LA-UR-21-21945
Approved for public release; distribution is unlimited.

Title: Adversarial Perturbations Are Not So Weird: Entanglement of Robust and Non-Robust Features in Neural Network Classifiers

Author(s): Springer, Jacob Mitchell
Mitchell, Melanie
Kenyon, Garrett

Intended for: Report

Issued: 2021-10-01 (rev.1)

**Disclaimer:**
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

arXiv:2102.05110v1 [cs.LG] 9 Feb 2021

# Adversarial Perturbations Are Not So Weird: Entanglement of Robust and Non-Robust Features in Neural Network Classifiers

**Jacob M. Springer** [1] **Melanie Mitchell** [2] **Garrett T. Kenyon** [1]

## Abstract

Neural networks trained on visual data are well-known to be vulnerable to often imperceptible adversarial perturbations. The reasons for this vulnerability are still being debated in the literature. Recently Ilyas et al. (2019) showed that this vulnerability arises, in part, because neural network classifiers rely on highly predictive but brittle "non-robust" features. In this paper we extend the work of Ilyas et al. by investigating the nature of the input patterns that give rise to these features. In particular, we hypothesize that in a neural network trained in a standard way, non-robust features respond to small, "non-semantic" patterns that are typically entangled with larger, robust patterns, known to be more human-interpretable, as opposed to solely responding to statistical artifacts in a dataset. Thus, adversarial examples can be formed via minimal perturbations to these small, entangled patterns. In addition, we demonstrate a corollary of our hypothesis: robust classifiers are more effective than standard (non-robust) ones as a source for generating transferable adversarial examples in both the untargeted and targeted settings. The results we present in this paper provide new insight into the nature of the non-robust features responsible for adversarial vulnerability of neural network classifiers.

## 1. Introduction

It is well-known that neural network classifiers trained on visual data are susceptible to adversarial examples—images that have been minimally perturbed so as to look unchanged to humans but are classified incorrectly, even though the original image is correctly classified. Many explanations have been offered for this susceptibility as well as for the transferability of adversarial examples across network architectures and even training sets; however, the ML community's understanding of these phenomena remains incomplete.

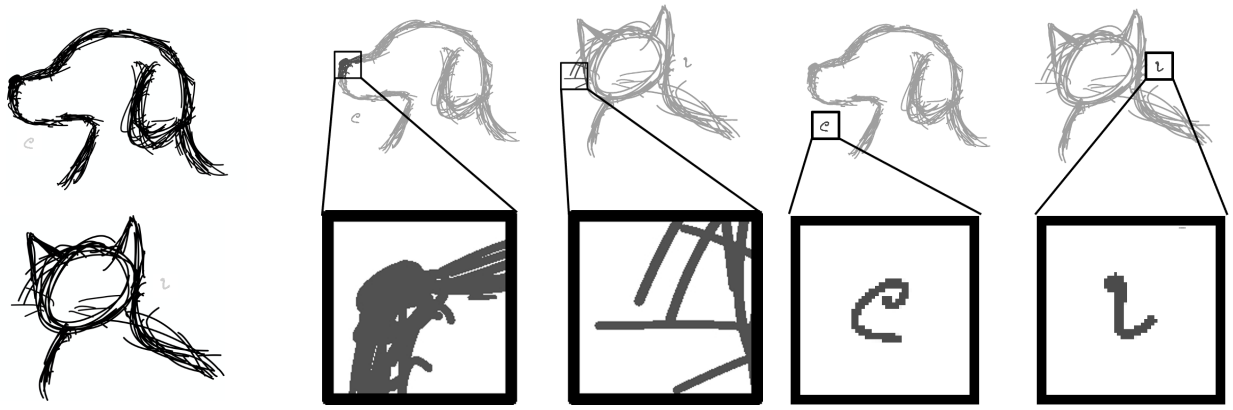
In this paper we extend the work of Ilyas et al. (2019) on understanding these phenomena as the result of highly predictive but brittle *non-robust* features that are learned by neural networks undergoing standard supervised training. Ilyas et al. proposed that adversarial examples are not "bugs" resulting from properties of a network that cause odd behavior on examples that are off the training manifold, but rather that they are due to learned "features" that respond to predictive—yet non-interpretable—patterns in the dataset. Ilyas et al. showed that such non-robust features correspond to patterns that are widespread in the data. Moreover, they showed that these non-robust features can account for much if not all of the high accuracy of neural networks on image datasets such as CIFAR-10 and ImageNet-9, and that perturbations targeting these non-robust features allow for transferable adversarial attacks.

Here, we investigate the nature of these non-robust features. In particular, we give empirical evidence that in a neural network trained in a standard way, predictive but non-robust features respond to small patterns that are typically entangled with larger, human-interpretable patterns, as opposed to responding to separate statistical artifacts in a dataset. We argue that the non-robust features of neural networks that can be exploited to construct seemingly uninterpretable adversarial perturbations in fact rely on the same underlying patterns captured by more robust features. While we use many of the experimental methods proposed by Ilyas et al. (2019) as tools, the question which we seek to answer and our results are distinct: we identify a relationship between robust and non-robust features and then demonstrate that our finding can motivate highly effective targeted transferable adversarial examples. Ilyas et al., on the other hand, consider robust and non-robust features separately.

By demonstrating that a classifier for which only non-robust features are useful can achieve well-above-chance accuracy on a test set in which only robust features are useful, we confirm that networks learn non-robust features that are closely entangled with robust features. Robust features have been shown to be substantially more interpretable than non-

[1]Los Alamos National Laboratory, Los Alamos, NM [2]Santa Fe Institute, Santa Fe, NM. Correspondence to: Jacob M. Springer <jacmspringer@gmail.com>.

Preprint.



**Type A:** Robust features. These features commonly respond to human-interpretable ("semantic") patterns (here of a dog or a cat).

**Type B:** Non-robust features that respond to small yet highly predictive patterns that by themselves appear non-semantic, yet are entangled with patterns associated with robust features (e.g., in (a)).

**Type C:** Non-robust features that respond to highly predictive patterns that are artifacts in the dataset, and are independent of robust features.

*Figure 1.* A sketch of the possible relationships between robust and non-robust features. Note that this is oversimplified as non-robust features could respond to combinations of (b) and (c).

robust features (Engstrom et al., 2019b; Kaur et al., 2019; Santurkar et al., 2019). By establishing the relationship between non-robust and robust features, we argue that non-robust features may be more interpretable than they appear through the lens of adversarial perturbations. In other words, non-robust features *may not be so weird, after all.*

The results of this paper have important implications. We explain conceptually why adversarial perturbations might appear non-semantic yet are related to semantic features. We present and test a corollary of our results: transferable examples (targeted and untargeted) can be generated more effectively by using robust classifiers rather than standard classifiers. Finally, our results provide new insights into the nature of adversarial vulnerabilities, and suggest directions of future research.

## 2. Terminology

In this section, following Ilyas et al. (2019), we define several important terms, in particular the notions of *robust* and *non-robust* features.

Deep learning classifiers, in particular convolutional neural networks (CNNs) are typically composed of a sequence of non-linear transformations called layers, the result of which is fed through a final linear classifier layer to select a class (Krizhevsky et al., 2012; LeCun et al., 1998). We refer to the penultimate layer as the *representation layer*, denoted $rep(x)$, where $x$ is the $n$ dimensional input to the network (e.g., an image). We refer to each unit of the representation layer as a *feature* that the network computes for classification. Each feature $f : \mathbb{R}^n \to \mathbb{R}$ maps the input to a real number—the feature's activation, given the input.

We adapt the following definitions from Ilyas et al. (2019), who considered the binary classification case, in which the dataset $\mathcal{D}$ consists of pairs $(x, y) \in \mathcal{D}, x \in \mathbb{R}^n, y \in \{1, -1\}$, and the final layer outputs either 1 or $-1$.

- A feature $f$ is *useful* for a dataset $\mathcal{D}$ when there exists a $\rho > 0$ such that,

$$\mathbb{E}_{(x,y)\in\mathcal{D}}[y \cdot f(x)] \geq \rho,$$

  or, more intuitively, when $f$ is correlated with the class $y$ of an input $x$.
- For a given input $x$, adversarial example $\hat{x} = x + \delta$, and $\varepsilon > 0$, $\delta$ is said to be a *permissible perturbation* when $\|\delta\|_2 < \varepsilon$.
- A feature $f$ is *robust* when, for some $\gamma > 0$,

$$\mathbb{E}_{(x,y)\in\mathcal{D}}[\min_{\|\delta\|_2\leq\varepsilon} y \cdot f(x + \delta)] \geq \gamma,$$

  More intuitively, a feature is *robust* when it is correlated with the class $y$ even under worst-case permissible perturbations of the input $x$.
- For simplicity, a feature is said to be *non-robust* when it is useful but not *robust*. (Here we do not consider non-useful features.)

A feature is a property of a classifier, and describes a way in which the classifier measures information in the input. However, it will also be conceptually useful to refer to the information itself. Thus we define the closely related notion of a *pattern* $P \subset \mathbb{R}^n$, a subset of inputs. We say that an image $x$ *contains* a pattern $P$ when $x \in P$. For example,

a pattern of stop signs would be the set of all images that contain a stop sign.

We are primarily concerned with the relationship between robust features and non-robust features.

### 2.1. Hypotheses Regarding The Relationship Between Robust and Non-Robust Features

The main question we address in this paper is, what is the nature of the *non-robust* features described by Ilyas et al. (2019)? We hypothesize that many *non-robust* features learned by standard (i.e., non-robust classifiers) respond to patterns that are entangled with human-interpretable *robust* features, rather than responding to dataset artifacts. Moreover, these entangled non-robust features can be exploited to create adversarial examples. If true, this hypothesis has important implications for defenses against, and transferability of, adversarial examples.

We illustrate our hypothesis with a conceptual diagram in Figure 1 consisting of three different types of patterns that features in a network could respond to. Robust features respond to Type A patterns—ones that are interpretable by humans as giving rise to a particular class. Type C illustrates a type of pattern that non-robust features might respond to: highly predictive non-semantic patterns in the dataset that are unrelated to the robust features in the image. These might be called "spurious correlations" or "artifacts" in the dataset. Type B illustrates another possibility for non-robust features: they might respond to small but highly predictive components of robust features (e.g., the nose of a dog or the whiskers of a cat) that could be easily exploited to yield adversarial examples.

We confirm in this paper that Type B features are indeed learned by CNNs, that these features can, in some cases, explain the high accuracy of CNNs, and that these features can be exploited in adversarial attacks. Thus the adversarial vulnerability of CNNs is not necessarily due to non-semantic artifacts, but can be explained in terms of non-robust features that are entangled with robust features that respond to semantically meaningful patterns.

## 3. Methods

### 3.1. Strategy for Testing Our Hypotheses

In order to test our hypothesis that Type B features are learned by standard CNN classifiers, we use the following strategy, inspired by Ilyas et al. (2019). We construct a neural network classifier in which only non-robust features are useful. We then construct a new test set for which only robust features are useful for classification.

We then show that that the classifier achieves substantially higher than chance accuracy on the constructed test set; this implies that the classifier must be using non-robust features that are entangled with robust features—namely, Type B features.

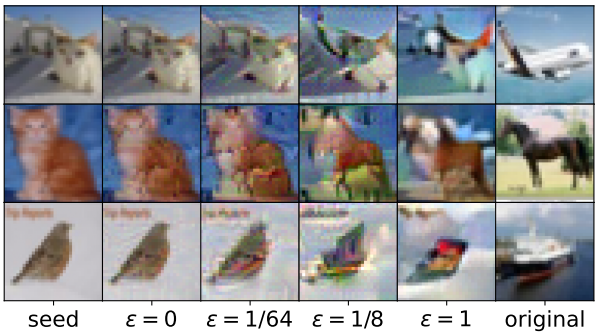



*Figure 2.* Examples of robustified images by $\varepsilon$ of robust classifier. The rightmost column depicts the original CIFAR-10 image; the leftmost column depicts the seed ($x_0$) from which the gradient descent of the robustification process began; the rest are robustified images. Since the representation layers of less robust classifiers are more easily perturbed, robustified images generated with a less robust model (smaller value of $\varepsilon$) will appear closer to the (random) CIFAR-10 image that seeded the gradient-descent process. (Best viewed in color.)

The following subsections describe how we construct this classifier and test set.

### 3.2. Constructing a Classifier For Which Only Non-Robust Features Are Useful

**Zero-Robust Classifier** We use the term *zero-robust classifier* for the classifier for which only non-robust features are useful. To train these classifiers, we follow the method of Ilyas et al. (2019). We first construct a training set $\mathcal{D}_{\text{zero}}$ from our original training set $\mathcal{D}$ such that under $\mathcal{D}_{\text{zero}}$, all robust features are non-useful, i.e., totally uncorrelated with the assigned label, while non-robust features remain useful. By training a standard classifier on this distribution, the standard classifier should only learn non-robust features.

The new training set $\mathcal{D}_{\text{zero}}$ can be constructed as follows: for each input-label pair $(x, y) \in \mathcal{D}$, choose a new label $\hat{y}$ uniformly at random from the possible labels. Construct a permissible adversarial example $\hat{x}$ (i.e., $\hat{x} = x + \delta$, where $\delta$ is a permissible perturbation) such that $\hat{x}$ is classified as $\hat{y}$. Then, let $(\hat{x}, \hat{y})$ be an element of $\mathcal{D}_{\text{zero}}$. Ilyas et al. (2019) argued that there should be almost no correlation between robust features and the assigned labels in this new dataset. Thus, a classifier trained on this dataset should not rely on robust features.

**Negative-Robust Classifier: Addressing Problem of "Robust Feature Leakage".** As pointed out by Engstrom et al. (2019a), there may be a small correlation between

robust features and the assigned labels in $\mathcal{D}_{\text{zero}}$, due to "robust feature leakage," which may account for some of the accuracy of a zero-robust classifier on a robustified dataset. We control for this possibility with the method proposed by Ilyas et al. (2019): we construct a *negative-robust* classifier, where robust features are anti-correlated with the label and non-robust features are correlated with the label. Thus, robust features should actively hurt accuracy and any positive accuracy can certainly be attributed to non-robust features. We construct a negative-robust classifier similar to the construction of the zero-robust classifier, but while $\hat{y}$ is selected uniformly at random to construct $\mathcal{D}_{\text{zero}}$, for the negative-robust classifier we deterministically permute the class labels, associating each label $y$ with the corresponding $\hat{y}$ in the permutation. Then for each $(x, y)$ pair in $\mathcal{D}$ we create $(\hat{x}, \hat{y})$, where $\hat{x}$ is an adversarial example based on $x$ targeting class $\hat{y}$. Using a deterministic permutation of the label classes makes robust features *anti-correlated* with the correct label class.

### 3.3. Constructing a Test Set For Which Only Robust Features Are Useful

In order to construct a test set for which only robust features are useful, we follow the approach of Ilyas et al. (2019). First we construct a *robust classifier*, and then use that classifier to construct the desired dataset.

**Robust Classifier.** We construct robust classifiers via standard adversarial training (Madry et al., 2017). Formally, we wish to find classifier parameters $\theta^*$ that minimize the following expression:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}}[\max_{\|\delta\|_2\leq\varepsilon} \mathcal{L}(\theta, x+\delta, y)]$$

where $\mathcal{L}(\theta, x, y)$ computes the cross-entropy loss of the classifier with parameters $\theta$ on the input $x$ and label $y$. The magnitude of adversarial examples to which the classifier is robust is characterized by the $\varepsilon$ parameter, which we call the *robustness parameter* of a classifier. Larger values of $\varepsilon$ correspond to more robust classifiers (i.e., they are robust to larger perturbations). The special case when $\varepsilon = 0$ corresponds to a standard classifier. We refer to a classifier with robustness parameter $\varepsilon$ as an *$\varepsilon$-robust classifier*.

The above saddle-point optimization problem can be solved by first solving the inner maximization problem with projected gradient ascent and then solving the outer maximization problem with standard back-propagation. We train with projected gradient descent as proposed by Madry et al. (2017). Since this problem is well studied, we defer to prior literature for discussion of the matter (Carlini et al., 2019; Madry et al., 2017).

**Test Set Robustification.** To construct a test set for which only robust features are useful for classification, we train a robust classifier $C$ as described above, and use it to create a new test set $\hat{\mathcal{D}}$ in which there is a one-to-one mapping between the elements of $\hat{\mathcal{D}}$ and the original test set $\mathcal{D}$: $(x, y) \mapsto (\hat{x}, y)$. In particular, an image $\hat{x} \in \hat{\mathcal{D}}$ is constructed so that any feature of $C$ that is useful in classifying $x$ is equally useful in classifying $\hat{x}$, in the sense that useful features are equally correlated with the label $y$, and no other possible feature is useful. In short, our construction will ensure that the *only* useful features for classifying $\hat{x}$ will be the features of C, which are, by definition, robust features.

Define $rep_C(x)$ as the output of the $m$-dimensional representation (penultimate) layer of robust classifier $C$ with input $x$. Then for $r \in \mathbb{R}^m$, ${rep_C}^{-1}(r)$ is a set of images with robust features $r$.

We cannot easily compute ${rep_C}^{-1}(r)$. However, for each element of $x$ our original test set, we can find an approximate inverse by solving the minimization problem

$$\min_{x_r} \|rep_C(x_r) - rep_C(x)\| .$$

We solve this via gradient descent, starting from an image $x_0$. In other words, starting from randomly chosen test-set image $x_0$, we search for an image $x_r$ whose representation $rep_C(x_r)$ is as close as possible to $rep_C(x)$.

Following Ilyas et al. (2019), if we choose the initial image $x_0$ uniformly from the test set, and the test set has a uniform distribution over labels, then all features in $rep_C(x_0)$ are uncorrelated (in expectation over $x_0$) with $x$'s label. This ensures (in expectation) that any features that respond to patterns in $x_0$ will not be correlated with $x$'s label, $y$. Since we apply gradient descent to $x_0$ in order to find an $\hat{x}$ whose representation is similar as possible to $C$'s representation for $x$, the only features that correlate with $y$ will be those in $C$, which are robust by definition. Thus only robust features are useful for classifying $\hat{x}$.

Following Ilyas et al. (2019), we refer to the inversion process the *robustification* of images. Given a test set $\mathcal{D}$, we can construct a new test set $\hat{\mathcal{D}}$ by computing this approximation of $rep^{-1}(rep(x))$ and labeling it $y$ for every $(x, y) \in D$.

If robust classifier $C$ has robustness parameter $\varepsilon > 0$, we say that $\hat{\mathcal{D}}$ is the *$\varepsilon$-robustification* of $\mathcal{D}$ and will be referred to as $\mathcal{R}_\varepsilon$. Figure 2 shows examples of the robustification process on CIFAR-10 test images.

### 3.4. Datasets & Model Consideration

We evaluate our methods on two datasets: CIFAR-10 (Krizhevsky et al., 2009) and ImageNet-9, a derivative of ImageNet (Russakovsky et al., 2015) with two key differences from the original ImageNet dataset: first, we use a

$64 \times 64$ pixel downsampled version of the ImageNet data for computational feasibility (Chrabaszcz et al., 2017), second, adopted from Ilyas et al. (2019), we reduce the number of classes to nine: dog, cat, frog, turtle, bird, primate, fish, crab, insect. See Ilyas et al. (2019) for further details.

For the CIFAR-10 dataset, we train ResNet50 classifiers (He et al., 2016a;b). However, due to the high computational cost of training robust models on ImageNet-9, we train a classifier with fewer parameters, ResNet20, on the ImageNet-9 dataset.

## 4. Results of Experiments

*Table 1.* Accuracy of zero-robust, negative-robust, standard, and robust ResNet50 classifiers on robustified CIFAR-10 test set. All classifiers are trained starting with random initial weights. The *Test* column is the accuracy on the original CIFAR-10 test set. The $\mathcal{R}_\varepsilon$ columns give accuracy on robustified CIFAR-10 test set, generated with respect to a robust ResNet50 with robustness parameter $\varepsilon$.

| Classifier | Test | $R_0$ | $\mathcal{R}_{1/16}$ | $\mathcal{R}_{1/4}$ | $\mathcal{R}_{1/2}$ | $\mathcal{R}_1$ |
|---|---|---|---|---|---|---|
| Standard | 94.1 | 58.3 | 79.9 | 75.1 | 69.2 | 56.3 |
| Robust | 86.5 | 9.93 | 18.2 | 70.5 | 86.5 | 70.0 |
| Zero-robust | 46.8 | 38.8 | 23.3 | 22.1 | 23.4 | 21.4 |
| Neg-robust | 21.9 | 31.8 | 11.1 | 12.8 | 14.3 | 13.2 |

*Table 2.* Accuracy of zero-robust, negative-robust, standard, and robust ResNet20 classifiers on robustified ImageNet-9 test set. Analogous to Table 1. Note: since there are nine classes, chance accuracy is approximately 11.1%.

| Classifier | Test | $R_0$ | $\mathcal{R}_{1/16}$ | $\mathcal{R}_{1/4}$ | $\mathcal{R}_{1/2}$ | $\mathcal{R}_1$ |
|---|---|---|---|---|---|---|
| Standard | 94.8 | 51.9 | 76.3 | 63.4 | 54.6 | 47.2 |
| Robust | 82.9 | 44.1 | 46.1 | 49.0 | 51.7 | 56.0 |
| Zero-robust | 49.2 | 24.8 | 24.3 | 24.6 | 28.6 | 31.4 |
| Neg-robust | 38.3 | 25.4 | 23.2 | 21.8 | 24.4 | 23.7 |

Recall our definition of Type B features: Non-robust features learned by a network that respond to small yet highly predictive patterns that by themselves appear non-semantic, yet are entangled with patterns associated with robust features. Our hypothesis is that Type B features are prevalent in standard (i.e., non-robust) image classifiers, and can be exploited to create adversarial examples. That is, Type C features (which respond to dataset artifacts) are not the only features responsible for adversarial vulnerability.

We present here the result of an experiment in which we evaluate zero-robust and negative-robust classifiers, representing classifiers that rely exclusively on non-robust features, on CIFAR-10 and ImageNet-9 test sets that have been robustified—that is, in which only robust features are useful.

The results, given in Tables 1 and 2, confirm our hypothesis: the zero-robust classifier performs above chance on all robustified test sets; this can only occur when Type B features are present. Similarly, the negative-robust classifier, which acts as a control for the problem of robust feature leakage (see Section 3.2), has above-chance accuracy on robustified test sets except for when $\varepsilon$ is large. We expect this result, since the presence of robust features decreases accuracy in negative-robust classifiers. In fact, in the absence of Type B features, we would expect the negative-robust classifier to perform at below-chance accuracy on the robustified test sets. As above-chance accuracy by the negative-robust can only be explained by the presence of Type B features, the accuracy of the negative-robust classifier serves as a lower-bound on the accuracy for which can be accounted by Type B features. The fact that the accuracy is above chance for many of the robustified test sets further confirms our hypothesis, and rejects the possibility that robust feature leakage entirely explains the above-chance accuracy of the zero-robust classifier on the robustified test sets. We include the accuracies of a standard classifier and an example robust classifier with robustness parameter $\varepsilon = 0.5$ for comparison with the zero- and negative-robust classifier results.

We observe that the accuracy of zero-robust and negative-robust classifiers decreases on the robustified test sets as the robustness parameter $\varepsilon$ increases. We have already discussed that we expect this in the negative-robust classifier due to the negative influence of robust features on accuracy. However, we hypothesize that the reason zero-robust accuracy decreases is similar to that of the well-established result that the accuracy of robust classifiers decreases on the original test set as the robustness parameter of the classifier increases (Tsipras et al., 2019; Zhang et al., 2019). We confirm this well-known result in Figure 4. We speculate that as the robustness increases, the classifier begins to ignore useful features, even some that are robust for smaller $\varepsilon$. Thus, the robustified test sets contain fewer useful features as $\varepsilon$ increases, and thus zero-robust accuracy decreases.

## 5. The Universality of Robust Features

The previous section established that non-robust features respond to patterns that are entangled with the patterns responded to by robust features. However, there may be many equally predictive "non-robust" patterns that can be entangled with a single "robust" pattern. For example, as illustrated in Figure 1, a non-robust Type B feature which responds to the texture of a dog nose may be a component of a robust Type A feature that identifies the shape of an entire dog. However, there may be many components of a particular robust feature that would equivalently predict "dog". Standard (non-robust) classifier $C_1$ might learn a particular subset of predictive non-robust (i.e., Type B) features, whereas another standard classifier $C_2$ with a different ar-

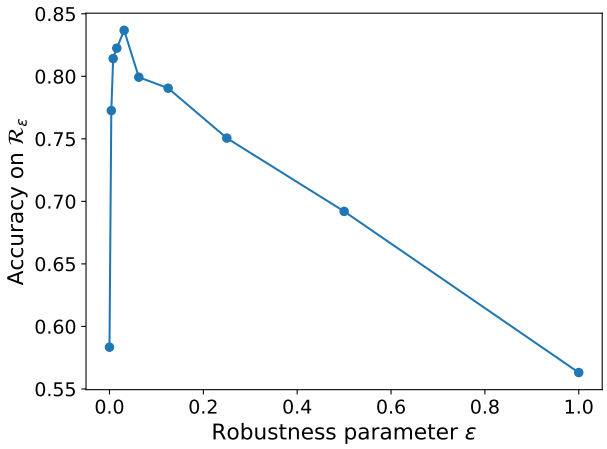



*Figure 3.* Accuracy of a standard ResNet50 classifier on robustifications of the CIFAR-10 test dataset, generated with respect to robust ResNet50 classifiers with varying robustness parameter $\varepsilon$. See Appendix for the analogous graph for ImageNet-9. (Best viewed in color.)

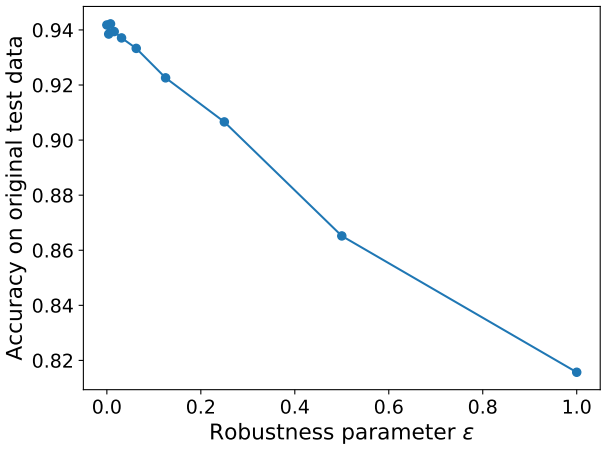



*Figure 4.* Accuracy of robust ResNet50 classifiers trained on the CIFAR-10 training data and evaluated on the CIFAR-10 original test data. See Appendix for the analogous graph for ImageNet-9. (Best viewed in color.)

chitecture or different initial weights might learn a different subset, with both classifiers exhibiting similar accuracy on a test set. The features of both classifiers would be responding to *different* subpatterns of the same underlying "robust" patterns; in this case, the features learned by $C_1$ would not strongly overlap the features learned by $C_2$.

The features learned by a classifier that tend to overlap with the features learned by other classifiers trained on the same dataset are characterized by *universality*, as they are roughly the same across classifiers (Olah et al., 2020). Li et al. (2015) called this phenomenon "convergent learning".

We hypothesize that robust features will be more universal than non-robust features, since universal features likely represent the underlying, better generalizing properties of the dataset that are captured by robust features. Analogous to the way we demonstrated the existence of Type B features entangled with Type A features in Section 3, here we give evidence for the hypothesis that robust features are more universal than non-robust features by evaluating standard classifiers on robustified test sets.

Recall that for a test set $D_T$ that has been robustified with respect to a robust classifier $C$, the only features that are useful for classification are robust features useful to $C$. If a different classifier $C'$ has high accuracy on $D_T$, then $C'$ must be using the same (or very similar) features as $C$. If many different classifiers have high accuracy on $D_T$, then we can say that $C$'s robust features are *universal*.

### 5.1. Comparing the Universality of Robust vs. Non-Robust Features

Figure 3 shows the accuracy of a standard (non-robust) ResNet50 classifier, trained on the CIFAR-10 training set, when tested on robustified CIFAR-10 test sets with varying robustness parameter $\varepsilon$. The figure shows that the the standard classifier, which we'll call $C'$, has significantly higher accuracy on robustified test sets with $0 < \varepsilon \leq 1/2$, with a peak at $\varepsilon = 1/32$.

To show why this supports the hypothesis that robust features are more universal than non-robust features, let $C_\varepsilon$ denote the $\varepsilon$-robust ResNet50 classifier used to create the robustified test set $R_\varepsilon$. Figure 3 shows that the accuracy of $C'$ is low on $R_0$, which was designed specifically so that the features of $C_0$ would be useful to classify it. This means that $C_0$'s features, which are likely to be mostly non-robust since $\varepsilon = 0$, tend not to be shared by $C'$. However, the accuracy of $C'$ on $R_\varepsilon$ for $0 < \varepsilon \leq 1/2$ is dramatically higher, meaning that the features that are useful to classifiers $C_\varepsilon$ are also useful to $C'$, for $\varepsilon$ in that range. Thus the robust features useful to these $C_\varepsilon$ tend to be shared by $C'$. Repeated evaluations with different initial weights for $C'$ exhibited the same behavior. Thus, the results shown in Figure 3 support the hypothesis that robust features are more universal than non-robust features.

It is noteworthy that accuracy is highest for a relatively small robustness parameter of $\varepsilon = 1/32$. We hypothesize that as robustness increases, while patterns associated with Type B features may remain present in the robustified dataset, patterns associated with Type C features, which were present in the original dataset, may be progressively removed, caus-

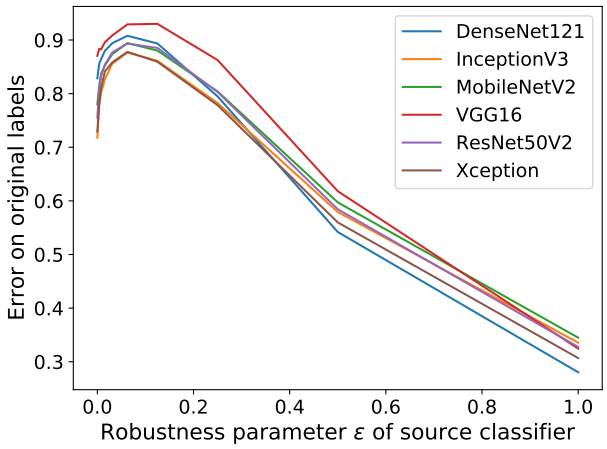



*Figure 5.* Error rate of different standard classifiers evaluated on untargeted transfer attacks. Each error rate is measured on a set of untargeted adversarial examples generated from the CIFAR-10 test set to attack a source classifier (ResNet50) with robustness parameter $\varepsilon$. Each adversarial example was generated with a perturbation $\delta$ where $\|\delta\|_2 < 2$. A higher error corresponds to a stronger transfer attack. See Appendix for the analogous graph for ImageNet-9. (Best viewed in color.)

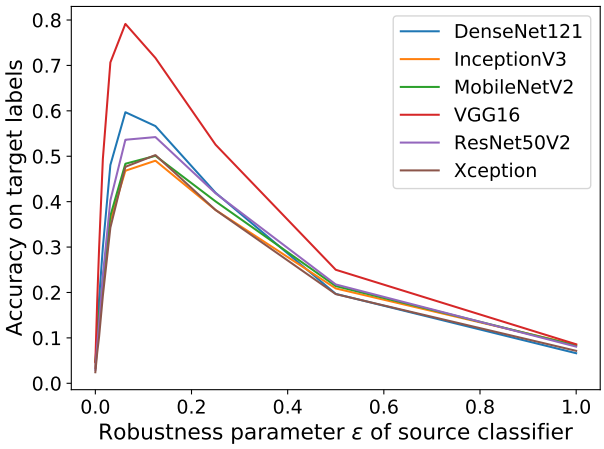



*Figure 6.* Accuracy of different standard classifiers on targeted transfer attacks. Each accuracy is measured as the fraction of times the target label $y_t$ is predicted for targeted adversarial example $x_t$. The evaluation is done on a set of targeted adversarial examples generated from the CIFAR-10 test set to attack a source ResNet50 classifier with robustness parameter $\varepsilon$. Each adversarial example was generated with a perturbation $\delta$ where $\|\delta\|_2 < 2$. A higher accuracy corresponds to a stronger transfer attack. See Appendix for the analogous graph for ImageNet-9. (Best viewed in color.)

ing a detriment to accuracy as the robustness parameter increases significantly. Similarly, we speculate that this may also explain the well-known decrease in accuracy on original test data associated with an increase in model robustness as shown in Figure 4 (Tsipras et al., 2019; Zhang et al., 2019).

Our results suggest that even when performance is critical and the drop in accuracy associated with adversarial training is unacceptable, adding even slight robustness ($\varepsilon = 1/32$) can drastically improve the universality of the learned features of a neural network. Networks can be improved in this way with little to no additional computational cost due to recent advances in improving the efficiency of adversarial training (Jeddi et al., 2020; Shafahi et al., 2019; Wong et al., 2020).

We repeat these experiments using ResNet20 classifiers trained on the ImageNet-9 dataset and find similar results, presented in the Appendix.

### 5.2. Transferability of Adversarial Examples

Adversarial examples designed to fool a given classifier are sometimes *transferable*: they can also successfully fool other classifiers, even those with different architectures from the original ("source") classifier. This is true for both *untargeted* adversarial examples $x_u$, which are considered to be successful if a classifier predicts any incorrect label when given $x_u$, and for *targeted* adversarial examples $x_t$, which are considered to be successful if a classifier predicts a specific targeted (incorrect) label $y_t$ when given $x_t$. It is typically much harder to design transferable targeted than untargeted adversarial examples (Liu et al., 2016).

Our earlier hypothesis—that robust features are more universal than non-robust features—suggests that adversarial examples designed to fool a robust classifier transfer (i.e., that exploit robust features) should be more transferable than adversarial examples designed to fool a non-robust classifier. In this section we give evidence for this conclusion: we show that if a robust classifier is used as a source model for creating adversarial examples, those examples transfer more effectively than if a non-robust classifier is used as the source. This is an important novel result for researchers trying to create transferable adversarial examples: one should use a robust model as the source, instead of a standard model.

To give evidence for these hypotheses, we train ResNet50 classifiers on the CIFAR-10 training set using adversarial training with varying robustness parameters $\varepsilon$ (in addition, we train ResNet20 on ImageNet-9 data, see Appendix). We then use these robust classifiers to generate targeted and untargeted adversarial examples via projected gradient descent (Madry et al., 2017), using the standard adversarial objectives. In the untargeted case, given an initial input $x$ and a parameter $\varepsilon$ to specify the maximum L2 norm of a permissible perturbation $\delta$, we seek to find an adversarial example $x_u = x + \hat{\delta}$ (where $x$ is the original example with

label $y$) such that

$$\hat{\delta} = \arg\max_{\|\delta\|_2 \leq \varepsilon} \mathcal{L}(\theta, x + \delta, y),$$

and for the targeted case, for a target label $y_t$, we seek to find an adversarial example $x_t = x + \hat{\delta}$ such that

$$\hat{\delta} = \arg\min_{\|\delta\|_2 \leq \varepsilon} \mathcal{L}(\theta, x + \delta, \hat{y}).$$

We evaluate the generated adversarial examples on standard classifiers with different architectures (Chollet, 2017; Howard et al., 2017; Huang et al., 2017; Simonyan & Zisserman, 2014; Szegedy et al., 2016). For the untargeted case, Figure 5 plots, for each classifier, the error rate on the original labels $y$, given adversarial examples $x_u$. This error rate is plotted as a function of the robustness parameter $\varepsilon$ of the source classifier used to generate the adversarial examples ($x_u$). The higher the error rate, the more successful the transfer attack. To evaluate targeted adversarial examples, in Figure 6 we plot the accuracy of each standard classifier on adversarial examples $x_t$ with respect to the target label $y_t$—i.e., the fraction of times the classifier predicted the target label $y_t$, given example $x_t$. The higher this accuracy, the more successful the transfer attack.

Figure 5 shows that untargeted adversarial examples benefit slightly if generated with respect to a robust classifier with a small robustness parameter, $0 < \varepsilon \leq 1/8$ with a peak at $\varepsilon \approx 1/16$. The success of transferable adversarial examples decreases as $\varepsilon$ is further increased.

Figure 6 shows that transferable targeted adversarial examples attain a more striking increase in success by increasing the robustness parameter of the source classifier. Our technique for generating transferable targeted adversarial examples is largely ineffective when the source classifier is equivalent to standard classifier (i.e., $\varepsilon = 0$). However, using a robust source classifier with $\varepsilon \approx 1/16$ dramatically increases the success of targeted transfer attacks.

Experiments on the ImageNet-9 dataset using ResNet20 as source models behave similarly (see Appendix for details).

Our goal here is not to generate state-of-the-art transfer attacks; we leave this to future work. Thus, we do not employ complex state-of-the-art transfer attacks and instead opt for the simple projected gradient descent approach. Nonetheless, our results confirm that robust features with small robustness parameter have high universality and can improve transfer attacks, especially in the targeted case, in which improvements are dramatic.

## 6. Related Work

Adversarial examples and associated adversarial robustness have been studied extensively in the machine learning literature (Allen-Zhu & Li, 2020; Athalye et al., 2018; Biggio et al., 2013; Carlini & Wagner, 2017a;b; Carlini et al., 2019; Cohen et al., 2019; Engstrom et al., 2019b; Feinman et al., 2017; Goodfellow et al., 2014; Kaur et al., 2019; Madry et al., 2017; Metzen et al., 2017; Moosavi-Dezfooli et al., 2016; Papernot et al., 2016b; Raghunathan et al., 2018; Santurkar et al., 2019; Shafahi et al., 2018; Stutz et al., 2019; Szegedy et al., 2014; Uesato et al., 2018; Warde-Farley & Goodfellow, 2016). Non-robust features have been studied directly by Ilyas et al. (2019) who finds that non-robust features are present in image training datasets. By contrast, we establish an entanglement relationship between robust and non-robust features.

We hypothesize that the overlap between robust and non-robust features may be explained in part from a simplicity bias (Arpit et al., 2017; De Palma et al., 2019; Nakkiran et al., 2019; Shah et al., 2020; Valle-Pérez et al., 2018; Wu et al., 2017), which suggests that neural networks learn simple functions more easily than complex functions, and from gradient starvation (Combes et al., 2018; Pezeshki et al., 2020), which suggests that once highly predictive features are learned, other features become increasingly difficult to learn due to a diminishing gradient. If non-robust features are learned more readily and support sufficiently low classification loss, such a combination may impede the learning of robust features, for instance, see Nakkiran (2019). Similarly, texture bias may be related (Hermann et al., 2020).

Many papers have shown that deep learning classifiers may rely on non-semantic cues or shortcuts (Geirhos et al., 2020; Jo & Bengio, 2017; McCoy et al., 2019; Wang et al., 2020; Wei, 2020) and that optimization-based feature visualization may be inadequate to visualize neural network features (Borowski et al., 2020; Hendrycks et al., 2019). On the other hand, there has been work on feature visualization (Aubry & Russell, 2015; Dosovitskiy & Brox, 2016; Mahendran & Vedaldi, 2015; Olah et al., 2017; Simonyan et al., 2013; Zhang & Zhu, 2018).

Related to our hypothesis that robust features are more universal than non-robust features, prior work has found that adversarial robustness serves as an effective prior for transfer learning (Liang et al., 2020; Salman et al., 2020; Terzi et al., 2020; Utrera et al., 2020). There has been some work on the universality hypothesis (Kornblith et al., 2019; Li et al., 2015; Olah et al., 2020; Raghu et al., 2017). Additionally, there has been significant work on constructing and evaluating transferable targeted adversarial examples (Eykholt et al., 2018; Kurakin et al., 2016; Liu et al., 2016; Moosavi-Dezfooli et al., 2017; Papernot et al., 2016a; 2017; Tramèr et al., 2017).

## 7. Conclusion

In this paper, we have presented an empirical study demonstrating that robust and non-robust features can be entangled in standard neural network classifiers. In particular, we have proposed three different types of features: robust features (Type A), non-robust features that are entangled with robust features (Type B), and non-robust features that are not entangled with robust features (Type C). Type A features are known to appear significantly more semantic than non-robust features (Engstrom et al., 2019b; Kaur et al., 2019; Santurkar et al., 2019). To our knowledge, no previous study has addressed the question of whether the non-robust features underlying adversarial vulnerability are the non-semantic yet highly predictive artifacts in image datasets, or if the features appear non-semantic for a different reason. We present evidence that at least some of the non-robust features indicate the presence of the same patterns of the dataset as robust features, suggesting that while appearing non-semantic when observed through the lens of their gradient and adversarial perturbations, these features may, in fact, indicate the presence of features that are aligned with human perception.

In addition, we provide evidence that robust features can be thought of as more universal than non-robust features, and as result, we find that robust classifiers are more effective as source classifiers for generating transferable adversarial examples in both the untargeted and targeted settings.

We believe that this work is an important step towards understanding the nature of non-robust features and answering the universality hypothesis (Olah et al., 2020). With a theory of robust and non-robust features, and thus of adversarial vulnerability and resilience, we hope to eventually understand how to build more interpretable classifiers and be able to better defend against adversarial attacks.

## Acknowledgements

Research presented in this article was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20210043DR.

The authors would like to thank Rory Soiffer for his helpful ideas and discussion.

## References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Allen-Zhu, Z. and Li, Y. Feature purification: How adversarial training performs robust deep learning. *arXiv preprint arXiv:2005.10190*, 2020.

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. *arXiv preprint arXiv:1706.05394*, 2017.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. In *International conference on machine learning*, pp. 284–293. PMLR, 2018.

Aubry, M. and Russell, B. C. Understanding deep features with computer-generated imagery. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2875–2883, 2015.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Borowski, J., Zimmermann, R. S., Schepers, J., Geirhos, R., Wallis, T. S., Bethge, M., and Brendel, W. Exemplary natural images explain cnn activations better than feature visualizations. *arXiv preprint arXiv:2010.12606*, 2020.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14, 2017a.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017b.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Chollet, F. et al. Keras. `https://keras.io`, 2015.

Chrabaszcz, P., Loshchilov, I., and Hutter, F. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320. PMLR, 2019.

Combes, R. T. d., Pezeshki, M., Shabanian, S., Courville, A., and Bengio, Y. On the learning dynamics of deep neural networks. *arXiv preprint arXiv:1809.06848*, 2018.

De Palma, G., Kiani, B., and Lloyd, S. Random deep neural networks are biased towards simple functions. In *Advances in Neural Information Processing Systems*, pp. 1964–1976, 2019.

Dosovitskiy, A. and Brox, T. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4829–4837, 2016.

Engstrom, L., Gilmer, J., Goh, G., Hendrycks, D., Ilyas, A., Madry, A., Nakano, R., Nakkiran, P., Santurkar, S., Tran, B., Tsipras, D., and Wallace, E. A discussion of 'adversarial examples are not bugs, they are features'. *Distill*, 2019a. doi: 10.23915/distill.00019. https://distill.pub/2019/advex-bugs-discussion.

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.

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *arXiv preprint arXiv:2004.07780*, 2020.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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, 2016a.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016b.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

Hermann, K., Chen, T., and Kornblith, S. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

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.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pp. 125–136, 2019.

Jeddi, A., Shafiee, M. J., and Wong, A. A simple fine-tuning is all you need: Towards robust deep learning via adversarial fine-tuning. *arXiv preprint arXiv:2012.13628*, 2020.

Jo, J. and Bengio, Y. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*, 2017.

Kaur, S., Cohen, J., and Lipton, Z. C. Are perceptually-aligned gradients a general property of robust classifiers? *arXiv preprint arXiv:1910.08640*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pp. 3519–3529. PMLR, 2019.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

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., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, Y., Yosinski, J., Clune, J., Lipson, H., and Hopcroft, J. E. Convergent learning: Do different neural networks learn the same representations? In *FE@ NIPS*, pp. 196–212, 2015.

Liang, K., Zhang, J. Y., Koyejo, O., and Li, B. Does adversarial transferability indicate knowledge transferability? *arXiv preprint arXiv:2006.14512*, 2020.

Liu, Y., Chen, X., Liu, C., and Song, D. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Mahendran, A. and Vedaldi, A. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5188–5196, 2015.

McCoy, R. T., Pavlick, E., and Linzen, T. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*, 2019.

Metzen, J. H., Genewein, T., Fischer, V., and Bischoff, B. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

Nakkiran, P. Adversarial robustness may be at odds with simplicity. *arXiv preprint arXiv:1901.00532*, 2019.

Nakkiran, P., Kalimeris, D., Kaplun, G., Edelman, B., Yang, T., Barak, B., and Zhang, H. Sgd on neural networks learns functions of increasing complexity. In *Advances in Neural Information Processing Systems*, pp. 3496–3506, 2019.

Olah, C., Mordvintsev, A., and Schubert, L. Feature visualization. *Distill*, 2(11):e7, 2017.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

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*, 2016a.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pp. 372–387. IEEE, 2016b.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519, 2017.

Pezeshki, M., Kaba, S.-O., Bengio, Y., Courville, A., Precup, D., and Lajoie, G. Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*, 2020.

Raghu, M., Gilmer, J., Yosinski, J., and Sohl-Dickstein, J. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *arXiv preprint arXiv:1706.05806*, 2017.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do adversarially robust imagenet models transfer better? *arXiv preprint arXiv:2007.08489*, 2020.

Santurkar, S., Ilyas, A., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Image synthesis with a single (robust) classifier. In *Advances in Neural Information Processing Systems*, pp. 1262–1273, 2019.

Shafahi, A., Huang, W. R., Studer, C., Feizi, S., and Goldstein, T. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018.

Shafahi, A., Najibi, M., Ghiasi, A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. The pitfalls of simplicity bias in neural networks. *arXiv preprint arXiv:2006.07710*, 2020.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Stutz, D., Hein, M., and Schiele, B. Disentangling adversarial robustness and generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6976–6987, 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

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.

Terzi, M., Achille, A., Maggipinto, M., and Susto, G. A. Adversarial training reduces information and improves transferability. *arXiv preprint arXiv:2007.11259*, 2020.

Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Uesato, J., O'donoghue, B., Kohli, P., and Oord, A. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, pp. 5025–5034. PMLR, 2018.

Utrera, F., Kravitz, E., Erichson, N. B., Khanna, R., and Mahoney, M. W. Adversarially-trained deep nets transfer better. *arXiv preprint arXiv:2007.05869*, 2020.

Valle-Pérez, G., Camargo, C. Q., and Louis, A. A. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

Wang, H., Wu, X., Huang, Z., and Xing, E. P. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8684–8694, 2020.

Warde-Farley, D. and Goodfellow, I. Adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics*, 311, 2016.

Wei, K.-A. A. *Understanding non-robust features in image classification*. PhD thesis, Massachusetts Institute of Technology, 2020.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

Wu, L., Zhu, Z., et al. Towards understanding generalization of deep learning: Perspective of loss landscapes. *arXiv preprint arXiv:1706.10239*, 2017.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pp. 7472–7482. PMLR, 2019.

Zhang, Q.-s. and Zhu, S.-C. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

# A. Experimental Setup

In this section, we describe details of the methods that we use to train our standard, robust, zero-robust, and negative-robust models, construct robustified images, and construct adversarial examples.

## A.1. Dataset Considerations

Our experiments involve training many models, a large portion of which require adversarial training. Similarly, our evaluation procedures involve constructing robust datasets, another highly computationally expensive operation. In order to ensure the computational feasibility of these problems, we conduct all experiments on two image datasets: CIFAR-10 (Krizhevsky et al., 2009) and a version of ImageNet (Russakovsky et al., 2015) which has been downsampled to $64 \times 64$ pixels (Chrabaszcz et al., 2017) and in which we only use images which fall under one of nine non-overlapping class labels: dog (classes 151–268), cat (classes 281–285), frog (classes 30–32), turtle (classes 33-37), bird (classes 80–100), primate (classes 365–382), fish (classes 389–397), crab (classes 118-121), insect (classes 300-319). This dataset is the same subset of the ImageNet dataset used by Ilyas et al. (2019), with the additional modification that our dataset is downsampled. CIFAR-10 has 50,000 training images and ImageNet-9 has approximately 200,000.

While we conduct our experiments only on these two datasets, we expect that our results should be general across all image datasets, and likely across other domains as well.

## A.2. Training & Evaluation Details

We describe the training procedure for the four different types of models that we construct: standard, robust, zero-robust, and negative-robust.

For each model, we train using the Adam optimizer (Kingma & Ba, 2014) with default TensorFlow parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) (Abadi et al., 2015). The learning rate for each model is specified in Table 3. Each model was trained for at most 100 epochs. Models with an epoch count marked with an asterisks (*) had their training stopped early in order to maximize the performance on a validation set. Models that are not marked with an asterisks were trained for the entire 100 epochs.

**Standard Models** To train standard models, we minimize expected cross-entropy loss over the training dataset. All parameters are specified in Table 3.

**Robust Models** To train robust models, we compute a set of adversarial examples for each batch during training. We use projected gradient descent to construct these adversarial examples (Madry et al., 2017). For adversarial examples constructed from CIFAR-10 images, we use 16 steps with a step size of 0.5. For adversarial examples constructed from ImageNet images, we use 8 steps with a step size of 1.0. For adversarial training, we train to minimize the expected cross-entropy loss of the network evaluated on adversarial examples labeled as the true label. All other parameters are specified in Table 3.

**Zero-Robust Models** To train zero-robust models, we apply the methods proposed in Section 3.2. In particular, we construct adversarial examples (see Table 4 for parameters) using projected gradient descent to minimize the targeted adversarial loss functions

$$\min_{\|\delta\|_2 \leq \varepsilon} \mathcal{L}(\theta, x + \delta, \hat{y})$$

where $x$ is the original image, $\varepsilon$ is the maximum allowed L2 norm difference between the adversarial example and the original image, and $\hat{y}$ is the targeted class. For each image in CIFAR-10, we construct ten adversarial examples, one targeting each of the ten classes, labeled as the target class. For each image in the ImageNet-9 dataset, we construct a single adversarial example with a target selected uniformly at random from the nine possible classes, similarly labeled as the target class. We train zero-robust models for both CIFAR-10 and ImageNet-9 using standard training on the appropriate constructed dataset. The hyperparameters for training the models can be found in Table 3.

**Negative-Robust Models** We train negative-robust models in the same way as we train zero-robust models, except with a slightly different dataset. For each image with label $y$ in the CIFAR-10 dataset (numbered 0–9 to represent each class in CIFAR-10), we construct a new dataset with an adversarial example targeting the class $y + 1 \mod 10$, and labeled as this

target class. We construct the ImageNet-9 version similarly, using $y + 1 \mod 9$ as the target label instead. The parameters for constructing the dataset can be found in Table 4 and the hyperparameters for training the models in Table 3.

**Victim Models** To test the transferability of adversarial examples generated on robust models, we train standard models with different architectures: DenseNet121, InceptionV3, MobileNetV2, VGG16, ResNet50V2, and Xception (Chollet, 2017; Howard et al., 2017; Huang et al., 2017; Simonyan & Zisserman, 2014; Szegedy et al., 2016). We initialize these models with pre-trained weights from training on the original ImageNet dataset (Chollet et al., 2015). For the hyperparameters, see Table 3.

*Table 3.* Model parameters for each model trained in this paper. Parameters were selected by hand. In addition to what is listed below, the zero- and negative- robust models for both ImageNet-9 and CIFAR-10 were stopped prior to 100 epochs in order to maximize accuracy on a validation set. Each model was trained on a IBM Power9 architecture machine with two NVIDIA Tesla V100 GPUs. Each standard model took approximately 24 hours to train. Each robust models took approximately 48 hours.

| Model | Architecture | Batch size | Epochs | Learning rate | LR decay | Data augmentation | Pretraining |
|---|---|---|---|---|---|---|---|
| Standard (CIFAR) | ResNet56 (v2) | 200 | 100 | 1e-3 | Yes | Yes | No |
| Robust (CIFAR) | ResNet56 (v2) | 200 | 100 | 1e-3 | Yes | Yes | No |
| Zero-robust (CIFAR) | ResNet56 (v2) | 200 | 100* | 1e-4 | Yes | No | No |
| Neg-robust (CIFAR) | ResNet56 (v2) | 200 | 100* | 1e-4 | Yes | No | No |
| Victim models (CIFAR) | Misc | 200 | 100 | 1e-4 | Yes | Yes | Yes |
| Standard (ImageNet) | ResNet20 (v2) | 128 | 100 | 1e-3 | Yes | Yes | No |
| Robust (ImageNet) | ResNet20 (v2) | 128 | 100 | 1e-3 | Yes | Yes | No |
| Zero-Robust (ImageNet) | ResNet20 (v2) | 128 | 100* | 1e-3 | Yes | No | No |
| Neg-robust (ImageNet) | ResNet20 (v2) | 128 | 100* | 1e-3 | Yes | No | No |
| Victim models (ImageNet) | Misc | 128 | 100 | 1e-4 | Yes | Yes | Yes |

*Table 4.* Parameters for the various datasets constructed in this paper. Parameters were selected by hand.

| Dataset | Maximum allowed change (L2 norm from seed) | Number of steps | Step size |
|---|---|---|---|
| Robustified (CIFAR) | $\infty$ | 400 | 0.05 |
| Zero-robust (CIFAR) | 0.5 | 200 | 0.1 |
| Neg-robust (CIFAR) | 0.5 | 200 | 0.1 |
| Untargeted Adversarial Examples (CIFAR) | 2.0 | 100 | 0.1 |
| Targeted Adversarial Examples (CIFAR) | 2.0 | 100 | 0.1 |
| Robustified (ImageNet) | $\infty$ | 200 | 0.1 |
| Zero-robust (ImageNet) | 0.5 | 200 | 0.1 |
| Neg-robust (ImageNet) | 0.5 | 200 | 0.1 |
| Untargeted Adversarial Examples (ImageNet) | 4.0 | 100 | 0.1 |
| Targeted Adversarial Examples (ImageNet) | 4.0 | 100 | 0.1 |

## B. Extended Results

In this section, we present extended results including the extended data from Tables 1 and 2 (Tables 5 and 6). In addition, we include ImageNet-9 versions of Figures 3, 5, and 6 (Figures 7, 9, and 8). We present the accuracy of each of our robust classifiers on the original ImageNet-9 testing set (Figure 10). We include extended examples of robustified images (Figures 11 and 12). Finally, we provide examples of adversarial examples generated on robust models (Figures 13 and 14).

*Table 5.* Extended data: Accuracy of zero-robust, negative-robust, standard, and robust ResNet50 classifiers on robustified CIFAR-10 test set. All classifiers are trained starting with random initial weights. The *Test* column is the accuracy on the original CIFAR-10 test set. The $\mathcal{R}_\varepsilon$ columns give accuracy on robustified CIFAR-10 test set, generated with respect to a robust ResNet50 with robustness parameter $\varepsilon$.

| Classifier | Test | $R_0$ | $\mathcal{R}_{1/256}$ | $\mathcal{R}_{1/128}$ | $\mathcal{R}_{1/64}$ | $\mathcal{R}_{1/32}$ | $\mathcal{R}_{1/16}$ | $\mathcal{R}_{1/8}$ | $\mathcal{R}_{1/4}$ | $\mathcal{R}_{1/2}$ | $\mathcal{R}_1$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Standard | 94.1 | 58.3 | 77.3 | 81.4 | 82.2 | 83.7 | 79.9 | 79.0 | 75.1 | 69.2 | 56.3 |
| Robust ($\varepsilon = 0.5$) | 86.5 | 9.93 | 9.97 | 9.96 | 10.0 | 11.1 | 18.2 | 44.5 | 70.5 | 86.5 | 70.0 |
| Zero-robust | 46.8 | 38.8 | 35.1 | 26.1 | 27.2 | 26.2 | 23.3 | 22.6 | 22.1 | 23.4 | 21.4 |
| Negative-robust | 21.9 | 31.8 | 15.8 | 14.6 | 16.6 | 11.3 | 11.1 | 11.0 | 12.8 | 14.3 | 13.2 |

*Table 6.* Extended data: Accuracy of zero-robust, negative-robust, standard, and robust ResNet20 classifiers on robustified ImageNet-9 test set. Analogous to Table 5. Note: since there are nine classes, chance accuracy is approximately 11.1%.

| Classifier | Test | $R_0$ | $\mathcal{R}_{1/256}$ | $\mathcal{R}_{1/128}$ | $\mathcal{R}_{1/64}$ | $\mathcal{R}_{1/32}$ | $\mathcal{R}_{1/16}$ | $\mathcal{R}_{1/8}$ | $\mathcal{R}_{1/4}$ | $\mathcal{R}_{1/2}$ | $\mathcal{R}_1$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Standard | 94.8 | 51.9 | 68.3 | 71.5 | 75.5 | 76.7 | 76.3 | 72.1 | 63.4 | 54.6 | 47.2 |
| Robust ($\varepsilon = 1.0$) | 82.9 | 41.9 | 42.1 | 42.5 | 42.5 | 42.5 | 44.1 | 46.1 | 49.0 | 51.7 | 56.0 |
| Zero-robust | 49.2 | 24.8 | 28.3 | 30.5 | 29.3 | 27.5 | 24.3 | 24.2 | 24.6 | 28.6 | 31.4 |
| Negative-robust | 38.3 | 25.4 | 18.8 | 18.4 | 19.2 | 20.2 | 23.2 | 23.6 | 21.8 | 24.4 | 23.7 |

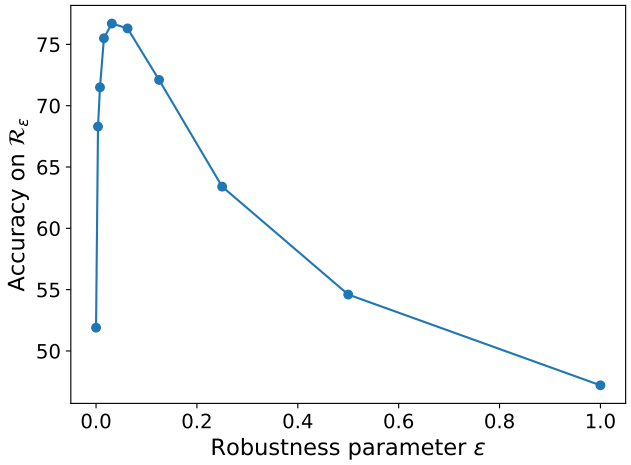



*Figure 7.* Accuracy of a standard ResNet20 classifier on robustifications of the ImageNet-9 test dataset, generated with respect to robust ResNet20 classifiers with varying robustness parameter $\varepsilon$. (Best viewed in color.)

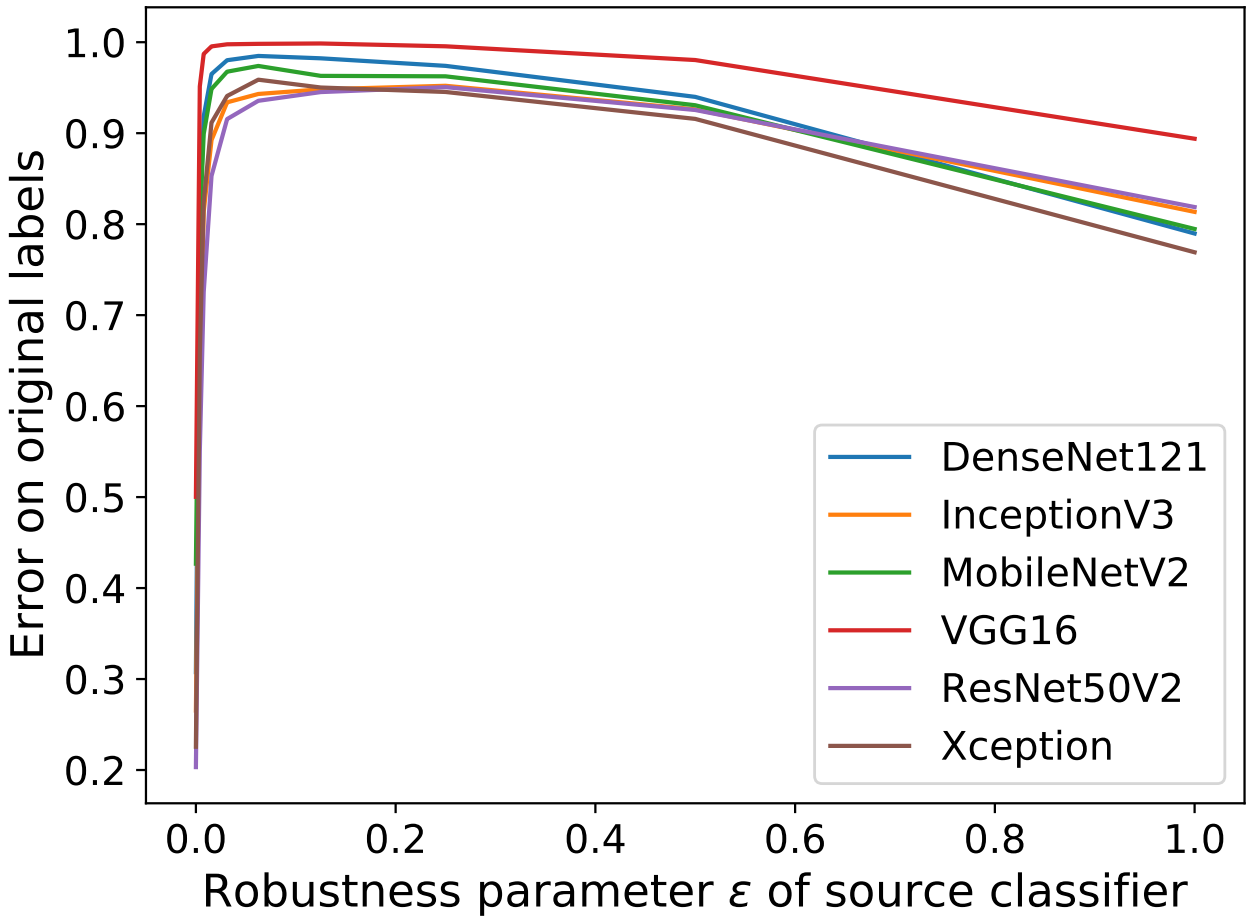



*Figure 8.* Error rate of different standard classifiers evaluated on untargeted transfer attacks. Each error rate is measured on a set of untargeted adversarial examples generated from the ImageNet-9 test set to attack a source classifier (ResNet20) with robustness parameter $\varepsilon$. Each adversarial example was generated with a perturbation $\delta$ where $\|\delta\|_2 < 2$. A higher error corresponds to a stronger transfer attack. (Best viewed in color.)

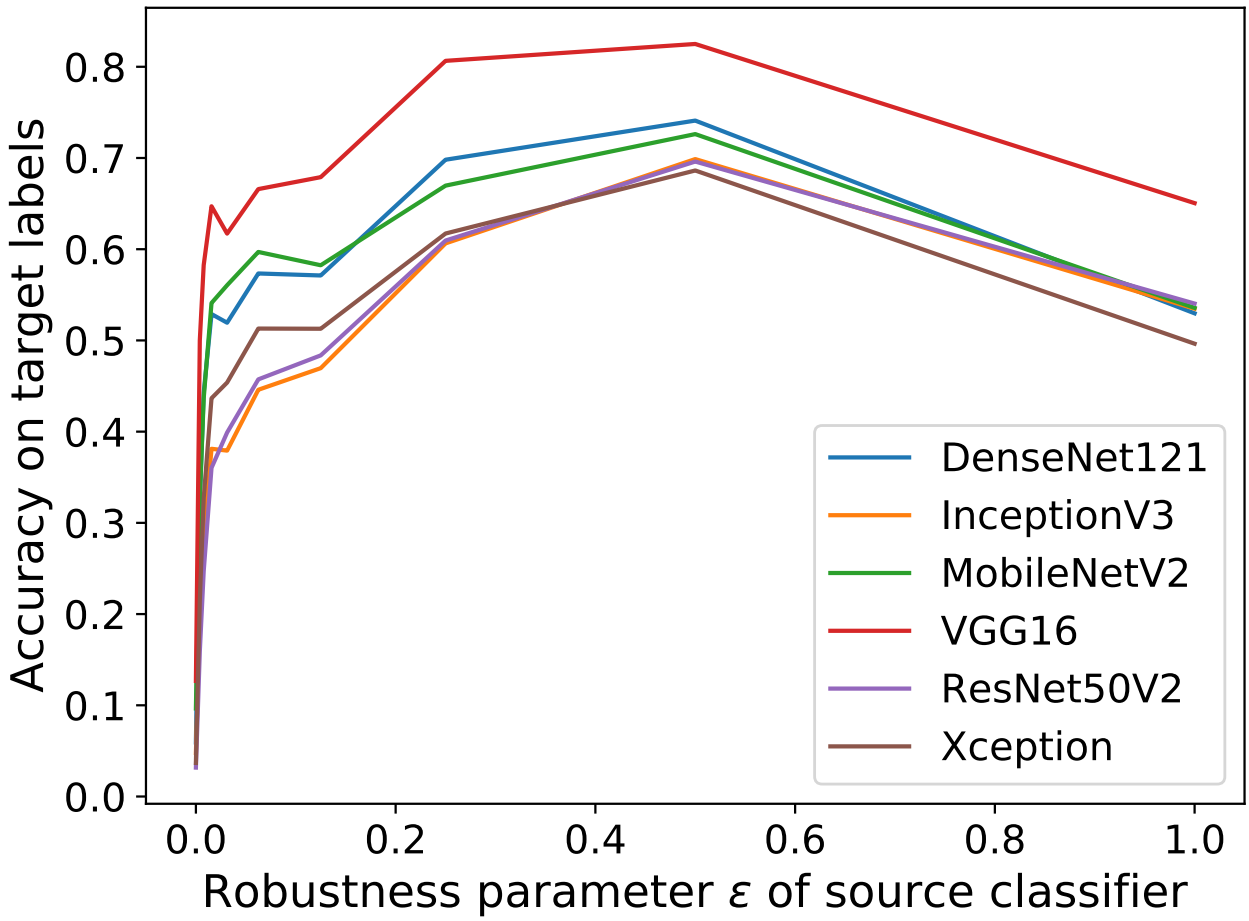



*Figure 9.* Accuracy of different standard classifiers on targeted transfer attacks. Each accuracy is measured as the fraction of times the target label $y_t$ is predicted for targeted adversarial example $x_t$. The evaluation is done on a set of targeted adversarial examples generated from the ImageNet-9 test set to attack a source ResNet20 classifier with robustness parameter $\varepsilon$. Each adversarial example was generated with a perturbation $\delta$ where $\|\delta\|_2 < 2$. A higher accuracy corresponds to a stronger transfer attack. (Best viewed in color.)

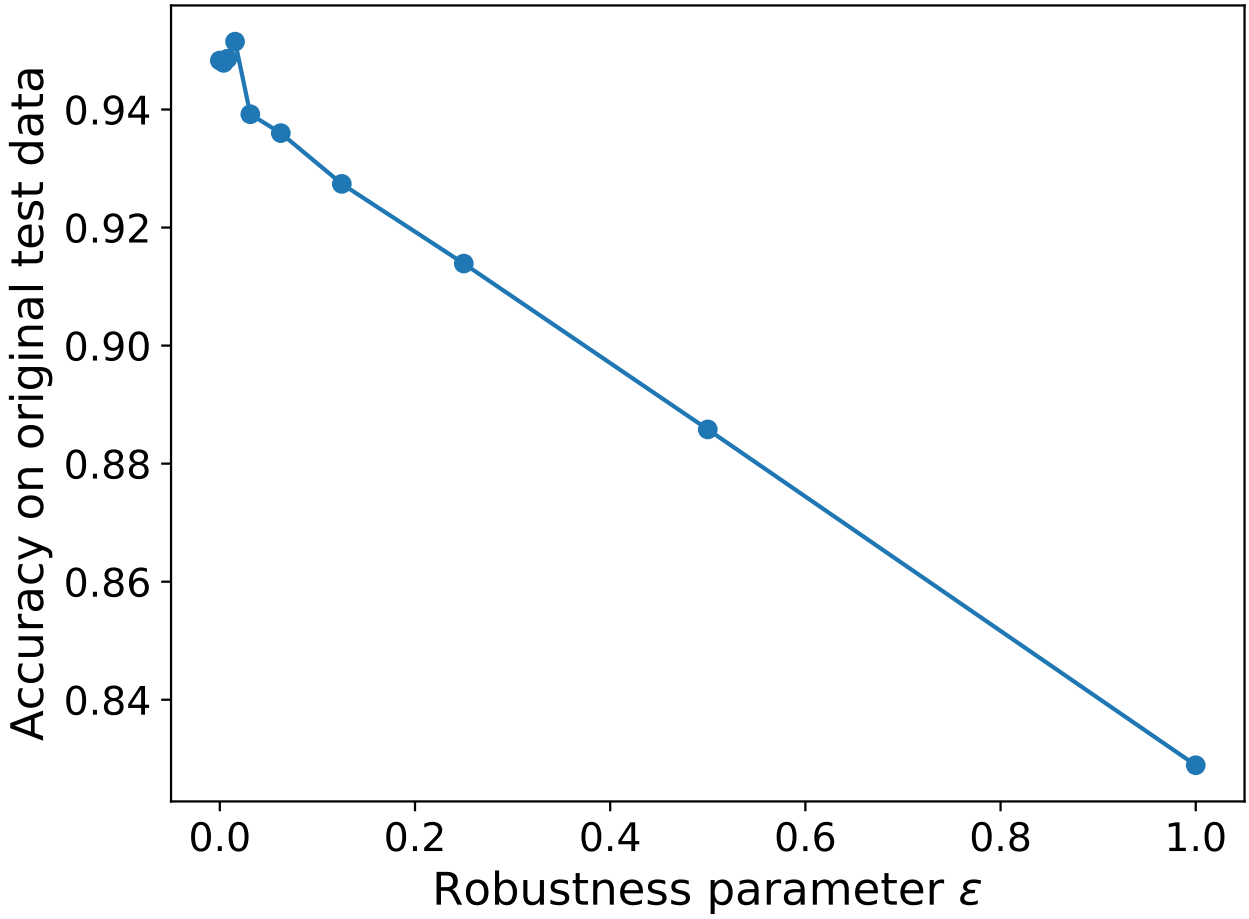



*Figure 10.* Accuracy of robust ResNet20s of varying robustness parameter trained on the ImageNet-9 training data and evaluated on the ImageNet-9 original test data. Confirms well known result: test accuracy decreases as robustness increases.

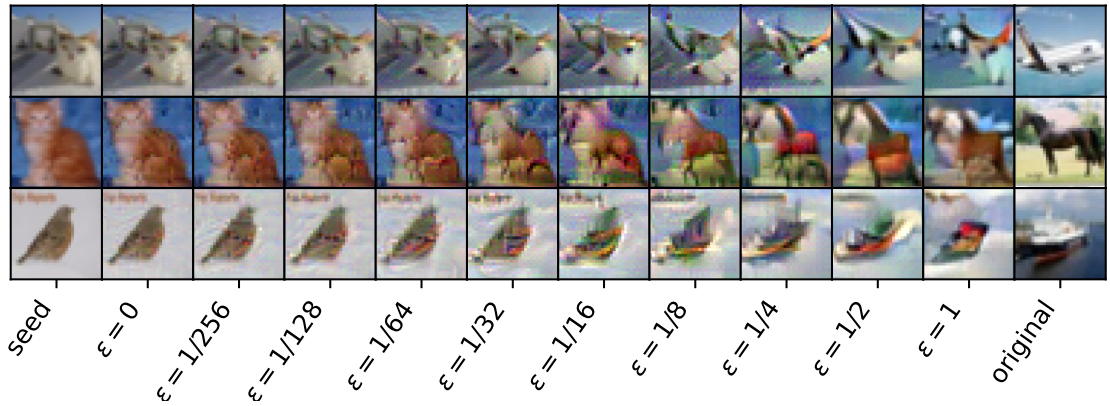



*Figure 11.* Extended data: Examples of robustified images by $\varepsilon$ of robust classifier. The rightmost column depicts the original CIFAR-10 image; the leftmost column depicts the seed ($x_0$) from which the gradient descent of the robustification process began; the rest are robustified images. The representation layer of an $\varepsilon$-robust classifier for each specified $\varepsilon$ is approximately the same for the robustified image and the associated original image. Since the representation layers of less robust classifiers are more easily perturbed, robustified images generated with respect to classifiers with a smaller robustness parameter appear closer to the (random) CIFAR-10 image that seeded the gradient-descent process. (Best viewed in color.)

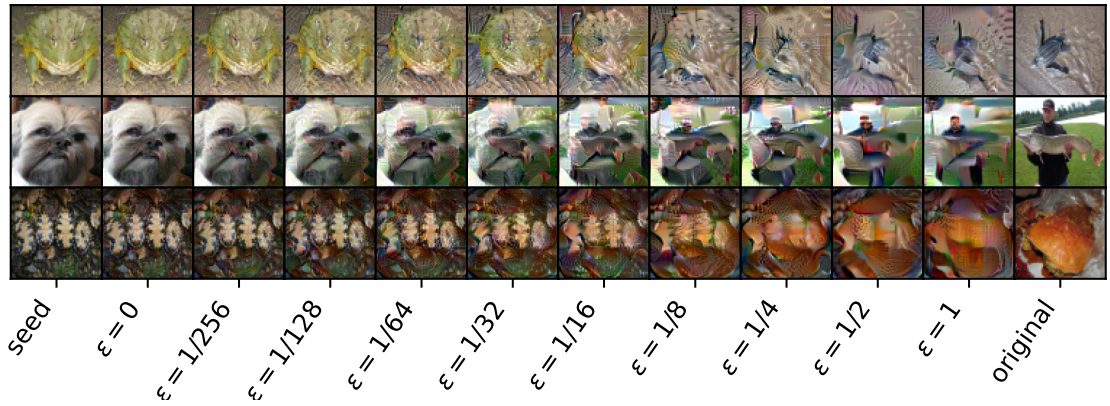



*Figure 12.* Extended data: Examples of robustified images by $\varepsilon$ of robust classifier. The rightmost column depicts the original ImageNet-9 image; the leftmost column depicts the seed ($x_0$) from which the gradient descent of the robustification process began; the rest are robustified images. The representation layer of an $\varepsilon$-robust classifier for each specified $\varepsilon$ is approximately the same for the robustified image and the associated original image. Since the representation layers of less robust classifiers are more easily perturbed, robustified images generated with respect to classifiers with a smaller robustness parameter appear closer to the (random) ImageNet-9 image that seeded the gradient-descent process. (Best viewed in color.)

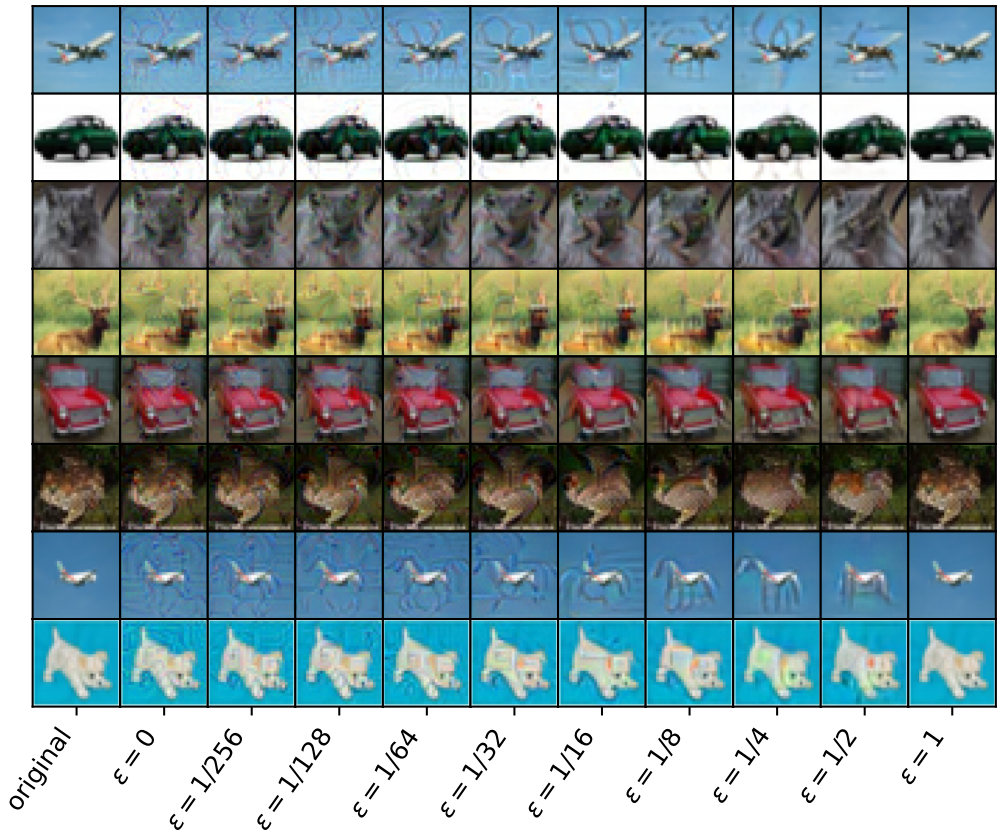



*Figure 13.* Extended data: Examples of adversarial examples for robust CIFAR-10 classifiers, of varying robustness. The left-most image is the original image from CIFAR-10. All other images are adversarial. Each adversarial example was generated via PGD from the original image $x$ such that the resulting adversarial example $\hat{x} = x + \delta$ where $\|\delta\|_2 \leq 2$ (Best viewed in color.)

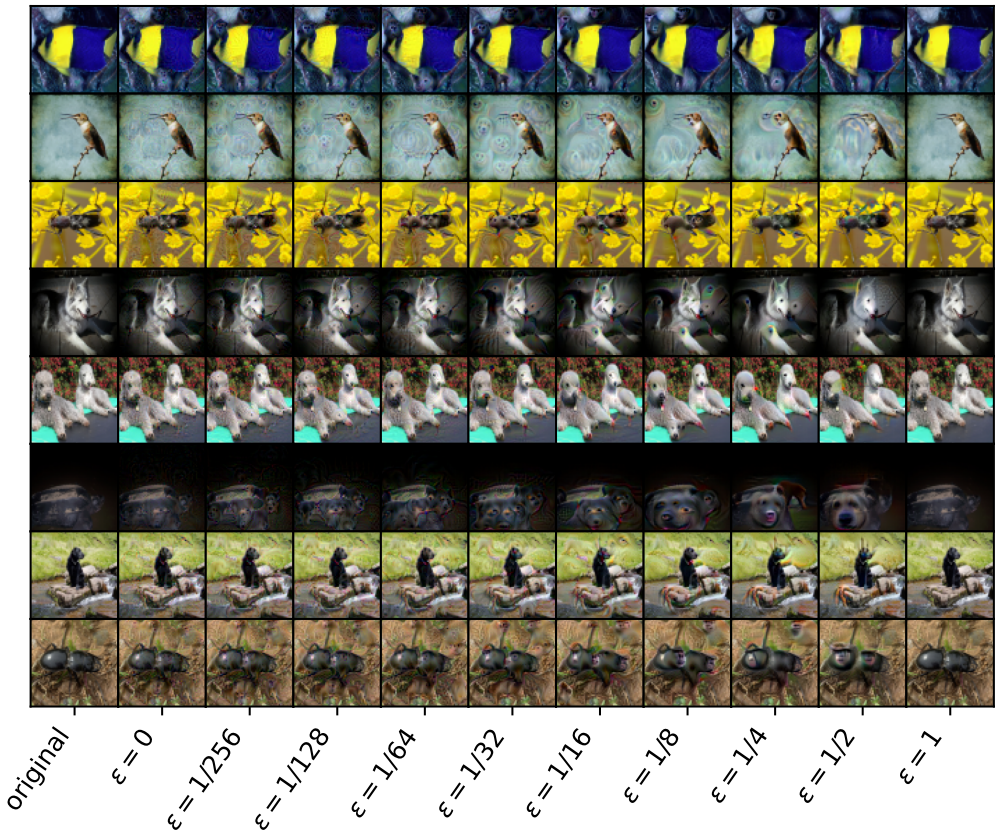



*Figure 14.* Extended data: Examples of adversarial examples for robust ImageNet-9 classifiers, of varying robustness. The left-most image is the original image from ImageNet-9. All other images are adversarial. Each adversarial example was generated via PGD from the original image $x$ such that the resulting adversarial example $\hat{x} = x + \delta$ where $\|\delta\|_2 \leq 4$ (Best viewed in color.)