

Corrupting Neuron Explanations of Deep Visual Features

Divyansh Srivastava
 UCSD CSE
 ddivyansh@ucsd.edu

Tuomas Oikarinen
 UCSD CSE
 toikarinen@ucsd.edu

Tsui-Wei Weng
 UCSD HDSI
 lweng@ucsd.edu

Abstract

The inability of DNNs to explain their black-box behavior has led to a recent surge of explainability methods. However, there are growing concerns that these explainability methods are not robust and trustworthy. In this work, we perform the first robustness analysis of Neuron Explanation Methods under a unified pipeline and show that these explanations can be significantly corrupted by random noises and well-designed perturbations added to their probing data. We find that even adding small random noise with a standard deviation of 0.02 can already change the assigned concepts of up to 28% neurons in the deeper layers. Furthermore, we devise a novel corruption algorithm and show that our algorithm can manipulate the explanation of more than 80% neurons by poisoning less than 10% of probing data. This raises the concern of trusting Neuron Explanation Methods in real-life safety and fairness critical applications.

1. Introduction

Deep neural networks (DNNs) have revolutionized many areas of Machine Learning and have been successfully used in various domains, including computer vision and natural language processing. However, one of the main challenges of DNNs is that they are black-box models with little insight into their inner workings. This lack of explainability can be problematic, particularly when using DNNs in safety and fairness critical real-world applications, such as autonomous driving, healthcare or loan/hiring decisions. Hence, it is imperative to explain the predictions of DNNs to increase human trust and safety of deployment.

The need for explainability has led to a recent surge in developing methods to explain DNN predictions. In particular, *Neuron Explanation methods* (NEMs) [2, 3, 10, 7] have attracted great research interest recently by providing global description of a DNN’s behavior. This type of method explains the roles of each neuron in DNNs using human-interpretable concepts (e.g. natural language descriptions). Representative methods include Network dis-

section [2, 3], Compositional Explanations [10] and MILAN [7].

There has been recent interest in using NEMs to explain DNNs in safety critical tasks such as medical tasks, including brain tumor segmentation [11] and medical image analysis [12]. However, despite of the excitement of NEMs, unfortunately the *explanations* provided by NEMs could be significantly corrupted and not trustworthy, as first shown in this work. The untrustworthiness of explanation may result in negative consequences and misuse. For example, imagine an auditor uses a NEM to monitor potential biases in computer vision models by inspecting whether there exists neurons activating for specific skin colors. Such auditor might depend on user-provided probing data as it is impossible for auditors to collect validation datasets for every task, and would rely on manual inspections of images to find corruption. This would incentivize the developer of a biased model to fool NEMs by providing a corrupted probing dataset that bypasses manual inspection since perturbations are imperceptible by human eyes. This raises serious concerns about utilizing these NEMs for trusting model prediction.

In this paper, we would like to bring this awareness to the research community by performing the first robustness evaluation of Neuron Explanation methods. Specifically, we unify existing NEMs under a generic pipeline and develop techniques to corrupt the neuron descriptions with a high success rate. We show that the neuron explanations could be significantly corrupted by both random noises and well-designed perturbation by only lightly corrupting the probing dataset in NEMs. We summarize our contributions in this work below:

1. We are the first to present a unified pipeline for Neuron Explanation methods and show that the pipeline is prone to corruptions on the input probing dataset.
2. We design a novel corruption on the probing dataset to manipulate the concept assigned to a neuron. Further, our formulation can manipulate the pipeline without explicit knowledge of the similarity function used by Neuron Explanation methods.

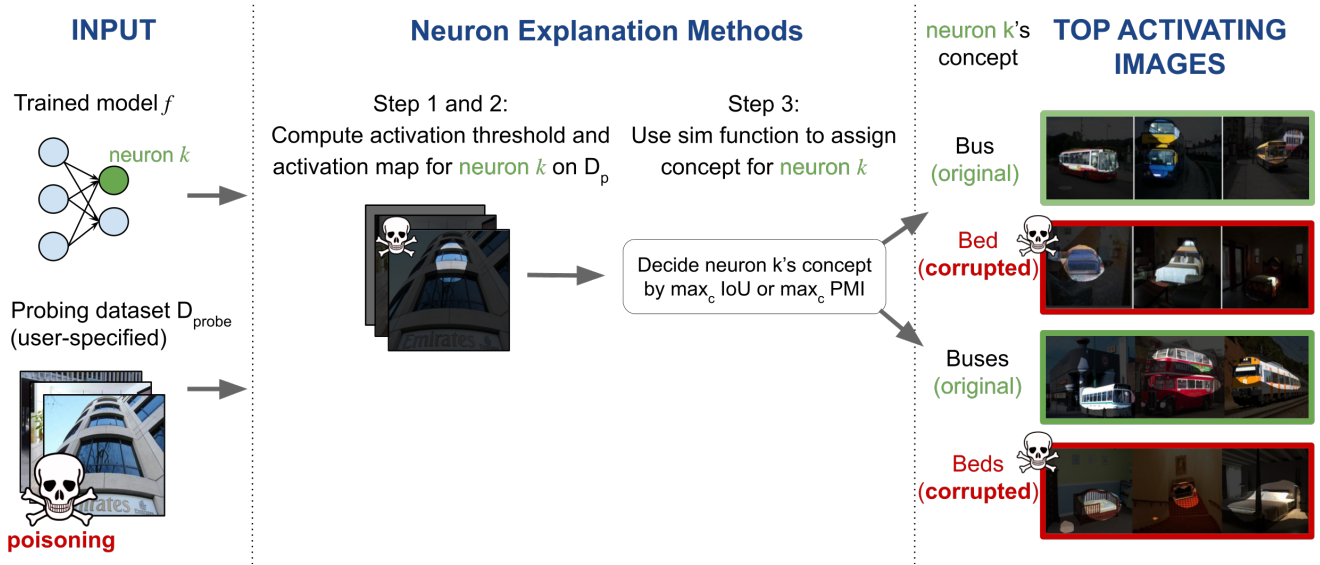


Figure 1: Pipeline depicting corruption of probing dataset to manipulate concepts of Neuron Explanation methods. Red words and outlines indicate parts of pipeline which have been manipulated due to corruption of the probing dataset.

3. We conduct the first large-scale study on the robustness of NEMs in terms of random noise and designed corruptions on the probing dataset. We show that even Gaussian random noise with a low standard deviation of 0.02 can manipulate up to 28% of neurons, and our proposed algorithm can further corrupt the concepts of more than 80% of neurons by poisoning less than 10% probing images. This raises concerns about deploying NEMs in practical applications.

2. Related Work

The mainstream interpretability methods to explain DNN’s behavior can be categorized into two branches: one is at the instance level and the other is at the model/neuron level. The NEMs that we focus on in this paper provide explanations at the neuron/model level, and we provide a brief introduction on the instance-level interpretability method below.

Feature-attribution Explanations for DNNs. Instance-level interpretability methods are also often known as feature-attribution methods, as they explain the model decision on each input instance by examining which input features (pixels) contribute the most towards a model’s output. This type of method includes CAM [18], Grad-CAM [13], Integrated Gradients [16], SmoothGrad [15], and DeepLIFT [14]. Unfortunately, recent work has shown that the interpretations provided by instance-level interpretability methods can be easily manipulated [17, 1, 5], which has increased concerns about utilizing these methods for trusting model prediction. It is worth noting that while

the robustness of the *Feature-attribution* methods is being investigated, the robustness of *Neuron Explanation methods* has remained unexplored, which is thus the main focus of this work.

Neuron Explanation methods (NEMs) for DNNs.

NEMs explain model behavior on a global level by assigning descriptions to *individual neuron*. Some of the representative methods in this class include Network Dissection [2, 3], Compositional Explanations [10], and the recent work MILAN [7]. Network Dissection [2] is the first work proposing understanding DNNs by inspecting the functionality of each neuron with a set of pre-defined *concepts*. The key idea of Network Dissection [2] is to detect the concept of a neuron by matching the neuron activations to the pattern of a ground-truth concept label mask. It assigns an atomic concept to a neuron and does not capture relationships between different concepts. The work Compositional Explanations [10] builds on Network Dissection and generates explanations by searching for logical forms defined by a set of composition operators over atomic concepts. The work MILAN [7] is a recent NEM that provides a natural language description of neurons. The idea of MILAN is to represent a neuron with a set of probing image that maximally activate it and obtain a description that maximizes pointwise mutual information of this set. Note that [3] is an extension of [2] with the modification of obtaining pixel-level concepts from a segmentation model. This setting also falls into the general NEMs pipeline that we have described in Sec 3.1, and can be handled by our unified pipeline in Sec 3.2. In this work, we focus on Network Dissection [2, 3] and

MILAN [7], which are representative methods in this class, but our formulation is general and applicable to all NEMs.

We highlight that this is the first work to study the sensitivity of neuron *explanations* from NEMs, while prior work mostly focused on manipulating DNN’s *predictions* and requires very different methods. Prior work on Cut-and-paste/copy-paste methods [4, 7] aim to manipulate a DNN’s *prediction* by adding patch of unrelated object on input image at the test time using information from NEMs explanation. In contrast, this work shows that the NEMs themselves are vulnerable, and can produce wrong explanations when their inputs are corrupted. As NEMs have started being adopted to explain DNNs at neuron level in safety-critical domains like healthcare [11, 12], it is critical to understand NEMs vulnerabilities and to caution the community against blindly trusting explanations from NEMs.

3. Our Approach

In this section, we describe our approach to evaluate the robustness of the *Neuron Explanation Methods* (NEMs). In Sec 3.1, we show that existing NEMs can be unified into a generic pipeline which is crucial to study the robustness of this class of methods. In Section 3.2, we present a novel dataset corruption algorithm and show that it is possible to exploit the vulnerability in the probing step of the unified pipeline to manipulate neuron explanations.

3.1. Unifying Neuron explanation pipelines

In this section, we present a unified pipeline for NEMs and show that the representative methods, including Network Dissection [2] and MILAN [7], are special instances of this generic pipeline. Define $X = \{x \mid x \in \mathbb{R}^{m \times n \times 3}\}$ as a set of images, Y as a set of labels, and $f : X \rightarrow Y$ to be a Convolution Neural Network (CNN) with L layers. Further, let i represents a neuron unit in f , and $f_i(x)$ denote i -th neuron’s activation map at input image $x \in \mathbb{R}^{m \times n \times 3}$, which is a three dimensional tensor. Network Dissection and MILAN describe individual channels in a convolutional layer and call them individual *neuron*, because features in the same channel correspond to the same (filter) weight. In the remaining paper, we will use *neuron*, *unit* or *channel* interchangeably to refer to the individual channels of a convolutional layer.

Let \mathcal{D}_{probe} represent the probing dataset used for computing activations of a neuron, and \mathcal{D}_c represent the set of concepts (i.e. explanations) that can be assigned to a neuron. The overall unified NEMs pipeline returns an explanation or concept $c_i^* \in \mathcal{D}_c$ for a neuron i in the following three steps:

- **Step 1.** Compute activation threshold T_i as the top η quantile level for probability distribution \mathbb{P} of activation

maps $f_i(x)$, $\forall x \in \mathcal{D}_{probe}$:

$$T_i = \left[\max_t \mathbb{P}_x [f_i(x) > t] \geq \eta \right] \quad (1)$$

- **Step 2.** Create binary activation map $M_i(x) \in \{0, 1\}^{m \times n}$ by upsampling $f_i(x)$ to match the size of input image $x \in \mathbb{R}^{m \times n \times 3}$ using a biliner interpolation function \mathcal{B} , and thresholding with T_i :

$$M_i(x) = [\mathcal{B}(f_i(x)) \geq T_i] \quad (2)$$

- **Step 3.** Assign concept with the maximum value on the similarity function sim to neuron i :

$$c_i^* = \underset{c \in \mathcal{D}_c}{\text{argmax}} \text{sim}(\{(x, M_i(x)) \mid x \in \mathcal{D}_{probe}\}, c) \quad (3)$$

As we can see, step 1 and 2 are method-agnostic, as they only associate with the network $f(x)$ to be dissected and the probing dataset \mathcal{D}_{probe} , while step 3 will use a similarity function, sim , that is specific to different methods as discussed below.

(I) sim of Network Dissection [2, 3]: For Network Dissection, let $L_c(x) \in \{0, 1\}^{m \times n}$ represent the binary ground-truth segmentation mask for a concept $c \in \mathcal{D}_c$, which indicate whether each pixel in x is associated with the concept c . In Network dissection, $L_c(x)$ can be obtained either by outsourcing experts to densely annotate probing images $x \in \mathcal{D}_{probe}$ with pixel-level concepts (e.g. broaden in [2]), or using a pre-trained segmentation model [3] to perform pixel-level segmentation for each probing image in order to obtain the pixel-level concept information. The concept set \mathcal{D}_c is a closed set containing all the possible explanations/concepts that can be assigned to a pixel. Further, Network Dissection uses $\eta = 0.005$ to calculate a neuron’s activation thresholds. The similarity function sim calculates *Intersection Over Union* (IOU) for a concept c and neuron i by examining the overlap between $M_i(x)$ and the ground truth segmentation mask $L_c(x)$ over \mathcal{D}_{probe} as

$$\begin{aligned} \text{sim}(\{(x, M_i(x)) \mid x \in \mathcal{D}_{probe}\}, c) \\ = \frac{\sum_{x \in \mathcal{D}_{probe}} |M_i(x) \cap L_c(x)|}{\sum_{x \in \mathcal{D}_{probe}} |M_i(x) \cup L_c(x)|} \end{aligned} \quad (4)$$

(II) sim of MILAN [7]: MILAN provides a natural language description for a neuron, and \mathcal{D}_c is an open set containing all possible natural language descriptions. The similarity function sim can be split into two steps. First, we calculate an exemplar set E_i containing k top-activating images $x \in \mathcal{D}_{probe}$ on a neuron i as given Eq 5a, and then calculate the pointwise mutual information (PMI) for concept c and exemplar set E_i as given in Eq 5b.

$$E_i = \{(x, M_i(x)) \mid M_i(x) = [\mathcal{B}(f_i(x)) \geq T_i]\} \quad (5a)$$

$$\begin{aligned} \text{sim}(\{(x, M_i(x)) \mid x \in \mathcal{D}_{probe}\}, c) \\ = \log p(c \mid E_i) - \log p(c) \end{aligned} \quad (5b)$$

where $p(c)$ is the probability that a human would use the description c for any neuron, and $p(c|E)$ is a distribution over image captions. In MILAN, these two terms are learned and approximated by pre-trained neural networks. Particularly, $p(c)$ is approximated by a two-layer LSTM network trained on the text of MILANNOTATION, and $p(c|E)$ is approximated by a modified Show-Attend-Tell image description model trained on MILANNOTATIONS dataset. Implementation details can be found in the Appendix of MILAN [7].

3.2. Formulation

As shown in the unified pipeline in Section 3.1, Neuron Explanation methods require probing datasets \mathcal{D}_{probe} to compute the activation map and threshold for each neuron in Step 1 and the similarity measure in Step 3. The probing dataset is domain-specific and generally collected by end users working with DNNs. This makes it vulnerable to noises and adversaries who can corrupt the probing dataset with the intent to manipulate concepts generated by NEMs. The data poisoning can easily be carried out in settings involving an exchange of the probing dataset over networks or by an adversary with backdoor access to the probing dataset. Until this work, it is not known if there indeed exists corruption that could fool NEMs. Following this discussion, we focus on analyzing the robustness of NEMs against random noise as well as crafted perturbations, where we design a novel data corruption algorithm on the probing dataset that can manipulate NEMs explanations with a high success rate.

(I) Corruption by Random noise. To evaluate the effect of random noise to the images in the probing dataset, we define $\mathcal{D}'_{probe} = \{x + \mathcal{N}(0, \sigma^2), \forall x \in \mathcal{D}_{probe}\}$ and run NEM with the corrupted probing dataset \mathcal{D}'_{probe} . This simple corruption method can manipulate concepts of upto 28% neuron with a low std of 0.02 as shown in Fig 2.

(II) Corruptions by designed perturbations. Given a neuron i , our goal is to compute the minimum corruption such that the corresponding concept c_i^* is changed:

$$\begin{aligned} \min_{\delta_j} \quad & \max_j \|\delta_j\|_\infty \\ \text{s.t.} \quad & c_i^*(\{x_j\}) \neq c_i^*(\{x_j + \delta_j\}) \\ & x_j + \delta_j \in [0, 1]^l, \forall x_j \in \mathcal{D}_{probe}. \end{aligned} \quad (6)$$

This optimization problem is a general non-convex function as deep neural network f is generally non-convex, and

so are its activation maps. In order to optimize it while considering the constraint with c_i^* , we have devised a differentiable objective function that is more amenable. To start with, recall that Eq 3 measures the similarity between a neuron’s activation map and the concepts, and assigns the concept with the highest similarity score to the neuron. We can thus manipulate the neuron’s concept by designing corruptions for each image in the probing dataset that can change the neuron’s activation map. Specifically, we define $act_{i,c}^{avg}(x)$ as the average of activation values of pixels that are associated with concept c and image $x \in \mathcal{D}_{probe}$ in the receptive field of neuron i . Formally, we define

$$act_{i,c}^{avg}(x) = \left(\frac{\sum_k \mathcal{B}(f_i(x))_k \times L_c(x)_k}{\sum_k L_c(x)_k} \right) \quad (7)$$

where k represents a location in the 2-D map $\mathcal{B}(f_i(x))$ and $L_c(x)$. We overload the definition of binary segmentation mask for an image x and concept c , $L_c(x)$, defined in the context of Network Dissection to all the NEMs. Simply, $L_c(x)$ is 1 at pixels that contain the concept c in image x and 0 elsewhere. $L_c(x)$ can be obtained in two ways:

- $L_c(x)$ from ground truth segmentations: This is the simplest of two scenarios where per-pixel segmentation data can be directly obtained. This is the typical setting for Network Dissection, where the ground truth segmentation is available to the concept assignment function sim either from the Broden dataset or a pre-trained segmentation model.
- $L_c(x)$ for general NEMs: Ground truth segmentation data might not always be available, for example, when segmentation data is not required for computing sim as in MILAN, or the concept assignment function sim including per-pixel segmentation data is not accessible to users. In this scenario, we can obtain a good approximation for $L_c(x)$ from the standard (non-corrupted) runs. Specifically, consider $\{x, M(x)\}$ pair that resulted in the assignment of label c_i^* for a neuron i . We define $L_{c_i^*}(x)$ as

$$L_{c_i^*}(x) = M(x) \quad (8)$$

This technique of defining $L_{c_i^*}(x)$ can be extended to all the NEMs if the ground truth segmentation data is unavailable. However, this reduces the scope of concepts that are available for manipulation of neurons in f , i.e., $\{c_i^* \mid \forall i \in f\}$.

Using $act_{i,c}^{avg}(x)$, we define our objective function for neuron i with concept c_i^* , probing image $x_j \in \mathcal{D}_{probe}$ and the target (manipulated) concept t as

$$\begin{aligned} \min_{\delta_j} \quad & act_{i,c_i^*}^{avg}(x_j + \delta_j) - act_{i,t}^{avg}(x_j + \delta_j) \\ \text{s.t.} \quad & \|\delta_j\|_\infty \leq \epsilon \\ & x_j + \delta_j \in [0, 1]^l \end{aligned} \quad (9)$$

Layer		Noise				
		0.01	0.02	0.03	0.04	0.05
conv1_2	mean	0.936	0.892	0.884	0.870	0.843
	std	0.172	0.208	0.222	0.196	0.203
conv2_2	mean	0.918	0.862	0.856	0.829	0.848
	std	0.169	0.202	0.212	0.214	0.206
conv3_3	mean	0.922	0.886	0.845	0.813	0.813
	std	0.181	0.214	0.226	0.235	0.238
conv4_3	mean	0.896	0.834	0.778	0.765	0.735
	std	0.207	0.238	0.259	0.259	0.263
conv5_3	mean	0.886	0.832	0.787	0.767	0.742
	std	0.223	0.251	0.269	0.269	0.279

Table 1: Average F1-BERT score between clean and manipulated descriptions after random noise data corruption. F1-BERT score between similar strings has its maximum value 1.0, and lower F1-BERT score indicates higher dissimilarity and stronger manipulation.

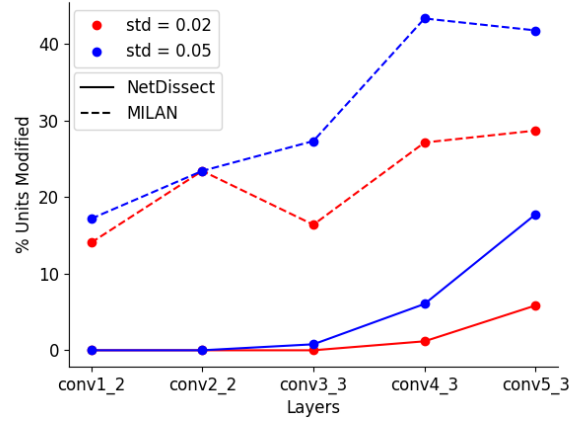
Therefore, instead of optimization Eq 6 directly, we use the designed surrogate function and solve Eq 7. The intuition is to decrease the activations of pixels that contain the source concept and increase the activation of pixels that contain the target concept for neuron i . Consequently, we can solve for the perturbation δ_j to poison each probing image x_j nicely by solving Eq 9 with projected gradient descent [6, 9].

4. Experiment

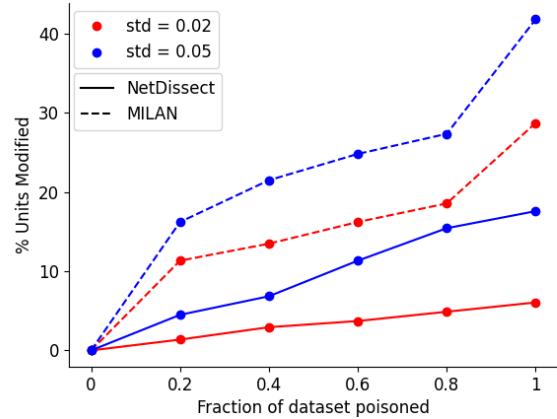
In this section, we perform the robustness analysis of Neuron Explanation Methods (NEMs) and show that they are not robust against various corruptions of probing dataset, from small random noises to well-designed perturbations which are imperceptible to human eyes. We give an overview of our experimental setup in Section 4.1, examine the effect of Gaussian random noise on neuron explanations in Section 4.2, present qualitative and quantitative analysis of our proposed corruption algorithm in Section 4.3, and show that our designed data corruption algorithm manipulates neuron explanations with a significantly higher success rate compared to random noise in Section 4.4. Note that we use the term *data poisoning* and *data corruption* in this section to refer to the corruption of probing dataset \mathcal{D}_{probe} .

4.1. Overview

Baselines. As described earlier, there are mainly two approaches of NEMs (Network dissection and MILAN) and we have unified them into the same pipeline in section 3.1. Therefore, in the following, we will show the result



(a)



(b)

Figure 2: Effect of Gaussian random noise on neuron explanations for VGG16-Places365. The red and blue lines visualize results for std = 0.02 and std = 0.05, respectively. Fig (a) visualizes the percentage of manipulated neurons explanation (y-axis) in layers of VGG16-Places365 (x-axis) for noise std of 0.02 and 0.05. Fig (b) visualizes the percentage of manipulated neurons explanation (y-axis) with increasing fraction of poisoned images (x-axis) for noise std of 0.02 and 0.05 in conv5_3.

of both standard (non-corrupted) and corrupted neuron explanations for both methods as a comprehensive study.

Setup. Our experiments run on a server with 16 CPU cores, 16 GB RAM, and 1 Nvidia 2080Ti GPU. Our codebase builds on the open source implementation of **Network Dissection** [2] and **MILAN** [7] released from their papers. We use pretrained **VGG16** on Places365[19] and pretrained Resnet50 on Imagenet, referred to as **VGG16-**



Figure 3: Targeted data corruption on Network Dissection (left) and MILAN (right) in VGG16-Places365 conv5.3 layer to change concept of unit 191 from *Buses* to *Beds*. First row in subfigure visualizes top activating images from \mathcal{D}_{probe} for neuron 191. Second row in each subfigure highlights pixels with value greater than activation threshold T_{191} .

Places365 and *Resnet50-Imagenet* respectively. The choice of these models follows the setting in the Network Dissection and MILAN paper for fair comparison. We modify projected gradient descent implementation available in Torchattacks[8] library for optimizing our objective function in Eq 9.

Probing dataset. NEMs require probing dataset to compute the activation threshold for neurons and generate an activation map, which is then used by the similarity function to assign concepts. In general, concept information does not need to be present in the probing dataset. For example, PNAS version of Network Dissection [3] uses validation dataset for probing the network and obtains per-pixel concept data from a pre-trained segmentation model. Similarly, MILAN uses held-out validation dataset of the trained model for probing DNNs, and uses a pre-trained model that generates descriptions for top activating maps. These are in contrast to the first version of Network Dissection [2] that use Broden dataset for probing DNNs and contains concept data. More precisely, Broden dataset contains pixel-level segmentation of concepts from six categories: (i) object, (ii) scenes, (iii) object parts, (iv) texture, (v) materials, and (vi) color in a variety of contexts. In this section, we will run our experiments on two NEMs, Network Dissection using Broden dataset and MILAN to cover both the ends of this spectrum.

Evaluation metric. The concept generated by Network Dissection is a term (e.g., “bus”) whereas MILAN generates

a natural language description for each neuron unit (e.g., “colorful dots and lines”). For Network Dissection, we define a neuron-unit as *manipulated* if the concept assigned by Network Dissection is changed after probing dataset is corrupted. For MILAN, we use F1-BERT score [5] to quantify the change in description between the clean and manipulated descriptions. Further, to compare MILAN and Network Dissection, we define a neuron unit as “manipulated” for MILAN if the F1-BERT score between clean description and manipulated description of a neuron ≤ 0.642 . Our estimate of this threshold is obtained by randomly selecting 500 clean and manipulated description pairs, manually evaluating whether the two descriptions have the same semantic meaning, and choosing a F1-BERT score cutoff where the number of false positives is equal to the number of false negatives. We have shown few examples in Appendix Fig A.4. We also report the F1-BERT scores alongside the manipulation success rate in the experiments to provide a complete picture.

4.2. Random noise corruption of probing dataset

In this section, we ask the following question: *How does the concept of a neuron unit change when probing dataset is corrupted with random noise?*

Fig 2a visualizes the percentage of neurons manipulated with Gaussian random noise in VGG16-Places365 for Network Dissection and MILAN. We observe that the higher layers of the network are more susceptible to being manipulated by Gaussian random noise. A noise with a low standard deviation of 0.05 can manipulate more than 17% neu-

VGG16-Places365							Resnet50-Imagenet						
Layer	Untargeted			Targeted			Layer	Untargeted			Targeted		
	$\epsilon = \frac{2}{255}$	$\frac{4}{255}$	$\frac{6}{255}$	$\frac{2}{255}$	$\frac{4}{255}$	$\frac{6}{255}$		$\epsilon = \frac{2}{255}$	$\frac{4}{255}$	$\frac{6}{255}$	$\frac{2}{255}$	$\frac{4}{255}$	$\frac{6}{255}$
conv1_2	47.73	52.27	54.55	2.27	4.55	4.55	layer1	16.25	76.25	87.5	9.68	20.97	32.26
conv3_3	34.88	58.14	67.44	4.65	13.95	25.58	layer2	29.17	85.42	91.67	1.90	20.00	42.86
conv4_3	57.45	95.74	100.0	0.0	29.79	46.81	layer3	75.34	98.63	98.63	3.90	27.63	46.05
conv5_3	70.59	96.08	96.08	7.84	49.02	56.86	layer4	79.35	98.63	98.63	2.70	22.97	32.43

Table 2: Percentage of units manipulated (higher means our corruption technique has stronger effects) in Network Dissection for VGG16-Places365(left) and Resnet50-Imagenet(right). More than 80% neurons can be manipulated with untargeted data corruption of less than 10% images in higher layers.

Layer	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$	$\epsilon = \frac{6}{255}$
layer1	0.73 ± 0.29	0.63 ± 0.30	0.55 ± 0.32
layer2	0.67 ± 0.22	0.56 ± 0.16	0.54 ± 0.20
layer3	0.46 ± 0.22	0.50 ± 0.25	0.45 ± 0.21
layer4	0.41 ± 0.29	0.42 ± 0.29	0.44 ± 0.31

Table 3: Average F1-BERT score (lower means our corruption technique has stronger effects) and standard deviation for untargeted data corruption on MILAN for **Resnet50-Imagenet**. F1-BERT score between similar strings has its maximum value 1.0, and lower F1-BERT score indicates higher dissimilarity and more successful untargeted data corruption.

rons for Network Dissection and over 40% of neuron descriptions for MILAN in the conv5_3 layer. Table 1 shows the mean and standard deviation of the F1-BERT score between the clean and manipulated descriptions of MILAN in the layers of VGG16. We observe a decrease in the F1-BERT score with increasing noise standard deviation in the higher layers suggesting that neurons are more prone to being manipulated in the higher layers.

Fig 2b visualize the percentage of neurons manipulated with the gradual addition of random noise for standard deviation 0.02 and 0.05. We observe that the number of neurons manipulated increases with an increasing standard deviation of random noise. A standard deviation of 0.05 with 60% data poisoning leads to manipulation of around 10% neurons for Network Dissection and 25% neurons for MILAN. This result is significant since noise addition is low cost and could happen naturally through noisy transmission, or probing dataset can be poisoned during exchange over the network.

4.3. Designed corruption of probing dataset

In this section, we try to answer the following question: *Can our designed corruption of the probing dataset manipulate the neuron explanations?* We visualize the results of our data corruption on Network Dissection and MILAN in Section 4.3.1, perform large-scale robustness analysis for NEMs with corrupted probing dataset in Sec 4.3.2, understand the susceptibility of categories in the Broden dataset to probing dataset corruption in section 4.3.3, and study the effect of probing dataset corruption on the model’s accuracy in Sec 4.3.4.

4.3.1 Qualitative analysis

Fig 3 (left) visualizes the data poisoning on Network Dissection to manipulate the concept of Unit 191 in layer conv5_3 from *Bus* to *Bed*. We poison a total of 2532 images out of 63296 images in the Broden dataset, which account for less than 5% of the total images. We observe that the concept gradually changes from *Bus* to *Bed*, first reducing the IOU of source concept *Bus* to 0.10 for $\epsilon = 2/255$, and then changing concept to *Bed* with a high IOU of 0.17 for $\epsilon = 3/255$.

On the other hand, Fig 3 (right) visualizes the data poisoning on MILAN to manipulate the description of Unit 191 in layer conv5_3 from *Buses* to *Beds*. We poison 800 out of 36500 images in the Places365 validation dataset, accounting for less than 3% of the total images. Similar to Network Dissection, we observe that the concept gradually changes to *Beds* for $\epsilon = 3/255$. Further, $\epsilon = 2/255$ reduces the activated area in the mask for *Bus*, suggesting that our objective function is indeed reducing the activation of the source concept. Table 4 shows the clean and manipulated descriptions of MILAN on a few sampled neurons. Our objective function is able to manipulate descriptions to very different concepts, for example, *a table* to *body parts*.

Layer-Unit	Clean description	Manipulated description	Score
VGG16-Places365			
conv3_3-42	“top of a building”	“red and white colored objects”	0.38
conv4_3-29	“a table”	“body parts”	0.45
conv5_3-46	“plants”	“green grass”	0.49
Resnet50-Imagenet			
layer2-6	“edges of objects”	“the color red”	0.45
layer3-29	“birds”	“human faces”	0.27
layer4-36	“items with straight features”	“rounded edges in pictures”	0.58

Table 4: Sampled F1-BERT scores with untargeted data poisoning of MILAN for VGG16-Places365 and Resnet50-Imagenet. It shows the neuron explanations with score below of our selected threshold of 0.642.

4.3.2 Large scale robustness analysis

Here, we perform a large-scale quantitative analysis of the robustness by performing targeted and untargeted corruption of probing dataset on the layers of VGG16-Places365 and Resnet50-Imagenet.

For Network Dissection, we consider an untargeted data corruption successful if the unit has been manipulated following our earlier definition, and consider a targeted data corruption successful if the manipulated concept matches the source concept. We poison less than 10% of \mathcal{D}_{probe} since the number of probing images for a concept is less than 5% of dataset size for most concepts. Table 2 shows the results of untargeted and targeted data corruption on all layers of VGG16-Places365 and Resnet50-Imagenet for PGD $\epsilon = 2/255, 4/255, 6/255$. Our objective function can successfully change the concepts of more than 80% neurons in the higher layers with PGD $\epsilon = 4/255$ in the untargeted setting. We can change the concept of a neuron to a desired target concept with a high success rate of more than 50% in higher layers with $\epsilon = 6/255$. Further, we are also able to manipulate more than 60% neuron concepts in the higher layers with $\epsilon = 6/255$ by obtaining $L_c(x)$ from the baseline runs as discussed in Appendix Sec A.2.1.

For MILAN, Table 3 and Appendix Table A.7 (for type U2) shows the mean and standard deviation of F1-BERT score between clean and manipulated descriptions for Resnet50-Imagenet and VGG16-Places365. We run our data corruption algorithm using projected gradient descent for different values of $\epsilon = \frac{2}{255}, \frac{4}{255}, \frac{6}{255}$. We observe a consistent decrease in the F1-BERT score with the mean score being less than 0.5 for a small perturbation of $\epsilon = \frac{4}{255}$ in the higher layers of the network. Further, we poison 800 images out of 36500 images in the \mathcal{D}_{probe} , which constitutes less than 3% of total images.

4.3.3 Effect of data corruption on Broden categories

Broden dataset used in Network Dissection consists of images with pixel-level segmentation of concepts from six cat-

egories, which allows us to study if certain categories are more vulnerable to data corruption. We follow the experimental setting discussed in Sec 4.3.2 for the untargeted data corruption on Network Dissection, and Fig 4 shows the percentage of concepts manipulated in different categories with untargeted corruption of probing dataset. Higher level concepts *objects*, *part*, *scene*, and *material* can be manipulated with a success rate of 100% by our objective function in the untargeted setting. In contrast, lower-level concepts *texture* and *color* are more robust to corruption.

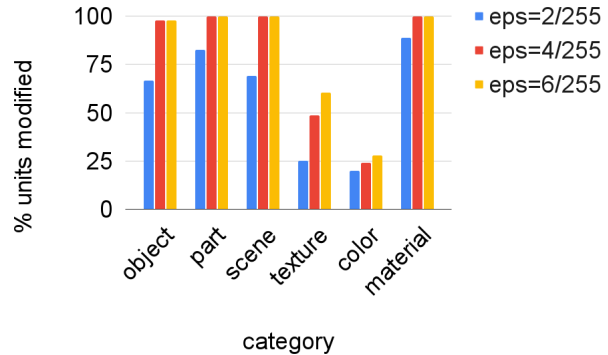


Figure 4: Percentage of concepts manipulated by category in Broden dataset with untargeted data corruption on VGG16-Places365. Higher level concepts *object*, *part*, *scene*, and *material* are more susceptible to data corruption than lower level concepts *scene* or *texture*

4.3.4 Effect of data corruption on model accuracy

MILAN uses the validation dataset as the probing dataset, which raises the question of using the accuracy on the validation dataset to detect data corruptions. We plot the validation accuracy of VGG16-Places365 with an increasing number of corrupted images for Unit 191-Conv 5.3 to manipulate the explanation from the concept *bus* to the concept *bed*. We consider a data corruption successful if the manip-

ulated MILAN description was semantically similar to *bed*, i.e., *Beds* or *Furniture*. The results are visualized in Fig 5. We see that our data corruption algorithm can succeed with minimal effect on the validation accuracy of the model, indicating that the data corruption would be harder to detect, and even the largest perturbation only reduce model accuracy by less than 2%.

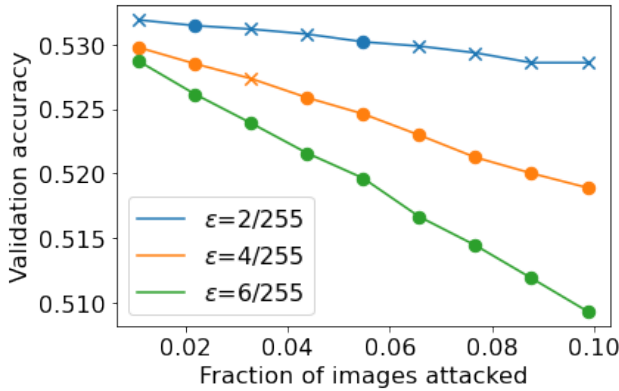


Figure 5: The effect of changing the number of poisoned images (x-axis) on the probing dataset accuracy (y-axis) for Unit 191 conv5_3 VGG16-Places365 in MILAN. Successfully manipulated concepts are marked as circles, while unsuccessful manipulations are crossed. Our data corruption algorithm has a minimal effect on classification accuracy, indicating that our corruption can be harder to detect.

4.4. Comparing random and designed corruption of probing dataset

In this section, we compare random and designed data corruption by keeping noise level and corruption magnitude ϵ set to $\frac{6}{255}$. Bernoulli random noise achieves higher neuron manipulation success rate compared to Gaussian random noise and uniform random noise as shown in Appendix Sec A.1.2. Following this result, we use Bernoulli noise for random data corruption and Eq 9 for designed data corruption. The results are shown in Fig 6. We observe that our data corruption algorithms achieves a significantly higher manipulation success rate than addition of random noise to the images in the probing dataset.

5. Conclusion

In this work, we made a significant effort to show that Neuron Explanation methods are not robust to corruptions including random noise and well-designed perturbations added to the probing dataset. Our experiments indicate that our objective function can manipulate assigned neuron concepts in both targeted and untargeted settings. Further, we show that it is easier to manipulate units in the top layer of a network that have higher-level concepts assigned to them.

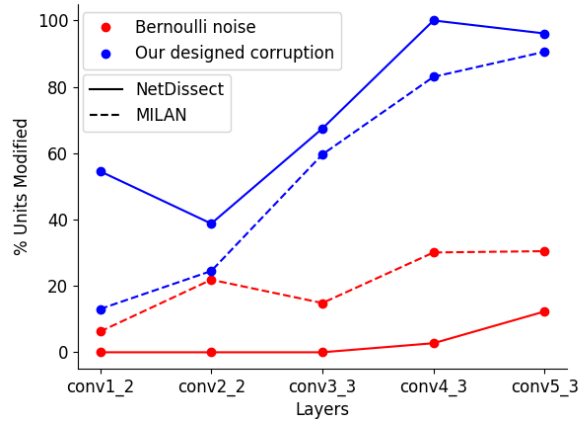


Figure 6: Comparing Bernoulli random noise and our data corruption algorithm in VGG16-Places365 with noise level and corruption magnitude ϵ set to $\frac{6}{255}$ for Network Dissection and MILAN. Higher percentage of units manipulated means that corruption technique has stronger effects. Our objective function achieves significantly higher success rate compared to Bernoulli random noise.

Acknowledgement

The authors would like to thank the anonymous reviewers for their valuable feedback. The authors also thank the computing resource supported in part by National Science Foundation awards CNS-1730158, ACI-1540112, ACI-1541349, OAC-1826967, OAC-2112167, CNS-2100237, CNS-2120019, the University of California Office of the President, and the University of California San Diego’s California Institute for Telecommunications and Information Technology/Qualcomm Institute with special thanks to CENIC for the 100Gbps networks. T. Oikarinen and T.-W. Weng are supported by National Science Foundation under grant no. 2107189.

References

- [1] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018. [2](#)
- [2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017. [1](#), [2](#), [3](#), [5](#), [6](#)
- [3] David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, 2020. [1](#), [2](#), [3](#), [6](#)
- [4] Stephen Casper, Max Nadeau, Dylan Hadfield-Menell, and Gabriel Kreiman. Robust feature-level adversaries are interpretability tools. *Advances in Neural Information Processing Systems*, 35:33093–33106, 2022. [3](#)
- [5] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3681–3688, 2019. [2](#), [6](#)
- [6] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. [5](#)
- [7] Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*, 2021. [1](#), [2](#), [3](#), [4](#), [5](#)
- [8] Hoki Kim. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*, 2020. [6](#)
- [9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. [5](#)
- [10] Jesse Mu and Jacob Andreas. Compositional explanations of neurons. *Advances in Neural Information Processing Systems*, 33:17153–17163, 2020. [1](#), [2](#)
- [11] Parth Natekar, Avinash Kori, and Ganapathy Krishnamurthi. Demystifying brain tumor segmentation networks: interpretability and uncertainty analysis. *Frontiers in computational neuroscience*, 14:6, 2020. [1](#), [3](#)
- [12] Zohaib Salahuddin, Henry C Woodruff, Avishek Chatterjee, and Philippe Lambin. Transparency of deep neural networks for medical image analysis: A review of interpretability methods. *Computers in biology and medicine*, 140:105111, 2022. [1](#), [3](#)
- [13] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, Oct 2019. [2](#)
- [14] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017. [2](#)
- [15] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. [2](#)
- [16] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017. [2](#)
- [17] Haizhong Zheng, Earlence Fernandes, and Atul Prakash. Analyzing the interpretability robustness of self-explaining models. *arXiv preprint arXiv:1905.12429*, 2019. [2](#)
- [18] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. [2](#)
- [19] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. [5](#)