Bounded Adversarial Attack on Deep Content Features

Qiuling Xu, Guanhong Tao, Xiangyu Zhang
Purdue University
{xu1230, taog, xyzhang}@purdue.edu

Abstract

We propose a novel adversarial attack targeting content features in some deep layer, that is, individual neurons in the layer. A naive method that enforces a fixed value/percentage bound for neuron activation values can hardly work and generates very noisy samples. The reason is that the level of perceptual 感知的 *variation entailed* 导致 *by a fixed value bound is non-uniform across neurons and even for the same neuron. We hence propose a novel distribution quantile* 分数位 *bound for activation values and a polynomial* 多项的 *barrier loss function. Given a benign* 适合的 *input, a fixed quantile bound is translated to many value bounds, one for each neuron, based on the distributions of the neuron's activations and the current activation value on the given input. These individualized bounds enable fine-grained* 粒度 *regulation, allowing content feature mutations with bounded perceptional variations. Our evaluation on ImageNet and five different model architectures demonstrates that our attack is effective. Compared to seven other latest adversarial attacks in both the pixel space and the feature space, our attack can achieve the state-of-the-art trade-off between attack success rate and imperceptibility.* 不可感知性 [1]

给定一个良性输入，一个固定的分位数边界被转换为多个值边界，每个神经元一个值边界，基于神经元的激活分布和给定输入上的当前激活值。这些个性化的边界支持细粒度的调节，允许内容特征的突变与有界的感知变化。

1. Introduction

Adversarial attack is a prominent security threat for Deep Learning (DL) applications. With a benign input, perturbation is applied to the input to derive an adversarial example, which causes the DL model to misclassify. An underlying assumption is that adversarial samples should be perceptually close to real inputs [10]. Without this assumption, adversarial samples could merely be too different from real inputs and become unseen samples, in which case misclassification is well expected. Traditionally, imperceptibility is ensured by having bounded perturbation *in the pixel space*. The bound is usually small, e.g., $[-4, 4]$ in the RGB range of $[0, 255]$, such that perturbations are imperceptible by humans.

Researchers have recently shown adversarial examples with large pixel distances (from the original inputs) can be generated. Such distances are usually way beyond the bounds that many existing defense and validation techniques aim to protect, providing a new attack vector. These techniques focus on mutating *meta-features* 元数据 of original inputs, such as colors and styles, due to the difficulty of harnessing perturbations on *content features*, such as shapes and local patterns, denoted by individual neurons. While using adversarial samples generated by these techniques can harden the model in meta-feature space, making the model robust to color and style changes, they offer limited protection when the attacker is able to mutate individual content-features/neurons in an imperceptible way. In addition, the perturbations generated by these methods are pervasive 普遍的 and hence more visible in human eyes, making them less desirable when being used in real attacks.

In this paper, we propose a new attack vector in the feature space that can perform imperceptible content feature perturbation. Such perturbations cannot be expressed by pixel bounds or meta-feature bounds (e.g., bounds on mean and standard deviation of activation values) and hence pose a new challenge to existing defense techniques. The essence of our technique is to bound the perturbations of individual neurons. Given a uniform bound at the perception 感知层 level (e.g., allowing 10% *perceptual perturbation* for each content-feature/neuron), it is projected 投影 to various value bounds for individual neurons which have different activation value ranges and distributions. Gradient back-propagation is used to mutate 突变 input pixels, just like in traditional adversarial attack, while the mutations are constrained by the internal bounds. A naive method is to first profile the activation value ranges of individual neurons and then limit the variation of each neuron to a fixed portion of its value range. However, this method does not work because *the perception of a fixed activation value perturbation varies substantially (even for a single feature) depending on the activation value itself.* For example, a change of 1.0 when the activation value is 0.0 admits a substantially 可观的 different level of perceptual variations compared to the same change when the value is 15.0. To achieve a uniform perceptual bound, we propose to use a *distribution quantile bound*. More specifically, given a model, we identify internal values that approximate normal 正态分布

[1] Code and Samples are available on Github [37].

A/255: 255像素中有a像素改变
conf: 攻击成功的自信程度
succ: 正的表示成功, 数值越大表示模型对预测越确定

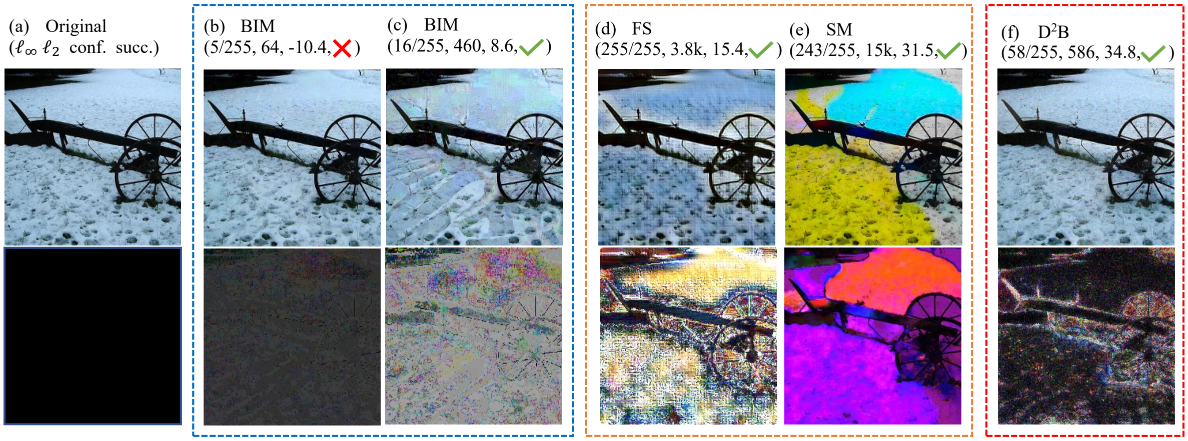


Figure 1. Adversarial examples of different attacks/attack-settings. The first column represents the original images. The second and third columns show examples from Basic Iterative Method (BIM) with a small and a large pixel distances, respectively. The following columns are samples from Feature Space Attack(FS), Semantic Attack (SM), and our **D**eep **D**istribution **B**ounded Attack (D^{2}B). For each set of images, the first row presents the adversarial examples. The second row shows the perturbations applied to the original image. We enlarge the perturbations of BIM by 10 times and the others' by 5 times for better illustration. On the bottom of each column, there is a quadruple representing the ℓ_∞, ℓ_2 distances, the attack confidence of each example and the attack success. A positive value implies a successful attack and a large value indicates the model is very confident about the (misclassification) result.

distributions. We call them a *throttle plane* 节流面 (see Section 3). After identifying the throttle plane, we collect the activation distribution over the training set for each neuron on the plane. Given a benign sample, its activation on each neuron (on the plane) is acquired. The bound for its value change is *dynamically* computed based on a fixed quantile of the distribution (e.g., 10%) and the activation itself. This is called the *quantile bound*. Our technique is hence called D^{2}B (**D**eep **D**istribution **B**ounded Attack). We then enforce the value bounds using a *polynomial internal barrier loss* (Section 3.2). Note that such value bounds are completely dynamic while they denote the same perceptual bound. Our results show that the method can mutate content features in a way that is less human perceptible. According to our human study in Section 4, our technique can achieve 95% attack success rate, and yet humans cannot easily distinguish the adversarial examples from the benign ones within a short time. In comparison to traditional pixel space attacks that generate noise-like perturbations, our attack generates perturbations piggy-backing on existing semantically meaningful features, making them difficult to detect. 附加在已有的语义特征上

Example. The second and third columns of Figure 1 (in the blue dashed box) show some samples with a small pixel bound (i.e., $\ell_\infty = 5/255$, meaning the maximum pixel value change is 5 out of 255) and a larger bound (i.e., $\ell_\infty = 16/255$) for the BIM attack[2]. Observe that with the larger bound, the adversarial perturbation is detectable by human eyes, suggesting using a large bound in the pixel space is undesirable. The third and fourth columns (in the orange dashed box) show some examples of attacks in the feature space, namely feature space attack [38] and semantic attack [2]. Details of these attacks can be found in Section 2. Observe that they have much larger ℓ_∞ and ℓ_2 distances (from the original inputs in the first column) than the examples generated by the BIM attacks. While their perturbations are less perceptible than the examples generated by pixel space attacks given a similar pixel distance, the perturbations are quite noticeable in human eyes due to their pervasive nature. This is confirmed by our human study (Section 4), in which we show that with an 80% attack success rate, humans can easily recognize the adversarial examples.

The last column of Figure 1 shows a typical example generated by our technique and its pixel-level contrast with the original image. Observe that the differences largely piggyback on the content features of the original example (by having similar shapes and local patterns as the original input), making them more human imperceptible. More importantly, these perturbations are quite different from those by existing pixel space and feature space attacks, suggesting a new threat.

We conduct experiments on ImageNet and five models, including both naturally and adversarially trained models. Our results show that existing adversarial training has little effect on our attack. Although comparing different attacks may be comparing apple with orange, we study the correlations between attack success rate and human imperceptibility for seven related attacks perturbing either the pixel or internal space, to understand the high level positioning of our attack. Our results show that our attack produces adversarial exam-

[2] We use BIM instead of other pixel space attacks such as PGD because we found that (compared to BIM) the random initialization of PGD degrades imperceptibility at a non-trivial scale, in exchange for just a slightly higher success rate. Hence, we consider BIM a more compelling baseline when considering the balance between attack success rate and imperceptibility.

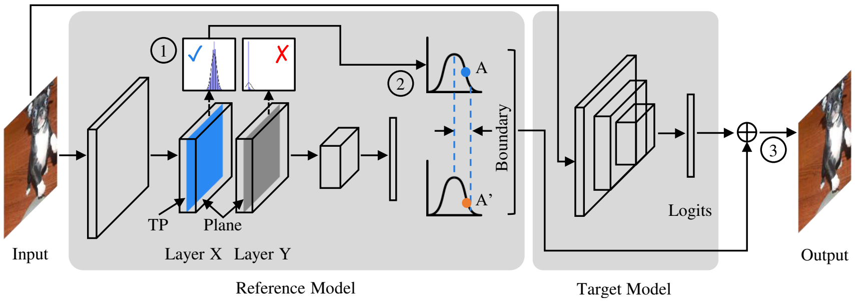


Figure 2. Workflow of our attack. It consists of three steps: ① throttle plane (TP) selection, ② internal distribution boundary constraint, and ③ adversarial sample generation with combined losses.

ples that are less human perceptible when achieving the same level of attack confidence/success-rate. Further evaluation against three different detection techniques demonstrates that our attack has better/comparable persistence while having better imperceptibility due to its new attack vector.

2. Related Work

White-box attacks, such as PGD [20], C&W [4], BIM [18], and FGSM [10], assume access to model internals and leverage gradient information in sample generation. Black-box attacks, such as ZOO [6], assume no access to model internals and directly 改变 mutate inputs based on classification outputs. Our work falls into the *white-box* attack category in the image classification domain.

Several works explored adversarial samples which are unrestricted in the pixel norm. Specifically, semantic attack [2] manipulates a benign image's color and texture. Xu et al. [38] leveraged style transfer [14] to mutate (implicit) styles of benign inputs. In particular, it perturbs the distribution (e.g., mean and variation) of feature maps. Xiao et al. [34] proposed to relocate pixels through flow optimization and constructed spatial adversarial samples. Song et al. [28] 使用 leveraged GAN to generate unbounded adversarial examples. Some works propose to uniformly change the colors and lightning conditions for constructing adversarial examples [12, 19]. To improve imperceptibility of adversarial samples, Croce et al. [7] proposed sparse attack. It defines a salience score for each pixel and avoids large perturbation to salient pixels. HotCold attack [23] constrains the perturbation of adversarial samples by utilizing SSIM [41], a score for measuring structure similarity. However, these scores can be unreliable for bounding perturbations [27]. Different from these approaches, our D^{2}B manipulates local content features, providing a novel attack vector. Analogous to pixel space attacks, D^{2}B allows changing individual features (just like changing individual pixels). To achieve the goal, a novel bounding technique is proposed. Our experiments show that our attack has high success rate while preserving imperceptibility.

Some existing works 利用 utilized internal representations to 促进, 加快 facilitate attack [17, 25]. Sabour et al. [25] tried to minimize the ℓ_2 distance of internal activations between normal and adversarial inputs. Kumari et al. [17] optimized inputs to have ℓ_∞-bounded internal perturbations in order to improve adversarial training. In Appendix F and L, we show that the uniform bound and two-step optimization used in these works are not effective for our purpose. Researchers also tried to perturb the 嵌入 embedding of GAN to generate adversarial samples [29, 30]. However, it is still an open problem to obtain 高保真 high-fidelity and content-preserving GAN based adversarial examples [9].

Some defense techniques utilize internal representations. For example, existing works [5, 21, 22] use distance of internal representations to detect adversarial samples. Defense-GAN [26] leverages the manifold of normal samples. It is however unclear whether these techniques can effectively defend against adversarial samples whose perturbations are bounded in deep layers [1, 31].

3. Attack Design

In our first attempt, we directly extended an existing pixel-space attack BIM to enforce a fixed internal activation value bound. However, we found that it does not work well. The reason is that a fixed value or percentage range makes sense in the pixel space as it directly reflects a fixed level of human perception variation. In contrast, a fixed internal range may imply various levels of pixel changes and hence various human perception levels. More details can be found in Appendix A. We hence propose a different design.

Our Design. Figure 2 describes the workflow of our attack. Given a reference model (from which a throttle plane is identified and used to constrain feature variations), a target model (for which adversarial samples are generated), and some training samples, we first perform throttle plane (TP) selection (step ①). Specifically, model execution can be considered as a directed 无环的 acyclic computational graph (DAG) from the input to the output. We run the reference model over some samples and collect the output distributions of all operations in its computational graph (e.g., the output of matrix multiplication). *The output values lie in a* 割集 *cut-set [33] of this computational graph are defined as a plane* (e.g., the blue and gray planes in Figure 2). 凭直觉的 Intuitively, a plane is a "slice" of a layer. *It contains values from all parallel operations in a particular layer across all neurons and channels* (and hence also a cut-set of the graph). A layer can be considered as a stack of multiple planes. For instance, assume a layer consists of three operations: kernel multiplication, addition with bias, and ReLU activation function. The values collected at the end of each operation constitute a plane. The plane could lie in the border between layers or even in between operations within a layer.

plane的构成

给定一个边界，将原值A变换为边界内的另一个值A'

用一个loss来计算调整限定边界

A plane whose value distribution approximates a normal distribution is a possible throttle plane (TP) to harness the adversarial perturbation (e.g., the blue plane in Figure 2). We use the normality criterion to select information-rich distributions and filter ill-defined distributions. With a (or multiple) selected throttle plane(s), we further inspect the possible distribution boundary for each neuron at step ②. That is, the perturbed value A' should be bounded within some distribution quantile range of the original value A. Finally, we model the constraint of distribution boundary by an internal barrier loss function (on the reference model), which is combined with the cross-entropy prediction loss (on the target model). During attack (step ③), a normal input is fed to both models and updated with respect to the combined attack loss, which produces a successful and imperceptible adversarial example. While the reference model and the target model could be the same model, empirically, we find that using a stand-alone reference model allows the best performance as it enables our attack even when a good throttle plane cannot be found in the target model.

独立的

3.1. Distribution Based Bounding and Throttle Plane Selection

总体的

The overarching design of our attack is to harness perturbation at the selected *throttle plane(s)* such that only small variations of abstract features are allowed. Note that the corresponding pixel space perturbations could be substantial as long as the inner value changes are within bound.

Challenges of Having Internal Throttle Plane. Traditional adversarial sample generation techniques simply place the perturbation throttle at the input plane. This makes the design simple as the perturbation happens strictly within the throttle plane. In contrast, placing the throttle in an inner plane poses new challenges.

输入层的输入有统一的语义特征，而中间层则没有

First of all, while in the input space values have uniform semantics (e.g., denoting the RGB values of individual pixels), values of the inner distributions do not have such a property. The different values on the same inner plane often represent various abstract features whose value ranges have diverse semantics. As such, a uniform perturbation bound across all these internal values is meaningless. Second, in our design, the perturbation occurs in the input space while the throttle is placed somewhere inside the reference model. Hence, the input perturbation is not directly controlled and could be substantial. An important hypothesis is that since the perturbations can only induce bounded inner value changes at the throttle plane, they denote small semantic mutations of the abstract features. However, given a particular inner value, the semantic mutation entailed by its changes is non-uniform within its range. Consider Figure 3c, which denotes the distribution of an inner value (across the training set). Observe that a variation of 0.5 when the value is 1 implies much more substantial semantic changes (indicated by the entailed substantial quantile change) than when the value is 4, which is at the very tail of the distribution, as the model is likely insensitive to such a large value.

特点

假设

导致

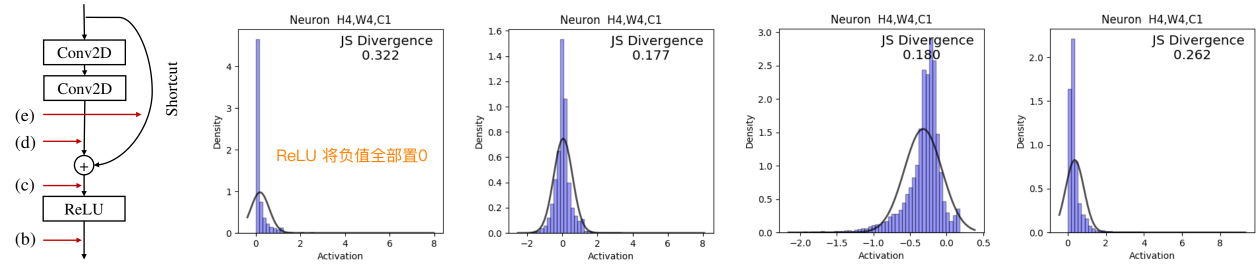
Our Method – Looking for Gaussian Planes. According to the above discussion, we cannot utilize a uniform value bound across the different inner values (on the plane); we cannot utilize the same bound even when the value varies (from one input to another). Therefore, we propose a novel idea of using a distribution based bound. Particularly, we collect the distributions for the individual values (on the plane). During perturbation, the bound for each inner value is based on its distribution quantile. As such, not only different values along the plane have different bounds, but also, the value may have different bounds when it varies from input to input. In particular, we *select the plane(s) whose values approximate Gaussian distributions* and *use a quantile bound based on the current value and its distribution*, instead of using a concrete value bound. These allow us to have precise and relatively easy control of the level of semantic mutation.

The intuition of looking for Gaussian distribution is to maximize entropy. A larger entropy in our case implies that the neuron contains more information, allowing more fine-grained semantic control. With the first and second momentums fixed, a Gaussian distribution has the largest entropy. In the following, we use a few examples to illustrate this point.

熵更大的神经元有更多数据，需要更细粒度的语义划分

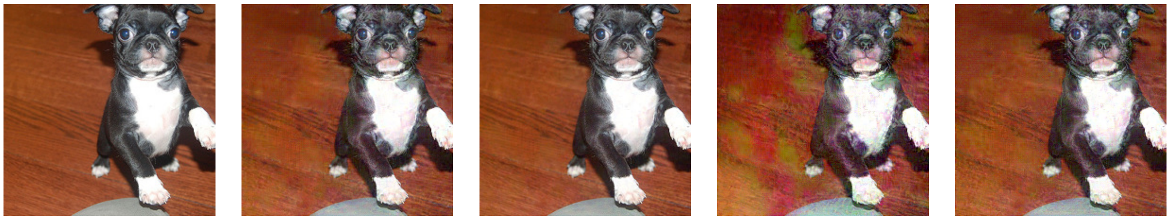
Consider a block of an adversarially trained ResNet152 (Figure 3a). If we set the throttle plane at the block boundary (i.e., right after the ReLU function), the distribution for some value on the plane (across the entire training set) is shown in Figure 3b. Observe that the density function is ill-defined at point zero (due to ReLU). It is hence not a good choice for the throttle plane. Intuitively, the reason is that substantial input perturbations would be admitted as long as their inner value remains negative before ReLU. Formally, the information loss of the negative side is reflected by the smaller entropy (4.0 after ReLu compared to 5.1 before) and results in degenerated effectiveness (see Figure 4b and 4c).

If we set the plane right before ReLU, according to Figure 3c, it approximates Gaussian distribution. The Gaussian distribution makes enforcing a quantile bound easy and hence allows generating imperceptible adversarial examples (see Figure 4c). Observe that the background has undergone much less perturbation (compared to others) as most pertur-



(a) ResNet V1 Block (b) After ReLU (c) Before ReLU (d) Main (e) Shortcut

Figure 3. Operations in the last block of group 1 of an adversarially trained ResNet152 and the corresponding typical distributions of the values after these operations. In (b)-(e), we present the estimated distance between the empirical distribution and a Gaussian through Jensen-Shannon(JS) divergence. A smaller value indicates more resemblance.



(a) Original (Entropy, Is Cut?) (b) After ReLU (4.0, Yes) (c) Before ReLU (**5.1, Yes**) (d) Main (5.8, No) (e) Main and Shortcut (4.5, Yes)

(average entropy, is TP a cut of the data flow?)

Figure 4. Adversarial images when the throttle plane is at different positions in Figure 3, under the same perturbation bound. In the bottom, we report the average entropy of underlying throttle plane and whether it forms a cut of the data flow.

bation is on the content features of the dog and hence not that visible. We also study the distributions of the values after the main output and along the shortcut (e.g., Figure 3d and 3e, respectively). Observe that although (d) resembles Gaussian distribution and has the largest entropy, placing the throttle there produces unnatural samples (see Figure 4d). The main reason is that the operation alone does not form a cut-set of the computational graph. Intuitively, it leaves a large part (i.e., the values along the shortcut) unconstrained. Figure 16 in Appendix R shows the distributions for a set of randomly selected values on the same plane. Observe that they approximately follow normal distributions. Also, observe that their distribution parameters are quite different, supporting our design of using different bounds for various values on a plane.

During sample generation, given a benign input, the selected throttle plane's inner values are collected. The bound of the value is then determined by its quantile of the value (on its density function). How to enforce such quantile bounds is discussed in the following section.

3.2. Enforcing Quantile Bound with Polynomial Barrier Loss

yi是符合正态分布的一维随机变量

Let $\mathcal{D}$ be a distribution on support $\mathcal{S}$. The activation y_i of neuron i on a selected throttle plane $\mathcal{I}$ is a random variable through mapping $f_i : \mathcal{S} \to \mathbb{R}$, $\mathrm{y}_i \sim f_i(\mathcal{D})$. We denote the cumulative distribution function of y_i as $C_i(x)$, and the corresponding quantile function as $C_i^{-1}(x)$. Let the original image be $\mathrm{x}^{\mathrm{nat}}$ and the adversarial sample be $\mathrm{x}^{\mathrm{adv}}$. Correspondingly, let ${\mathrm{y}_i}^{\mathrm{adv}}$ and ${\mathrm{y}_i}^{\mathrm{nat}}$ be the respective activations for $\mathrm{x}^{\mathrm{nat}}$ and $\mathrm{x}^{\mathrm{adv}}$. Assume the allowed quantile change is less than a threshold ϵ. The corresponding value bound for $\mathrm{y}_i^{\mathrm{adv}} \in [\mathrm{low}_i, \mathrm{high}_i]$ is hence the following.

$$\begin{aligned}\mathrm{y}_i^{\mathrm{adv}} \in \Big[&C_i^{-1}\left(\max(C_i(\mathrm{x}^{\mathrm{nat}})-\epsilon, 0)\right), \\ &C_i^{-1}\left(\min(C_i(\mathrm{x}^{\mathrm{nat}})+\epsilon, 1)\right)\Big]\end{aligned} \quad (1)$$

Note that we translate the quantile bound to a value bound, over which we can define a loss function.

Polynomial Barrier Loss. *Interior point method* or *barrier method* [3] is a standard technique for constrained optimization. It is widely used in linear programming applications [32]. It utilizes a negative log function in the loss function by default. However, it was intended to be used in problems where the bound is hard, meaning the values must not exceed the bound as the loss becomes infinitely large when the value infinitely approaches the bounds. In our context, a hard bound does not work well with ReLU functions. Specifically, input changes guided by gradients may activate some previously inactive neurons, leading to the inner values to exceed their bounds, causing numerical exceptions (on the log function). Another naive design is to introduce a ReLU kind of bound, that is, the loss is 0 while the value is in bound and some large value otherwise. However, it does not apply penalty 惩罚 when the value is approaching the bound. Therefore, we devise 设计 a polynomial barrier loss as follows.

In bound: loss>0
Out of bound: loss<0

$$\mathcal{L}_i(\mathrm{y}_i{}^{\mathrm{adv}}) = k\left[\frac{\mathrm{ReLU}(\mathrm{y}_i{}^{\mathrm{adv}} - \mathrm{y}_i{}^{\mathrm{nat}})}{\mathrm{high}_i - \mathrm{y}_i{}^{\mathrm{nat}}} + \frac{\mathrm{ReLU}(\mathrm{y}_i{}^{\mathrm{nat}} - \mathrm{y}_i{}^{\mathrm{adv}})}{\mathrm{y}_i{}^{\mathrm{nat}} - \mathrm{low}_i}\right]^b \tag{2}$$

Intuitively, the loss function applies a polynomially increasing penalty when the inner value induced by adversarial perturbation approaches the bound. Please refer to Appendix J for details and the comparison with other loss choices.

Optimization Method. With the polynomial barrier loss, we use a standard gradient sign method [4] for optimization. There are other design choices. For example, in [17], a two-step optimization was proposed to facilitate adversarial example generation by leveraging internal values. However, we found that the method is not effective when a strict internal boundary is enforced. Another simple method is clipping, which clips the inner values (on a throttle plane) and prevents gradient propagation if they are beyond bounds as we mentioned in Section 3. We conduct experiments to compare the three methods. Our method can better enforce the internal bound and generate adversarial examples with one order of magnitude smaller average boundary size. Details can be found in Appendix F.

Identifying an appropriate quantile bound value is important. We address the problem by profiling the quantile changes at the throttle plane under other attacks. Specifically, we use the average observed internal value ℓ_∞ quantile change (at a throttle plane) for the adversarial examples by BIM4. Identifying the learning rate is discussed in Appendix B. Occasionally, we observe the generated adversarial examples exhibit checkerboard patterns. We hence add a feature smoothing loss during optimization. Details and an ablation study can be found in Appendix C and G.

4. Experiments

We conduct experiments on ImageNet [24] and five types of DNN models. We show that pre-trained models hardened by state-of-the-art adversarial training methods cannot defend our attack. Furthermore, although attacks in different spaces may not be comparable in general, we study the tradeoff between attack effectiveness and imperceptibility for a large set of eight attacks in both pixel and feature spaces, including D^{2}B. The goal of the study is not to say one attack is superior to another, but rather to provide an intuitive understanding of D^{2}B's positions in these generic metrics. Finally, we evaluate D^{2}B against three popular adversarial attack detection approaches.

Attacks In Study. We study D^{2}B and seven other existing attack methods: BIM [18], hot cold attack [23], sparse attack [7], feature space attack [38], semantic attack [2], latent attack [17] and spatial attack [34]. We use BIM as the representative of classic pixel space adversarial attacks such as PGD and C&W (due to the reason explained in Section 1). Feature space attack uses auto-encoder based on VGG16 and performs bounded perturbation of the mean and variance of 嵌入 internal embeddings [38]. Semantic attack manipulates input color and texture. Sparse attack applies small perturbation to 主要的 salient pixels and large perturbation to less salient pixels. Hot-cold attack uses the SSIM score [41] to constrain perturbation. Spatial attack optimizes flow of pixels to generate adversarial sample. Latent attack uses two-step optimization. We use these eight attack methods to generate adversarial examples for a naturally trained ResNet50 model [11] and an adversarially trained ResNet152 model [35] (ResNet152-adv). For BIM, sparse attack, hot cold attack, latent attack, feature space attack and our attack, we stop the attack optimization when 收敛 convergence is reached (no confidence increase). For semantic attack and spatial attack, we use a preset number of optimization steps. Note that they are unbounded and the optimization step controls the perturbation and the attack success rate. For the sparse attack, we are unable to scale it up to an untargeted attack on ResNet152-Adv.

Evaluation Metrics. We measure attack success rate versus imperceptibility for untargeted attacks on the adversarially trained model (Figures 5b and 5d). In contrast, we measure attack confidence score versus naturalness for targeted attacks on the normally trained model (Figures 5a and 5c). Note that we cannot use attack success rate for the normally trained model as it is almost 100% for all the attacks. We do not conduct untargeted attacks on the normally trained model or targeted attacks on the adversarially trained model as the former is too easy and the latter is too hard for a comparative study. The confidence score of a sample x is defined on the logits (the pre-softmax) value $L(x)$ [4]. Specifically, for a targeted attack, suppose the target label is t, the confidence score of a sample x is defined as follows.

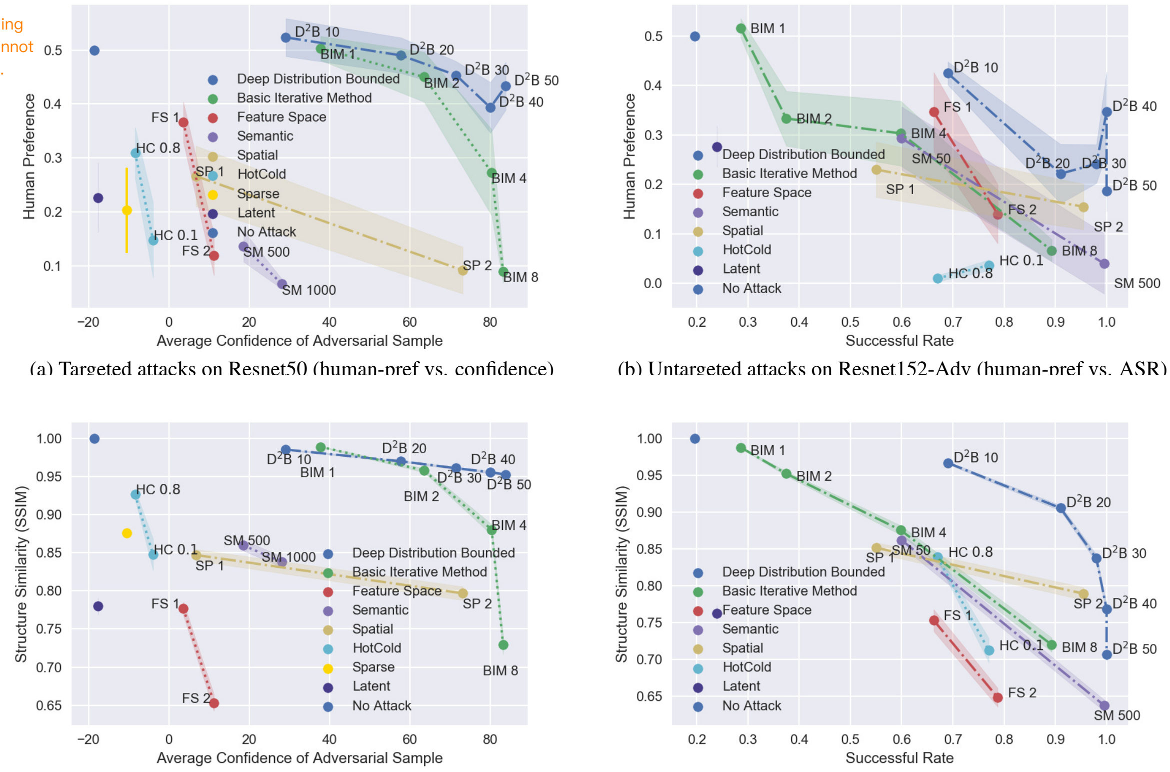
$$L_t(x) - \max_{i \neq t} L_i(x) \tag{3}$$

Intuitively, it is the logits gap between the target label and another label with the maximum logits and hence measures the success level of an attack when the attack can always induce misclassification [35].

In order to measure the imperceptibility of generated examples, we perform a human study using Amazon Turk. We employ a similar setting as that in [40]. Specifically, for each attack, users are given 100 pairs of images, each consisting of a real image and its adversarial counterpart. They are asked to choose the one that looks unreal. Each user is given 5 test-drives before the study starts. Each pair of image appears on screen for 5 seconds and is evaluated by three different users. More about the user study can be

50% meaning humans cannot distinguish.

SSIM score (with value 1 indicating an adversarial sample and its benign version are identical)



(a) Targeted attacks on Resnet50 (human-pref vs. confidence)

(b) Untargeted attacks on Resnet152-Adv (human-pref vs. ASR)

(c) Targeted attacks on Resnet50 (SSIM vs. confidence)

(d) Untargeted attacks on Resnet152-Adv (SSIM vs. ASR)

Figure 5. Quality of the generated adversarial examples. These figures present the level of naturalness (y axis) versus the level of attack success (x axis). Results in the top-right corner denote the best trade-off. Figures (a) and (c) denote targeted attacks on a normally trained Resnet50; (b) and (d) denote untargeted attacks on an adversarially trained Resnet152. For naturalness, we report SSIM score (with value 1 indicating an adversarial sample and its benign version are identical), and human preference rate collected in human studies, which denotes the rate that humans consider an adversarial example real (compared to its benign version), with 50% meaning humans cannot distinguish. To measure attack success level, we use attack success rate and confidence score. The latter is for attacks on normally trained models which always cause misclassification, rendering attack success rate an invalid metric. We regard an untargeted attack successful if the true label does not appear in the top-5 predicted labels, which is consistent with the literature [36]. The shaded area on a data point (i.e., an attack setting) represents the standard error of the human preference rate or the SSIM score.

Table 1. Pixel and quantile distances for ResNet50, with Conf. meaning attack confidence

Attack	Conf.	Pixel Dist.		ℓ_∞ Quantile Dist.		
		ℓ_2	ℓ_∞	Plane 1	Plane 2	Plane 3
BIM4	81.91	10.81	0.04	0.62	0.83	0.90
D^{2}B10	30.29	4.26	0.07	0.06	0.08	0.09
D^{2}B20	58.82	6.27	0.09	0.12	0.16	0.17
D^{2}B30	72.43	7.28	0.10	0.18	0.24	0.26
D^{2}B40	80.06	7.88	0.11	0.24	0.32	0.35
D^{2}B50	84.52	8.25	0.11	0.30	0.41	0.44

Table 2. Pixel and quantile distances for ResNet152-Adv, with Succ. meaning attack success rate

Attack	Succ.	Pixel Dist.		ℓ_∞ Quantile Dist.		
		ℓ_2	ℓ_∞	Plane 1	Plane 2	Plane 3
BIM4	0.58	14.49	0.04	0.65	0.82	0.89
D^{2}B10	0.61	6.65	0.11	0.06	0.08	0.09
D^{2}B20	0.86	15.43	0.24	0.13	0.16	0.17
D^{2}B30	0.86	16.68	0.19	0.19	0.24	0.26
D^{2}B40	0.97	21.84	0.22	0.26	0.33	0.35
D^{2}B50	0.98	27.08	0.26	0.33	0.41	0.44

found in Appendix D. In addition to the human study, we also use Structure Similarity Index (SSIM) [41] to quantify the perceptual distance of the adversarial samples. SSIM ranges from -1 to 1, with a larger value indicating more similarity, while a 0 score indicating no similarity.

Results. The results are summarized in Figure 5. These figures show the tradeoffs between imperceptibility (y axis) and attack effectiveness (x axis) for various attacks. Figures (a) and (c) present targeted and untargeted attacks for Resnet50, respectively, and (b) and (d) for the adversarial trained Resnet152. Each point in these figures denotes an attack setting and each curve shows the variations of different settings of an attack. Points at the top-right corner are desirable, i.e., effective attacks with imperceptibility.

BIMx denotes BIM with an ℓ_∞ bound $x\%$ of 255. For example, BIM4 means the ℓ_∞ bound is $255 \times 4\% \approx 10$. FS1 and FS2 are feature space attacks using the relu2_1 and relu3_1 layers of VGG16, respectively, as the embedding layer. SMx is semantic attack with an optimization step of x. SP1 and SP2 are spatial attacks [34] with the flow constraint set to 0.005 and 0.0005 respectively. HCx is the hot-cold attack using a SSIM score x as the constraint. D^{2}Bx denotes that we allow $x\%$ of the average quantile change observed in BIM4 at the throttle plane. The reason we use percentage relative to BIM4 is for simplicity. Otherwise, one needs to fine tune the magnitude among different throttle planes. Overall, we have studied 37 attack settings, each entailing one user study. In these user studies, we involve 3700 samples and 252 users in total.

From Figure 5, we have the following observations.

(1) *Adversarial training has less effect on* D^{2}B. Observe in (a), without adversarial training, D^{2}B40 has a very high 80 confidence with 0.43 human preference, meaning that the attack is quite effective while humans can hardly tell the adversarial samples from the benign ones. BIM2 and BIM4 have similar performance. In (b), with adversarial training, D^{2}B40 still has close to 1.0 attack success rate and a 0.36 human preference rate. Even the lightest attack D^{2}B10 has close to 0.7 ASR and 0.42 human preference. In contrast, adversarial training is quite effective for defending BIM. Observe only BIM2 can achieve a similar human preference rate as D^{2}B40. But its ASR is as low as 0.37. The results on SSIM, i.e., (c) and (d), disclose similar observations.

(2) D^{2}B *is much more effective and imperceptible than other feature space attacks.* Observe that in (a), all the feature space attack data points are distant from the D^{2}B curve, mostly falling in the left-bottom quadrant. In other words, they have low attack confidence and humans can tell the adversarial samples. In (b), although the gap becomes narrower due to adversarial training, the differences are still prominent. For example, SM500 can achieve 1.0 ASR, just like D^{2}B40. However, its human preference rate is as low as 0.04. FS1 has a comparable human preference score as D^{2}B40, but its ASR is 0.66 (compared to 1.0 for D^{2}B40).

(3) D^{2}B *has the highest attack confidence/success rate at the same level of human preference/SSIM score (more towards the top-right corner than others). And with the same confidence/success rate, our adversarial examples are consistently more favored by the testers/SSIM score (for being more imperceptible) than those by other attacks.* Our adversarial examples with the most aggressive settings (e.g., D^{2}B40) have a similar human preference to pixel space attack with a very small bound (BIM2 and BIM4), indicating our attack is indeed imperceptible. With the increase of quantile change, perturbation bound, or optimization step, all the attacks achieve a higher success rate, and our attack is increasingly more imperceptible than others.

We further study the pixel distance and quantile distance of the generated adversarial examples by different attacks. The ℓ_∞ quantile distance is defined as $\max_{i \in \mathcal{I}} |C_i(\mathrm{x}^{\mathrm{adv}}) - C_i(\mathrm{x}^{\mathrm{nat}})|$. Table 1 and Table 2 show that with a similar level of attack confidence or attack success rate, our attack has a smaller ℓ_2 pixel distance and ℓ_∞ quantile distance. This indicates that our generated examples can achieve a similar level of attack effectiveness with less perturbation (and hence better imperceptibility). In other words, it can tolerate more aggressive perturbation without degrading imperceptibility as much, demonstrating the benefits of bounding deep layers. The larger ℓ_∞ pixel distance and the smaller ℓ_2 pixel distance (compared to BIM4) indicate our perturbations are more diverse, heavily piggy-backing on original features. The attack effectiveness and pixel/internal distances for other models are similar. Details can be found in Appendix H. Adversarial examples generated by the different settings of our attack and other attacks can be found in Figure 13 and Figure 14 in Appendix Q.

Other Experiments. We evaluate D^{2}B against three popular detection approaches and find that our attack is more or comparably persistent in the presence of detection (Appendix M). We study the transferability of our attack and observe that D^{2}B has a comparable/slightly-better transferability than others (Appendix O). We conduct a study about the essence of D^{2}B by studying the places that it aims to attack (Appendix N). Comparison of different reference models and deep bounds can be found in Appendix K and L. Details about the throttle planes used are discussed in Appendix E.

5. Conclusion

We propose a novel adversarial attack that perturbs individual content-features/neurons. It leverages a per-neuron normal distribution quantile bound and a polynomial barrier loss to address the non-uniformity of internal values regarding perception. Our evaluation on ImageNet, five models, and comparison with seven other state-of-the-art attacks demonstrates that the examples generated by our attack are more imperceptible while achieving better attack effectiveness. This poses new challenges to existing defense. It is also more persistent in the presence of various detection techniques. We discuss ethical considerations in Appendix P.

Acknowledgement

We thank the anonymous reviewers for their constructive comments. This research was supported, in part by IARPA TrojAI W911NF-19-S-0012, NSF 1901242 and 1910300, ONR N000141712045, N000141410468 and N000141712947. Any opinions, findings, and conclusions in this paper are those of the authors only and do not necessarily reflect the views of our sponsors.

References

[1] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *CoRR*, abs/1802.00420, 2018. 3

[2] Anand Bhattad, Minjin Chong, Kaizhao Liang, Bo Li, and David Forsyth. Unrestricted adversarial examples via semantic manipulation. In *International Conference on Learning Representations (ICLR)*, 2020. 2, 3, 6, 18

[3] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 5

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Proceedings of 38th IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017. 3, 6, 11

[5] Fabio Carrara, Rudy Becarelli, Roberto Caldelli, Fabrizio Falchi, and Giuseppe Amato. Adversarial examples detection in features distance spaces. In Laura Leal-Taixé and Stefan Roth, editors, *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part II*, volume 11130 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 2018. 3

[6] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26, 2017. 3

[7] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. pages 4723–4731, 10 2019. 3, 6

[8] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E. Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression. *CoRR*, abs/1705.02900, 2017. 17, 18

[9] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *CoRR*, abs/1907.02544, 2019. 3

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 3, 11

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 6

[12] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1614–1619, 2018. 3

[13] Shengyuan Hu, Tao Yu, Chuan Guo, Wei-Lun Chao, and Kilian Q Weinberger. A new defense against adversarial images: Turning a weakness into a strength. In *Advances in Neural Information Processing Systems*, pages 1633–1644, 2019. 17, 18

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1501–1510, 2017. 3

[15] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features, 2019. 16

[16] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, and Mingli Song. Neural style transfer: A review. *CoRR*, abs/1705.04058, 2017. 16

[17] Nupur Kumari, Mayank Singh, Abhishek Sinha, Harshitha Machiraju, Balaji Krishnamurthy, and Vineeth N. Balasubramanian. Harnessing the vulnerability of latent layers in adversarially trained models. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 2779–2785. ijcai.org, 2019. 3, 6, 13

[18] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *Proceedings of 5th International Conference on Learning Representations (ICLR)*, 2017. 3, 6, 11, 18

[19] Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 10408–10418, 2019. 3

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of 6th International Conference on Learning Representations (ICLR)*, 2018. 3, 11

[21] Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. Rethinking softmax cross-entropy loss for adversarial robustness. *CoRR*, abs/1905.10626, 2019. 3

[22] Nicolas Papernot and Patrick D. McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *CoRR*, abs/1803.04765, 2018. 3

[23] Andras Rozsa, Ethan M. Rudd, and Terrance E. Boult. Adversarial diversity and hard positive generation. *CoRR*, abs/1605.01775, 2016. 3, 6

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej

Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6

[25] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J. Fleet. Adversarial manipulation of deep representations. In *ICLR (Poster)*, 2016. 3

[26] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *CoRR*, abs/1805.06605, 2018. 3

[27] Mahmood Sharif, Lujo Bauer, and Michael K. Reiter. On the suitability of l_p-norms for creating and preventing adversarial examples. *CoRR*, abs/1802.09653, 2018. 3

[28] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. In *Advances in Neural Information Processing Systems*, pages 8312–8323, 2018. 3

[29] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8312–8323. Curran Associates, Inc., 2018. 3

[30] David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. *CoRR*, abs/1812.00740, 2018. 3

[31] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses, 2020. 3

[32] Robert J Vanderbei et al. *Linear programming*. Springer, 2015. 5

[33] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001. 3

[34] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. *CoRR*, abs/1801.02612, 2018. 3, 6, 8

[35] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 501–509, 2019. 6

[36] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[37] Qiuling Xu. D2B Code and Samples. https://github.com/qiulingxu/D2B, 2022. 1, 12

[38] Qiuling Xu, Guanhong Tao, Siyuan Cheng, Lin Tan, and Xiangyu Zhang. Towards feature space adversarial attack. *arXiv preprint arXiv:2004.12385*, 2020. 2, 3, 6, 18

[39] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Proceedings of 25th Annual Network and Distributed System Security Symposium (NDSS)*, 2018. 17, 18

[40] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016. 6, 12

[41] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 3, 6, 7