# SGD ON NEURAL NETWORKS LEARNS ROBUST FEATURES BEFORE NON-ROBUST

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Neural networks are known to be vulnerable to adversarial attacks - small, imperceptible perturbations that cause the network to misclassify an input. A recent line of work attempts to explain this behavior by positing the existence of non-robust features - well-generalizing but brittle features present in the data distribution that are learned by the network and can be perturbed to cause misclassification.

In this paper, we look at the dynamics of neural network training through the perspective of robust and non-robust features. We find that there are two very distinct "pathways" that neural network training can follow, depending on the hyperparameters used. In the first pathway, the network initially learns only *predictive*, robust features and *weakly predictive* non-robust features, and subsequently learns predictive, non-robust features. On the other hand, a network trained via the second pathway eschews predictive non-robust features altogether, and rapidly overfits the training data. We provide strong empirical evidence to corroborate this hypothesis, as well as theoretical analysis in a simplified setting. Key to our analysis is a better understanding of the relationship between predictive non-robust features and adversarial transferability. We present our findings in light of other recent results on the evolution of inductive biases learned by neural networks over the course of training.

Finally, we digress to show that rather than being "quirks" of the data distribution, predictive non-robust features might actually occur across datasets with different distributions drawn from independent sources, indicating that they perhaps possess some meaning in terms of human semantics.

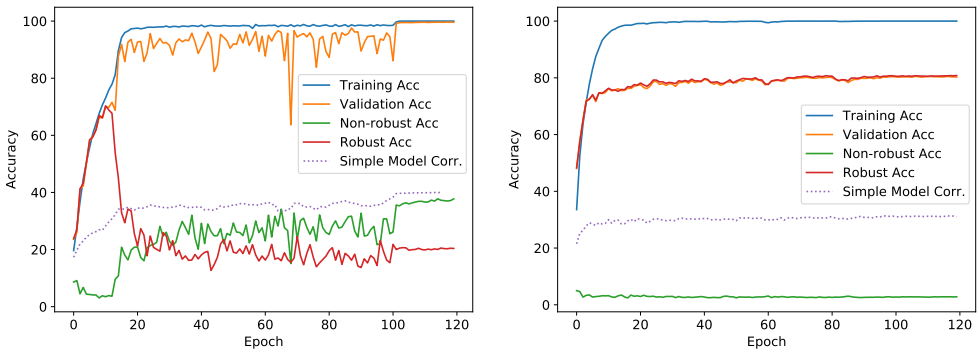
## 1 INTRODUCTION

Neural networks have achieved state of the art performance on tasks spanning an array of domains like computer vision, translation, speech recognition, robotics, and playing board games (Krizhevsky et al. (2012); Vaswani et al. (2017); Graves et al. (2013); Silver et al. (2016)). However in recent years, their vulnerability to adversarial attacks - small, targeted input perturbations, has come under sharp focus (Szegedy et al. (2013); Papernot et al. (2017); Carlini & Wagner (2017); Athalye et al. (2018); Schmidt et al. (2018)).

Ilyas et al. (2019) propose that neural network vulnerability is at least partly due to neural networks learning *well-generalizing* but *brittle* features that are properties of the data distribution. From this point of view, an adversarial example would be constructed by modifying an input of one class slightly such that it takes on the non-robust features of another class.

They provide empirical evidence for their theory by training a model on adversarially perturbed examples labeled as the *target class*, and showing that this model generalizes well to the original, unperturbed distribution.

Another unrelated line of work (Brutzkus et al. (2018); Ji & Telgarsky (2019); Li & Liang (2018)) aims to study the properties of the functions learned by gradient descent over the course of training. Nakkiran et al. (2019) and Mangalam & Prabhu (2019) independently showed that Stochastic Gradient Descent (SGD) learns simple, almost linear functions to start out, but then learns more complex functions as training progresses. Li et al. (2019) showed that models trained with a low learning rate learn well-generalizing but hard-to-fit features first, and thus perform poorly on easier-to-fit patterns.




(a) Pathway 1 (Normal Training)

(b) Pathway 2 (Overfitting)

Figure 1: (Best viewed in color). Neural network training follows two very different pathways based on the choices of hyperparameters. These are training graphs of two Resnet50 models trained on relabeled CIFAR-10 adversarial examples. See Section 4 for more details.

In this paper, we study gradient descent on neural networks from the perspective of robust and non-robust features. Our main thesis is that based on choices of hyperparameters, neural network training follows one of two pathways, :

- **Pathway 1 (Informal) :** The neural network first learns *predictive* robust features and *weakly predictive* non-robust features. As training progresses, it learns predictive *non-robust* features, and having learned both robust and non-robust predictive features, achieves good performance on held-out data. This is the pathway that Ilyas et al. (2019) used to prove their theory.
- **Pathway 2 (Informal) :** The neural network learns *predictive* robust features and *weakly predictive* non-robust features (as in Pathway 1). But thereafter, it begins to fit the *noise* in the training set, and quickly achieves zero training error. In this scenario, the network learns only the *robust* predictive features and shows modest generalization on held-out data.

Through a series of experiments, we validate our two-pathway hypothesis, investigate the specific circumstances under which Pathway 1 and Pathway 2 occur, and analyze some properties of the two pathways. We will also develop a closer understanding of the relationship between adversarial transfer and *predictive* non-robust features, which will aid our analysis of the two pathways.

The rest of this paper is organized as follows. Section 2 sets up the notation and definitions we use. In Section 3, we investigate the link between adversarial features and transferability. In Section 4 we provide empirical evidence for the two-pathway hypothesis and analyze some characteristics of each pathway. Section 5 presents a theoretical analysis of gradient descent on a toy linear model. We show that for different choices of initial parameters, the linear model exhibits properties of the first and second pathways. We digress to explore whether non-robust features can occur *across datasets* in Section 6, and discuss future research directions in Section 7.

## 2 Definitions and Preliminaries

Consider the binary classification setting, where $\mathcal{D}$ is a joint distribution over the input space $\mathcal{X}$ and the labels $\{-1, 1\}$. In this setting, Ilyas et al. (2019) define a *feature* as any function $f : \mathcal{X} \to \mathbb{R}$, scaled such that $\mathbb{E}_{(x,y)\in\mathcal{D}}[f(x)] = 0$ and $\mathbb{E}_{(x,y)\in\mathcal{D}}[f(x)^2] = 1$. A feature is said to be *$\rho$-useful* if

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

for some $\rho > 0$, and *$\gamma$-robust* if

$$\mathbb{E}_{(x,y)\in\mathcal{D}}\left[\inf_{\delta\in\Delta(x)} y \cdot f(x+\delta)\right] > \gamma$$

for some $\gamma > 0$ and some family of perturbations $\Delta$. For brevity, we sometimes refer to a $\rho$-useful, $\gamma$-robust feature ($\rho, \gamma > 0$) simply as a *robust feature*. Let $\rho_{\mathcal{D}}(f)$ be the largest $\rho$ for which $f$ is $\rho$-useful. A feature $f$ is said to be (highly) predictive or weakly predictive if $\rho_{\mathcal{D}}(f)$ is high or low respectively.

**A better definition of a non-robust feature :** A useful, non-robust feature as defined by Ilyas et al. (2019) is one that is $\rho$-useful for some $\rho > 0$, but is not $\gamma$-robust for any $\gamma > 0$. Consider the following scenario :

$$\mathbb{E}[y \cdot f_{NR}(x)] > 0 \qquad\qquad \mathbb{E}\left[\inf_{\delta \in \Delta(x)} y \cdot f_{NR}(x+\delta)\right] \ll 0$$

Note that we omit the $(x, y) \in \mathcal{D}$ for brevity while taking the expectation. When one trains a model on $(x + \delta, -y)$ pairs, the model can be expected to learn $f_{NR}$ since

$$\mathbb{E}\left[\inf_{\delta \in \Delta(x)} -y \cdot f_{NR}(x+\delta)\right] \gg 0$$

Hence, such a model would generalize well on the original distribution. This is the experiment that Ilyas et al. (2019) perform with their $\mathcal{D}_{det}$ and $\mathcal{D}_{rand}$ datasets to demonstrate the existence of non-robust features. Note that the fact that the model learns $f_{NR}$ is crucial, otherwise it would fail to generalize.

Goh (2019a) points out a subtle flaw with the above definition - highly predictive non-robust features can arise from "contamination" of a robust feature with a non-robust feature, instead of something meaningful. To see how this can happen, consider a *highly predictive* robust feature $f_R$ and a *weakly predictive* non-robust feature $f_{NR}$. Let $f_C$ be a "contaminated" feature that is a simple sum of $f_R$ and $f_{NR}$ (appropriately normalized). Then,

$$\mathbb{E}[y \cdot f_R(x)] > 0 \qquad\qquad \mathbb{E}\left[\inf_{\delta \in \Delta(x)} y \cdot f_R(x+\delta)\right] > 0$$

$$\mathbb{E}[y \cdot f_{NR}(x)] \gtrsim 0 \qquad\qquad \mathbb{E}\left[\inf_{\delta \in \Delta(x)} y \cdot f_{NR}(x+\delta)\right] \ll 0$$

$$\mathbb{E}[y \cdot f_C(x)] > 0 \qquad\qquad \mathbb{E}\left[\inf_{\delta \in \Delta(x)} y \cdot f_C(x+\delta)\right] < 0$$

$f_C$ is thus a highly predictive non-robust feature. Now when you train a model on $(x + \delta, -y)$ pairs, $f_C = f_R + f_{NR}$ is still correlated with $-y$. But $f_{C'} = -f_R + f_{NR}$ is more correlated, so the model will learn this combination in preference to $f_C$ and will not generalize on the original distribution. In fact, thanks to learning $-f_R$, it will generalize to the distribution with flipped labels, i.e., $y \to -y$. To exclude such "contaminants", we propose the following alternate definition of a non-robust feature.

**Definition 1** (Non-robust feature)**.** *A feature $f$ is a useful, non-robust feature if it is $\rho$-useful for some $\rho > 0$, not $\gamma$-robust for any $\gamma > 0$, and there does not exist a $\rho$-useful, $\gamma$-robust feature $g$ such that*

$$\mathbb{E}_{(x,y)\in\mathcal{D}}[f(x) \cdot g(x)] \geq 1 - \epsilon$$

*for some small $\epsilon > 0$.*

Intuitively, if a feature is not $\gamma$-robust but its performance on the data distribution can be explained to a large extent by a robust feature, then we classify it as a contaminated robust feature and not a true non-robust feature. Note that $f$ and $g$ are normalized, so the expectation of their product is upper-bounded by 1.

This definition also excludes cases where the non-robust component of the feature is very predictive but just *happens* to be correlated strongly with a robust feature. While we acknowledge this as a flaw in our definition, we reason that if the two features are indeed so similar, then there is no real utility to the non-robust feature and it can be safely excluded.

**Remark :** A network that learns predictive robust and weakly predictive non-robust features can still be very vulnerable to adversarial attacks, as the above example shows. The weakly predictive non-robust features can be highly predictive of the *opposite* label on out-of-distribution examples.

## 3 NON-ROBUST FEATURES AND TRANSFERABILITY

The phenomenon of adversarial transferability (Papernot et al. (2016)), where adversarial examples generated for one model are still adversarial (to a limited but non-trivial extent) to other models trained independently on the same data, can be readily explained in terms of non-robust features. If the data contain highly predictive non-robust features, then any model trained on these data can be expected to learn all or some of these features. One would construct an adversarial example by modifying an input belonging to one class such that it takes on the predictive non-robust features of another class.

Nakkiran (2019) showed that by explicitly penalizing transferability, one can construct adversarial examples that do not transfer, and from which it is not possible to learn a generalizing model. So these adversarial examples must not contain predictive non-robust features.

Here we provide a simpler experiment that constructs non-transferable adversarial examples without explicitly penalizing transferability, and also strengthens the connection between predictive non-robust features and transferability.

We take two trained Resnet50 models (He et al. (2016)) on CIFAR-10 and construct targeted adversarial examples for the first network by shifting the labels by one, i.e., $y \to (y+1)\%10$. We use standard Projected Gradient Descent (PGD) (Madry et al. (2018)) to construct these examples. We split them into two categories - those that transfer to the second network *with* their target labels, and those that do not. We relabel all adversarial examples with their target label, and train a Resnet50 model on relabeled adversarial examples from each category.

Our results are in Figure 2. We can see that the network trained on the examples that transfer generalizes well to the true distribution, but the network trained on the examples that do not transfer does not generalize. In fact, it generalizes well to the *shifted* distribution, indicating that it has learned the robust feature association. The configuration in the figure is as a result of a thorough grid search over hyperparameters with the metric for success being performance on the true distribution.

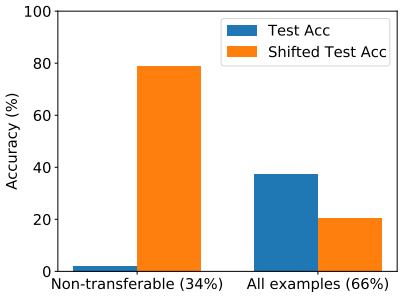



Figure 2: Accuracy of a Resnet50 model trained on subsets of *relabeled* adversarial examples

This indicates that the neural network is unable to find any predictive non-robust features of the original distribution in the examples that don't transfer. Hence, standard PGD even without an adversarial transfer penalty finds some adversarial examples with identical characterstics to those found by Nakkiran (2019).

If we accept a neural network being unable to learn a generalizing model as evidence that there are no highly predictive non-robust features, then our experiments strengthen the connection between adversarial transferability and highly predictive non-robust features. We shall use this connection in our analysis in the following section.

## 4 THE TWO PATHWAY HYPOTHESIS

### 4.1 EXPERIMENTAL SETUP

Train a neural network on the CIFAR-10 dataset, and generate adversarial examples by perturbing each example to the next class, i.e., $y \to (y+1)\%10$. Verify that some non-trivial fraction of these adversarial examples transfer to another neural network with their target labels, indicating that they contain predictive non-robust features. Relabel each adversarial example with its target label and call this $\mathcal{D}_{det}$. Split $\mathcal{D}_{det}$ into training and validation sets and train a new neural network on the training set.

At an intermediate stage in the training of this model, if it is able to generalize to the true unperturbed test set, then it must have learned the *predictive non-robust features* present in the adversarial examples.

Also, note that the robust features in the adversarial examples are predictive of the *shifted* label. Hence if the model is able to generalize to the shifted un-perturbed test set with $y \to (y+1)\%10$ but $x$ unchanged, then it has learned *predictive robust features* (possibly with non-robust contaminants).

We use the accuracy on the test set (respectively, shifted test set) as a proxy for how much of the model's performance can be attributed to its learning predictive non-robust (respectively, robust) features. We refer to these as "non-robust accuracy" and "robust accuracy".

Finally, the accuracies on the training and validation splits of $\mathcal{D}_{det}$ tell us how well the model has *fit the training data*, and whether the model is *overfitting*. We train the network using SGD for 120 epochs using different combinations of learning rate and regularization, and plot the evolution of these four metrics over the course of training.

### 4.2 Results and Discussion

We observe that training follows two very distinct regimes or pathways depending on the choice of hyperparameters.

The **first pathway**, illustrated in Figure 1a occurs when the model is trained with regularization of some sort - either in the form of a high initial learning rate (LR), $L_2$ weight decay, or data augmentation. The model starts out by learning only predictive robust features (possibly with contaminants), but at some point switches to learning a combination of robust and non-robust predictive features. Training and validation accuracy steadily increase, and the model ends with both training and validation accuracy close to $100\%$.

The **second pathway**, illustrated in Figure 1b occurs when the model is trained with a low starting learning rate, little or no $L_2$ weight decay and no data augmentation. The model starts out similar to the first pathway, but then starts overfitting the training data before it can learn non-robust predictive features. At this point, validation accuracy stagnates. The model finishes with a training accuracy of $100\%$ but a validation accuracy of $81\%$.

We observe that the non-robust accuracy remains stubbornly close to zero throughout, and that the robust accuracy is very close to the validation accuracy. As indicated by its high robust accuracy, this model generalizes well to the original unperturbed test set. This suggests that nearly all the performance of the model can be attributed to its learning predictive robust features.

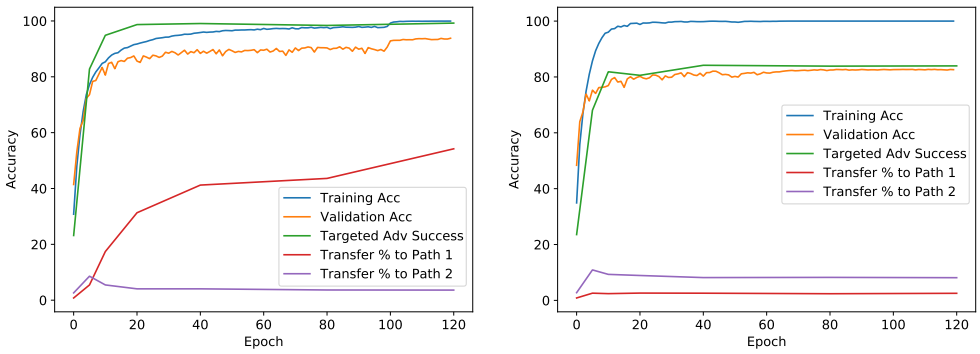
**Is there a middle ground? :** At least on the $\mathcal{D}_{det}$ datasets, we observe that the model makes a *sharp* transition from Pathway 1 to Pathway 2 in the space of hyperparameters, and there does not seem to be a middle ground. For details, refer to Section C of the Appendix.

**Transferability :** If our assumptions are correct and consistent, then a model trained via the second pathway does not learn *any* predictive non-robust features. Given the strong connection between adversarial transferability and predictive non-robust features, adversarial examples generated for this model should not transfer. Sure enough, we find that we can construct adversarial examples for this model with a very high targeted success rate of **99.96%**, but these examples transfer to a model trained on the original CIFAR-10 dataset with a targeted success rate of only **0.6%**.

**On clean data :** A natural question to ask here is, are these two pathways peculiar to our perturbed and shifted dataset? Do models trained on real-world datasets follow these two pathways too? We train two models on the unaltered CIFAR-10 dataset, one with both data augmentation and some weight decay, and one with neither. The training and validation accuracies over the course of training for these two models are shown in Figure 3a and Figure 3b.

Simultaneously, we also plot the targeted adversarial attack success, as well as the transfer accuracy to two other models trained (with different random initializations) via Pathway 1 and Pathway 2. We observe that adversarial examples generated for the Pathway 1 model have a very high attack success rate on the same model, and transfer to the other Pathway 1 model with reasonable success. Adversarial examples generated for the Pathway 2 model fail to transfer to any of the other models.

Once again, we use the connection between adversarial transferability and non-robust features to conclude that the model trained by Pathway 1 has learned predictive non-robust features, and the model trained by Pathway 2 has not. We remark that learning predictive non-robust features seems to be essential for good generalization on the validation set, and robust features alone do not suffice.




(a) Pathway 1 (Normal Training)

(b) Pathway 2 (Overfitting)

Figure 3: (Best viewed in color). Training and validation accuracy for a model trained on CIFAR-10, along with accuracy of targeted adversarial attacks and transfer accuracy to other identical models trained via Pathway 1 and Pathway 2.

**Remark 1 :** Although by our definitions, a model that learns only robust features can still be very vulnerable due to the presence of non-robust contaminants, in practice training via Pathway 2 seems to confer some non-trivial robustness.

**Remark 2 :** Although Figure 1a suggests that the model eventually generalizes *better* to the non-robust feature mapping than the robust, this is not a generally applicable rule over different datasets, architectures and combinations of hyperparameters. Table 4 in the Appendix illustrates this point.

### 4.3 Relation to Other Results about Neural Network Training

**Low and high learning rates :** The regularizing effect of a high initial learning rate has been studied in detail by Li et al. (2019). They classify the features learned by a network into two types - those that are *hard to fit* but *well-generalizing*, and those that are *easy to fit* but *not so well-generalizing*.

They show that a neural network trained with small learning rate first focuses on the hard-to-fit features, and because they are well-generalizing, quickly overfits to the training set. As a result, it is not able to learn easy-to-fit features effectively later on. In contrast, a model that starts out with a high learning rate learns easy-to-fit features first, and since these are not as well generalizing, doesn't overfit the training set. Later on, once the learning rate is annealed, the model is able to effectively learn the harder-to-fit features.

These two cases can be crudely mapped onto our two pathways. The model in Pathway 2, trained with a low LR, learns only robust features to start out, indicating that these features are hard to fit but well-generalizing. It overfits the training set and thereafter is unable to learn the non-robust features, which are easy to fit.

The model in Pathway 1, trained with a high LR, quickly begins to learn the non-robust features which are easy to fit. However, it learns the robust features too side-by-side, indicating that this mapping from the low and high LR scenarios to our two pathway theory is not perfect.

**Complexity of learned functions :** Another perspective on the training of neural networks is given by Nakkiran et al. (2019). They define a performance correlation metric between two models that captures *how much of the performance of one model can be explained by the other*, and show that as training progresses, the peformance correlation between the current model and the best simple model decreases. This indicates that the functions learned by a neural network become increasingly complex as training progresses.

Although their metric is defined for a binary classification setting, we adapt it for the multi-class setting, and use a multi-class logistic regression classifier as the "best simple classifier". We measure the performance correlation between the model trained on $\mathcal{D}_{det}$ and the simple classifier as training

progresses. The performance correlation, scaled by 100 and smoothed, is shown in Figure 1 along with the training curves.

We observe in Figure 1a that the point at which the performance correlation plateaus corresponds with when the robust accuracy decreases sharply. Similarly in Figure 1b, the correlation levels off along with the robust accuracy. We conjecture that the initial robust features learned by the model are simple, linear functions, and the non-robust features are more complex and non-linear. This is in line with the findings of Tramèr et al. (2017) that transferable adversarial examples occur in a space of high dimensionality.

## 5 Theoretical Analysis

In this section, we present some results for gradient descent on a toy linear model on a distribution with robust and non-robust predictive features. We get results that mirror our two pathways, for different choices of initial conditions. The setting we use is an adaptation of the one used by Nakkiran et al. (2019).

Define a data distribution $P$ over $\mathbb{R}^d \times \{-1, 1\}$ as follows :

$$y \overset{u.a.r}{\sim} \{-1, 1\}, \quad q \overset{u.a.r}{\sim} \{3, ..., d\}$$

$$\lambda_1 \sim \text{Bernoulli}(p) \text{ over } \{\pm 1\}, \quad \lambda_2 \sim \text{Bernoulli}(p/k) \text{ over } \{\pm 1\}, \quad p \in (0, 1/2), \quad k > 1$$

$$\boldsymbol{x} = y\lambda_1 \boldsymbol{e_1} + \epsilon y \lambda_2 \boldsymbol{e_2} + \boldsymbol{e_q}$$

where $\epsilon < 1$ is some small positive constant, and $\boldsymbol{e_i}$ denotes the $i^{\text{th}}$ natural basis vector. Now sample a "training set" $(X_1, Y_1), ..., (X_n, Y_n)$ from this distribution. Consider the overparameterized setting, where $n \ll d$. So with high probability, the coordinates $\{3, ..., d\}$ of the training data are orthogonal for all the training points.

The idea is that the data consists of a "robust" feature given by the first coordinate, a "non-robust" feature (one that can become anti-correlated with a perturbation of $2\epsilon$) given by the second coordinate, and a noisy component that comprises the rest of the coordinates, making it possible for a model to fit the data exactly. The robust component is predictive of the true label with probability $1 - p$, and the non-robust component is predictive of the true label with probability $1 - (p/k)$.

Let $A = [X_1^T; ...; X_n^T] \in \mathbb{R}^{n \times d}$ and $B = [Y_1, ..., Y_n]^T \in \mathbb{R}^n$. Consider training a linear classifier using gradient descent with a learning rate of $\eta$ to minimize the squared loss :

$$\mathcal{L}(\boldsymbol{w}) = \frac{1}{2n} \|B - A\boldsymbol{w}\|_2^2, \quad \boldsymbol{w} \in R^d$$

For simplicity, assume that the initial weight vector $\boldsymbol{w}_0 = \mathbf{0}$, and that $n$ is sufficiently large.

**Theorem 1** (Robust before Non-robust)**.** *If $\epsilon \leq \sqrt{(1-2p)/(1-2p/k)}$, then at the end of the first step, with high probability, the model will rely on the robust feature for classification,, i.e., $\boldsymbol{w}_1^{(1)} \geq \epsilon \boldsymbol{w}_1^{(2)}$, and will have a population accuracy of $1 - p$.*

**Theorem 2** (Two Pathways)**.** *Define*

$$k_t = \frac{2p(1 + n\epsilon^2) - 2p(1 - \epsilon)}{2p(1 + n\epsilon^2) - (1 - \epsilon)}$$

*Then if $\eta \leq 2/(1 + \epsilon^2 + (1/n))$, as the number of gradient steps goes to infinity,*

- *if $k \geq k_t$, sample accuracy approaches $1$ and population accuracy approaches $1 - (p/k)$ with high probability.*
- *if $k < k_t$, sample accuracy approaches $1$ and population accuracy approaches $1 - p$ with high probability.*

*Discussion :* The two cases of Theorem 2 very roughly correspond to Pathways 1 and 2. Since this is a strongly convex problem, gradient descent with a small enough learning rate will converge to a fixed solution, so we cannot mimic the setting where different training hyperparameters lead to Pathway 1 or 2. But we can see that if the non-robust feature is predictive enough, the model learns the non-robust feature, otherwise it learns the robust feature.

For proofs of these theorems, refer to Section A of the Appendix.

| Trained On | On CIFAR-minus-Flickr | On CINIC-10 |
|---|---|---|
| CIFAR-minus-Flickr | 93.4 | 65.7 |
| CINIC-10 | 82.1 | 84.6 |
| CIFAR-minus-Flickr $\mathcal{D}_{det}$ | 33.7 | **20.5** |
| CINIC-10 $\mathcal{D}_{det}$ | **19.6** | 30.1 |

Table 1: Accuracy of Resnet50 models on the CIFAR-minus-Flickr and CINIC-10 test sets. The two numbers in bold are the ones to focus on.

## 6 DIGRESSION : CROSS-DATASET TRANSFER

Our current understanding of non-robust features is that they are peculiarities or quirks in the data distribution. We provide evidence that allows us to tentatively refute that assumption by showing that one can construct two datasets from completely independent sources, and a model that learns only the predictive non-robust features of one dataset can achieve non-trivial generalization on the other dataset.

The CINIC-10 dataset Darlow et al. (2018) is a distribution-shifted version of CIFAR-10 constructed by sampling from the ImageNet synsets corresponding to each of the CIFAR-10 classes. Although it may seem like CIFAR-10 and CINIC-10 could be candidates for two datasets drawn from independent sources, ImageNet is constructed by querying Flickr, and Flickr is also one of the seven sources for the 80 million TinyImages dataset (Torralba et al. (2008)) that was used to construct CIFAR-10 (Krizhevsky et al. (2009)). So roughly one in seven CIFAR-10 images is from Flickr.

To be even more certain that there are no spurious correlations creeping in because of a common source, we construct the CIFAR-minus-Flickr dataset that consists of those CIFAR-10 images that haven't been sourced from Flickr. This comprises 52172 out of the 60,000 CIFAR-10 images.

We construct $\mathcal{D}_{det}$ datasets as described in Section 4 for CIFAR-minus-Flickr and CINIC-10, and train Resnet50 models on them. These models can only learn non-robust features to help them generalize to the original unperturbed datasets, because the robust features are correlated with the *shifted* labels.

*Remark :* Goh (2019b) showed that the $D_{rand}$ dataset constructed by perturbing each image to a *random* target class and labeling it as that class, suffers from a sort of "robust feature leakage". PGD introduces faint robust cues in the generated adversarial example that can be learned by the model. But on the $\mathcal{D}_{det}$ dataset, the robust features are correlated with the *shifted* label, which is enough to drown out any correlation with the target label through leakage. Thus, we can be reasonably confident that the only features available to the model are non-robust.

The results are shown in Table 1. Both $\mathcal{D}_{det}$ trained models achieve an accuracy of close to 20% on the other dataset, which is a long way from the expected 10% accuracy of a random model.

## 7 CONCLUSION AND FUTURE DIRECTIONS

In this paper, we've shown that from the perspective of predictive robust and non-robust features, neural network training follows two very different pathways, corresponding to the *training* and *overfitting* regimes. In both regimes, the model starts out by learning predictive robust features first.

Some questions that arise naturally from our work are understanding precisely why this behavior occurs. What characteristics do predictive non-robust features that ensure that they are learned only subsequent to predictive robust features? We pose finding a more precise definition of non-robust features that will allow us to theoretically analyze and explain these properties as an important direction for future work.

Finally, as we show in the last section, predictive non-robust features can occur across datasets sampled from independent sources. Although this needs to be investigated more thoroughly, our results allow us to speculate that they could have a meaning in terms of human semantics. As an illustrative example of such a kind of non-robust feature, suppose that most dogs' eyes were a particular shade of brown, but most cats' eyes were a *slightly* lighter shade of brown.

## REFERENCES

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pp. 274–283, 2018.

Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns overparameterized networks that provably generalize on linearly separable data. In *International Conference on Learning Representations*, 2018.

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

Luke N. Darlow, Elliot J. Crowley, Antreas Antoniou, and Amos J. Storkey. Cinic-10 is not imagenet or cifar-10, 2018.

Gabriel Goh. A discussion of 'adversarial examples are not bugs, they are features': Two examples of useful, non-robust features. *Distill*, 2019a. doi: 10.23915/distill.00019.3. https://distill.pub/2019/advex-bugs-discussion/response-3.

Gabriel Goh. A discussion of 'adversarial examples are not bugs, they are features': Robust feature leakage. *Distill*, 2019b. doi: 10.23915/distill.00019.2. https://distill.pub/2019/advex-bugs-discussion/response-2.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. IEEE, 2013.

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

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 125–136. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8307-adversarial-examples-are-not-bugs-they-are-features.pdf.

Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pp. 1772–1798, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pp. 8157–8166, 2018.

Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 11674–11685. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9341-towards-explaining-the-regularization-effect-of-initial-large-learning-rate-in-training-neural-networks.pdf.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Karttikeya Mangalam and Vinay Uday Prabhu. Do deep neural networks learn shallow learnable examples first? 2019.

Preetum Nakkiran. A discussion of 'adversarial examples are not bugs, they are features': Adversarial examples are just bugs, too. *Distill*, 2019. doi: 10.23915/distill.00019.5. https://distill.pub/2019/advex-bugs-discussion/response-5.

Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity. *arXiv preprint arXiv:1905.11604*, 2019.

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

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 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.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pp. 5014–5026, 2018.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

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

Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008.

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

## A Proofs of Theorems in Section 5

Consider using Gradient Descent with a learning rate of $\eta$ to minimize the squared loss as described in Section 5. Then,

$$\boldsymbol{w}_{t+1} := \boldsymbol{w}_t + \frac{1}{n}\eta A^T(B - A\boldsymbol{w}_t)$$

Letting $\alpha = \eta/n$, it can be proved by induction that

$$\boldsymbol{w}_t = \boldsymbol{w}_0 + \alpha A^T \left[\sum_{k=0}^{t-1}(I - \alpha AA^T)^k\right](B - A\boldsymbol{w}_0)$$

Let the largest eigenvalue of $AA^T$ be $\lambda_{max}$. If $|1 - \alpha\lambda_{max}| \leq 1$, then

$$w_t = w_0 + A^T(AA^T)^{-1}\left[I - (I - \alpha AA^T)^t\right](B - Aw_0)$$

$$\implies \boldsymbol{w}_T = A^T(AA^T)^{-1}(B - A\boldsymbol{w}_0) + \boldsymbol{w}_0 \tag{1}$$

for some very large number of steps $T$. This achieves zero empirical training error, as we can verify that $A\boldsymbol{w}_T = B$.

Let the first and second column of $A$ be $\boldsymbol{s}$ and $\boldsymbol{r}$ respectively. Since (with high probability) coordinates 3 to $d$ are orthogonal for all training points,

$$AA^T = I + \boldsymbol{s}\boldsymbol{s}^T + \boldsymbol{r}\boldsymbol{r}^T$$

Other than 1, the eigenvalues of this matrix are

$$1 + \frac{(s^Ts + r^Tr) \pm \sqrt{(s^Ts - r^Tr)^2 + 4(s^Tr)^2}}{2}$$

And the corresponding eigenvectors are

$$(s^Tr)\boldsymbol{s} + \left[\frac{(-s^Ts + r^Tr) \pm \sqrt{(s^Ts - r^Tr)^2 + 4(s^Tr)^2}}{2}\right]\boldsymbol{r}$$

It's easy to see that $\boldsymbol{s}^T\boldsymbol{s} = n$, $\boldsymbol{r}^T\boldsymbol{r} = n\epsilon^2$. Let $\boldsymbol{s}^T\boldsymbol{r} = \boldsymbol{r}^T\boldsymbol{s} = n\epsilon\beta$. Then since $\epsilon$ is small,

$$\begin{aligned}\sqrt{(s^Ts - r^Tr)^2 + 4(s^Tr)^2} &= n\sqrt{(1-\epsilon^2)^2 + 4(\epsilon\beta)^2} \\ &\approx n(1 + \epsilon^2(2\beta^2 - 1))\end{aligned}$$

Then the eigenvalues and eigenvectors are

$$\lambda_1 = 1 + n(1 + \epsilon^2\beta^2), \quad \lambda_2 = 1 + n\epsilon^2(1 + \beta^2)$$

$$\boldsymbol{v}_1 = n(\epsilon\beta)\boldsymbol{s} + n(\epsilon^2\beta^2)\boldsymbol{r}, \quad \boldsymbol{v}_2 = n(\epsilon\beta)\boldsymbol{s} + n(-1 + \epsilon^2 - \epsilon^2\beta^2)\boldsymbol{r}$$

**Theorem 1 :** *If $\epsilon \leq \sqrt{(1-2p)/(1-2p/k)}$, then at the end of the first step, with high probability, the model will rely on the robust feature for classification,, i.e., $\boldsymbol{w}_1^{(1)} \geq \epsilon\boldsymbol{w}_1^{(2)}$, and will have a population accuracy of $1-p$.*

*Proof.*

$$\boldsymbol{w}_1^{(1)} = \boldsymbol{w}_0^{(1)} + \alpha s^T(B - A\boldsymbol{w}_0)$$
$$\boldsymbol{w}_1^{(2)} = \boldsymbol{w}_0^{(2)} + \alpha r^T(B - A\boldsymbol{w}_0)$$

$$\boldsymbol{w}_1^{(1)} \geq \epsilon\boldsymbol{w}_1^{(2)} \implies \alpha s^TB \geq \epsilon\alpha r^TB$$

It is easy to see that

$$\mathbb{E}\left[\frac{1}{n}s^TB\right] = (1-2p), \quad \mathbb{E}\left[\frac{1}{n}\epsilon r^TB\right] = \epsilon^2\left(1 - \frac{2p}{k}\right)$$

Since $n$ is sufficiently large, these random variables are close to their mean with high probability. So,

$$(1-2p) \geq \epsilon^2\left(1 - \frac{2p}{k}\right)$$

This is true by the assumed bound on $\epsilon$. □

**Theorem 2 :** *Define*

$$k_t = \frac{2p(1+n\epsilon^2) - 2p(1-\epsilon)}{2p(1+n\epsilon^2) - (1-\epsilon)}$$

*Then if $\eta \leq 2/(1 + \epsilon^2 + (1/n))$, as the number of gradient steps goes to infinity,*

- *if $k \geq k_t$, sample accuracy approaches 1 and population accuracy approaches $1 - (p/k)$ with high probability.*
- *if $k < k_t$, sample accuracy approaches 1 and population accuracy approaches $1 - p$ with high probability.*

*Proof.* Gradient descent will converge if

$$\alpha\lambda_1 \leq 2 \implies \eta \leq \frac{2}{1+\epsilon^2\beta^2+(1/n)}$$

Using the fact that $\beta \leq 1$ gives us the bound on learning rate in the theorem statement.

$$(I+\boldsymbol{s}\boldsymbol{s}^T+\boldsymbol{r}\boldsymbol{r}^T)^{-1} = \left(I - \frac{(1+\boldsymbol{s}^T\boldsymbol{s})(\boldsymbol{r}\boldsymbol{r}^T)+(1+\boldsymbol{r}^T\boldsymbol{r})(\boldsymbol{s}\boldsymbol{s}^T)-(\boldsymbol{s}^T\boldsymbol{r})(\boldsymbol{s}\boldsymbol{r}^T)-(\boldsymbol{r}^T\boldsymbol{s})(\boldsymbol{r}\boldsymbol{s}^T)}{(1+\boldsymbol{s}^T\boldsymbol{s})(1+\boldsymbol{r}^T\boldsymbol{r})-(\boldsymbol{s}^T\boldsymbol{r})(\boldsymbol{r}^T\boldsymbol{s})}\right)$$

$$\begin{aligned}
\implies \boldsymbol{w}_T^{(1)} &= \boldsymbol{s}^T(I+ss^T+rr^T)^{-1}(B-A\boldsymbol{w}_0)+\boldsymbol{w}_0^{(1)} \\
&= \left[\boldsymbol{s}^T - \frac{(1+n)(n\epsilon\beta)\boldsymbol{r}^T+n(1+n\epsilon^2)\boldsymbol{s}^T-(n^2\epsilon\beta)\boldsymbol{r}^T-(n^2\epsilon^2\beta^2)\boldsymbol{s}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B \\
&= \left[\frac{(1+n\epsilon^2)\boldsymbol{s}^T-(n\epsilon\beta)\boldsymbol{r}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{w}_T^{(2)} &= \boldsymbol{r}^T(I+\boldsymbol{s}\boldsymbol{s}^T+\boldsymbol{r}\boldsymbol{r}^T)^{-1}(B-A\boldsymbol{w}_0)+\boldsymbol{w}_0^{(2)} \\
&= \left[\boldsymbol{r}^T - \frac{(1+n)(n\epsilon^2)\boldsymbol{r}^T+(1+n\epsilon^2)(n\epsilon\beta)\boldsymbol{s}^T-(n^2\epsilon^2\beta^2)\boldsymbol{r}^T-(n^2\epsilon^3\beta)\boldsymbol{s}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B \\
&= \left[\frac{(1+n)\boldsymbol{r}^T-(n\epsilon\beta)\boldsymbol{s}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B
\end{aligned}$$

Now suppose we sample a new point $(X, Y)$ from the data distribution. Let $q_i$ denote the index of the noise coordinate of $X_i$ and let $q$ denote the index of the noise coordinate of $X$. With high probability, $q \neq q_i \;\; \forall i$. So,

$$\begin{aligned}
X^T w_T &= X^{(1)}w_T^{(1)} + X^{(2)}w_T^{(2)} + X^{(q)}w_T^{(q)} \\
&= X^{(1)}w_T^{(1)} + X^{(2)}w_T^{(2)} + w_0^{(q)} \\
&= X^{(1)}w_T^{(1)} + X^{(2)}w_T^{(2)}
\end{aligned}$$

We want to analyze the case when the first and second coordinate *disagree*. Let $X^{(1)} = -1$ and $X^{(2)} = \epsilon$. In this scenario if the model always predicts $X^T w_T \geq 0$, it will match the prediction of the *second* coordinate and achieve a population accuracy of $1 - p/k$. On the other hand if it always predicts $X^T w_T < 0$, it will match the prediction of the *first* coordinate and achieve a population accuracy of $1 - p$.

$$X^T w_T > 0 \implies \epsilon w_T^{(2)} > w_T^{(1)}$$

$$\implies \epsilon\left[\frac{(1+n)\boldsymbol{r}^T-(n\epsilon\beta)\boldsymbol{s}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B \geq \left[\frac{(1+n\epsilon^2)\boldsymbol{s}^T-(n\epsilon\beta)\boldsymbol{r}^T}{(1+n)(1+n\epsilon^2)-n^2\epsilon^2\beta^2}\right]B$$

With high probability, $\boldsymbol{s}^T B = n(1-2p)$, $\boldsymbol{r}^T B = n\epsilon(1-2p/k)$, and $\beta = (1-p(k+1)/k+4p^2/k)$.

$$\implies \epsilon\left[(1+n)n\epsilon\left(1-\frac{2p}{k}\right)-(n\epsilon\beta)n(1-2p)\right] \geq \left[(1+n\epsilon^2)n(1-2p)-(n\epsilon\beta)n\epsilon\left(1-\frac{2p}{k}\right)\right]$$

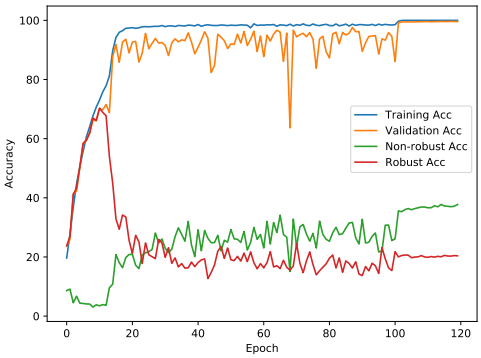
$$\implies n(\epsilon^2-1)+2pn\left(1-\frac{\epsilon^2}{k}\right)+2pn^2\epsilon^2\left(1-\frac{1}{k}\right) \geq 0$$

$$\implies k \geq \frac{2p(1+n\epsilon^2)-2p(1-\epsilon)}{2p(1+n\epsilon^2)-(1-\epsilon)}$$

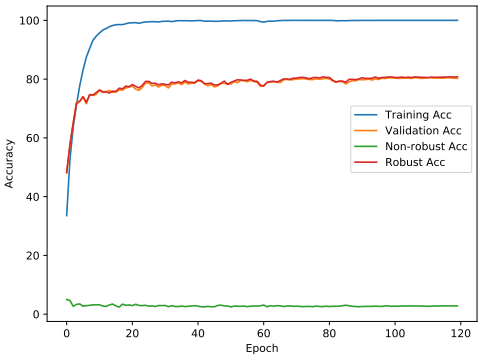
□

## B OTHER DATASETS AND ARCHITECTURES

In this section, we provide training graphs illustrating Pathway 1 and 2 for **Resnet18** and **Resnet50** models trained on the $\mathcal{D}_{det}$ versions of the **CIFAR-10** and **CINIC-10** (Darlow et al. (2018)) datasets. Along with each graph, we mention the hyperparameters used.

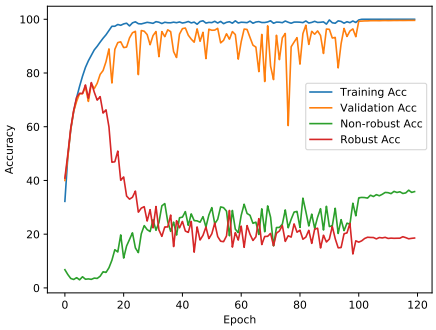



(a) Pathway 1 (No aug, LR 0.1, L2 5e-4)

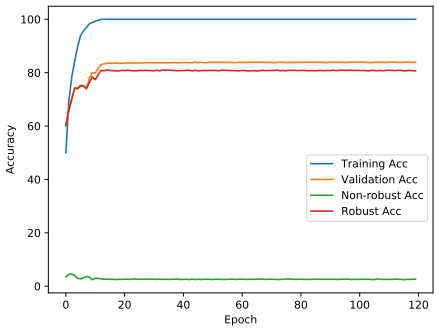



(b) Pathway 2 (No aug, LR 0.01, L2 0)

Figure 4: Resnet50, CIFAR-10 $\mathcal{D}_{det}$

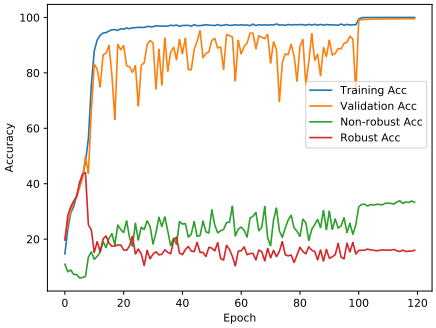



(a) Pathway 1 (No aug, LR 0.1, L2 5e-4)

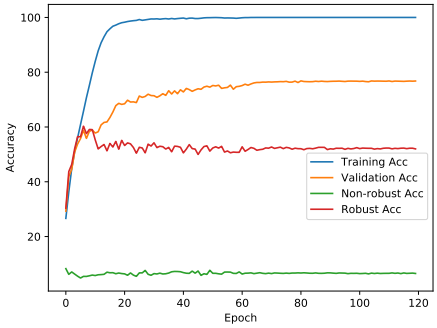



(b) Pathway 2 (No aug, LR 0.01, L2 0)

Figure 5: Resnet18, CIFAR-10 $\mathcal{D}_{det}$

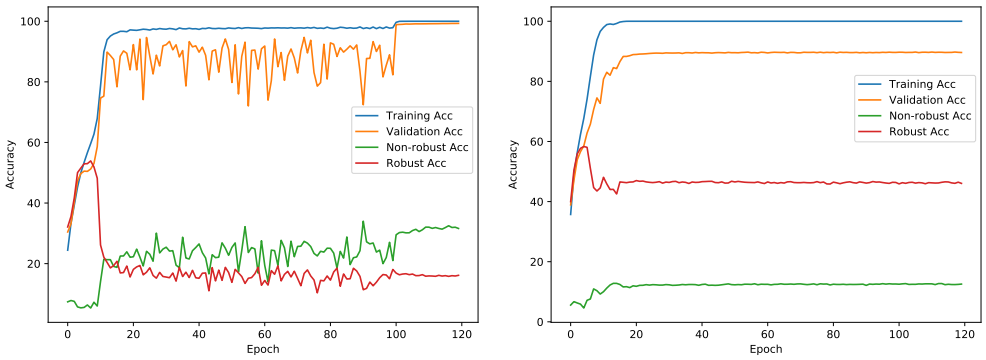



(a) Pathway 1 (Aug, LR 0.1, L2 5e-4)




(b) Pathway 2 (Aug, LR 0.1, L2 5e-4)

Figure 6: Resnet50, CINIC-10 $\mathcal{D}_{det}$




(a) Pathway 1

(b) Pathway 2

Figure 7: Resnet18, CINIC-10 $\mathcal{D}_{det}$

## C HYPERPARAMETERS

In this section, we list the hyperparameters used in our experiments. We also take the case of a Resnet18 trained on $\mathcal{D}_{det}$ CIFAR-10 and look at which hyperparameters lead to Pathway 1 and which lead to Pathway 2. As we note in Section 4, there is a *sharp* transition between the two pathways in the space of hyperparameters.

**Common Parameters :**

| Parameter | Value |
|---|---|
| Batch Size | 128 |
| LR Annealing | $\times 0.1$ at Epoch 100 |
| Optimizer | SGD |
| Momentum | 0.9 |
| PGD norm | $L_2$ |
| PGD epsilon | 0.5 |
| PGD steps | 100 |
| Step size | 0.1 |

Table 2: These parameters were used throughout all our experiments

**Training on Non-transferable examples (Section 3)**

We trained the model using 120 epochs of SGD, and did a grid search over the following combinations of hyperparameters.

| Parameter | Values |
|---|---|
| Learning Rate | $\{0.01, 0.1\}$ |
| Data Augmentation | {True, False} |
| L2 Weight Decay | {0, 1e-5, 5e-4} |

Table 3: Hyperparameter grid for training on non-transferable adversarial examples

**CIFAR $\mathcal{D}_{det}$ (Figure 1)** :

- Pathway 1 : LR 0.1, No data augmentation, L2 5e-4
- Pathway 2 : LR 0.01, No data augmentation, L2 0

**CIFAR (Figure 3)** :

- Pathway 1 : LR 0.01, Data augmentation, L2 5e-4
- Pathway 2 : LR 0.01, No data augmentation, L2 0

**Which hyperparameters lead to Pathway 1 and Pathway 2? :**

We pick one model (Resnet18) and one dataset (CIFAR-10), and explore which hyperparameters lead to Pathway 1 and which lead to Pathway 2. To re-iterate, Pathway 1 is characterized by the model learning almost exclusively only the robust features. All models were trained with SGD for 120 epochs, and the reported accuracies are after the final epoch.

| Data Aug | L2 decay | Learning Rate | |
|---|---|---|---|
| | | 0.1 | 0.01 |
| Yes | 0 | 30.9 / **43.0** | 24.4 / **50.66** |
| | 5e-4 | **38.3** / 26.2 | 30.9 / **43.0** |
| No | 0 | 2.9 / **77.8** | 2.6 / **80.7** |
| | 1e-5 | 4.2 / **79.1** | – |
| | 5e-5 | 2.7 / **80.1** | – |
| | 1e-4 | 2.6 / **79.7** | – |
| | 5e-4 | **35.9** / 18.6 | 20.9 / **43.7** |

Table 4: (Test / Shifted Test) accuracy for a Resnet18 model trained on CIFAR-10 $\mathcal{D}_{det}$. Corresponds to (Non-robust Accuracy / Robust Accuracy). The higher of the two is **bold**, and the configurations which follow Pathway 2 have been highlighted in green.

*Remark :* As we can see in the case with LR 0.1 and no data augmentation, the network exhibits a *sharp* transition from Pathway 2 to Pathway 1 in the space of hyperparameters. There does not seem to be a "middle ground".

**Cross-Dataset Transfer (Table 1)** :

- CIFAR-minus-Flickr (clean) : *Adam optimizer*, LR 1e-3, L2 1e-5, with data augmentation.
- CINIC-10 (clean) : LR 0.01, data augmentation, L2 5e-4.
- CIFAR-minus-Flickr $\mathcal{D}_{det}$ : LR 0.1, No data augmentation, L2 5e-4.
- CINIC-10 $\mathcal{D}_{det}$ : LR 0.1, No data augmentation, L2 5e-4.