity, like the FCNet5 and FCNet10 models, the correlations differ in both strength and direction, reinforcing the unpredictability of NC.

Table 4 shows the results of all configurations broken down by an attack algorithm, network, and t threshold. Only 2 out of 64 correlations satisfy the hypothesis that NC is both positively and strongly correlated with defect detection. Independent of direction, 58% of experimental configurations show a weak correlation, while 25% are merely moderate. The correlation is positive in only 36% of configurations, negative in 52%, and non-existent in 12%.

> **Defect Detection.** Our findings reject the hypothesis that NC is strongly and positively correlated with defect detection. Only 3% of the configurations supported this.

4.2 Naturalness

DL systems are designed to solve real-world problems and therefore a test suite must have realistic and natural inputs. In fact, several prior techniques are motivated by this naturalness goal and state this requirement. For example, DeepXplore [47] uses domain-specific constraints to generate test images that are *valid and realistic*. DeepTest also states that it seeks to apply well-behaved transformations to preserve realism [58, 63]. We explicitly investigate whether maximizing NC can generate test suites reflecting the naturalness of the expected input space.

4.2.1 Study Method. Appraising the visual quality of an image can be highly subjective and there is still no definitive solution on how to formalize its naturalness. Fortunately, research into generative adversarial networks (GANs) [13] has produced several popular metrics for this purpose. We select the two most highly cited metrics from the GAN literature to objectively measure naturalness.

The **Inception Score (IS)** [4, 51] formalizes the concept of naturalness by decomposing it into the following two sub-concepts:

- *Salience.* Of the possible class labels that could be applied to an individual image, only one has a high probability and the others are very low. This corresponds to the image being highly recognizable.
- *Diversity.* There are many different kinds of classes present across all images in the set.

The **Frèchet Inception Distance (FID)** [17, 42] is a measure of similarity between two datasets of images. It is calculated by computing the Frèchet distance between two Gaussians fitted to feature representations of the final average pooling layer within the InceptionV3 network [56]. The inventors, Heusel *et al.*, find evidence that FID captures the similarities of generated images better than IS and that FID correlates well with human judgement of visual quality. Unlike IS, the lower the FID value, the more realistic the images are, since the distance from the original images is smaller. Therefore, we investigate whether NC has a strong *negative* correlation with FID.

In the ML community, ImageNet [10] is considered as a comprehensive data set for image classifications. Thus, the authors of IS and FID derived these metrics based on the models trained on ImageNet and demonstrated generalizability to other datasets such as SVHN [43], CelebA [35], CIFAR10 [27], and LSUN Bedrooms [60].

Table 5: Correlation between NC & Naturalness: Gray indicates a p-value > 0.05

	CW - IS / FID Correlations				PGD - IS / FID Correlations			
DNNs	**NC$_{t=0}$**	**NC$_{t=0.2}$**	**NC$_{t=0.5}$**	**NC$_{t=0.75}$**	**NC$_{t=0}$**	**NC$_{t=0.2}$**	**NC$_{t=0.5}$**	**NC$_{t=0.75}$**
ResNet56	0.09 / 0.27	-0.87 / 0.76	-0.81 / 0.75	-0.59 / 0.59	0.34 / -0.03	-0.38 / 0.42	-0.52 / 0.46	0.06 / -0.14
DenseNet121	0.57 / -0.23	**0.73** / 0.13	0.63 / 0.24	0.46 / 0.35	-0.15 / 0.26	0.16 / -0.06	-0.08 / 0.16	0.19 / -0.12
Dave2	-0.62 / 0.61	-0.32 / -0.22	0.02 / -0.31	0.21 / -0.18	-0.55 / 0.49	-0.89 / 0.97	-0.53 / 0.60	-0.29 / 0.29
Dave2-N	0.56 / 0.88	-0.48 / 0.50	-0.53 / 0.41	-0.50 / 0.49	-0.94 / 0.96	-0.67 / 0.78	-0.28 / 0.38	-0.23 / 0.32
Average	0.15 / 0.38	-0.23 / 0.29	-0.17 / 0.27	-0.10 / 0.31	-0.33 / 0.42	-0.45 / 0.53	-0.35 / 0.40	-0.07 / 0.08

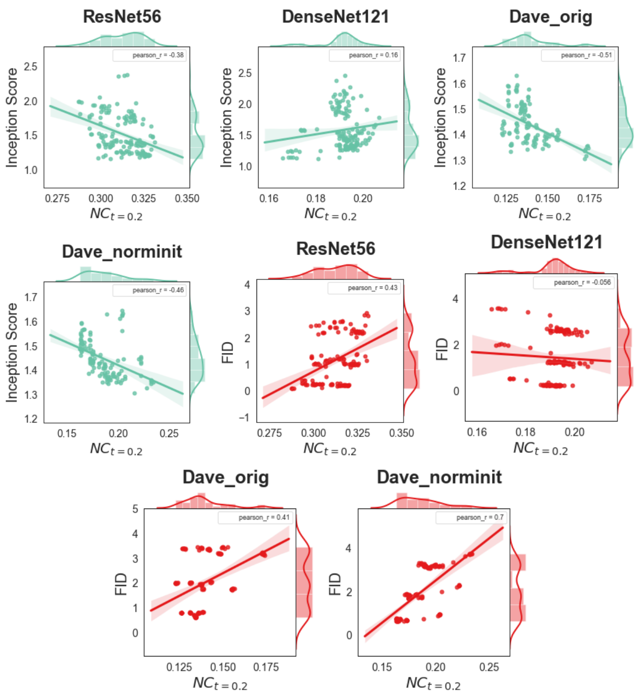

Figure 4: NC$_{t=0.2}$ vs Naturalness (IS / FID): the results show both strongly negative and strongly positive correlations.

Therefore, we use the same method that the authors of FID and IS used. In our experiments, we exclude MNIST from the measurement of IS and FID, since it is inapplicable to discuss naturalness of highly, pre-processed MNIST digit recognition. Therefore, we use only CIFAR10 and driving datasets for examining the relationship between NC and naturalness.

4.2.2 Results. Figure 4 depicts the relationship between NC and IS and FID, broken down by metric, model for the PGD attack. Once again, the wide fluctuation of strongly negative and strongly positive correlations underscore the volatility of NC. Table 5 shows the results for each attack algorithm, model, and t threshold. Only 1 out of 64 correlations satisfy the hypothesis that NC is both positively and strongly correlated with improving input naturalness. Independent of direction, 38% of configurations show a weak correlation while another 45% are merely moderate. Independent of strength, the correlation is positive in only 31% of cases.

Unlike the mixed results for IS, increasing NC invariably increases FID, making the inputs less natural. In fact, **not a single** configuration in the FID experiment supports the hypothesis.

More than half of the PGD results across both IS and FID are statistically insignificant. This is because PGD attacks enforce a more strict ϵ perturbation limit, while the perturbations of CW attacks are theoretically unbounded and thus minimize the distortion as much as possible. Since this limit tightly constrains the range of measurements, it is difficult to assess the correlation with NC.

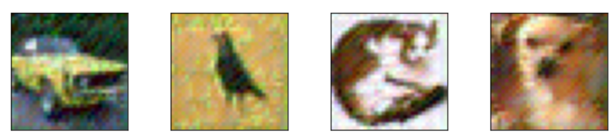

Figure 5: Test Suite #33. NC$_{t=0.2}$: 0.29 - IS: 1.97 - FID: 0.10

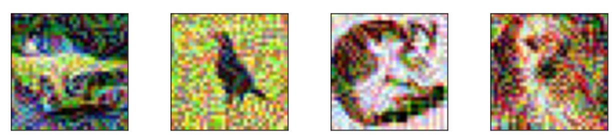

Figure 6: Test Suite #140. NC$_{t=0.2}$: 0.33 - IS: 1.48 - FID: 2.96

Figures 5 and 6 show a sample of two test suites with a 14% NC difference. While both sets of images are noticeably distorted, test suite # 140 is clearly more unnatural. Test suite # 33 has an IS about 33% higher and an FID about 29x smaller, both confirming the intuition that Figure 5 with $NC = 0.29$ is more natural than Figure 6 with $NC = 0.33$. Here, increasing NC makes noisier and more noticeably perturbed inputs, thus a less valuable test suite.

> **Naturalness.** Only 3% of all experimental results supported the hypothesis that NC is strongly and positively correlated with naturalness. 56% of the test suites are actually negatively correlated, implying that maximizing neuron coverage is likely to undermine naturalness.

4.3 Output Impartiality

The final dimension of our investigation probes the relationship between NC and the bias in model predictions. This idea of measuring the impartiality of model predictions is motivated by the

Table 6: Correlation between NC & Output Impartiality: Gray indicates a p-value > 0.05

	CW - OI Correlations				PGD - OI Correlations			
DNNs	$\mathbf{NC}_{t=0}$	$\mathbf{NC}_{t=0.2}$	$\mathbf{NC}_{t=0.5}$	$\mathbf{NC}_{t=0.75}$	$\mathbf{NC}_{t=0}$	$\mathbf{NC}_{t=0.2}$	$\mathbf{NC}_{t=0.5}$	$\mathbf{NC}_{t=0.75}$
FCNet5	0.33	0.43	0.36	0.11	0.37	**0.74**	0.52	0.26
FCNet10	**0.77**	-0.70	-0.76	-0.02	0.42	0.34	0.25	0.02
Conv1DNet	0.39	0.08	0.10	0.15	-0.57	0.18	-0.28	-0.15
Conv2DNet	-0.22	-0.02	0.29	0.34	0.34	0.11	-0.04	-0.08
ResNet56	-0.27	0.45	0.43	0.43	-0.02	0.07	0.09	0.08
DenseNet121	**0.79**	0.11	-0.04	-0.18	-0.09	-0.18	-0.06	0.08
Dave2	NA	NA	NA	NA	-0.36	0.13	0.41	0.34
Dave2-N	0.66	0.01	-0.04	0.00	-0.05	0.19	0.26	0.22
Average	0.35	0.05	0.05	0.12	0.01	0.20	0.14	0.09

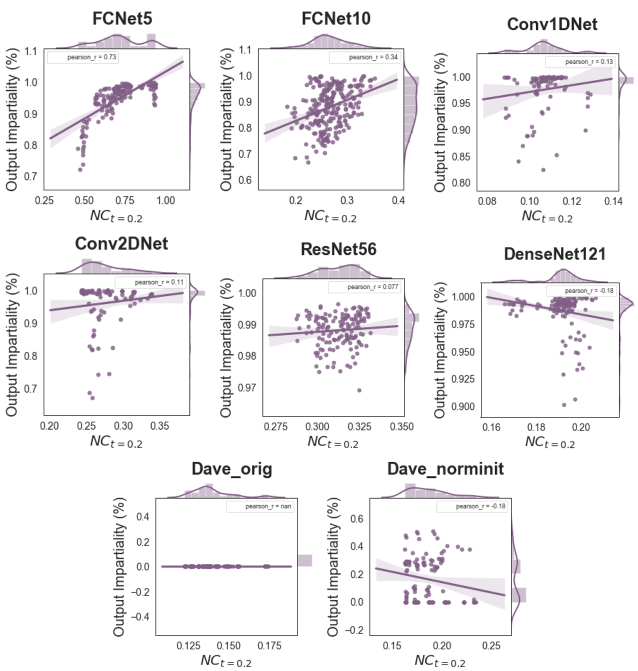

Figure 7: $\mathbf{NC}_{t=0.2}$ vs Output Impartiality: the results show that increasing NC creates bias in output behavior.

output-uniqueness test selection criteria [2] in traditional software testing, which argues that a test suite must exercise diverse output behavior and should not prefer only a few output values. Investigating the relationship between NC and output impartiality is also motivated by several observations about DNN behavior by prior work. Ilyas *et al.* [21] found that adversarial examples can be created by incorporating unnoticeable features of other classes to confuse the DL model. Similarly, Pei *et al.* found that different classes are associated with distinctive neuron activation patterns [47].

Consider a balanced test suite comprised of inputs evenly drawn from multiple classes. Suppose that the test suite is fed to a model and the model predicts always the same class label. This indicates output skew. Since one important aspect of testing is to exercise as much diverse output behavior as possible, we investigate the relationship between NC and the impartiality of predicted outcomes.

4.3.1 Study Method. We take inspiration for measuring impartiality from adjacent work on ecological biodiversity [34]. Instead of considering a distribution of species, we recast *impartiality* as a measure of the distribution of class predictions under a uniform input distribution (i.e. the initial test suite contains an equal number of inputs from each class). We use Pielou's *evenness* score [49], a theoretically grounded measure of biodiversity [36] to assess the skew of the output class distribution. It uses a normalized Shannon's entropy [54] scaled to a range of 0 and 1 by dividing the entropy of each test suite's output distribution by the maximum entropy given the total number of classes. A high evenness score entails high impartiality (low bias). We define an output impartiality metric for a test suite T with $|C|$ possible classes, indexed by k:

$$output\_impartiality(T) = \frac{\sum_{t \in C_k} P_{t=C_k} \log P_{t=C_k}}{\log |C|},$$

where $|C|$ is the cardinality of classes and $P_{t=C_k}$ represents the percentage of the test cases t predicted to belong to class C_k. For the regression models, we use the same discretization method as before to enable the use of this metric.

4.3.2 Results. Figure 7 visualizes the relationship between NC and output impartiality by DNN for CW. The results show that increasing NC creates bias in output behavior. Table 6 shows the results of all configurations by an attack algorithm and t threshold. Only 3 out of 64 configurations show that NC is both positively and strongly correlated with output impartiality. Independent of strength, the correlation is negative in 33% of correlations. Independent of direction, 62% of experimental configurations show a weak correlation while 32% are moderate.

4.3.3 Investigating Output Bias Caused by NC. In addition to the previous section's correlation analysis, we design another experiment to investigate which classes are likely to be over-represented in the outputs after a test suite has been perturbed to maximize NC.

The idea of maximizing NC during test suite generation does not take into account that different classes of inputs can already have different baseline NC levels. For example, it may be the case that a set of inputs containing only the "dog" class in CIFAR10 has $NC_{t=0} = 0.9$ while another set of inputs containing the same number of "cars" has $NC_{t=0} = 0.6$. Increasing NC may then bias the perturbations—and therefore the output predictions—towards

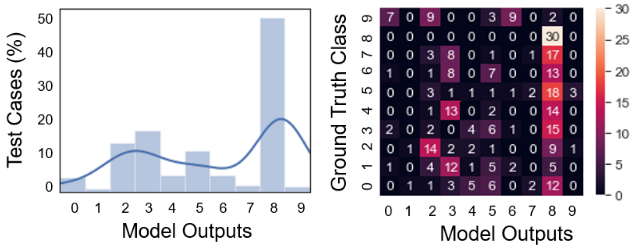

Figure 8: Output Prediction Distribution Histogram (left) and Cross-Class Prediction Heatmap (right)

the class "car" with the higher NC baseline instead of "dog". Below we describe an experiment conducted with the MNIST dataset to investigate this further.

We generate 10 partitions of the test data—one partition for each class—by randomly selecting 100 instances of that class from the test set. These partitions are then used to calculate a class-specific NC baseline. Since NC depends on the choice of t, we repeat NC baseline calculation for each class label, while varying t from 0 to 0.9 in an increment of 0.1. This process reveals which class label has the highest NC baseline, the second highest NC baseline, and so on. In other words, we rank class labels from the highest NC (Rank 1) to the lowest NC (Rank 10).

Suppose that class label 8 has a rank {1, 3, 3} and class label 1 has a rank {8, 9, 10}, respectively for $t \in \{0, 0.5, 0.9\}$. A low average rank for class 8 (2.3) indicates that class 8 tends to have a high NC baseline regardless of t. On the other hand, a high average rank for class 1 (9) indicates that class 1 tends to have a low NC baseline. Therefore, during NC maximization, the perturbation process may favor over-representing class 8 in the output predictions. However, suppose that class 3 has an average ranking closer to 5—the midpoint of 10 possible labels. That implies that class 3 may have a high NC baseline under a certain threshold, but may have a low NC baseline under another threshold, or places the fifth for all t, etc. Thus, it would be unlikely for NC maximization to *consistently prefer* over-representation of outputs associated with class label 3 in the resulting test suite.

Class	**8**	**5**	**2**	**0**	**7**	**3**	**4**	**9**	**1**	**6**
Average Rank	2.1	2.9	3.1	4.1	5.0	6.2	7.0	7.4	8.2	8.6

Table 7: MNIST Class and Average Rank of NC Baseline

Concretely, an average rank closer to 1 indicates a greater likelihood of being over-represented in the output distribution through NC-maximization. Table 7 reports the average class ranks for the 10 class labels of MNIST. Here, we can see that class label 8 tends to have a high NC baseline and that class label 6 tends to have a low NC baseline across different thresholds. Therefore in the NC-maximized test suite, class 8 is likely to be over-represented and class 6 is likely to be under-represented in the output distribution.

We first construct a group of inputs with an output impartiality 1 by drawing 30 inputs per class label—every class is equally represented in the output distribution, because all are correctly predicted by the Conv2DNet trained for MNIST. We then use our test generation algorithm with a diversity-promoting regularizer to perturb the input set to increase its NC. The histogram on the left in Figure 8 shows the percentage of model predictions for each class. The heatmap on the right details how many of the inputs belonging to each class in the original group were perturbed into to predicting another class label. This perturbed test suite had a $NC_{t=0.2}$ of 0.34—about 40% higher than the original images, but 45% of all predictions are now for class 8, demonstrating output skew (a low impartiality score of 0.26). As expected, the classes with the low NC baselines (e.g., 6, 1, and 9) are among the most under-represented. This shows that, when NC maximization is used as a guidance criterion, a test generation technique can easily satisfy this criterion by simply perturbing inputs towards the class label with the highest NC baseline.

> **Output Impartiality.** Only 5% of all experimental results support the hypothesis that NC is both strongly and positively correlated with output impartiality. When a few class labels have higher NC baselines than the other class labels, increasing NC biases the test suite to predominantly incorporate the features of this preferred subset.

5 DISCUSSION

This section includes additional evidence and rationale that questions the meaningfulness of neuron coverage.

5.1 DeepXplore & DeepTest Comparison

It is certainly possible that another method may create a natural test suite with high NC. Therefore we perform similar analysis on the test suites generated by DeepXplore [47] and DeepTest [58] to see whether similar trade-offs exist. We utilize the authors' publicly available implementations to generate tests for the MNIST and Driving datasets. For DeepXplore, **not a single** correlation is sufficiently strong enough to support the three hypotheses that NC is positively related with defect detection, naturalness, and output impartiality. For DeepTest, only one correlation for output impartiality at $NC_{t=0.5}$ is strongly positive. In fact, our investigation finds that many images generated by DeepXplore and DeepTest turn rich driving scenes into completely white images, yet retain their original labels. No human or program can predict a steering angle from such an unnatural input.

While the results from DeepXplore and DeepTest may have been sufficient to warrant skepticism about NC, our NC-maximizing approach is easily applicable and systematic. First, it can probe the behavior of a single model, while DeepXplore's differential testing requires multiple models. As DNNs become large and costly to train [7], differential testing may become less practical. Second, by directly extending adversarial attacks that maximize defective behavior and minimize the norm-distance from their original sources, the generated test suite is orders of magnitude more natural. For instance, our test suites have average FID scores 458× and 3,887× higher than those created by DeepXplore and DeepTest respectively. While it is certainly possible that yet another method may create

Figure 9: Visualization of neuron activations shows mixed concepts—cats, foxes, and cars [45]

a natural test suite with high NC, our comprehensive experimentation of 2095 test configurations suggests that increasing NC is unlikely to correlate with defect detection, naturalness, and output impartiality. Triangulation between these approaches increases confidence about the external validity of our findings.

5.2 How Meaningful is a Neuron?

The viability of NC as a DNN testing metric is underpinned by the idea that "each neuron independently extracts a specific input feature" [47] rather than collaborating with other neurons. However, recent research into DNN visualization techniques [44, 45, 61] has demonstrated that this is not so—neuron independence and local feature extraction do not accurately characterize DNN behavior. Instead, the neurons in a layer interact with one another to pass information to subsequent layers and NC does not capture the richness of such neuron interactions. While the probability that a neuron distinctly encodes a specific feature increases the deeper it is situated in the DNN, many of the neurons represent an amalgam of very different abstract concepts, like the visualization of pixels leading to high activations of certain neurons in Figure 9 [45]. This observation raises serious doubts about whether neurons are even the right semantic units for understanding DNN behavior, further questioning the viability of NC as a meaningful test metric.

5.3 Does NC maximization make sense for testing DNNs?

Assuming the best case scenario of neurons independently encoding specific features, is maximization of NC even desirable? Consider each neuron in a DNN as a binary classifier checking for the presence or absence of a specific feature within the input. For n neurons in a DNN, there are 2^n possible activation patterns. In general, establishing a single objective to maximize NC could easily target having *one* possible pattern, where all neurons are activated. As a simplified example, consider two neurons in a DNN trained for autonomous driving. Suppose that one detects the presence of vehicles and the other detects stop signs. NC maximization as a single objective in test generation can be easily satisfied with a single image containing both a vehicle and a stop sign together. Subsequently, such limited focus on NC could easily produce a test suite that does not cover other interesting portions of the potential input space.

6 CONCLUSION

Recent effort to test deep learning systems has produced an intuitive testing adequacy metric, called neuron coverage (NC) and its several variants. Prior work has also produced several test generation techniques that use NC as a guidance criterion and some has found evidence that adding adversarial inputs to an existing test suite tends to increase NC.

To systematically incorporate NC maximization to existing adversarial attack algorithms, we designed a novel diversity promoting regularizer that can be plugged into existing attack algorithms to increase NC. We then assessed the quality of the resulting test suites in terms of defect detection, naturalness, and output impartiality. From our evaluation of 2,095 experimental configurations involving 8 DNNs, 2 datasets, and 2 adversarial attack algorithms, we conclude that *NC should not be blindly trusted as a guidance metric for DNN testing*. While we do not claim that NC is useless, increasing NC actually has a harmful effect by producing less natural inputs and by creating a skew in output distribution. This result is aligned with recent skepticism that code coverage in traditional software testing is not strongly correlated with test suite effectiveness and thus may not be a meaningful metric by itself [22].

We therefore advocate incorporating other test objectives such as *naturalness* and *output impartiality* and use multi-objective search techniques for testing DL systems. Our experience of adapting existing adversarial attack algorithms for test generation has shown that it is fairly easy to create inputs that lead to misprediction by sacrificing naturalness, and that it is also fairly easy to perturb a test suite to produce a high NC score by skewing the output distribution. Our results call for more systematic research on how to generate *realistic* inputs that reveal *meaningful*, undesired behavior in DL systems. Such a research direction may require new methods to empower users to easily specify domain specific constraints expressively and to leverage those constraints to guide test generation.

Per open science policy, the code and data is available at https://doi.org/10.5281/zenodo.4021473.

ACKNOWLEDGEMENT

We thank anonymous reviewers for their comments. The participants of this research are in part supported by NSF grants CCF-1764077, CCF-1527923, CCF-1723773, SaTC-1717950; ONR grant N00014-18-1-2037; Intel CAPA grant; Samsung grant; Google PhD Fellowship; and the Alexander von Humboldt Foundation.

REFERENCES

[1] 2016. *Using Deep Learning to Predict Steering Angles.* https://github.com/udacity/self-driving-car
[2] Nadia Alshahwan and Mark Harman. 2014. Coverage and Fault Detection of the Output-Uniqueness Test Selection Criteria. In *Proceedings of the 2014 International Symposium on Software Testing and Analysis* (San Jose, CA, USA) *(ISSTA 2014)*. Association for Computing Machinery, New York, NY, USA, 181–192. https://doi.org/10.1145/2610384.2610413
[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015.*
[4] Shane Barratt. 2018 (accessed August 8, 2019). *Inception Score for GANs in Pytorch.* https://github.com/sbarratt/inception-score-pytorch
[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. 2016. End to End Learning for Self-Driving Cars. *CoRR* abs/1604.07316 (2016). arXiv:1604.07316 http://arxiv.org/abs/1604.07316
[6] Benjamin Brosgol. 2011. Do-178C: The Next Avionics Safety Standard. In *Proceedings of the 2011 ACM Annual International Conference on Special Interest Group on the Ada Programming Language* (Denver, Colorado, USA) *(SIGAda '11)*. ACM, New York, NY, USA, 5–6. https://doi.org/10.1145/2070337.2070341
[7] T. Brown, B. Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, P. Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, G. Krüger, Tom Henighan, R. Child, Aditya Ramesh, D. Ziegler,

Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, E. Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, J. Clark, Christopher Berner, Sam McCandlish, A. Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *ArXiv* abs/2005.14165 (2020).

[8] Nicholas Carlini and David A. Wagner. 2016. Towards Evaluating the Robustness of Neural Networks. *2017 IEEE Symposium on Security and Privacy (SP)* (2016), 39–57.

[9] Alexandra Chouldechova and Aaron Roth. 2018. The frontiers of fairness in machine learning. *arXiv preprint arXiv:1810.08810* (2018).

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[11] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Jianye Hao, Xinyu Wang, Li Wang, Jin Song Dong, and Dai Ting. 2019. There is Limited Correlation between Coverage and Robustness for Deep Neural Networks. *arXiv preprint arXiv:1911.05904* (2019).

[12] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness Testing: Testing Software for Discrimination. 498–510. https://doi.org/10.1145/3106237.3106277

[13] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 2672–2680. http://dl.acm.org/citation.cfm?id=2969033.2969125

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. In *ICLR*.

[15] Kelly J Hayhurst. 2001. *A practical tutorial on modified condition/decision coverage.* DIANE Publishing.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), 770–778.

[17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *NIPS*.

[18] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29, 6 (2012), 82–97.

[19] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2016. Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 2261–2269.

[20] Yerlan Idelbayev. 2018 (accessed August 8, 2019). *Proper ResNet Implementation for CIFAR10/CIFAR100 in pytorch.* https://github.com/akamaster/pytorch_resnet_cifar10

[21] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial Examples Are Not Bugs, They Are Features. *ArXiv* abs/1905.02175 (2019).

[22] Laura Inozemtseva and Reid Holmes. 2014. Coverage is Not Strongly Correlated with Test Suite Effectiveness. In *Proceedings of the 36th International Conference on Software Engineering* (Hyderabad, India) *(ICSE 2014)*. ACM, New York, NY, USA, 435–445. https://doi.org/10.1145/2568225.2568271

[23] ISO 26262-6:2011(en) 2011. *Road vehicles — Functional safety — Part 6: Product development at the software level.* Standard. International Organization for Standardization, Geneva, CH.

[24] Kyle Julian, Mykel Kochenderfer, and Michael Owen. 2018. Deep Neural Network Compression for Aircraft Collision Avoidance Systems. *Journal of Guidance, Control, and Dynamics* 42 (11 2018), 1–11. https://doi.org/10.2514/1.G003724

[25] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding Deep Learning System Testing Using Surprise Adequacy. *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)* (2019), 1039–1049.

[26] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[27] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. [n.d.]. CIFAR-10 (Canadian Institute for Advanced Research). ([n. d.]). http://www.cs.toronto.edu/~kriz/cifar.html

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[29] S. Kullback and R. A. Leibler. 1951. On Information and Sufficiency. *Ann. Math. Statist.* 22, 1 (03 1951), 79–86. https://doi.org/10.1214/aoms/1177729694

[30] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. 2018. Adversarial Examples in the Physical World. In *Artificial Intelligence Safety and Security*. Chapman and Hall/CRC, 99–112.

[31] Kiran Lakhotia, Mark Harman, and Phil McMinn. 2007. A multi-objective approach to search-based test data generation. *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference*, 1098–1105. https://doi.org/10.1145/1276958.1277175

[32] Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. (2010). http://yann.lecun.com/exdb/mnist/

[33] Zenan Li, Xiaoxing Ma, Chang Xu, and Chun Cao. 2019. Structural Coverage Criteria for Neural Networks Could Be Misleading. In *Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results* (Montreal, Quebec, Canada) *(ICSE-NIER '19)*. IEEE Press, 89–92. https://doi.org/10.1109/ICSE-NIER.2019.00031

[34] X. Liu, L. Zhang, and S. Hong. 2010. Global biodiversity research during 1900–2009: a bibliometric analysis. *Biodiversity and Conservation* 20 (2010), 807–826.

[35] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

[36] Jost Lou. 2010. The Relation between Evenness and Diversity. *Diversity* 2 (02 2010). https://doi.org/10.3390/d2020207

[37] 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. 2018. DeepGauge: Multi-granularity Testing Criteria for Deep Learning Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (Montpellier, France) *(ASE 2018)*. ACM, New York, NY, USA, 120–131. https://doi.org/10.1145/3238147.3238202

[38] Wei Ma, Mike Papadakis, Anestis Tsakmalis, Maxime Cordy, and Yves Le Traon. 2019. Test Selection for Deep Learning Systems. *ArXiv* abs/1904.13195 (2019).

[39] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJzIBfZAb

[40] William M McKeeman. 1998. Differential testing for software. *Digital Technical Journal* 10, 1 (1998), 100–107.

[41] Lubin Meng, Chin-Teng Lin, Tzyy-Ping Jung, and Dongrui Wu. 2019. White-Box Target Attack for EEG-Based BCI Regression Problems. In *International Conference on Neural Information Processing*. Springer, 476–488.

[42] mseitzer. 2018 (accessed August 9, 2019). *Frèchet Inception Distance (FID score) in PyTorch.* https://github.com/mseitzer/pytorch-fid

[43] Yuval Netzer, T. Wang, A. Coates, Alessandro Bissacco, B. Wu, and A. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning.

[44] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. 2016. Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks. *ArXiv* abs/1602.03616 (2016).

[45] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. 2017. Feature Visualization. *Distill* (2017). https://doi.org/10.23915/distill.00007 https://distill.pub/2017/feature-visualization.

[46] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 372–387.

[47] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China) *(SOSP '17)*. ACM, New York, NY, USA, 1–18. https://doi.org/10.1145/3132747.3132785

[48] Huy Phan. 2019 (accessed October 4, 2019). *PyTorch models trained on CIFAR-10 dataset.* https://github.com/huyvnphan/PyTorch-CIFAR10

[49] E. C. Pielou. 1966. The measurement of diversity in different types of biological collections.

[50] Bruce Ratner. 2009. The correlation coefficient: Its values range between +1/1, or do they? *Journal of Targeting, Measurement and Analysis for Marketing* 17, 2 (01 Jun 2009), 139–142. https://doi.org/10.1057/jt.2009.5

[51] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. *ArXiv* abs/1606.03498 (2016).

[52] Sergio Segura, Gordon Fraser, Ana B. Sánchez, and Antonio Ruiz-Cortés. 2016. A Survey on Metamorphic Testing. *IEEE Transactions on Software Engineering* 42 (09 2016), 1–1. https://doi.org/10.1109/TSE.2016.2532875

[53] Jasmine Sekhon and Cody Fleming. 2019. Towards Improved Testing For Deep Learning. *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)* (2019), 85–88.

[54] C. E. Shannon. 2001. A Mathematical Theory of Communication. *SIGMOBILE Mob. Comput. Commun. Rev.* 5, 1 (Jan. 2001), 3–55. https://doi.org/10.1145/584091.584093

[55] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.

[56] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), 2818–2826.

[57] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR*.

[58] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. DeepTest: Automated Testing of Deep-neural-network-driven Autonomous Cars. In *Proceedings of the 40th International Conference on Software Engineering* (Gothenburg, Sweden) *(ICSE '18)*. ACM, New York, NY, USA, 303–314. https://doi.org/10.1145/3180155.3180220

[59] Luís Torgo and Joo Gama. 1999. Regression by Classification. (07 1999).

[60] Jian xiong Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. 2010. SUN database: Large-scale scene recognition from abbey to zoo. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), 3485–3492.

[61] Jason Yosinski, Jeff Clune, Anh Mai Nguyen, Thomas J. Fuchs, and Hod Lipson. 2015. Understanding Neural Networks Through Deep Visualization. *ArXiv* abs/1506.06579 (2015).

[62] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2019. Machine Learning Testing: Survey, Landscapes and Horizons. *CoRR* abs/1906.10742 (2019). arXiv:1906.10742 http://arxiv.org/abs/1906.10742

[63] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (Montpellier, France) *(ASE 2018)*. ACM, New York, NY, USA, 132–142. https://doi.org/10.1145/3238147.3238187