Robust Feature-Level Adversaries are Interpretability Tools

Anonymous Author(s)
Affiliation
Address
email

Abstract

The literature on adversarial attacks in computer vision typically focuses on pixel-level perturbations. These tend to be very difficult to interpret. Recent work that manipulates the latent representations of image generators to create "feature-level" adversarial perturbations gives us an opportunity to explore perceptible, interpretable adversarial attacks. We make three contributions. First, we observe that feature-level attacks provide useful classes of inputs for studying the representations in models. Second, we show that these adversaries are versatile and highly robust. We demonstrate that they can be used to produce targeted, universal, disguised, physically-realizable, and black-box attacks at the ImageNet scale. Third, we show how these adversarial images can be used as a practical interpretability tool for identifying bugs in networks. We use these adversaries to make predictions about spurious associations between features and classes which we then test by designing "copy/paste" attacks in which one natural image is pasted into another to cause a targeted misclassification. Our results indicate that feature-level attacks are a promising approach for rigorous interpretability research. They support the design of tools to better understand what a model has learned and diagnose brittle feature associations.

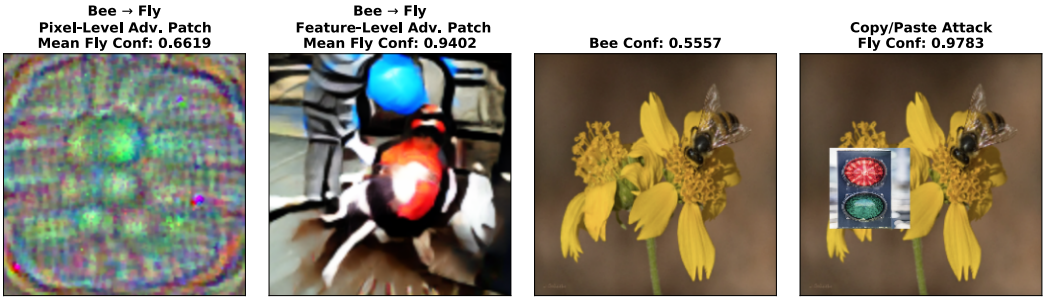
1 Introduction

State-of-the-art neural networks are vulnerable to adversarial examples. Conventionally, adversarial inputs for visual classifiers take the form of small-norm perturbations to natural images [64, 17]. These perturbations reliably cause confident misclassifications. However, to a human, they typically appear as random or mildly-textured noise. Consequently, it is difficult to interpret these attacks. Beyond the observation that such attacks are possible, it is hard to learn about the underlying victim network from these pixel-level perturbations.

In contrast, many real-world failures of biological vision are caused by perceptible, human-describable features. For instance, the ringlet butterfly's predators are stunned by adversarial "eyespots" on its wings (Appendix A.1, Fig. 7). This falls outside the scope of conventional adversarial examples because the misclassification results from a feature-level change to an object/image. The adversarial eyespots are robust in the sense that the same attack works across a variety of different backgrounds and viewing conditions. Furthermore, because the attack relies on high-level features, it is easy for a human to describe it.

This work takes inspiration from the ringlet butterfly's eyespots and similar examples in which a model is fooled in the real world by an interpretable feature (e.g. [45]). Our goal is to design adversaries that reveal easily-understandable weaknesses of the victim network. We focus on two desiderata for adversarial perturbations: attacks must be (1) interpretable (i.e. describable) to a human, and (2) robust so that interpretations generalize. We refer to these types of attacks as "feature-

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) (b) (c) (d)

Figure 1: Our feature-level adversaries are useful for interpreting deep networks (we used a ResNet50 [20]). (a) A pixel-level adversarial patch trained to make images of bees misclassified as flies. (b) An analogous feature-level adversarial patch. (c) A correctly-classified image of a bee. (d) A successful copy/paste attack whose design was guided by adversarial examples like the one in (b).

level" adversarial examples. Several previous works have created attacks by perturbing the latent representations of an image generator (e.g., [23]), but thus far, approaches have been small in scale, limited in robustness, and not interpretability-driven (See Section 2).

We build on this prior work to propose a method that generates feature-level attacks against computer vision models. Our attack method works on ImageNet scale models and creates robust, feature-level adversarial examples. We test three methods of introducing adversarial features into source images either by modifying the generator's latents and/or inserting a generated patch into natural images. In contrast to previous works that have enforced the "adversarialness" of attacks only by inserting small features or restricting the distance between an adversary and a benign input, we also introduce new methods that regularize the feature to be perceptible yet disguised to resemble something other than the target class.

We show that our method produces robust attacks that provide actionable insights into a network's learned representation. Fig. 1 demonstrates the interpretability benefits of this type of feature-level attack. It compares a conventional, pixel-level, adversarial patch, created using the method from [6], with a feature-level attack using our method. While both attacks attempt to make a network misclassify a bee as a fly, the pixel-level attack exhibits high-frequency patterns and lacks visually-coherent objects. On the other hand, the feature-level attacks displays easily describable features: the colored balls. We can validate this insight by considering the network performance when a picture of a traffic light is inserted into the image a bee. In this example, the image classification goes from a 55% confidence that the image is a bee to a 97% confidence that the image is of a fly. Fig. 6 shows several of these "copy/paste" attacks and the feature-level attacks that inspired them.

Our contributions are threefold.

1. **Conceptual:** We observe that robust feature-level adversaries can used to produce useful types of inputs for studying the representations of deep networks
2. **Robust Attacks:** We introduce methods for generating feature-level adversaries that are uniquely versatile and able to produce targeted, universal, disguised, physically-realizable, and black-box attacks at the ImageNet scale. See Table 1.
3. **Interpretability:** We generalize from our adversarial examples to design copy/paste attacks, verifying that our adversaries help us understand the network well enough to exploit it.

The following sections contain background, methods, experiments, and discussion. Appendix A.11 has a high-level summary for a lay-audience. We are making code openly available.

2 Related Work

Here, we contextualize our approach with others related to improving on conventional adversarial examples [64, 17]. Table 1 summarizes capabilities.

	Targeted	Universal	Disguised	Physically-Realizable	Transferable/Black-Box	Copy/Paste	ImageNet Scale
Szegedy et al. (2013) [64], Goodfellow et al. (2014) [17]	✔	✗	✗	✗	✗	✗	✔
Natural mimics, e.g. Peacock, Ringlet Butterfly	✔	✔	✗	✔	✔	✗	N/A
Hayes et al. (2018)[19]	✔	✔	✗	✗	✔	✗	✔
Mopuri et al. (2018)a[40]	✔	✔	✗	✗	✔	✗	✔
Mopuri et al. (2018)b[41]	✔	✔	✗	✗	✔	✗	✔
Poursaeed et al. (2018) [50]	✔	✔	✗	✗	✔	✗	✔
Xiao et al. (2018) [70]	✔	✗	✗	✗	✔	✗	✔
Hashemi et al. (2020) [18]	✔	✔	✗	✗	✔	✗	✔
Wong et al. (2020) [69]	✔	✗	✗	✗	✗	✗	✗
Liu et al. (2018) [37]	✔	✗	✔	✗	✗	✗	✗
Samangouei et al. (2018) [53]	✔	✗	✗	✗	✗	✗	✗
Song et al. (2018) [61]	✔	✗	✔	✗	✔	✗	✗
Joshi et al. (2018) [28]	✔	✗	✗	✗	✗	✗	✗
Joshi et al. (2019) [27]	✔	✗	✔	✗	✗	✗	✗
Singla et al. (2019) [58]	✔	✗	✗	✗	✔	✗	✗
Hu et al. (2021) [23]	✔	✔	✔	✔	✗	✗	✗
Wang et al. (2020) [67]	✔	✔	✔	✗	✗	✗	✗
Kurakin et al. (2016) [33]	✔	✗	✗	✔	✔	✗	✔
Sharif et al. (2016) [55]	✔	✗	✗	✔	✔	✗	✔
Brown et al. (2017) [6]	✔	✔	✗	✔	✔	✗	✔
Eykholt et al. (2018) [14]	✔	✗	✔	✔	✗	✗	✗
Athalye et al. (2018) [2]	✔	✗	✗	✔	✗	✗	✔
Liu et al. (2019) [36]	✔	✗	✗	✔	✔	✗	✔
Thys et al. (2019) [65]	✔	✔	✗	✔	✗	✗	✗
Kong et al. (2020) [31]	✔	✗	✗	✔	✗	✗	✗
Komkov et al. (2021) [30]	✔	✔	✗	✔	✗	✗	✗
Dong et al. (2017) [10]	✔	✗	✗	✗	✔	✗	✔
Geihros et al. (2018) [16]	✔	✗	✗	✗	✗	✗	✔
Leclerc et al. (2021) [34]	✔	✗	✔	✗	✗	✔	✔
Carter et al. (2019) [7]	✔	✗	✔	✗	✗	✔	✔
Mu et al. (2020) [42]	✗	✗	✔	✗	✗	✔	✔
Hernandez et al. (2022) [22]	✗	✗	✔	✗	✗	✔	✔
Ours	✔	✔	✔	✔	✔	✔	✔

Table 1: Our feature-level attacks are uniquely versatile. Each row represents a related work (in the order in which they are presented in Section 2.) Each column indicates a demonstrated capability of a method. Note that two methods each having a ✔for a capability does not imply they do so equally well. *Targeted*=working for an arbitrary target class. *Universal*=working for any source example. *Disguised*=Perceptible and resembling something other than the target class. *Physically-realizable*=working in the physical world. *Transferable/black-box*=transferring to other classifiers. *Copy/Paste*=useful for designing attacks in which a natural feature is pasted into a natural image.

Inspiration from Nature: Mimicry is common in nature, and sometimes, rather than holistically imitating another species, a mimic will only display particular features. For example, many animals use adversarial eyespots to confuse predators [62] (see Appendix A.1 Fig. 7a). Another example is the mimic octopus which imitates the patterning, but not the shape, of a banded sea snake. We show in Figure 7b that a ResNet50 classifies an image of one as a sea snake.

Generative Modeling: An approach related to ours has been to train a generator or autoencoder to produce small adversarial perturbations that are applied to natural inputs. This has been done to synthesize imperceptible attacks that are transferable, universal, or efficient to produce [19, 40, 41, 50, 70, 18, 69]. Rather than training a generator, ours and other works have perturbed the latents of pretrained generative models to produce perceptible alterations. [37] did this with a differentiable image renderer. Others [53, 61, 28, 27, 58, 23] have used deep generative networks, and [67] aimed to create more semantically-understandable attacks by using an autoencoder with a "disentangled" embedding space. Our work is different in four ways. (1) These works focus on small classifiers trained on simple datasets (MNIST [35], Fashion MNIST [71], SVHN [43], CelebA [38], BDD [73], INRIA [8], and MPII [1]) while we work at the ImageNet [51] scale. (2) We do not simply rely on using small features or restricting the distance to a benign image to enforce the adversarialness of attacks. We introduce techniques that regularize the adversarial feature to be perceptible yet disguised to resemble something other than the target class. (3) We evaluate three distinct ways of inserting adversarial features into images. (4) Our work is interpretability-oriented.

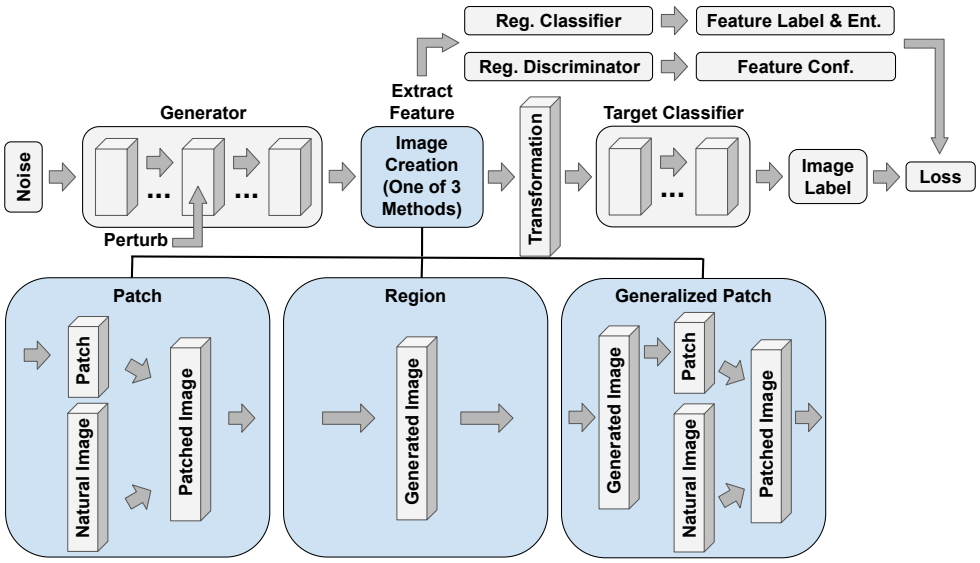


Figure 2: Our fully-differentiable pipeline for creating feature-level attacks. In each experiment, we create either "patch," "region," or "generalized patch" attacks. The regularization terms in the loss based on an external classifier and discriminator are optional and are meant to make the inserted feature appear disguised as some non-target class.

Attacks in the Physical World: Physical-realizability demonstrates robustness. We show that our attacks fool a classifier when printed and photographed. This directly relates to [33] who found that pixel-space adversaries could do this to a limited extent in controlled settings. More recently, [55, 6, 14, 2, 36, 65, 31, 30] created adversarial clothing, stickers, or objects. In contrast with these, we also produce attacks in the physical world that are disguised as a non-target class.

Adversaries and Interpretability: Using adversarial examples to better interpret networks has been proposed by [10] and [66]. We use ours to discover human-describable feature/class associations learned by a network. This relates to [16] and [34] who debug networks by searching over feature insertions, transformations, and textural changes. More similar to our work are [7, 42, 22], who use feature visualization [46] and network dissection [3] to interpret the network. Each use their interpretations to design "copy/paste" attacks in which one natural image pasted inside another causes an unrelated misclassification. We add to this work with a new method to identify such adversarial features. Unlike any previous approach, ours does so in a way that allows for targeted attacks that take into account an arbitrary distribution of source images.

3 Methods

We adopt the "unrestricted" adversary paradigm [61] under which an attack is successful if the network's classification differs from an oracle's (e.g., a human). Our adversaries can only change a small, fixed portion of either the generator's latent or the image. We use white-box access to the network, though we present black-box attacks based on transfer from an ensemble in Appendix A.6.

Our attacks involve perturbing the latent representation in some layer of an image generator to produce an adversarial feature-level alteration. Fig. 2 depicts our approach. We test three types of attacks, "patch", "region" and "generalized patch" (plus a fourth in Appendix A.5 which we call "channel" attacks). We find patch attacks to generally be the most successful.

Patch: We use the generator to produce a square patch that is inserted into a natural image [56].

Region: Starting with some generated image, we randomly select a square column of the latent in a generator layer which spans the channel dimension and replace it with a learned insertion. This is analogous to a square patch in the pixel representation. We keep insertion location fixed over training. The modified latent is passed through the rest of the generator, producing the adversarial image.

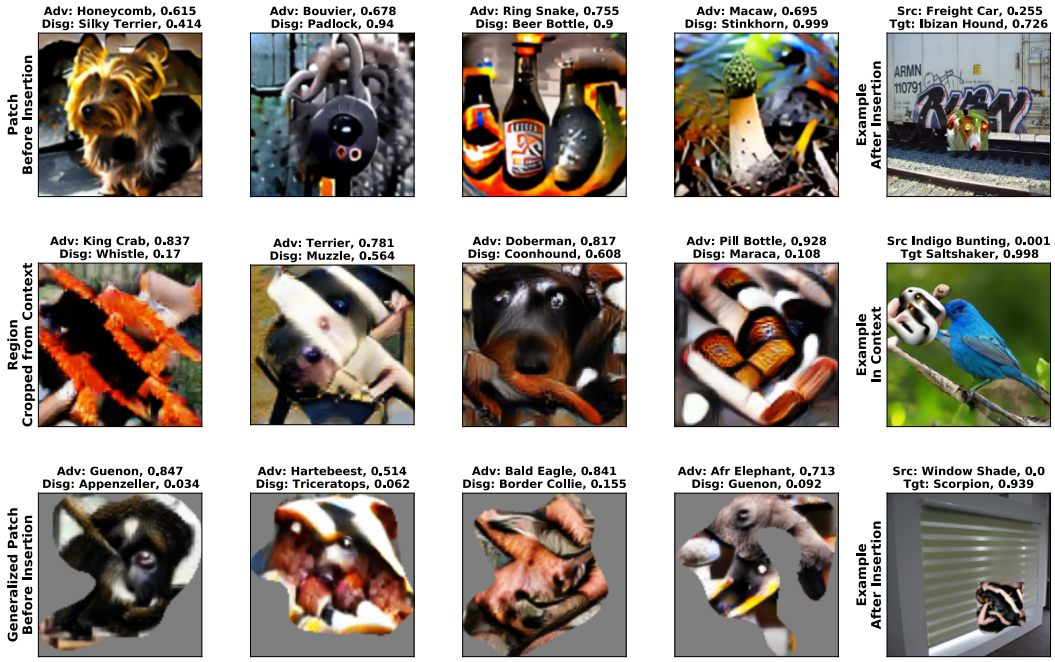


Figure 3: Examples of targeted, universal feature-level adversaries from patch (top), region (middle), and generalized patch (bottom) attacks. The first four columns show the adversarial features. The fooling confidence is labeled 'Adv.' and is calculated under random source images (and random insertion locations for patch and generalized patch attacks). The target network's disguise class confidence for each patch or extracted generalized patch is labeled 'Disg.' The final column shows examples of the features applied to images. The example image for each is labeled with its source and target class confidences.

Generalized Patch: These patches can be of any shape, hence the name "generalized" patch. We first generate a region attack and then extract a generalized patch from it. We do this by taking the absolute-valued pixel-level difference between the original and adversarial image, applying a Gaussian filter for smoothing, and creating a binary mask from the top decile of these pixel differences. We apply this mask to the generated image to isolate the region that the perturbation altered. We can then treat this as a patch and overlay it onto an image in any location.

Basic Attacks: For all attacks, we train a perturbation δ to the latent of the generator to minimize a loss that optimizes for both fooling the classifier and appearing interpretable:

$$\arg\min_{\delta} \mathbb{E}_{x\sim\mathcal{X},t\sim\mathcal{T},l\sim\mathcal{L}} \quad L_{\text{x-ent}}[C(A(x,\delta,t,l)), y_{\text{targ}}] + L_{\text{reg}}[A(x,\delta,t,l)] \tag{1}$$

with $\mathcal{X}$ a distribution over source images (e.g., a dataset or generation distribution), $\mathcal{T}$ a distribution over transformations, $\mathcal{L}$ a distribution over insertion locations (this only applies for patches and generalized patches), C the target classifier, A an image-generating function, $L_{\text{x-ent}}$ a targeted crossentropy loss for fooling the classifier, y_{targ} the target class, and L_{reg} a regularization loss. The adversary has no control over $\mathcal{X}$, $\mathcal{T}$, or $\mathcal{L}$, so it must learn features that fool the network independent of any particular source image, transformation, or insertion location. For all of our attacks, L_{reg} contains a total variation loss, $TV(a)$, to discourage high-frequency patterns.

"Disguised" Attacks: Ideally, a feature-level adversarial example should appear to a human as easily-describable but should not resemble the attack's target class. We call such attacks "disguised." Here, the main goal is not to fool a human, but to help them *learn* about what types of realistic features might cause the model to make a mistake. To train these disguised attacks, we use additional terms in L_{reg} as proxies for these two criteria. We differentiably resize the patch or the extracted generalized patch and pass it through a GAN discriminator and auxiliary classifier. We then add weighted terms to the regularization loss based on the discriminator's (D) logistic loss for classifying the input as fake, the output entropy (H) of some classifier (C'), and/or the negative of the classifier's

crossentropy loss for labeling the input as the attack's target class. Note that C' could either be the same or different from C. With all of these terms, the regularization objective is

$$L_{\text{reg}}(a) = \lambda_1 TV(a) + \underbrace{\lambda_2 L_{\text{logistic}}[D(P(a))] + \lambda_3 H[C'(P(a))] - \lambda_4 L_{\text{x-ent}}[C'(P(a), y_{\text{targ}})]}_{\text{“Disguise” Regularizers}}. \quad (2)$$

Here, $P(a)$ returns the extracted and resized patch from adversarial image a. In order, these three new terms encourage the adversarial feature to (1) look realistic, and (2) look like some specific class, but (3) not the target class. The choice of disguise class is left entirely to the training process.

4 Experiments

We use BigGAN generators from [5, 68], and perturb the post-ReLU outputs of the internal 'GenBlocks.' We used the BigGAN discriminator and adversarially trained classifiers from [12] for disguise regularization. By default, we attacked a ResNet50 [20], restricting patch attacks to 1/16 of the image and region and generalized patch attacks to 1/8. Appendix A.2 has additional details. First, we show that these feature level adversaries are highly robust to suggest that interpretations based on them are generalizable. Second, we demonstrate that these adversaries can help one understand a network well enough to exploit it.

4.1 Robust Attacks

Figure 3 shows examples of targeted, universal, and disguised feature-level patch (top), region (middle), and generalized patch (bottom) attacks which were each trained with all of the disguise regularization terms from Eq. 2. We find the disguises to be effective, particularly for the patches (top row), but imperfect. Appendix A.3 discusses this and what it may suggest about networks, human visual cortex, and size bias.

Performance versus Disguise: Here, we study our patch attacks in depth to test how effective they are at fooling the network and how successfully they can help to identify non-target-class features that can fool the network. We compared seven different approaches. The first was our full approach using the generator and all disguise regularization terms from Eq. 2. The rest were ablation tests in which we omitted the generator (No Gen), the discriminator (No Disc) regularization term (No Reg), the entropy regularization term (No Ent), the crossentropy regularization term (No Patch X-ent), all three regularization terms (Only Gen), and finally the discriminator and all three regularization terms (Brown et al. '17). This final unregularized, pixel-level method resulted in the same approach as Brown et al. (2017) [6]. For each test, all else was kept identical including penalizing total variation, training under transformations, and initializing the patch as a generator output.

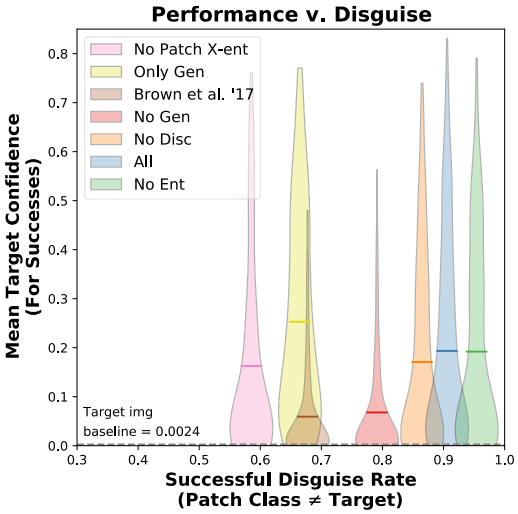


Figure 4: Targeted, universal patch attacks compared. Successful disguise success rate (x axis) shows the proportion of attacks in which the patch was not classified by the network as the target class when viewed on its own. Mean target confidence (y axis) gives the empirical fooling confidences of 250 patch attacks. Each is an average over 100 source images. The proportion of each distribution above 0.5 gives a lower-bound for the top-1 attack success rate. The mean target class confidence for using randomly-sampled natural target class images as patches is 0.0008 and is shown as a thin dotted line at the bottom.

For each method, we generated universal attacks with random target classes until we obtained 250 successfully "disguised" ones in which the resulting adversarial feature was not classified by the network as the target class when viewed on its own. Fig. 4 plots the success rate versus the distribution of target class mean fooling confidences for

Figure 5: Examples of targeted, disguised, universal, and physically-realizable feature-level attacks. See Appendix A.10 Fig. 13 for full-sized versions of the patches.

each type of attack. For all methods, these universal attacks have variable fooling confidences due in large part to the random selection of target class. Some attacks are stochastically Pareto dominated by others. For example, the pixel-space Brown et al. (2017) attacks were the least effective at fooling the target network and had the third least disguise rate. In other cases, there is a tradeoff between fooling performance and disguise which can be controlled using the regularization terms from Eq. 2. We also compare all attacks to a baseline in which we used randomly-sampled natural images from the target class as patches. These resulted in a mean target class confidence of 0.0024.

Notably, Fig. 4 does not capture everything that one might care about in these attacks. It does not show any measure of how "realistic" the resulting patches look. In Appendix A.4, Fig. 9 plots the same fooling confidence data from the y axis in Fig. 4 versus the disguise class label confidence from an Inception-v3 which we use as a proxy for how realistic a human would find the patch. It suggests that the best attacks for producing patches that appear realistic are the "All" and "No Disc" methods. In Appendix A.10, Figs. 13, 14, and 15 give examples of successful "All", "Only Gen", and "Brown et al. '17" attacks respectively. Because they were initialized from generator outputs, some of the "Brown et al. '17" attacks have a veneer-like resemblance to non-target class features. Nonetheless, they contain higher-frequency patterns and less coherent objects in comparison to the two sets of feature-level attacks. We subjectively find the "All" attacks to be the best disguised.

Physical-Realizability: To test their ability to transfer to the physical world, we generated 100 additional targeted, universal, and disguised adversarial patches. We used the generator and all regularization terms (the "All" condition from above). We selected the 10 with the best mean fooling confidence, printed them, and photographed each next to 9 different ImageNet classes.[1] We confirmed that photographs of each object were correctly classified without a patch. Figure 5 shows successful examples. Meanwhile, resizable and printable versions of all these patches and others are in Appendix A.10. The mean and standard deviation of the fooling confidences for our attacks in the physical world were 0.312 and 0.318 respectively ($n = 90$, not i.i.d.). This means that these patches' mean effectiveness dropped by less than $\frac{1}{2}$ when transferring to the physical world.

Black-Box Attacks: In Appendix A.6, we show that our universal attacks can transfer from an ensemble to a held-out model.

4.2 Interpretability

If an adversarial feature successfully fools the victim network, this suggests that the network associates that feature in context of a source image with the target class. We find that our adversaries can suggest both beneficial and harmful feature-class associations. In Appendix A.7, Fig. 11 provides a simple example of each.

One challenge in the explainable AI literature is to develop interpretations that go beyond seeming-plausible and stand up to scrutiny [72]. Observing robust feature level adversarial patches can easily be used to develop hypotheses about the network's behavior, e.g. "The network thinks that bee features plus colorful balls implies a fly." But are these valid, useful interpretations of the network? In other words, are our adversaries adversarial because of their interpretable qualities, or is it because of hidden motifs? A rigorous approach to interpretability requires verification. We do this by using our attacks to make and validate predictions about how to fool the target network with natural objects.

[1] Backpack, banana, bath towel, lemon, jeans, spatula, sunglasses, toilet tissue, and toaster.

Figure 6: Feature-level adversaries can guide the design of class-universal copy/paste adversarial attacks. Patch adversary pairs are on the left, region in the middle, and generalized patch on the right of each odd row. Attack examples are on each even row. We do not claim that the traffic light $\wedge$ bee $\rightarrow$ fly examples on row 4 are necessarily adversarial, but they demonstrate alongside the bee $\wedge$ traffic light $\rightarrow$ fly adversaries that the adversarial features are sensitive to the the source images. For each attack except traffic light $\rightarrow$ fly, we limited ourselves to attempting 10 natural patches.

Validating Interpretations with Copy/Paste Attacks: A "copy-paste" attack is created by inserting one natural image into another to cause an unexpected misclassification. They are more restricted than patch attacks because the features pasted into an image must be natural objects. As a result, they are of high interest for physically-realizable attacks because they suggest combinations of real objects that yield unexpected classifications. They also have precedent in the real world. For example, subimage insertions into pornographic images have been used to evade NSFW content detectors [74].

To develop copy/paste attacks, we select a source and target class, generate class-universal adversarial features, and manually analyze them for motifs that resemble natural objects. Here, we used basic attacks without the disguise regularization terms from Eq. 2. We then paste images of these objects into natural images and pass them through the classifier.

Fig. 6 shows four types of copy/paste attacks. In each odd row, we show six patch, region, and generalized patch adversaries that were used to guide the design of a copy/paste attack. In each even row are the copy/paste adversaries for the 6 (of 50) images for the source class for which the insertion resulted in the highest target confidence increase along with the mean target class confidences before and after patch insertion for those 6. The success of these attacks shows their usefulness for interpreting the target network because they require that a human understands the mistake the model is making like "Bee $\wedge$ Traffic Light $\rightarrow$ Fly" well enough to manually exploit it. Given the differences in the adversarial features that are produced in the Bee $\rightarrow$ Fly and Traffic Light $\rightarrow$ Fly attacks, Fig. 6 also demonstrates how our attacks take the distribution of source images into account.

Three prior works [7, 42, 22] have developed copy/paste attacks, also via interpretability tools. Unlike [42, 22], our approach allows for targeted attacks. And unlike all three, rather than simply identifying features associated with a class, our adversaries generate fooling features for a target class *conditional* on any distribution over source images (i.e. the source class) with which the adversaries are trained. Little work has been done on copy/paste adversaries, and thus far, methods have either not allowed for targeted attacks or have required a human in the loop. This makes objective comparisons difficult. However we provide examples of a feature-visualization based method inspired by [7] in Appendix A.8 to compare with ours. For the Indian $\rightarrow$ African Elephant attack, the source and target class share many features, and we find no evidence that feature visualization is able to suggest useful features for copy/paste attacks. This suggests that our attacks' ability to take the source image distribution into account may be more helpful for discovering certain weaknesses compared to the baseline inspired by [7].

5 Discussion and Broader Impact

Here we use feature-level adversarial examples to fool and interpret deep networks in order to contribute to a more practical understanding of network vulnerabilities. As an attack method, our approach is versatile. It can produce targeted, universal, disguised, physically-realizable, black-box, and copy/paste attacks at the ImageNet scale. Furthermore, this attack method can be used as an interpretability tool to help diagnose flaws in models. We ground the notion of interpretability in the ability to make predictions about when a model will fail. Finally, we demonstrate this through the design of targeted copy/paste attacks for any distribution over source inputs.

Like any work on adversarial attacks, our methods could be used maliciously to make a system fail, but we emphasize their diagnostic value. Understanding threats is a prerequisite to avoiding them. Given the robustness and versatility of our attacks, we argue that they may be valuable for continued work to address threats that systems may face in practical applications. There are at least two ways in which these methods can be useful.

The first is for adversarial training. Training networks on adversarial images has been shown to improve their robustness to the attacks that are used [12]. But this does not guarantee robustness to other types of adversarial inputs (e.g. [21]). Our feature-level attacks are categorically different from conventional pixel-level ones, and our copy/paste attacks show how networks can be fooled by novel combinations of natural objects, failures that are outside the conventional paradigm for adversarial robustness (e.g., [12]). Consequently, we expect that adversarial training on broader classes of attacks such as the one we propose here will be valuable for designing more robust models. As a promising sign, we show in Appendix A.9 that adversarial training is useful for our attacks.

The second is for rigorously diagnosing flaws. We show that feature-level adversaries aid the discovery of exploitable spurious feature/class associations (Fig. 6) and a socially-harmful bias (Appendix A.7 Fig. 11). Our approach could also be extended beyond what we have demonstrated here. For example, our methods may be useful for feature visualization [46] of a network's internal neurons. An analogous approach to ours can also be used in Natural Language Processing [60, 49], and we are currently working on a method for this. Furthermore, it may be valuable to use these adversaries to identify generalizable flaws in networks that humans can easily understand but with minimal

human involvement. This would be much more scalable and prevent human priors from influencing interpretations. Toward this goal, we are currently working on methods for the fully-automated discovery of copy/paste attacks

A limitation of our approach is that when multiple desiderata are optimized for at the same time (e.g., universality + transformation robustness + disguise), attacks are generally less successful, more time-consuming, and require more screening to find good ones. This could be a bottleneck for large-scale adversarial training. Ultimately, this type of attack is limited by the efficiency and quality of the generator, so future work should leverage advances in generative modeling. Our evaluation method is also limited to be a proof-of-concept for the design of copy/paste attacks. Future work should evaluate this more rigorously. We are currently working toward developing a benchmark for interpretability tools based on their ability to aid a human in rediscovering trojans [15] that have been implanted into a model.

As AI becomes increasingly capable, it becomes more important to design models that are reliable. Each of the 11 proposals for building safe AI outlined in [25] explicitly call for adversarial robustness and/or interpretability tools, and recent work from [76] on high-stakes reliability in AI found that interpretability tools strengthened their ability to produce inputs for adversarial training. Given the close relationship between interpretability and adversarial robustness, continued study of the connections between them will be key for building safer AI systems.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[3] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

[4] Anand Bhattad, Min Jin Chong, Kaizhao Liang, Bo Li, and David A Forsyth. Unrestricted adversarial examples via semantic manipulation. *arXiv preprint arXiv:1904.06347*, 2019.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[6] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

[7] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas. *Distill*, 4(3):e15, 2019.

[8] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[9] Joel Dapello, Tiago Marques, Martin Schrimpf, Franziska Geiger, David D Cox, and James J DiCarlo. Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations. *BioRxiv*, 2020.

[10] Yinpeng Dong, Hang Su, Jun Zhu, and Fan Bao. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv preprint arXiv:1708.05493*, 2017.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019.

[13] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*, 2019.

[14] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

[15] Arturo Geigel. Neural network trojan. *Journal of Computer Security*, 21(2):191–232, 2013.

[16] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[18] Atiye Sadat Hashemi, Andreas Bär, Saeed Mozaffari, and Tim Fingscheidt. Transferable universal adversarial perturbations using generative models. *arXiv preprint arXiv:2010.14919*, 2020.

[19] Jamie Hayes and George Danezis. Learning universal adversarial perturbations with generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 43–49. IEEE, 2018.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021.

[22] Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. Natural language descriptions of deep visual features. *arXiv preprint arXiv:2201.11114*, 2022.

[23] Yu-Chih-Tuan Hu, Bo-Han Kung, Daniel Stanley Tan, Jun-Cheng Chen, Kai-Lung Hua, and Wen-Huang Cheng. Naturalistic physical adversarial patch for object detectors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7848–7857, 2021.

[24] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[25] Evan Hubinger. An overview of 11 proposals for building safe advanced ai. *arXiv preprint arXiv:2012.07532*, 2020.

[26] Jessica Hullman and Nick Diakopoulos. Visualization rhetoric: Framing effects in narrative visualization. *IEEE transactions on visualization and computer graphics*, 17(12):2231–2240, 2011.

[27] Ameya Joshi, Amitangshu Mukherjee, Soumik Sarkar, and Chinmay Hegde. Semantic adversarial attacks: Parametric transformations that fool deep classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4773–4783, 2019.

[28] Shalmali Joshi, Oluwasanmi Koyejo, Been Kim, and Joydeep Ghosh. xgems: Generating examplars to explain black-box models. *arXiv preprint arXiv:1806.08867*, 2018.

[29] Lim Kiat. Lucent. `https://github.com/greentfrapp/lucent`, 2019.

[30] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 819–826. IEEE, 2021.

[31] Zelun Kong, Junfeng Guo, Ang Li, and Cong Liu. Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14254–14263, 2020.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[33] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[34] Guillaume Leclerc, Hadi Salman, Andrew Ilyas, Sai Vemprala, Logan Engstrom, Vibhav Vineet, Kai Xiao, Pengchuan Zhang, Shibani Santurkar, Greg Yang, et al. 3db: A framework for debugging computer vision models. *arXiv preprint arXiv:2106.03805*, 2021.

[35] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[36] Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1028–1035, 2019.

[37] Hsueh-Ti Derek Liu, Michael Tao, Chun-Liang Li, Derek Nowrouzezahrai, and Alec Jacobson. Beyond pixel norm-balls: Parametric adversaries using an analytically differentiable renderer. *arXiv preprint arXiv:1808.02651*, 2018.

[38] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[39] Luke Melas. Pytorch-pretrained-vit. https://github.com/lukemelas/PyTorch-Pretrained-ViT, 2020.

[40] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018.

[41] Konda Reddy Mopuri, Phani Krishna Uppala, and R Venkatesh Babu. Ask, acquire, and attack: Data-free uap generation using class impressions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[42] Jesse Mu and Jacob Andreas. Compositional explanations of neurons. *arXiv preprint arXiv:2006.14032*, 2020.

[43] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[44] Mark D Norman, Julian Finn, and Tom Tregenza. Dynamic mimicry in an indo–malayan octopus. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1478):1755–1758, 2001.

[45] National Transportation Safety Board NTSB. Collision between vehicle controlled by developmental automated driving system and pedestrian. 2018.

[46] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

[47] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[49] Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*, 2022.

[50] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018.

[51] 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.

[52] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? In *ArXiv preprint arXiv:2007.08489*, 2020.

[53] Pouya Samangouei, Ardavan Saeedi, Liam Nakagawa, and Nathan Silberman. Explaingan: Model explanation via decision boundary crossing transformations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 666–681, 2018.

[54] Martin Schrimpf, Jonas Kubilius, Michael J Lee, N Apurva Ratan Murty, Robert Ajemian, and James J DiCarlo. Integrative benchmarking to advance neurally mechanistic models of human intelligence. *Neuron*, 108(3):413–423, 2020.

[55] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[56] Abhijith Sharma, Yijun Bian, Phil Munz, and Apurva Narayan. Adversarial patch attacks and defences in vision-based tasks: A survey, 2022.

[57] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[58] Sumedha Singla, Brian Pollack, Junxiang Chen, and Kayhan Batmanghelich. Explanation by progressive exaggeration. *arXiv preprint arXiv:1911.00483*, 2019.

[59] Uchida So and Ihor Durnopianov. Lpips pytorch. https://github.com/S-aiueo32/lpips-pytorch, 2019.

[60] Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. Universal adversarial attacks with natural triggers for text classification. *arXiv preprint arXiv:2005.00174*, 2020.

[61] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. *arXiv preprint arXiv:1805.07894*, 2018.

[62] Martin Stevens and Graeme D Ruxton. Do animal eyespots really mimic eyes? *Current Zoology*, 60(1), 2014.

[63] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[64] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[65] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019.

[66] Richard Tomsett, Amy Widdicombe, Tianwei Xing, Supriyo Chakraborty, Simon Julier, Prudhvi Gurram, Raghuveer Rao, and Mani Srivastava. Why the failure? how adversarial examples can provide insights for interpretable machine learning. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 838–845. IEEE, 2018.

[67] Shuo Wang, Shangyu Chen, Tianle Chen, Surya Nepal, Carsten Rudolph, and Marthie Grobler. Generating semantic adversarial examples via feature manipulation. *arXiv preprint arXiv:2001.02297*, 2020.

[68] Thomas Wolf. Pytorch pretrained biggan. https://github.com/huggingface/pytorch-pretrained-BigGAN, 2018.

[69] Eric Wong and J Zico Kolter. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020.

[70] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

[71] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[72] Fan Yang, Mengnan Du, and Xia Hu. Evaluating explanation without ground truth in interpretable machine learning. *arXiv preprint arXiv:1907.06831*, 2019.

[73] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

[74] Kan Yuan, Di Tang, Xiaojing Liao, XiaoFeng Wang, Xuan Feng, Yi Chen, Menghan Sun, Haoran Lu, and Kehuan Zhang. Stealthy porn: Understanding real-world adversarial images for illicit online promotion. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 952–966. IEEE, 2019.

[75] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[76] Daniel M Ziegler, Seraphina Nix, Lawrence Chan, Tim Bauman, Peter Schmidt-Nielsen, Tao Lin, Adam Scherlis, Noa Nabeshima, Ben Weinstein-Raun, Daniel de Haas, et al. Adversarial training for high-stakes reliability. *arXiv preprint arXiv:2205.01663*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix A.2 and supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Fig. 4
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section A.2.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] We only used open source assets plus ImageNet. We are license compliant but do not discuss this explicitly.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We are license compliant for ImageNet but do not discuss this explicitly.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] Data we use is limited to ImageNet.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]