
One Thing to Fool them All: Generating Interpretable, Universal, and Physically-Realizable Adversarial Features

Stephen Casper^{*123} Max Nadeau^{*234} Gabriel Kreiman²³

Abstract

It is well understood that modern deep networks are vulnerable to adversarial attacks. However, conventional attack methods fail to produce adversarial perturbations that are intelligible to humans, and they pose limited threats in the physical world. To study feature-class associations in networks and better understand their vulnerability to attacks in the real world, we develop feature-level adversarial perturbations using deep image generators and a novel optimization objective. We term these *feature-fool* attacks. We show that they are versatile and use them to generate targeted feature-level attacks at the ImageNet scale that are simultaneously interpretable, universal to any source image, and physically-realizable. These attacks reveal spurious, semantically-describable feature/class associations that can be exploited by novel combinations of objects. We use them to guide the design of “copy/paste” adversaries in which one natural image is pasted into another to cause a targeted misclassification.¹

1. Introduction

State-of-the-art neural networks are vulnerable to adversarial inputs, which cause the network to fail yet only differ from benign inputs in subtle ways. Adversaries for visual classifiers conventionally take the form of a small-norm perturbation to a benign source image that causes misclassification (Szegedy et al., 2013; Goodfellow et al., 2014). These perturbations are effective attacks, but to a human, they typically appear as random or mildly-textured noise. As

^{*}Equal contribution ¹MIT CSAIL ²Boston Children’s Hospital, Harvard Medical School ³Center for Brains, Minds, and Machines ⁴Harvard College, Harvard University. Correspondence to: Stephen casper <scasper@csail.mit.edu>, Max Nadeau <mnadeau@college.harvard.edu>.

Submitted to the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

¹https://github.com/thestephencasper/feature_fool.

such, analyzing these adversaries does not reveal information about the network relevant to how it will function—and how it may fail—when presented with human-interpretable features. Another limitation is the extent to which conventional adversaries work in the physical world. While conventional attacks can retain some effectiveness when photographed in a controlled setting (Kurakin et al., 2016), they are less effective in uncontrolled ones (e.g., Kong et al. (2020)) and cannot be created from natural objects.

Several works discussed in Section 2 have aimed to produce adversaries that are universal to any source image, interpretable (i.e., appear comprehensible to a human), or physically-realizable. But to the best of our knowledge, no method exists to accomplish all three. To better understand deep networks and what practical threats they face, we set out to create adversaries meeting all of these desiderata.

Because pixel-space optimization produces non-interpretable perturbations, a way to manipulate images at a higher level of abstraction is needed. We take inspiration from advancements in generative modeling (e.g., Brock et al. (2018)) at the ImageNet scale. Instead of pixels, we perturb the latent representations inside of a deep generator to manipulate an image in feature-space. In doing so, we produce adversarial features which are inserted into source images either by modifying the latents that generate them or by inserting a generated patch into natural images. We combine this with a loss function that uses an external discriminator and classifier to regularize the adversarial feature into appearing interpretable.

We use this strategy to produce what we term *feature-fool* attacks in which a high-level feature added to an image causes a misclassification yet appears intelligible to a human without resembling the target class. Fig. 1 shows an example in which a universal adversarial patch depicting an owl is printed and physically placed near jeans to fool a network into classifying the image as a loogerhead turtle. We show that our universal attacks are more coherent and better disguised than pixel-space ones while transferring to the physical world. To further demonstrate their potential for interpretable and physically-realizable attacks, we also use these adversaries to design *copy/paste* attacks in which one natural image is pasted into another to induce an un-

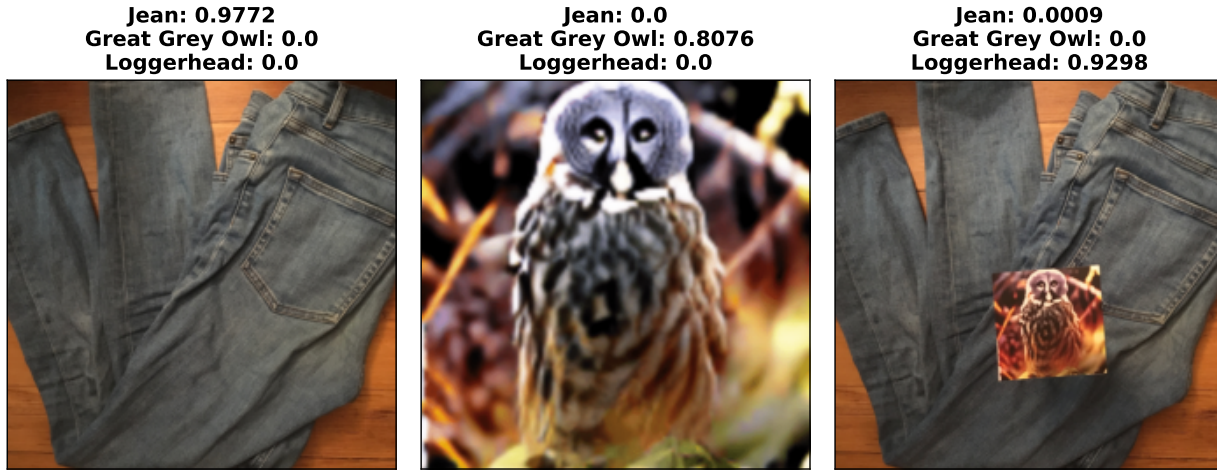


Figure 1. An example of an interpretable, universal, and physically realizable feature-fool attack against a ResNet50. The patch depicts an owl, however, when printed, physically inserted into a scene with jeans, and photographed, it causes a misclassification as a loggerhead turtle. The patch was created by perturbing the latent of a generator to manipulate the image in a feature-space and training with a loss that jointly optimizes for fooling the classifier and resembling some non-target disguise class.

related misclassification. The following sections contain background, methods, experiments, and discussion. Appendix A.7 has a jargon-free summary. Code is available at https://github.com/thestephencasper/feature_fool.

2. Related Work

Conventional adversaries (Szegedy et al., 2013; Goodfellow et al., 2014) tend to be non-interpretable pixel-level perturbations with limited ability to transfer to the physical world. Here, we contextualize our approach with other work and natural examples related to overcoming these challenges.

Inspiration from Nature: Mimicry is common in nature, and sometimes, rather than holistically imitating another species’ appearance, a mimic will only exhibit particular features. For example, many animals use adversarial eyespots to confuse predators (Stevens & Ruxton, 2014). Another example is the mimic octopus which imitates the patterning, but not the shape, of a banded sea snake. We show in Figure 2 using a photo of a mimic octopus from Norman et al. (2001) that a ResNet50 classifies it as a sea snake.

Generative Modeling: In contrast to pixel-space attacks, our method uses a generator to manipulate images in feature-space. One similar approach has been to train a generator or autoencoder to produce adversarial perturbations that are subsequently applied to natural inputs. This has been done by Hayes & Danezis (2018); Mopuri et al. (2018a;b); Poursaeed et al. (2018); Xiao et al. (2018); Hashemi et al. (2020); Wong & Kolter (2020) to synthesize attacks that are transferable, universal, or efficient to produce. Unlike these, however, we also explicitly focus on physical-realizability

and human-interpretability. Additionally, rather than training an adversary generator, ours and other related works have skipped this step and simply trained adversarial latent perturbations to pretrained models. Liu et al. (2018) did this with a differentiable image renderer. Others (Song et al., 2018; Joshi et al., 2019) have used deep generative networks, and Wang et al. (2020) aimed to create more semantically-understandable attacks by training an autoencoder with a “disentangled” embedding space. However, these works focus on small classifiers trained on simple datasets (MNIST (LeCun et al., 2010), SVHN (Netzer et al., 2011), CelebA (Liu et al., 2015) and BDD (Yu et al., 2018)). In contrast, we work at the ImageNet (Russakovsky et al., 2015) scale. Hu et al. (2021) also do this, but only with adversarial patches, while we use three types of attacks discussed in the following section. Finally, compared to all of these works, ours is also unique in that we regularize adversaries for interpretability and disguise with our training objective rather than relying on a generator alone.

Physically-Realizable Attacks: Our first contribution related to physical realizability is human-interpretable adversaries that fool a classifier when printed and photographed. This directly relates to the work of Kurakin et al. (2016) who found that conventional pixel-space adversaries could do this to a limited extent in controlled settings. More recently, Sharif et al. (2016); Brown et al. (2017); Eykholt et al. (2018); Athalye et al. (2018); Liu et al. (2019); Kong et al. (2020); Komkov & Petiushko (2021) used optimization under transformation to create adversarial clothing, stickers, patches, or objects. In contrast to each of these, we generate attacks that are not only physically-realizable but also inconspicuous in the sense that they are both human-interpretable



Figure 2. Adversarial features in nature. (a) A peacock and butterfly with adversarial eyespots (unfortunately, peacock and ringlet butterfly are ImageNet classes, so one cannot meaningfully test how ImageNet networks might be misled by them). (b) A mimic octopus from Norman et al. (2001) is classified as a sea snake by a ResNet50.

and disguised. Our second contribution related to physical realizability is “copy-paste” attacks discussed next.

Interpretable Adversaries: In addition to fooling models, our adversaries provide a method for discovering semantically-describable feature/class associations learned by a network. This relates to work by Geirhos et al. (2018) and Leclerc et al. (2021) who debug networks by searching over features, transformations, and textural changes that cause misclassification. More similar to our work are Carter et al. (2019) and Mu & Andreas (2020) who develop interpretations of networks using feature visualization (Olah et al., 2017) and network dissection (Bau et al., 2017) respectively. Both find cases in which their interpretations suggest a “copy/paste” attack in which a natural image of one object is pasted inside another to cause an unrelated misclassification. We add to this work with a new method to identify such adversarial features, and unlike either previous approach, ours does so in a context-conditional fashion.

3. Methods

3.1. Threat Model

We adopt the “unrestricted” adversary paradigm of Song et al. (2018), under which an attack is successful if the network’s classification differs from that of an oracle (e.g., a human). The adversary’s goal is to produce a feature that causes a targeted misclassification, does not resemble the target class to a human, is universally effective across a distribution of source images, and is physically-realizable. The adversary can only change a certain portion of either the latent or the image depending on the type of attack we use. We assume that the adversary has access to a differentiable image generator, a corresponding discriminator, and an auxiliary classifier. We use white-box access to the target network, though we present black-box attacks based on transfer from an ensemble in Appendix A.2.

3.2. Training Process

Our attacks involve manipulating the latent representation in a generator layer to produce an adversarial feature. Fig. 3 depicts our approach. We test three attacks, *patch*, *region* and *generalized patch* (with a fourth in Appendix A.1). We find that patch attacks are generally the most successful.

Patch: We use the generator to produce a square patch that is inserted in a natural image.

Region: Starting with some generated image, we randomly select a square portion of the latent representation in a layer of the generator spanning the channel dimension but not the height or width dimensions and replace it with a learned insertion. This is analogous to a patch in the image’s pixel representation. The modified latent is then passed through the rest of the generator, producing the adversarial image.

Generalized Patch: These patches can be of any shape, hence the name “generalized” patch. First, we generate a region attack. Second, we extract a generalized patch from it. We do this by (1) taking the absolute-valued pixel-level difference between the original and adversarial image, (2) applying a Gaussian filter for smoothing, and (3) 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.

Objective: 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 as an interpretable, inconspicuous feature:

$$\arg \min_{\delta} \mathbb{E}_{x \sim \mathcal{X}, t \sim \mathcal{T}, l \sim \mathcal{L}}$$

$$[L_{x\text{-ent}}(C(A(x, \delta, t, l)), y_{\text{targ}}) + L_{\text{reg}}(A(x, \delta, t, l))]$$

with \mathcal{X} a distribution over images (e.g., a dataset or generation distribution), \mathcal{T} a distribution over transformations, \mathcal{L}

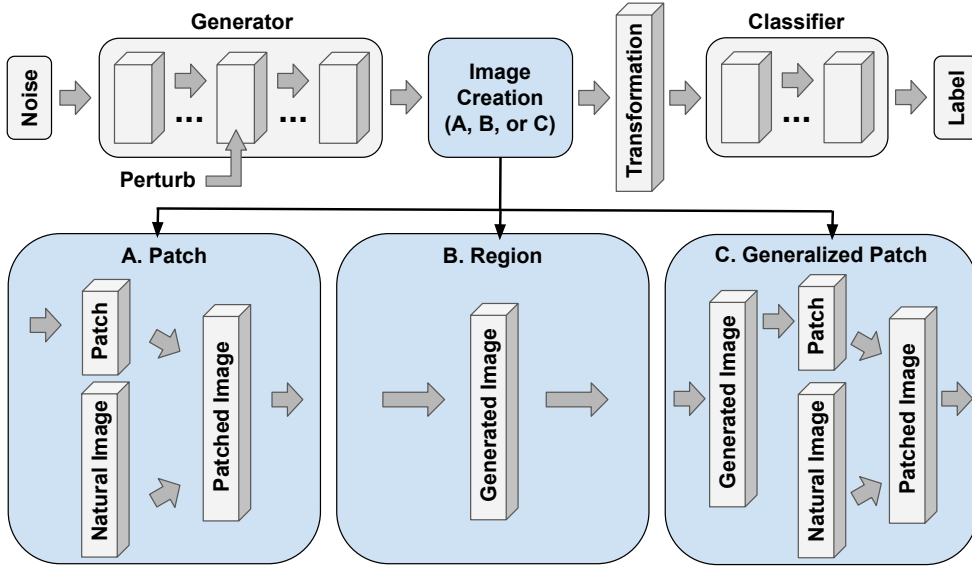


Figure 3. Our fully-differentiable pipeline for creating *patch*, *region*, and *generalized patch* attacks.

a distribution over insertion locations (this only applies for patches and generalized patches), C the target classifier, A an image-generating function, $L_{x\text{-ent}}$ a targeted crossentropy loss for fooling the classifier, y_{targ} the target class, and L_{reg} a regularization loss for interpretability and disguise.

L_{reg} contains several terms. Our goal is to produce features that are interpretable and disguised to a human, but absent the ability to scalably or differentially have a human in the loop, we instead use L_{reg} as a proxy. L_{reg} includes three terms calculated using a discriminator and an auxiliary classifier. For all attacks, we differentially resize the patch or the extracted generalized patch and pass it through the discriminator and auxiliary classifier. We then add weighted terms to the regularization loss based on (1) the discriminator’s (D) logistic loss for classifying the input as fake, (2) the softmax entropy of a classifier’s (C') output, and (3) the negative of the classifier’s crossentropy loss for classifying the input as the attack’s target class. Thus, we have:

$$L_{\text{reg}}(a) = L_{\text{logistic}}[D(P(a))] \\ + H[C'(P(a))] - L_{x\text{-ent}}[C'(P(a), y_{\text{targ}})]$$

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

4. Experiments

4.1. Attack Details

We use BigGAN generators from Brock et al. (2018); Wolf (2018), and perturb the post-ReLU activations of the internal ‘GenBlocks’. Due to self-attention in the generator, for region attacks, the change to the output image is not square even though the perturbation to the latent is. By default, we attacked a ResNet50 (He et al., 2016), restricting patch attacks to 1/16 of the image and region and generalized patch attacks to 1/8. We found that performing our crossentropy and entropy regularization using adversarially-trained auxiliary classifiers produced subjectively more interpretable results. This aligns with findings that adversarially-trained networks tend to learn more interpretable representations (Engstrom et al., 2019b; Salman et al., 2020) and better approximate the human visual system (Dapello et al., 2020). Thus, for crossentropy and entropy regularization, we used an $\epsilon = 4 L_2$ and $\epsilon = 3 L_\infty$ robust ResNet50s from Engstrom et al. (2019a). For discriminator regularization, we use the BigGAN class-conditional discriminator with a uniform class vector input (as opposed to a one-hot vector). For patch adversaries, we train under colorjitter, Gaussian blur, Gaussian noise, random rotation, and random perspective transformations to simulate real world perception. For region and generalized patch ones, we only use Gaussian blurring and horizontal flipping. Also for region and generalized patch adversaries, we promote subtlety by penalizing the difference from the original image using the LPIPs per-

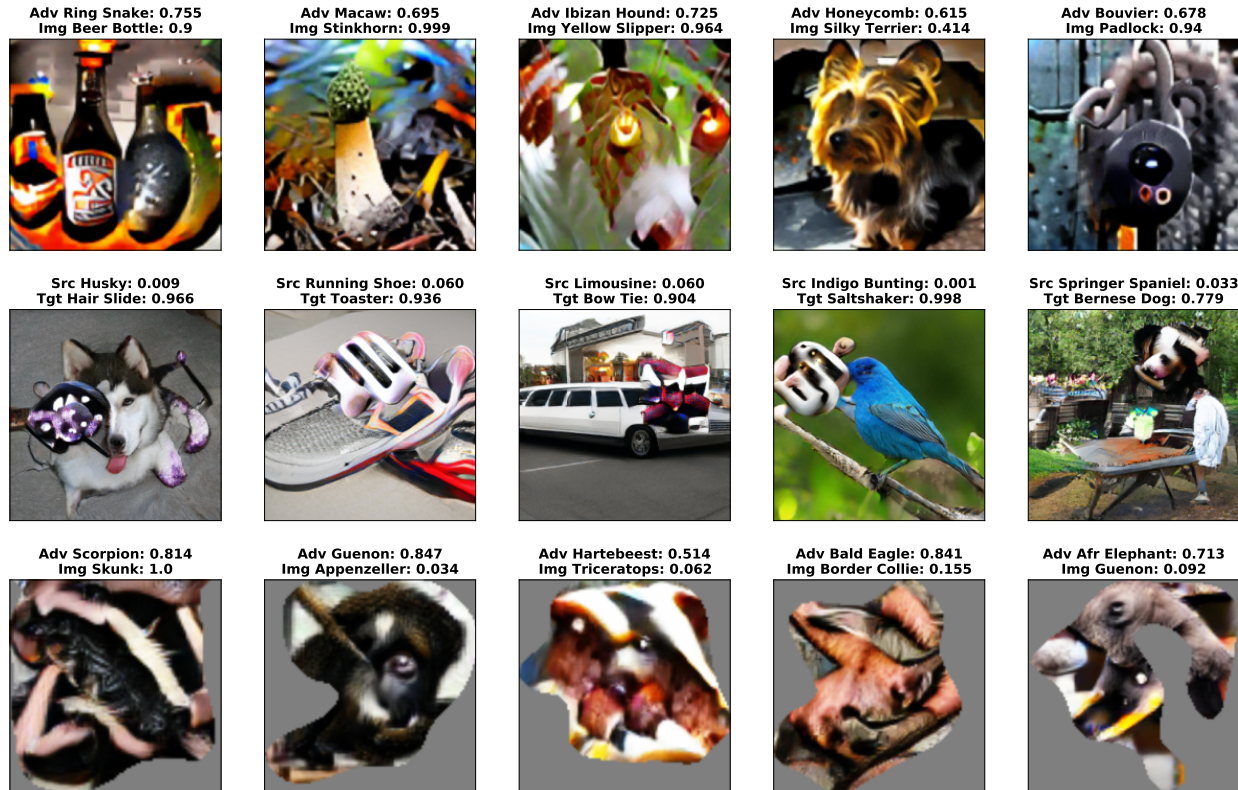


Figure 4. Adversarial features from universal patch (top), region (middle), and generalized patch (bottom) feature-fool attacks. Each patch and generalized patch is labeled with its mean fooling confidence under random insertion in source images (labeled ‘Adv’) and the confidence for the disguise class (labeled ‘Img’). Target and disguise confidences in each patch and generalized patch subfigure title come from different inputs, so they can add to > 1 . Region attacks are labeled with their source (‘Src’) and target (‘Tgt’) class confidences.

ceptual distance (Zhang et al., 2018; So & Durnopianov, 2019). Finally, for all adversaries, we apply a penalty on the total variation of the patch or change induced on the image to discourage high-frequency patterns. All experiments were implemented with PyTorch (Paszke et al., 2019).

Figure 4 shows examples of universal feature-level patch (top), region (middle), and generalized patch (bottom) attacks. In particular, the patches on the top row are effective at resembling a disguise distinct from the target class. However, when inserted as a patch into another image, the network sees them as the target. To the extent that humans also find these patches to resemble the target, this may suggest similar properties in visual cortex. However, it is key to note framing effects when analyzing these images: recognizing target-class features unconditionally and given the class are different tasks (Hullman & Diakopoulos, 2011).

4.2. Interpretable, Universal, Physically-Realizable Attacks

To demonstrate that feature-fool adversaries are interpretable and versatile, we generate adversarial patches

which appear as one object to a human, cause a targeted misclassification by the network as another, do so universally regardless of the source image, and are physically-realizable. We generated feature-fool patches using our approach and compared them to five alternatives, four of which were single-ablation tests in which no generator was used in favor of optimization in pixel space and in which each of the three regularization terms discussed in Section 3.2 were omitted. In the final, we omitted the generator and all three of the regularization terms, resulting in the same method as Brown et al. (2017). For each test, all else was kept identical including training under transformation and initializing the patch as an output from the generator. This initialization allowed for the pixel-space controls to be disguised and was the same as the method for generating disguised pixel-space patch attacks in Brown et al. (2017).

In Silico: Before testing in the physical world, we did so in silico with 250 universal attacks of each type with random target classes. Fig. 5 plots the results. On the x axis are target class fooling confidences. On the y axis are the labeling confidences from an Inception-v3 (Szegedy et al., 2016) which we use as a proxy for human interpretability. It was

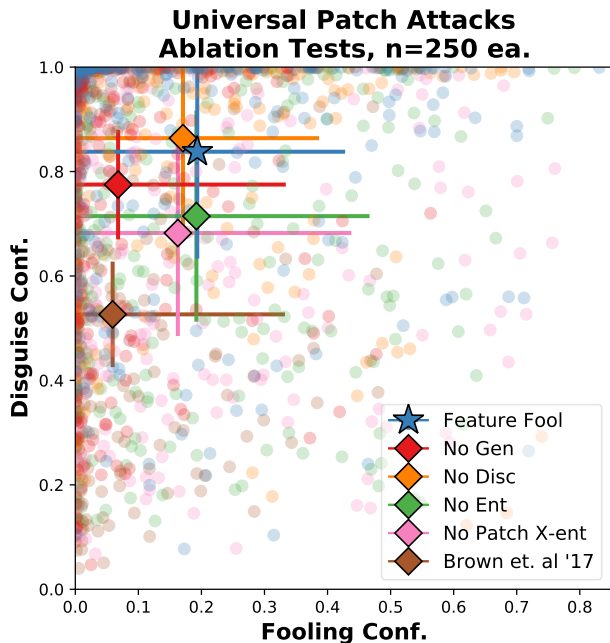


Figure 5. Feature-fool patch attacks compared to ablation tests. Fooling conf. gives target class confidence. Disguise conf. shows an Inception-v3’s label class confidence for the patch. Attacks further up and right are better. Centroids are shown with error bars giving the standard deviation.

not involved in regularization. For all tests, these universal attacks have variable success for both fooling and disguise due in large part to the random selection of target class and random insertion of the patches. Centroids are shown with standard deviations and suggest that our feature-fools are equal or better at fooling and appearing interpretable on average than all others except for the controls without discriminator regularization which seem to trade fooling confidence for disguise confidence.

Furthermore, see Appendix A.6 for examples of feature-fool and Brown et al. controls which used optimization in pixel space. Because they were initialized from generator outputs, some of the Brown et al. controls have a veneer-like resemblance to non-target class features. Nonetheless, they contain higher-frequency patterns and less coherent objects in comparison to ours.

In the Physical World: Next, we compared the physical realizability of our attacks with a control method Brown et al. (2017) which used optimization in pixel space. We generated 100 additional adversarial patches for each, selected the 10 with the best mean fooling confidence, printed them, and photographed them next to 9 different ImageNet

classes of common household items.² We confirmed that photographs of each object with no patch were correctly classified and analyzed the outputs of the classifier when the adversarial patches were added in the physical scene.

Figure 6 shows successful examples of these physically-realizable feature-fool and Brown et al. controls which used optimization in pixel space. Meanwhile, resizable and printable versions of all 10 feature-fool and Brown et al. controls which used optimization in pixel space are in Appendix A.6. The mean and standard deviation of the fooling confidence for the feature-fool attacks in the physical world were 0.312 and 0.318 respectively ($n = 90$) while for the Brown et al. controls which used optimization in pixel space, they were 0.474 and 0.423 ($n = 90$). We do not attempt any hypothesis tests due to nonindependence between the results across classes due to the same set of patches being used for each. These tests in the physical world show that the feature-fool attacks were often effective but that there is high variability. The comparisons to Brown et al. controls provide some evidence that, unlike with our results in silico, the feature-fool attacks may be less reliably successful in the real world than the controls. Nonetheless, the overall level of fooling success between both groups was comparable in these tests.

4.3. Interpretability and Copy/Paste Attacks

Using adversarial examples to better interpret networks has been proposed by Dong et al. (2017) and Tomsett et al. (2018). Unlike conventional adversaries, ours can be used to understand how networks may respond to human-comprehensible features. Here, our end goal is not to produce a disguised attack but rather to identify spurious feature-class associations, so we omit the regularization terms from Section 3.2. We find that inspecting the resulting adversaries suggests both useful and harmful feature-class associations. In the Appendix, Fig. 10 provides a simple example of each.

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 the attacks in Section 4.2 because the features pasted into an image must be natural objects rather than ones whose synthesis can be controlled. 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, feature insertions into pornographic images have been used to evade NSFW content detectors (Yuan et al., 2019).

To develop copy/paste attacks, we select a source and target class, develop class-universal adversarial features, and

²Backpack, banana, bath towel, lemon, jeans, spatula, sunglasses, toilet tissue, and toaster.



Figure 6. Successful examples of universal, physically-realizable feature-fool attacks (top) and Brown et al. attacks (bottom). See Appendix A.6 for full-sized versions of the patches.

manually analyze them for motifs that resemble natural objects. Then we paste images of these objects into natural images and pass them through the classifier. Two other works have previously developed copy/paste attacks, also via interpretability tools: Carter et al. (2019) and Mu & Andreas (2020). However, our technique has a unique advantage for producing germane fooling features. Rather than simply producing features associated with the target class, these adversaries generate fooling features *conditional* on the distribution over source images (i.e. the source class) with which the adversaries are trained. This allows any source/target classes to be selected, but we find the clearest success in generating copy/paste attacks when they are somewhat related (e.g., bee and fly).

Fig. 7 gives three examples. We show two example images for each of the patch, region, and generalized patch adversaries. Below these are the copy/paste adversaries with average target class confidence before and after feature insertion for the 6 (out of 50) images for the source class in the ImageNet validation set for which the insertion resulted in the highest target confidence. Overall, little work has been done on copy/paste adversaries, and thus far, methods have always involved a human in the loop. This makes objective comparisons difficult. However we provide examples of a feature-visualization based method inspired by Carter et al. (2019) in Appendix A.4 to compare with ours.

5. Discussion and Broader Impact

By using a generative model to synthesize adversarial features, we contribute to a more practical understanding

of deep networks and their vulnerabilities. As an attack method, our approach is versatile. Across experiments here and in the Appendix, we show that it can be used to produce targeted, interpretable, disguised, universal, physically-realizable, black-box, and copy/paste attacks at the ImageNet level. To the best of our knowledge, we are the first to introduce a method with *all* of these capabilities. As an interpretability method, this approach is also effective as a targeted means of searching for adversarially exploitable feature-class associations.

Conventional adversaries reveal intriguing properties of the learned representations in deep neural networks. However, as a means of attacking real systems, they pose limited threats outside of the digital domain (Kurakin et al., 2016). Given feature-fool attacks, copy/paste attacks, and related work, a focus on adversarial features and robust, physically-realizable attacks will be key to understanding practical threats. Importantly, even if a deep network is adversarially trained to be robust to one class of perturbations, this does not guarantee robustness to others that may be used to attack it in deployment. Consequently, we argue for pragmatic threat models, the use of more robust networks (e.g., Engstrom et al. (2019a); Dapello et al. (2020)), and exercising caution with deep networks in the real world. As a promising sign, we show in Appendix A.5 that adversarial training is useful against feature-fool attacks.

A limitation is that when more constraints are applied to the adversarial generation process (e.g., universality + physical-realizability + disguise), attacks are generally less successful, and more screening is required to find good ones. They also take more time to generate which could be a bottle-



Figure 7. Patch, region, and generalized patch adversaries being used to guide three class-universal copy/paste adversarial attacks. Patch adversary example pairs are on the left, region adversaries in the middle, and generalized patch adversaries on the right of each odd row. Six successful attack examples are on each even row.

neck for adversarial training. Further still, while we develop disguised adversarial features, we find that they often have somewhat unnatural, suspicious forms typical of generated images. In this sense, our disguised attacks may nonetheless be detectable by a human. Ultimately, this type of attack is limited by the efficiency and quality of the generator.

Future work should leverage advances in generative modeling. One possibly useful technique could be to develop fooling features adversarially against a discriminator which is trained specifically to recognize them from natural features. We also believe that studying human responses to feature-level adversaries and the links between interpretable

representations, robustness, and similarity to the primate visual system (Dapello et al., 2020) are promising directions for better understanding both networks and biological brains. Robustness against only certain aspects of attacks (e.g., feature-based, universal, physical-realizable) will not be sufficient to develop broadly-reliable networks, so studying attacks like ours which have many such properties seems promising. Ultimately, given that each of the 11 proposals for building safe, advanced AI outlined in Hubinger (2020) directly call for interpretability tools and/or adversarial robustness, we argue that judiciously continuing this type of work will be valuable for safe AI.

Acknowledgments

We thank Dylan Hadfield-Menell, Cassidy Laidlaw, Miles Turpin, Will Xiao, and Alexander Davies for insightful discussions and feedback. This work was conducted in part with funding from the Harvard Undergraduate office of Research and Fellowships.

References

- Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. In *International conference on machine learning*, pp. 284–293. PMLR, 2018.
- Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.
- Bhattad, A., Chong, M. J., Liang, K., Li, B., and Forsyth, D. A. Unrestricted adversarial examples via semantic manipulation. *arXiv preprint arXiv:1904.06347*, 2019.
- Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.
- Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.
- Carter, S., Armstrong, Z., Schubert, L., Johnson, I., and Olah, C. Activation atlas. *Distill*, 4(3):e15, 2019.
- Dapello, J., Marques, T., Schrimpf, M., Geiger, F., Cox, D. D., and DiCarlo, J. J. Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations. *BioRxiv*, 2020.
- Dong, Y., Su, H., Zhu, J., and Bao, F. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv preprint arXiv:1708.05493*, 2017.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., and Tsipras, D. Robustness (python library), 2019a. URL <https://github.com/MadryLab/robustness>.
- Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Tran, B., and Madry, A. Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*, 2019b.
- Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1625–1634, 2018.
- Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.
- Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- Hashemi, A. S., Bär, A., Mozaffari, S., and Fingscheidt, T. Transferable universal adversarial perturbations using generative models. *arXiv preprint arXiv:2010.14919*, 2020.
- Hayes, J. and Danezis, G. Learning universal adversarial perturbations with generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 43–49. IEEE, 2018.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Hu, Y.-C.-T., Kung, B.-H., Tan, D. S., Chen, J.-C., Hua, K.-L., and Cheng, W.-H. Naturalistic physical adversarial patch for object detectors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7848–7857, 2021.
- Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.
- Hubinger, E. An overview of 11 proposals for building safe advanced ai. *arXiv preprint arXiv:2012.07532*, 2020.
- Hullman, J. and Diakopoulos, N. Visualization rhetoric: Framing effects in narrative visualization. *IEEE transactions on visualization and computer graphics*, 17(12): 2231–2240, 2011.
- Joshi, A., Mukherjee, A., Sarkar, S., and Hegde, C. Semantic adversarial attacks: Parametric transformations that fool deep classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4773–4783, 2019.

- Kiat, L. Lucent. <https://github.com/greentfrapp/lucent>, 2019.
- Komkov, S. and Petiushko, A. Advhat: Real-world adversarial attack on arcface face id system. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 819–826. IEEE, 2021.
- Kong, Z., Guo, J., Li, A., and Liu, C. Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14254–14263, 2020.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.
- Kurakin, A., Goodfellow, I., Bengio, S., et al. Adversarial examples in the physical world, 2016.
- Leclerc, G., Salman, H., Ilyas, A., Vemprala, S., Engstrom, L., Vineet, V., Xiao, K., Zhang, P., Santurkar, S., Yang, G., et al. 3db: A framework for debugging computer vision models. *arXiv preprint arXiv:2106.03805*, 2021.
- LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2, 2010.
- Liu, A., Liu, X., Fan, J., Ma, Y., Zhang, A., Xie, H., and Tao, D. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1028–1035, 2019.
- Liu, H.-T. D., Tao, M., Li, C.-L., Nowrouzezahrai, D., and Jacobson, A. Beyond pixel norm-balls: Parametric adversaries using an analytically differentiable renderer. *arXiv preprint arXiv:1808.02651*, 2018.
- Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- Melas, L. Pytorch-pretrained-vit. <https://github.com/lukemelas/PyTorch-Pretrained-ViT>, 2020.
- Mopuri, K. R., Ojha, U., Garg, U., and Babu, R. V. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 742–751, 2018a.
- Mopuri, K. R., Uppala, P. K., and Babu, R. V. Ask, acquire, and attack: Data-free uap generation using class impressions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–34, 2018b.
- Mu, J. and Andreas, J. Compositional explanations of neurons. *arXiv preprint arXiv:2006.14032*, 2020.
- Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.
- Norman, M. D., Finn, J., and Tregenza, T. Dynamic mimicry in an indo-malayan octopus. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268 (1478):1755–1758, 2001.
- Olah, C., Mordvintsev, A., and Schubert, L. Feature visualization. *Distill*, 2(11):e7, 2017.
- Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance.pdf>.
- Poursaeed, O., Katsman, I., Gao, B., and Belongie, S. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4422–4431, 2018.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252, 2015.
- Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do adversarially robust imagenet models transfer better? In *ArXiv preprint arXiv:2007.08489*, 2020.
- Sharif, M., Bhagavatula, S., Bauer, L., and Reiter, M. K. 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*, pp. 1528–1540, 2016.
- Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

- So, U. and Durnopianov, I. Lpips pytorch. <https://github.com/S-aiueo32/lpips-pytorch>, 2019. *on computer vision and pattern recognition*, pp. 586–595, 2018.
- Song, Y., Shu, R., Kushman, N., and Ermon, S. Constructing unrestricted adversarial examples with generative models. *arXiv preprint arXiv:1805.07894*, 2018.
- Stevens, M. and Ruxton, G. D. Do animal eyespots really mimic eyes? *Current Zoology*, 60(1), 2014.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- Tomsett, R., Widdicombe, A., Xing, T., Chakraborty, S., Julier, S., Gurram, P., Rao, R., and Srivastava, M. Why the failure? how adversarial examples can provide insights for interpretable machine learning. In *2018 21st International Conference on Information Fusion (FUSION)*, pp. 838–845. IEEE, 2018.
- Wang, S., Chen, S., Chen, T., Nepal, S., Rudolph, C., and Grobler, M. Generating semantic adversarial examples via feature manipulation. *arXiv preprint arXiv:2001.02297*, 2020.
- Wolf, T. Pytorch pretrained biggan. <https://github.com/huggingface/pytorch-pretrained-BigGAN>, 2018.
- Wong, E. and Kolter, J. Z. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020.
- Xiao, C., Li, B., Zhu, J.-Y., He, W., Liu, M., and Song, D. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.
- Yu, F., Xian, W., Chen, Y., Liu, F., Liao, M., Madhavan, V., and Darrell, T. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.
- Yuan, K., Tang, D., Liao, X., Wang, X., Feng, X., Chen, Y., Sun, M., Lu, H., and Zhang, K. Stealthy porn: Understanding real-world adversarial images for illicit online promotion. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 952–966. IEEE, 2019.
- Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference*



Figure 8. Examples of original images (top) alongside class-universal channel adversaries (bottom). Each image is labeled with the source and target class confidence.

A. Appendix

A.1. Channel Attacks

In contrast to the region attacks presented in the main paper, we also experiment with *channel* attacks. For region attacks, we optimize an insertion to the latent activations of a generator’s layer which spans the channel dimension but not the height and width. This is analogous to a patch attack in pixel-space. For channel attacks, we optimize an insertion which spans the height and width dimensions but only involves a certain proportion of the channels. This is analogous to an attack that only modifies the R, G, or B channel of an image in pixel-space. Unlike the attacks in Section 4.1, we found that it was difficult to create universal channel attacks (single-image attacks, however, were very easy). Instead, we relaxed this goal and created class-universal ones which are meant to cause any generated example from one random source class to be misclassified as a target. We also manipulate $1/4^{\text{th}}$ of the latent instead of $1/8^{\text{th}}$ as we do for region attacks. Mean fooling rates and examples from the top 5 attacks out of 16 are shown in Fig. 8. They induce textural changes somewhat like adversaries crafted by Geirhos et al. (2018) and Bhattad et al. (2019)).

A.2. Black-Box Attacks

Adversaries are often created using first order optimization on an input to a network which requires that the network’s parameters are known. However, adversaries are often transferrable between models (Papernot et al., 2016), and one method for developing black-box attacks is to train against a different model and then transfer to the intended target. We do this for our adversarial patches and generalized patches

by attacking a large ensemble of AlexNet (Krizhevsky et al., 2012), VGG19 (Simonyan & Zisserman, 2014), Inception-v3 (Szegedy et al., 2016), DenseNet121 (Huang et al., 2017), ViT(Dosovitskiy et al., 2020; Melas, 2020), and two robust ResNet50s (Engstrom et al., 2019a), and then transferring to ResNet50 (He et al., 2016). Otherwise, these attacks were identical to the ones in Section 4.1 including a random source/target class and optimization for disguise. Many were unsuccessful, but a sizable fraction were able to fool the ResNet50 with a mean confidence of over 0.1 for randomly sampled images. The top 5 out of 64 of these attacks for patch and generalized patch adversaries are shown in Fig. 9.

A.3. Discovering Feature-Class Associations

Fig. 10 shows two simple examples of using feature-fool attacks to identify feature-class associations. It shows one positive example in which the barbershop class is desirably associated with barber-pole-stripe-like features and one negative example in which the bikini class is undesirably associated with caucasian-colored skin. Notably though, the ability to identify feature-class associations such as this is not a unique capability of feature-fool attacks and could also be achieved with feature visualization (Olah et al., 2017).

A.4. Copy/Paste Attacks with Class Impressions

Mu & Andreas (2020) and Carter et al. (2019) both used interpretability methods to guide the development of copy/paste adversaries. Mu & Andreas (2020), used network dissection (Bau et al., 2017) to develop interpretations of neurons and fit a semantic description in compositional logic over the network using weight magnitudes. This allowed



Figure 9. Black-box adversarial patches (top) and generalized patches (bottom) created using transfer from an ensemble. Patches are displayed alongside their target class and mean fooling confidence.

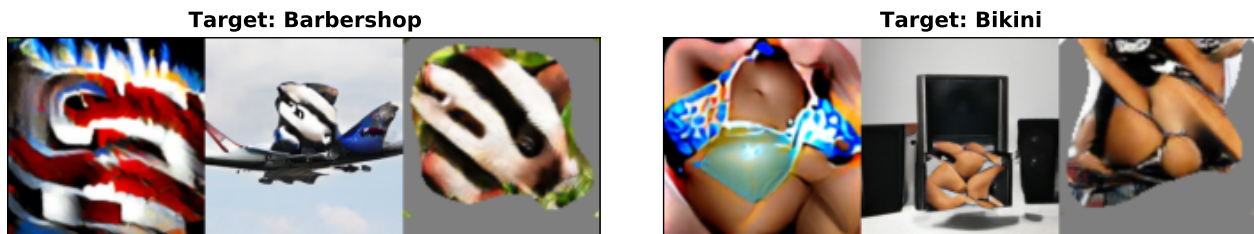


Figure 10. Examples of good and bad features class associations in which barber pole stripes are associated with a barbershop and caucasian-colored skin is associated with a bikini. Patch (left), region (middle), and generalized patch adversaries (right) are shown.

them to identify cases in which the networks learned undesirable feature-class associations. However, this approach cannot be used to make a targeted search for copy/paste attacks that will cause a given source class to be misclassified as a given target.

More similar to our work is Carter et al. (2019) who found inspiration for successful copy/paste adversaries by creating a dataset of visual features and comparing the differences between ones which the network assigned the source versus target label. We use inspiration from this approach to create a baseline against which to compare our method of designing copy/paste attacks in Section 4.3. Given a source and target class such as a bee and a fly, we optimize a set of inputs to a network for each class in order to maximize the activation of the output node for that class. Mopuri et al. (2018b) refers to these as *class impressions*. We train these inputs under transformations and with a decorrelated frequency-space parameterization of the input pixels using the Lucent (Kiat, 2019) package. We do this for the same

three class pairs as in Fig. 7 and display 6 per class in Fig. 11. In each subfigure, the top row gives class impressions of the source class, and the bottom gives them for the target. Each class impression is labeled with the network’s confidences for the source and target class. In analyzing these images and comparing to the ones from Fig. 11, we find no evidence of more blue coloration in the African elephant class impressions than the Indian ones. However, we find it plausible that some of the features in the fly class impression may resemble traffic lights and that those for the Lionfish may resemble an admiral Butterfly’s wings. Nonetheless, these visualizations are certainly different in appearance from our adversarial ones.

These class impressions seem comparable but nonredundant with our adversarial method from Section 4.3. However, our adversarial approach may have an advantage over the use of class impressions in that it is equipped to design features that look like the target class *conditional* on the rest of the image being of the source class. Contrastingly, a class

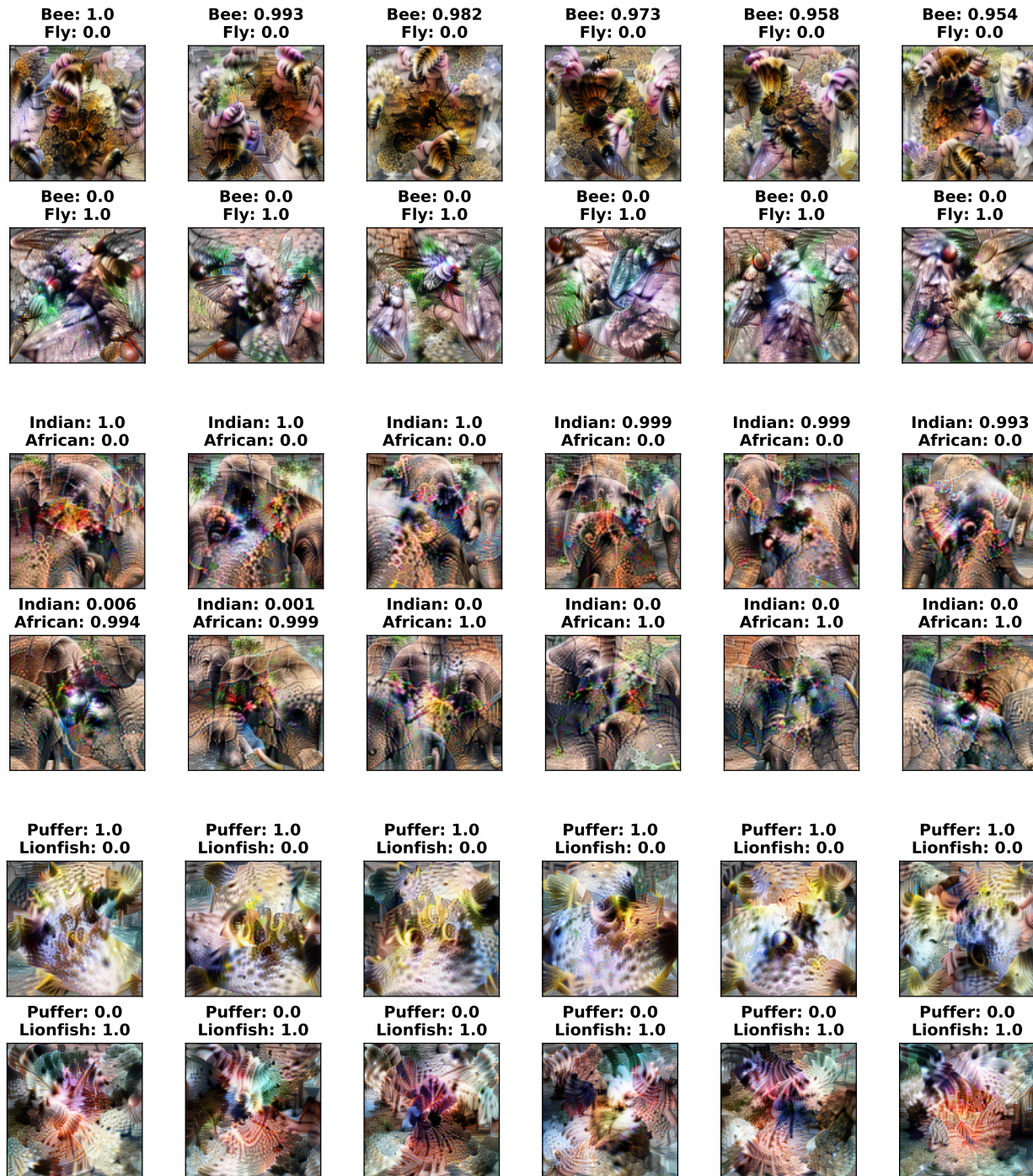


Figure 11. Class impressions for three pairs of classes. These could be used for providing insight about copy/paste attacks in the same way as the examples from Fig. 7. Each subfigure is labeled with the network’s output confidence for both classes.

impression is only meant to visualize features typical of the target class. It is possible that this is why our adversarial attacks were able to show that inserting a blue object into an image of an Indian elephant can cause a misclassification as an African elephant – two very similar classes – while

the class impressions for the two appear very similar and suggest nothing of the sort.

A.5. Defense via Adversarial Training

Adversarial training is a common and broadly effective means for improving robustness. Here, to test how effective it is for our attacks, for 5 pairs of similar classes, we generate datasets of 1024 images evenly split between each class and images with/without adversarial perturbations. We do this separately for channel, region, and patch adversaries before treating the victim network as a binary classifier and training on the examples. We report the post-training minus pre-training accuracies in Tbl. A.5 and find that across the class pairs and attack methods, the adversarial training improves binary classification accuracy by a mean of 42%.

	Channel	Region	Patch	Mean
Great White/Grey Whale	0.49	0.29	0.38	0.39
Alligator/Crocodile	0.13	0.29	0.60	0.34
Lion/Tiger	0.29	0.28	0.63	0.40
Frying Pan/Wok	0.32	0.39	0.68	0.47
Scuba Diver/Snorkel	0.42	0.36	0.69	0.49
Mean	0.33	0.32	0.60	0.42

Table 1. Binary classification accuracy improvements from adversarial training for channel, region, and patch adversaries across 5 class pairs.

A.6. Resizable, Printable Patches

See Figs. 12 and 13 for feature-fool and control adversarial images respectively. We encourage readers to experiment with these images (which were optimized to fool a ResNet50) or with their own which can be created using our provided code. In doing so, one might find a mobile app to be convenient. We used Photo Classifier.³

A.7. Jargon-Free Summary

AI and related fields are making rapid progress, but there exists a communication gap between researchers and the public which too-often serves as a barrier to the spread of information outside the field. For readers who may not know all of the field’s technical concepts and jargon, we provide a more readable summary here.

Historically, it has proven difficult to write conventional computer programs that accurately classify real-world images. But this task has seen revolutionary success by neural networks which can now classify images into hundreds or thousands of categories, sometimes with higher accuracy than humans. Despite the impressive performance, we still don’t fully understand the features that these networks use to classify images, and we cannot be confident that they will always do so correctly. In fact, past research has demonstrated that it is usually very easy to take an image that the network classifies correctly and perturb its pixel values by a

tiny amount – often imperceptibly to a human – in such a way that the network will misclassify it with high confidence as whatever target class the attacker desires. For example, we can take a cat, make minute changes in a few pixels, and make the network believe that it is a dog. Researchers have also discovered perturbations that can be added onto a wide range of images to cause them to be misclassified, making those perturbations “universal”. In general, this process of designing an image that the network will misclassify is called an “adversarial attack” on the network.

Unfortunately, conventional adversarial attacks tend to produce perturbations that are not interpretable. To a human, they usually just appear as pixelated noise (when exaggerated to be visible). As a result, they do not help us to understand how networks will process sensible inputs, and they do not reveal weaknesses that could be exploited by adversarial features in the real world. To make progress toward solving these problems, we focus on developing adversarial features that are interpretable (i.e., appear comprehensible to a human). In one sense, this is not a new idea. Quite the opposite, in fact – there are already examples in the animal kingdom. Figure 2 shows examples of adversarial eyespots on a peacock and butterfly and adversarial patterns on a mimic octopus.

To generate interpretable adversarial features, we introduce a method that uses “generative” modeling. In addition to classifying images, networks can also be used to great effect for generating them. These networks are often trained with the goal of producing images that are so realistic that one cannot tell whether they came from the training set or not, and modern generation methods are moving closer to this goal. Typically, these networks take in a random input and form it into an image. Inside this process are intermediate “latent” representations of the image at each “layer” inside the generator that gradually shift from abstract, low-dimensional representations of high-level features of the image to the actual pixel values of the final image.

In order to create interpretable adversarial images, we take images created by the generator and use them for adversarial attacks. In the simplest possible procedure, one can generate an image and then modify the generator’s representations to change the generation of the image in such a way that the classifier is fooled by it. We also found that optimizing under transformations to our images (like blurring, cropping, flipping, rotating, etc.) and adding in some additional terms into our optimization objective to encourage more interpretable and better-disguised images greatly improved results. Ultimately, we produce adversarial images that differ from normal ones in higher-level, more intelligible ways than conventional adversaries.

These adversaries are useful and informative in two main ways. First, they allow us to create patches that are simulta-

³<https://apps.apple.com/us/app/photo-classifier/id1296922017>



Figure 12. Printable examples of the disguised, transformation-robust, physically-realizable feature-fool adversarial patches from Section 4.2. Patches can be resized before printing.



Figure 13. Printable examples of transformation-robust, physically-realizable pixel-space adversarial patches from Section 4.2. Patches can be resized before printing.

neously “disguised”, “physically-realizable”, and “universal” at the same time. By “disguised”, we mean that they look like one thing to a human but cause an unrelated misclassification. By “physically-realizable”, we mean that these images can be printed and physically placed in a scene with some other object, causing a photo of the scene to be misclassified. And by “universal”, we mean that these images can cause photos of a wide range of objects to be misclassified as the target class. As an example, Fig. 1 shows a patch of an owl that can be physically inserted next to any real world object (such as jeans) in order to cause a misclassification as a loggerhead turtle.

Second, these adversaries allow us to interpret networks by revealing feature-class associations they have learned. We even find that this can be used to create an additional type of attack. We show that this process can guide the creation of “copy/paste” attacks in which one natural image is pasted as a patch into another in order to cause a particular misclassification. Some of these are unexpected. For example, in Fig. 7, we find that a traffic light can make a bee look like a fly. These copy/paste attacks also have implications for physically-realizable attacks because they suggest combinations of real objects that could yield unexpected classifications.

Together, our findings offer potential for better understanding network representations and better predicting the ways that they may fail. We join others in the AI community in calling for caution and adversarial robustness when deploying networks in the real world.

A.8. Epitaph

One thing to fool them all,
One class to assign them,
One thing to see it all,
And in the real world find them.⁴

⁴Adapted from J.R.R. Tolkien’s *Ring Verse*.