The Geometry of Adversarial Subspaces

Anonymous authors
Paper under double-blind review

Abstract

Artificial neural networks (ANNs) are constructed using well-understood mathematical operations, and yet their high-dimensional, non-linear, and compositional nature has hindered our ability to provide an intuitive description of how and why they produce any particular output. A striking example of this lack of understanding is our inability to design networks that are robust to adversarial input perturbations, which are often imperceptible to a human observer but cause significant undesirable changes in the network's response. The primary contribution of this work is to further our understanding of the decision boundary geometry of ANN classifiers by utilizing such adversarial perturbations. For this purpose, we define adversarial subspaces, which are spanned by orthogonal directions of minimal perturbation to the decision boundary from any given input sample. We find that the decision boundary lies close to input samples in a large subspace, where the distance to the boundary grows smoothly and sub-linearly as one increases the dimensionality of the subspace. We undertake analysis to characterize the geometry of the boundary, which is more curved within the adversarial subspace than within a random subspace of equal dimensionality. To date, the most widely used defense against test-time adversarial attacks is adversarial training, where one incorporates adversarial attacks into the training procedure. Using our analysis, we provide new insight into the consequences of adversarial training by quantifying the increase in boundary distance within adversarial subspaces, the redistribution of proximal class labels, and the decrease in boundary curvature.

1 Introduction

Artificial neural networks (ANNs) have been highly performant on common machine learning tasks, but their application space is limited by their susceptibility to adversarial attacks (Szegedy et al., 2014), which underscores a general lack of understanding for how and why they make their decisions. Examples produced by adversarial attacks are a worst-case demonstration of an ANN's inability to gracefully cope with identity-preserving shifts or distortions of its inputs. To construct one, an adversary must perturb the input in a small but specific way such that the output of the network changes significantly. Adversarial perturbations are defined to have minimal length, resulting in inputs that are as close to the network's decision boundary as possible. Here, we present a method that uses an untargeted adversarial attack objective to find the multi-dimensional subspace where an ANN's decision boundary is closest to any given input sample. The resulting *adversarial subspace* allows us to visualize and understand ANN decision boundaries in terms of their curvature, their shape, and their proximity to the input.

One existing technique to explain ANN classifications has relied on two-dimensional visualizations of the decision surface, although it suffers from a number of shortcomings, most notably high-dimensional points resolving to the same projected location, distortions of distances, and a lack of invertibility from the two-dimensional space back to the high-dimensional decision space (Féraud & Clérot, 2002; Rauber et al., 2016; Rodrigues et al., 2019). A popular alternative strategy is to investigate the decision boundary in the high-dimensional input space itself, early attempts of which (Golland, 2001; Baehrens et al., 2010) lead the way to the modern interest in adversarial attacks (Biggio et al., 2013; Szegedy et al., 2014). For example, one can measure the distance to the decision boundary for random perturbation

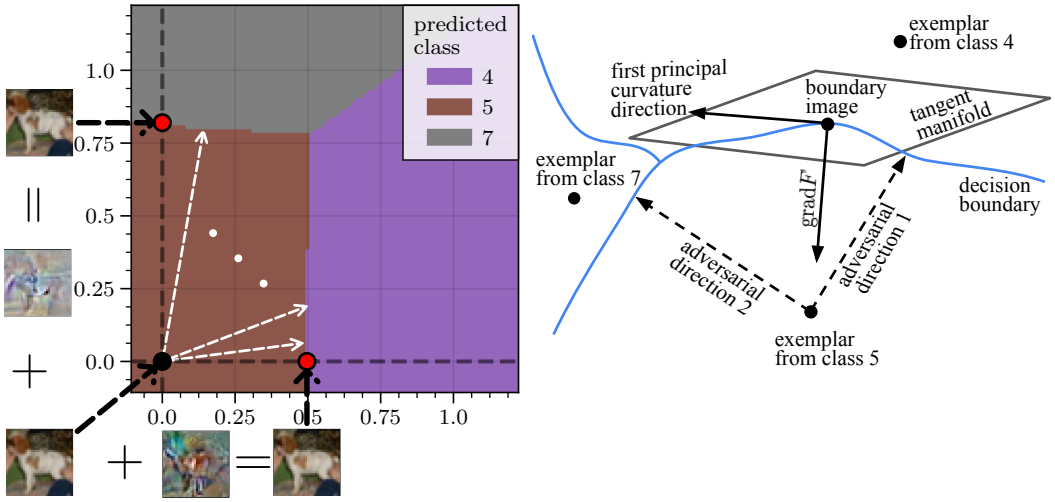

Figure 1: **An adversarial subspace**. (Left) We define a subspace around any given input example where the boundary is closest to that input. The white arrows depict our sampling scheme, which uniformly samples directions where we measure the distance to the decision boundary within the adversarial subspace. The perturbation images are rescaled for visualization, but their ℓ_2 length is indicated on the axes. Each color corresponds to the class categories for deer (class 4), dogs (class 5), and horses (class 7). This visualization is for an adversarially trained neural network. (Right) We then measure the curvature of the decision boundary, both in the high-dimensional input space and in the adversarial subspace.

directions away from adversarial examples (Tabacof & Valle, 2016; He et al., 2018), or for orthogonal directions which are proximal to an initial adversarial example (Tramèr et al., 2017). Alternatively, much work has demonstrated the utility of visualizing two-dimensional cross-sections of the decision space between adversarial examples and random orthogonal directions (Warde-Farley & Goodfellow, 2016; Yu et al., 2019a) or orthogonal test set examples (Swirszcz et al., 2019). Unlike all of the existing approaches to characterize decision boundaries in the input space, which rely on at most one adversarial direction per input, we propose jointly defining a subspace of adversarial directions. The resulting analysis leads to a general explanation of classifier decision boundaries, as well as adversarial susceptibility, in the subspace where the boundary is closest to the input example.

As a further demonstration of the utility of our approach, we investigate the most widely used technique for defending against adversarial attacks, so-called adversarial training. The defense method requires including adversarial perturbations in the training process by either augmenting the dataset with pre-computed examples (Ilyas et al., 2019) or by incorporating an attack model into the training procedure (Goodfellow et al., 2014; Madry et al., 2018). It has been recently demonstrated that adversarial training performs well on the CIFAR-10 machine learning dataset when enough resources are dedicated to hyperparameter tuning (Gowal et al., 2020), although the performance is still far from that on unperturbed images. We provide a new perspective on this defense by observing that it increases the distance to the decision boundary in the entire adversarial subspace, even though the method utilizes only a single adversarial perturbation per input sample. We additionally find that adversarial training increases the number of alternative label regions in adversarial subspaces, and changes the relative distribution of nearby alternate classes. Herein, we will focus on defining the method and summarizing the subspace with a continued application of understanding adversarial training.

2 Methods

In the following we will outline the novel methods employed in this study. Namely, the method for finding adversarial subspaces is first described, and then we provide an overview of how we measure decision boundary curvature (with additional derivations given in Appendix A). Our primary investigation was conducted using the CIFAR-10 dataset (Krizhevsky &

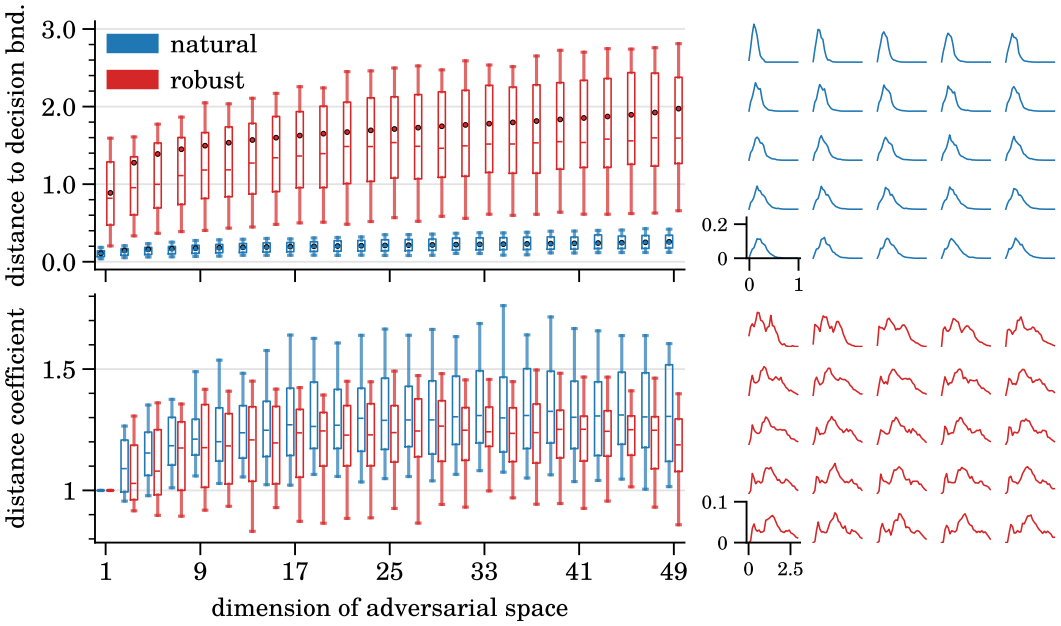

Figure 2: **Distances to the decision boundary.** The distance to the decision boundary for random perturbations within an adversarial subspace grows sub-linearly with increased dimensionality. (Upper left) The dots indicate the adversarial perturbation length for that dimension (i.e. along the perturbation axis), averaged across images. The box plots show the distance to the decision boundary for uniformly-sampled directions within the n-dimensional adversarial subspace, where n is increasing along the horizontal axis. The box indicates the interquartile range, the whiskers indicate the 10% and 90% data percentiles, and the solid horizontal lines indicate the mean. (Lower left) Same as the above, except now the vertical axis indicates the ratio of the largest distance to the decision boundary divided by the length of the given subspace dimension,. That this is small indicates that we rarely sampled a boundary point more than 50% farther away than the largest adversarial dimension. (Right) Distribution of boundary distances as one increases subspace dimensionality, i.e the probability (vertical axis) of a sampled direction having a specific distance (horizontal axis). Observe that the mean shifts to the right, and the variance increases proportionately, which suggests that the distance smoothly increases from one dimension to the next.

Hinton, 2009) and compared two identical ResNet-50 ANNs (He et al., 2016) trained with and without adversarial examples. In Appendix G we provide results on the MNIST dataset, which demonstrates largely consistent trends across datasets and across weight initialization seeds. All distances measured are with respect to the ℓ_2-norm. Additional dataset and model details can be found in Appendix B.

2.1 Finding adversarial subspaces

We aim to find the set of orthonormal vectors that span the subspace where an input sample has minimal distance to the decision boundary. In order to achieve this, we follow a greedy optimization approach to span an adversarial subspace up to a desired number of dimensions. More precisely, consider some neural network, f, that receives an n-pixel input image, $x \in \mathbf{R}^n$, and produces an output vector of probabilities associated with l label categories, $\hat{y} \in \mathbf{R}^l$. For a given input, we wish to find an ordered set of orthogonal vectors, i.e. $\Delta = [\delta_1, \delta_2, \ldots, \delta_m], m \leq n$, such that $\langle \delta_k, \delta_m \rangle = 0, \quad \forall k < m$. Thus for each vector, we want to solve an optimization objective that minimizes $||\delta_m||_p$, where $||\delta||_p = (\sum_i |\delta|^p)^{\frac{1}{p}}$ is the l_p norm. We expect the optimizer to explicitly and jointly consider the boundedness of the input domain, force the perturbations to be orthogonal, and produce perturbations that result in new (incorrect) classifier outputs. Such an objective is intractable to solve for deep

neural networks, but is well approximated by the untargeted adversarial attack objective[1], which modifies the conditional to be subject to $\max_{j \neq i} f_j(x+\delta) - f_i(x+\delta) > 0$, where i is the correct class index. The length of these vectors is nearly synonymous with distances to the decision boundary, where the actual decision boundary would be an infinitesimal step towards x from $x+\delta$. We must further relax the objective to a differentiable approximation for finding a gradient based solution. While there are many such approximations, we opted for a modification of that which was used by Szegedy et al. (2014) to find singular adversarial directions, which is to minimize the inverse of the cross entropy loss (the training objective of the model), weighed with a distance penalization term. Specifically, $\ell = \frac{1}{L_{CE}(x+\delta)} + \kappa \cdot ||\delta||_p$, with the cross entropy loss defined as $L_{CE}(x) = -\log \frac{\exp f_i(x)}{\exp \sum_j f_j(x)}$. Using this loss term, we can now define our objective function as implemented:

$$
\begin{aligned}
\text{minimize} \quad & \frac{1}{L_{CE}(x+\delta_m)} + \kappa \cdot ||\delta_m||_p \\
\text{subject to} \quad & \max_{j \neq i} f_j(x+\delta_m) - f_i(x+\delta_m) > 0 \\
& x + \delta_m \in [0,1]^n \\
& \langle \delta_k, \delta_m \rangle = 0, \quad \forall k < m.
\end{aligned}
\tag{1}
$$

Intuitively, this maximizes the model loss with respect to the true class label while keeping the perturbation as small as possible. The first constraint in Equation 1 is to ensure that the image is adversarial[2], the second keeps the perturbed images within the allowable pixel bounds, and the last one enforces orthogonality of all found adversarial subspace dimensions. We found the optimal value of constant κ in equation (1) with a binary search performed during run time. The optimization problem was solved using the greedy interior-point optimizer implemented by Ipopt (Wächter & Biegler, 2006).

Once the adversarial vectors are found, we conducted experiments to measure the distance to the decision boundary in the space spanned by the vectors (i.e. within the adversarial subspace). The experiment was repeated on 100 test images that were chosen to have equal label representation (10 images per label) and to be correctly classified by both networks. For each test image we found 50 orthonormal adversarial vectors, which allowed us to define a sequentially growing list of adversarial subspaces with dimensionality increasing from 1 to 50. For a given image and subspace we sampled directions via a linear combination of normally distributed variables, which has been previously proven to provide an even sampling (Muller, 1959). Finally, for each random direction we performed a binary search to find the decision boundary.

2.2 Measuring decision boundary curvature

As before, let $f_i, f_j : \mathbb{R}^m \to \mathbb{R}$ be the functions corresponding to i and j-th logits of a neural network, f. Let $F_{ij} \stackrel{\text{def}}{=} f_i - f_j$. Then the (i,j)-*decision boundary* is the set

$$
B_{ij} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \mid F_{ij}(x) = 0\} = \{x \in \mathbb{R}^n \mid f_i(x) = f_j(x)\}. \tag{2}
$$

The set B_{ij} is defined independently of the sign of F_{ij} (that is, $B_{ij} = B_{ji}$), although doing so will produce a change of sign in our calculations. Therefore we will continue to follow the above convention by identifying, at a point x, i with the correct label associated to x, and j with the incorrect label. This convention (which agrees with Moosavi-Dezfooli et al., 2018), ensures that positive (resp. negative) curvature of B_{ij} near x corresponds to B_{ij} bowing

[1] Many works use "untargeted" adversarial attacks to refer to attacks that are actually targeted to the second-most-likely output class (for example, Carlini & Wagner, 2017; Kurakin et al., 2017; He et al., 2018; Chen et al., 2018). While these may be practically equivalent for a single perturbation, they are not when finding multiple orthogonal directions. Therefore, we will use a more general definition of untargeted attack, following (Tabacof & Valle, 2016; Finlay et al., 2019).

[2] Since we are not targeting any specific class, the first constraint avoids a completely uniform output distribution (i.e. no class wins)

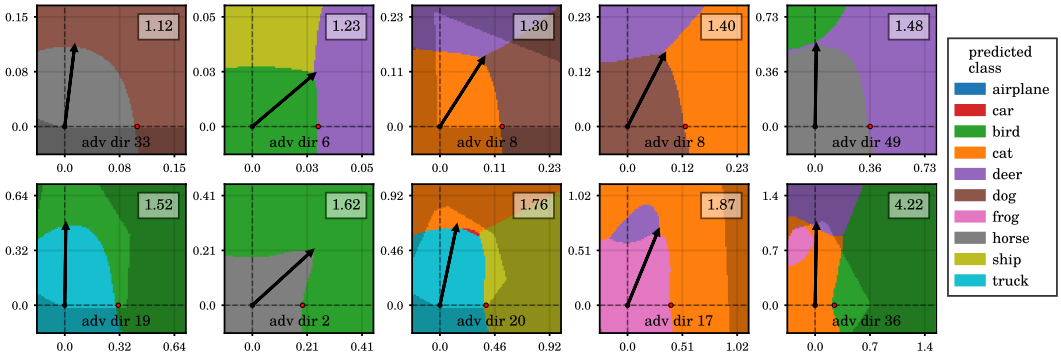

Figure 3: **Boundary visualizations.** Each subplot visualizes a two dimensional cross-section of the decision space for a naturally trained network. Each cross-section is spanned by the vectors corresponding to the largest distance coefficient found for a given input image (visualized as black arrows) and the according adversarial perturbation of the largest dimension of the subspace from which the distance coefficient was retrieved. The red dots indicate the position of the described adversarial perturbation. The number in the top right corner depicts the distance coefficient of the cross-section (as reported in Figure 2, bottom left panel). Areas of the cross-section that lie outside the image bounds are shaded.

inward towards (resp. away from) x (see Figure 1 for an example of positive curvature). If the curvature along B_{ij} is zero near x then B_{ij} is affine. The complete details of how our calculations were derived, and numerical considerations therein, are relegated to Appendix A, however we give a brief intuitive account below.

In our curvature analysis we use a method known as principal curvature decomposition to understand the degree of linearity of the decision boundary. Since the decomposition is performed on a level-set of the decision space, which in general is of codimension 1, it yields $n-1$ scalars and vectors, called the *principal curvatures* and *principle curvature directions*, respectively. The curvature directions point in the directions of maximal and minimal surface curvature, while the principal curvatures themselves indicate the signed magnitude of the curvature. Similar to principal component analysis (PCA), which organizes the space as an orthogonal basis of directions of maximal variance, principal curvature analysis organizes the (tangential) space of the decision boundary into orthogonal directions of maximum curvature. The principle curvatures and associated directions are the eigenvectors and eigenvalues of a linear operator $s : \mathbb{R}^m \to \mathbb{R}^m$ called the *shape operator*.

In order to study the curvature over an adversarial (or random) subspace we introduce the *subspace pullback* of s

$$\left(P^{\mathsf{T}}\right)^{*} s \stackrel{\text{def}}{=} P s P^{\mathsf{T}}, \tag{3}$$

where $P \in \mathbb{R}^{m \times n}$ is an orthogonal projection, so that the image of P^{T} spans the adversarial subspace of $\mathbb{R}^m$ in question. This definition of $\left(P^{\mathsf{T}}\right)^{*} s$ essentially restricts the shape operator to a linear subspace of the tangential space and is motivated by the principles of differential geometry, details of which can be found in Remark 1 and the surrounding material in Appendix A. The pullback shape operator allows us to compare the amount of curvature in the adversarial subspace to that of the entire response space.

3 Results

By definition, the distance to the decision boundary increases for each additional dimension of an adversarial subspace. The circles in the top left panel of Figure 2 indicate these distances, averaged across the test images, which are found by greedily minimizing Equation 1. The distance to the decision boundary grows sub-linearly as one increases the dimensionality of the subspace, with the fastest growth occurring in the first couple of dimensions. The leftmost box plots of the top left panel in figure 2 as well as the Appendix Table 2 confirms that the mean initial perturbation distances are comparable to more traditional adversarial attacks (Madry et al., 2018) and decision boundary distance estimates (Ding et al., 2020)).

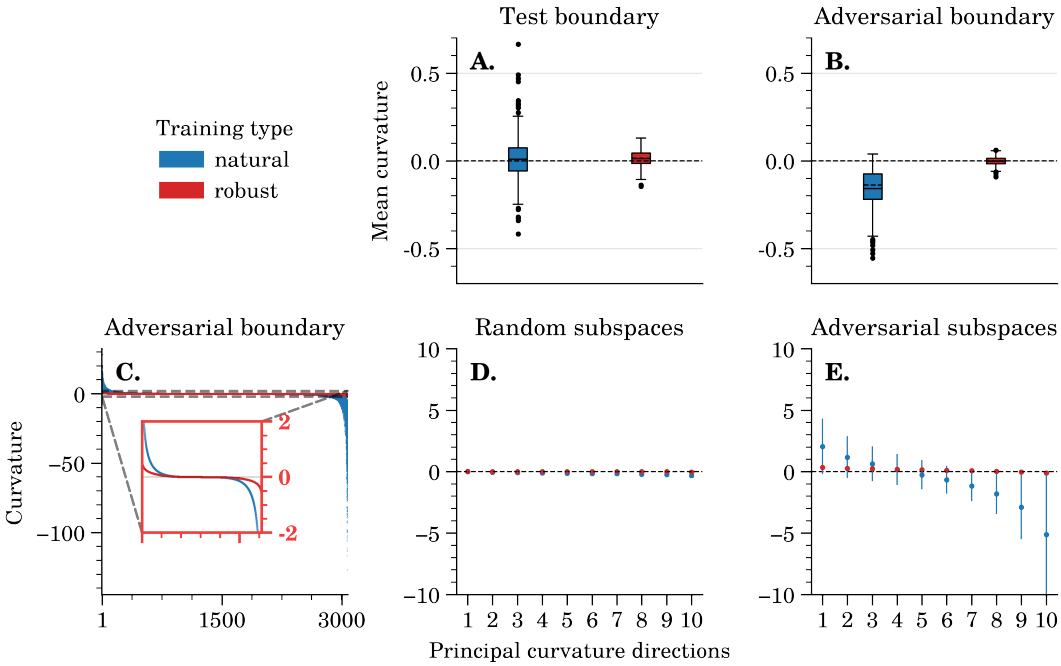

Figure 4: **Decision boundary curvature.** All plots include data for 50 correctly classified test images with evenly distributed labels and the first 10 adversarial subspace dimensions. (A,B) The mean curvatures per boundary point are smaller when the boundary is separating two test images ("test boundary") than when it is separating a test image and an adversarial image ("adversarial boundary"). (C) Scatter plot of the curvature profiles for all images and directions with the adversarial boundary condition. The red inlay shows the full dimensionality (i.e. same horizontal axis as the parent plot), but zoomed in (i.e. different vertical axis limits) to emphasize that while both models have positive and negative curvature, the adversarially trained boundary is more flat (i.e. linear). The data includes all adversarial subspace boundary points per test image. (D,E) Curvature of adversarial point boundaries when restricted to random or adversarial subspaces. Curvature in the adversarial subspace is more extreme than when in an equal sized random subspace. The points indicate the mean across test images and the error lines indicate the standard deviation.

While this gives an idea of the lower-bound of distances to the decision boundary, it does not provide information about the behavior of the boundary elsewhere in the subspace. To better understand this, we uniformly sampled 100 random directions *within a given subspace* and performed a binary search to find the decision boundary along those directions. We only report distances for decision boundary points that are within the ℓ_{inf} allowable pixel box, which was never fewer than 40% of the subspace samples (see Appendix C Figure 7). The box plots in the top left panel of Figure 2 indicate the distribution of decision boundary distances across the sampled directions and test images. Using a small sample of images, we found that the perturbation was discernible at after an ℓ_2-distance of 1.5, which occurs after about 20 dimensions for the adversarially trained network, and after greater than 50 dimensions for the naturally trained network.

The bottom left panel of Figure 2 plots a distance coefficient that is defined as the maximum distance to the decision boundary within an adversarial subspace divided by the longest vector defining the subspace (which is always associated with the highest dimension number). For example, a value of 2 would indicate that the largest sampled distance is twice as far away as the largest found adversarial perturbation. Finally, a complete description of the boundary distances is conveyed as histograms in the rightmost panels, with dimensionality increasing from left to right, then top to bottom. Together, this provides further evidence that the decision boundary grows consistently, without large variations in distance between the found adversarial subspace axes.

In Figure 3 we visualize two-dimensional cross-sections of the decision surface. The horizontal axis of each cross-section corresponds to an adversarial subspace perturbation, the black

arrow indicates the farthest decision boundary point, and the vertical axis is found by orthogonalizing the direction of the farthest decision boundary point with respect to the adversarial perturbation using one step of the Gram-Schmidt process. The cross-sections are ordered according to the distance coefficient, which is displayed in the upper-right corner. We emphasize that because these are cross-sections, they avoid the shortcomings of typical low-dimensional decision space projections noted in the introduction. Among the visualizations one can see examples of when the boundary increases smoothly from one axis to the next, and also the more rare (as indicated by Figure 2) cases when the boundary protrudes quite far from the origin when between the axes. From this analysis we conclude that adversarial examples are not isolated or due to sharp protrusions into an otherwise homogeneously labeled region. Rather, there exists a smooth increase in distance to the decision boundary from the extreme case of an initially found adversarial example to semantically differentiated images.

In Figure 4 we provide an analysis of boundary curvature for several experimental conditions that vary the type of training, the location of the boundary, and subspace defining vectors. This analysis reveals that there is a relationship between the boundary flatness (i.e. linearity) and the existence of adversarial examples. Panels A and B report the mean curvature (i.e. the mean of the eigenvalues of the shape operator) per point on the decision boundary. Both model types have larger boundary curvature for boundary points separating clean images from adversarial images (adversarial boundary; Panel B) when compared to boundary points separating test set examples (test boundary; Panel A). We additionally found that adversarial training tends to linearize the decision boundary, both between correctly labeled test images and near adversarial examples. Panel C is a scatter plot of the principal curvatures (i.e. eigenvalues of the shape operator) in the high-dimensional input space for all of the adversarial boundary points. We consistently find both positive and negative curvature at each boundary point and for both training types. Using the pullback defined in Equation 3, we can measure the subspace principal curvatures for random (Panel D) and adversarial (Panel E) subspaces. The curvature within adversarial subspaces is notably larger than in random subspaces. The difference in curvature values between naturally and adversarially trained networks is less pronounced when limited to random subspaces – suggesting that much of the linearization occurring during adversarial training is along directions aligned with adversarial examples.

In addition to the earlier comparisons between natural and adversarial network training, we measured the diversity of adversarial classes near test samples. The leftmost subplot of Figure 5 gives the number of unique adversarial classes found in the first 50 dimensions of the adversarial subspaces, averaged across a uniform sampling of correctly labeled test images. The other plots show the distribution of adversarial classes present in the subspaces with respect to the origin label. These plots suggest that adversarial training results in a more even distribution of alternate classes near any given test sample, which we further quantify by measuring the discrete entropy of each distribution in Table 1. It is apparent from our analysis that adversarial training pushes the decision boundary away from the origin in all dimensions tested, even though few adversarial perturbations are provided per training sample. In the top left panel of Figure 2, the dots indicate that the largest increase in boundary distance from one dimension to the next is after the first, although the adversarial images corresponding to the first several dimensions were often still imperceptibly changed from the original. Furthermore, in the bottom left panel of Figure 2 one can see that the adversarial trained network consistently produces smaller coefficients, indicating that the boundary size is less variable between the subspace axes. In addition to (or more probably as a consequence of) increasing the distance to the boundary, adversarial training further modifies the density of alternative class labels near test images. This is evident in Figure 5,

Table 1: Entropy of unique proximal adversarial labels per class, averaged across a balanced sampling of correctly labeled test images.

Label	Natural	Robust
airplane	1.70	**1.81**
car	1.01	**1.26**
bird	**1.72**	1.59
cat	1.56	**1.97**
deer	1.41	**1.98**
dog	1.51	**1.82**
frog	1.12	**1.51**
horse	1.63	**1.78**
ship	**1.74**	1.39
truck	1.79	**1.88**

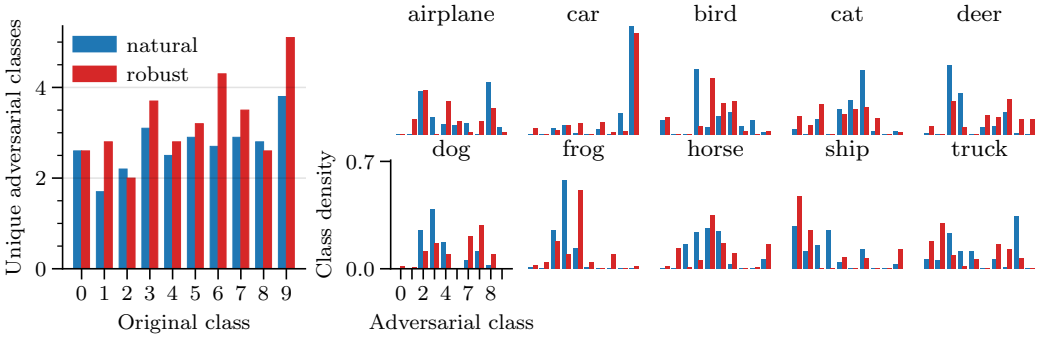

Figure 5: **Adversarial Class Composition.** (Left) Within the adversarial subspaces of randomly chosen test images we quantified how many unique adversarial classes exist. In most cases adversarial training increases the diversity of alternate classes close to any test image. (Right) Each plot shows the distribution of nearby alternate classes for each origin class with natural and adversarial training. Adversarial training results in a more even distribution of nearby classes for all categories except bird and ship.

which demonstrates that adversarial training increases the diversity of alternative classes near test images on the left and modifies the proximal class distribution on the right. While some categories, like "car" and "truck" are intuitively adjacent, the distribution of classes generally failed to match any clear intuitions. For example, observe the lack of symmetry between cars and trucks, or that adversarial training switches the "frog" category's most frequent alternate class from "bird" to "cat". Indeed, Table 1 suggests that adversarial training spreads out the distribution among classes, suggesting that the decision surface is more evenly utilizing the high-dimensional space.

4 Related Work

Related adversarial subspace studies: The work of Tramèr et al. (2017) is most similar to our adversarial decomposition, in that they reported the dimensionality of a subspace in the proximity of a given adversarial perturbation, although do not comment on the number or proximity of alternative adversarial examples. Work from Ma et al. (2018) measured the local intrinsic dimensionality (LID) of adversarial subspaces, which indirectly estimates adversarial subspace dimensionality by measuring the distance from an adversarial example to a batch of nearest unperturbed dataset examples. This idea has been extended to algorithms for detecting adversarial examples (Ma et al., 2018; Mao et al., 2020), although Athalye et al. (2018) generated adversarial examples with indistinguishable LID scores from natural examples. In terms of understanding ANN classifiers, such an approach leaves open the questions of how proximal the decision boundary is from input samples in greater than one dimension or the actual dimensionality of adversarial subspaces. Our approach is the first to pursue a more global examination of the decision boundary where it is nearest to test samples by defining adversarial subspaces using multiple orthogonal adversarial directions. Our emphasis on the subspace where the decision boundary is closest to test images is also unique when compared to work that characterizes the decision boundary in random directions, which will be considerably farther away from the origin (e.g. Fawzi et al., 2016; Tabacof & Valle, 2016; Liu et al., 2017; He et al., 2018). Finally, most of these studies perform a clipping operation *after* defining or finding the desired directions, which results in a non-orthogonal set (see Appendix E for a more detailed discussion). We overcome the problem of clipping pixel values outside the valid pixel range by including the valid range as inequality constraints in the optimization process. Our work is complementary to previous studies that investigate the relationship between adversarial robustness and *input* dimensionality (Amsaleg et al., 2017; Simon-Gabriel et al., 2019), since the comparative trends observed between robust and natural networks are relatively unchanged with small (MNIST) or large (CIFAR) dimensional inputs.

Neuron response curvature: The use of response curvature to understand nonlinear functions has a long history in neuroscience (for a review, see Gollisch & Herz, 2012) as well as models of neural computation (e.g. Zetzsche & Krieger, 2001; Rust et al., 2005; Golden et al.,

2016). The principal curvature decomposition was previously employed in this context by Golden et al. (2019), who used it to understand invariance and selectivity of individual ANN neurons, and by Paiton et al. (2020), who first found a relationship between individual neuron response curvature and adversarial robustness. Our work is distinct from the aforementioned studies in that we investigate the curvature of the decision boundary, as opposed to that of individual neurons. However, the relationship between decision boundary curvature and adversarial robustness has been previously studied Moosavi-Dezfooli and colleagues, who provided a number of compelling results identifying "an increasing [adversarial] vulnerability with respect to the curvature of the decision boundary" (Moosavi-Dezfooli et al., 2018)(for a complete account, see the thesis of Moosavi-Dezfooli, 2019). As we will discuss in the next Section, our analysis provides support for hypotheses that go beyond the correlation between decision boundary curvature and network robustness, in that we measure boundary curvature in the adversarial subspace dataset samples. We additionally note that while previous work has utilized Riemannian curvature analysis to understand neural networks (viz. Fawzi et al., 2016; Poole et al., 2016; Moosavi-Dezfooli et al., 2018; 2019; Golden et al., 2019), each study measures different quantities without complete details of the derived formulas. We demonstrate in Appendix F that these formulas do not find the correct level-set curvature of simple manifolds. This together with our rigorous derivation and discussion in Appendix A constitutes, to the best of our knowledge, the first principled and analytically supported measurement of decision boundary curvature of trained nonlinear neural networks.

5 Conclusion

There is an ongoing debate on the cause, prevalence, and uniformity of adversarial examples. Several studies have supported a hypothesis that most adversarial examples inhabit relatively dense regions of the input space (Goodfellow et al., 2014; Tabacof & Valle, 2016; Moosavi-Dezfooli et al., 2017a; He et al., 2018). However, other works have suggested that adversarial examples live in isolated pockets or thin "spikes" of incorrect classification regions (Szegedy et al., 2014; Yu et al., 2019b). Here we provide quantitative evaluation supporting the hypothesis that decision boundaries are close to input samples in a large subspace. Additional work has suggested that it is the linearity of deep neural networks that leads to adversarial examples (Goodfellow et al., 2014), and that adversarial training leads to a less linear decision boundary (Madry et al., 2018). However, more recent work has proposed the alternative: that increasing decision boundary linearity causes increased robustness (Li et al., 2019b; Moosavi-Dezfooli et al., 2019; Terjék, 2020; Sarkar & Iyengar, 2020). Our curvature analysis confirms that adversarial training linearizes the boundary, which is also congruent with hypotheses about the robustness benefits of smoothing the decision surface (Zhang et al., 2019; Wu et al., 2020). We further compare the boundary curvature in adversarial subspaces and random subspaces to show an increase in curvature in adversarial subspaces for both training methods. Since we do not report the causal relationship between adversarial directions and boundary curvature, our results have no impact on hypotheses for adversarial susceptibility due to overfitting (Tanay & Griffin, 2016) or the perceptual quality of adversarial features (Ilyas et al., 2019), although we see this as an interesting future direction of study.

Adversarial subspaces allow for a quantitative description of the nonlinear geometry and class composition of the network decision boundary in the subspace spanned by its most susceptible perturbation directions. They also assist in explaining the prevalence of adversarial examples as well as the effects of adversarial training. Understanding this space better can improve neural network performance with semi-supervised relabeling (Benato et al., 2018), architecture design (Rauber et al., 2018), and adversarial robustness (Tramèr et al., 2017; Moosavi-Dezfooli et al., 2019). Importantly, we also provide a quantitative and falsifiable interpretability approach for understanding ANN decisions (Ribeiro et al., 2016; Vlassopoulos et al., 2020), which is a necessary prerequisite for the technology to be applied to many industries (Doshi-Velez & Kim, 2017; Leavitt & Morcos, 2020). We have identified an interesting avenue for future work in performing a large-scale analysis of the adversarial subspaces of various neural network architectures, as well as estimating decision boundary geometry of non-smooth (e.g. ReLU or stochastic networks, Li et al., 2019a; Dapello et al., 2020) neural networks (which requires adding a costly Hessian approximation step).

Author Contributions

Acknowledgments

References

Laurent Amsaleg, James Bailey, Dominique Barbe, Sarah Erfani, Michael E Houle, Vinh Nguyen, and Miloš Radovanović. The vulnerability of learning to adversarial perturbation increases with intrinsic dimensionality. In *2017 IEEE Workshop on Information Forensics and Security (WIFS)*, pp. 1–6. IEEE, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/athalye18a.html`.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research, JMLR*, 11:1803–1831, 2010.

Barbara Caroline Benato, Alexandru Cristian Telea, and Alexandre Xavier Falcão. Semi-supervised learning with interactive label propagation guided by feature space projections. In *2018 31st Conference on Graphics, Patterns and Images, SIBGRAPI*, pp. 392–399. IEEE, 2018.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 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.

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

Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

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

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations, ICLR*, 2020.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems, NeurIPS*, volume 29. Curran Associates, Inc., 2016. URL `https://proceedings.neurips.cc/paper/2016/file/7ce3284b743aefde80ffd9aec500e085-Paper.pdf`.

Raphael Féraud and Fabrice Clérot. A methodology to explain neural network classification. *Neural networks*, 15(2):237–246, 2002.

Chris Finlay, Aram-Alexandre Pooladian, and Adam Oberman. The logbarrier adversarial attack: making effective use of decision boundary information. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4862–4870, 2019.

J R Golden, K P Vilankar, M C K Wu, and D J Field. Conjectures regarding the nonlinear geometry of visual neurons. *Vision research*, 120:74–92, 2016.

James R Golden, Kedarnath P Vilankar, and David J Field. Selective and invariant features of neural response surfaces measured with principal curvature. *bioRxiv*, 2019. URL https://doi.org/10.1101/2019.12.26.888933.

Polina Golland. Discriminative direction for kernel classifiers. In *Advances in neural information processing systems, NeurIPS*, volume 14, pp. 745–752, 2001.

Tim Gollisch and Andreas VM Herz. The iso-response method: measuring neuronal stimulus integration with closed-loop experiments. *Frontiers in neural circuits*, 6:104, 2012.

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

Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

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, CVPR*, pp. 770–778, 2016.

Warren He, Bo Li, and Dawn Song. Decision boundary analysis of adversarial examples. In *International Conference on Learning Representations, ICLR*, 2018.

Andrew Ilyas, Shibani Santurkar, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in neural information processing systems, NeurIPS*, volume 32, 2019.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/cifar.html.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations, ICLR*, 2017.

Matthew L Leavitt and Ari Morcos. Towards falsifiable interpretability research. *arXiv preprint arXiv:2010.12016*, 2020.

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

John M. Lee. *Introduction to Smooth Manifolds*. Number 218 in Graduate Texts in Mathematics. Springer, 2 edition, 2013. ISBN 978-1-4419-9981-8 978-1-4419-9982-5.

John M. Lee. *Introduction to Riemannian Manifolds*. Springer International Publishing, 2 edition, 2018. ISBN 978-3-319-91754-2.

Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. *Advances in Neural Information Processing Systems*, 32:9464–9474, 2019a.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. *Advances in neural information processing systems, NeurIPS*, 32:15390–15402, 2019b.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations, ICLR*, 2017.

Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *International Conference on Learning Representations*, 2018.

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, ICLR*, 2018.

Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley, 1999. ISBN 978-0-471-98632-4 978-0-471-98633-1.

Xiaofeng Mao, Yuefeng Chen, Yuhong Li, Yuan He, and Hui Xue. Learning to characterize adversarial subspaces. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2438–2442, 2020. doi: 10.1109/ICASSP40776.2020.9052933.

Seyed-Mohsen Moosavi-Dezfooli. *Geometry of adversarial robustness of deep networks: methods and applications*. PhD thesis, EPFL, 2019. URL `https://infoscience.epfl.ch/record/271933`.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition, CVPR*, pp. 1765–1773, 2017a.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Analysis of universal adversarial perturbations. *ArXiv e-prints*, pp. arXiv–1705, 2017b.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *International Conference on Learning Representations, ICLR*, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 9078–9086, 2019.

Mervin E. Muller. A note on a method for generating points uniformly on n-dimensional spheres. *Commun. ACM*, 2(4):19–20, April 1959. ISSN 0001-0782. doi: 10.1145/377939.377946. URL `https://doi.org/10.1145/377939.377946`.

Dylan M Paiton, Charles G Frye, Sheng Y Lundquist, Joel D Bowen, Ryan Zarcone, and Bruno A Olshausen. Selectivity and robustness of sparse coding networks. *Journal of Vision*, 20(12):10–10, 2020.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29:3360–3368, 2016.

Paulo E Rauber, Samuel G Fadel, Alexandre X Falcao, and Alexandru C Telea. Visualizing the hidden activity of artificial neural networks. *IEEE transactions on visualization and computer graphics*, 23(1):101–110, 2016.

Paulo E Rauber, Alexandre X Falcao, and Alexandru C Telea. Projections as visual aids for classification system design. *Information Visualization*, 17(4):282–305, 2018.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Francisco Rodrigues, Mateus Espadoto, Roberto Hirata, and Alexandru C Telea. Constructing and visualizing high-quality classifier decision boundary maps. *Information*, 10(9):280, 2019.

Nicole C Rust, Odelia Schwartz, J Anthony Movshon, and Eero P Simoncelli. Spatiotemporal elements of macaque v1 receptive fields. *Neuron*, 46(6):945–956, 2005.

Anindya Sarkar and Raghu Iyengar. Enforcing linearity in dnn succours robustness and adversarial image generation. In Igor Farkaš, Paolo Masulli, and Stefan Wermter (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2020*, pp. 52–64, Cham, 2020. Springer International Publishing. ISBN 978-3-030-61609-0.

Carl-Johann Simon-Gabriel, Yann Ollivier, Leon Bottou, Bernhard Schölkopf, and David Lopez-Paz. First-order adversarial vulnerability of neural networks and input dimension. In *International Conference on Machine Learning*, pp. 5809–5817. PMLR, 2019.

Grzegorz Swirszcz, Brendan O'Donoghue, and Pushmeet Kohli. Visualizations of decision regions in the presence of adversarial examples. In *ICLR Debugging Machine Learning Models Workshop*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014. URL `https://openreview.net/forum?id=kklr_MTHMRQjG`.

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks, IJCNN*, pp. 426–433. IEEE, 2016.

Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

Dávid Terjék. Adversarial lipschitz regularization. In *International Conference on Learning Representations, ICLR*, 2020.

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

Georgios Vlassopoulos, Tim van Erven, Henry Brighton, and Vlado Menkovski. Explaining predictions by approximating the local decision boundary. *arXiv preprint arXiv:2006.07985*, 2020.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

David Warde-Farley and Ian J Goodfellow. Adversarial perturbations of deep neural networks. In Tamir Hazan, George Papandreou, and Daniel Tarlow (eds.), *Perturbations, Optimization, and Statistics*, pp. 311. MIT Press, 2016.

Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020.

Fuxun Yu, Zhuwei Qin, Chenchen Liu, Liang Zhao, Yanzhi Wang, and Xiang Chen. Interpreting and evaluating neural network robustness. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 4199–4205. International Joint Conferences on Artificial Intelligence Organization, 7 2019a. doi: 10.24963/ijcai.2019/583. URL `https://doi.org/10.24963/ijcai.2019/583`.

Tao Yu, Shengyuan Hu, Chuan Guo, Wei-Lun Chao, and Kilian Q Weinberger. A new defense against adversarial images: Turning a weakness into a strength. In *Advances in Neural Information Processing Systems, NeurIPS*, volume 32, 2019b.

Christoph Zetzsche and Gerhard Krieger. Nonlinear mechanisms and higher-order statistics in biological vision and electronic image processing: review and perspectives. *Journal of Electronic Imaging*, 10(1):56–100, 2001.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pp. 7472–7482. PMLR, 2019.

A Boundary curvature analysis

The following definitions are common. Our notation follows Lee (2013; 2018) for differential geometry, and Magnus & Neudecker (1999) for linear algebra.

A.1 Introduction

In order to derive our main curvature expressions for a graph submanifold, we first introduce some notation and recall some concepts.

Tangent Spaces If M is an n-dimensional smooth manifold then the *tangent space at* $p \in M$ is a vector space $\mathrm{T}_p M$ of derivations at p, which is isometric to $\mathbb{R}^n$. In particular, $\mathrm{T}_p M$ is the set of directional derivative operators acting on the smooth real functions $f : M \to \mathbb{R}$.

The disjoint union of all of the tangent spaces is the *tangent bundle* $\mathrm{T}M \stackrel{\text{def}}{=} \{\{p\} \times \mathrm{T}_p M \mid p \in M\}$. If we equip M with a smooth, bilinear, positive definite, symmetric form $g : \mathrm{T}M \times \mathrm{T}M \to \mathbb{R}$ then (M, g) is said to be a *Riemannian manifold* and g the *Riemannian metric tensor*.

When it is unambiguous, denote by $\Gamma(E)$ the smooth mappings $M \to E$ (for some topological space E). Most often we will refer to $\mathfrak{X}(M) \stackrel{\text{def}}{=} \Gamma(\mathrm{T}M)$, and represent elements of $\mathfrak{X}(M)$ with the letters X, Y, Z.

Differentials If M and N are manifolds and $F : M \to N$ is differentiable, then the *differential of F at $p \in M$* is the linear mapping denoted $\mathrm{d}F_p : \mathrm{T}_p M \to \mathrm{T}_p N$, where the action on tangent vectors is defined by

$$\forall_{p \in M} \forall_{v \in \mathrm{T}_p M} \forall_{f \in \mathrm{C}^1(N)} : \ \mathrm{d}F_p(v) f = v(f \circ F)(p).$$

That is, the rate of change of the function $f \circ F$ in the direction $v \in \mathrm{T}_p M$ at the point p. As is common practice, we often omit the p subscripts for clarity.

The linear operator $\mathrm{d}f$ is a linear functional when N in the above definition is $\mathbb{R}$, and at each $p \in M$ there exists a vector $\operatorname{grad} f_p \in \mathrm{T}_p$. So that

$$\forall_{p \in M} \forall_{v \in \mathrm{T}_p M} : \ \mathrm{d}f_p(v) = g(\operatorname{grad} f_p, v),$$

and $\operatorname{grad} f \in \mathfrak{X}(M)$ is the associated vector field.

Connections A *connection* is an operator $\nabla : \mathfrak{X}(M) \times \mathfrak{X}(M) \to \mathfrak{X}(M)$, linear in its first argument, and satisfying a product rule in its second. Surprisingly, a connection uniquely defines a connection in each tensor bundle,[3] (which we overload with ∇) (Lee, 2018, Prop. 4.15). Importantly for us, we have $\nabla_X f = Xf$ for all $X \in \mathfrak{X}(M)$ (for any connection).[4] A Riemannian manifold always posesses a *Levi–Cevita connection* (Lee, 2018, §5), denoted by ∇, which is the unique connection satisfying

$$\forall_{X,Y,Z \in \mathfrak{X}(M)} : \ \nabla_X g(Y, Z) = g(\nabla_X Y, Z) + g(Y, \nabla_X Z).$$

To $f \in \mathrm{C}^2(M)$ we associate the Hessian operator $\nabla^2 f$, which is a symmetric $(0, 2)$-tensor field (Lee, 2018, Ex. 4.22) with $\nabla^2 f(X, Y) \stackrel{\text{def}}{=} \nabla_X \, \mathrm{d}f(Y)$ for $X, Y \in \mathfrak{X}(M)$. We have (via Lee, 2018, Prop. 4.21)

$$\forall_{X,Y \in \mathfrak{X}(M)} : \ \nabla_X \, \mathrm{d}f(Y) = \nabla_X(\nabla_Y f) - \nabla_{(\nabla_X Y)} f = X(Yf) - (\nabla_X Y) f. \tag{4}$$

[3] Including the $(0, 0)$-tensor bundle, that is, the smooth real functions on M.

[4] That is, the action of a vector field through the connection agrees with the ordinary directional derivative.

Using the compatibility of ∇ with the metric we arrive at the following expression for the Hessian, which will be useful later

$$\begin{aligned}\nabla^2 f(X,Y) &\overset{(4)}{=} X(Yf) - (\nabla_X Y)f \\ &= \underbrace{Xg(\operatorname{grad} f, Y)}_{\nabla_X g(\operatorname{grad} f, Y)} - g(\operatorname{grad} f, \nabla_X Y) \\ &= g(\nabla_X \operatorname{grad} f, Y) + g(\operatorname{grad} f, \nabla_X Y) - g(\operatorname{grad} f, \nabla_X Y) \\ &= g(\nabla_X \operatorname{grad} f, Y). \end{aligned} \tag{5}$$

A.1.1 Submanifolds

If (M, g), $(\tilde{M}, \tilde{g})$ are two Riemanian manifolds with $M \subseteq \tilde{M}$, then an injection $\iota : M \hookrightarrow \tilde{M}$ is called an *inclusion mapping*. If the inclusion is a smooth isometry, that is

$$\forall_{u,v \in \mathrm{T}M} : \; g(u,v) = \tilde{g}(\mathrm{d}\iota(u), \mathrm{d}\iota(v)),$$

equivalently, $g = \iota^* \tilde{g}$, then M is called a *Riemanian submanifold of* $\tilde{M}$.

Using the inclusion map differential, we embed $\mathrm{T}M$ in $\mathrm{T}\tilde{M}$. If M, $\tilde{M}$ are of dimensions $m < n$, then $\mathrm{T}_p M$ occupies an m dimensional subspace of $\mathrm{T}_p\tilde{M}$ for each $p \in M$. The *normal bundle* is formed from the orthogonal compliment of $\mathrm{T}_p M$ at each $p \in M$, and is defined in a similar way to the tangent bundle, denoted $\mathrm{N}M$ and having a dimension of $n - m$. For vectors $v \in \mathrm{T}\tilde{M}$ we have the decomposition $v = v^\top + v^\perp$ with $v^\top \in \mathrm{T}M$ and $v^\perp \in \mathrm{N}M$.

Some care must be taken when taking covariant derivatives in $\mathrm{T}M$ with the ambient connection, formally

$$\forall_{X,Y \in \mathrm{T}M} : \; \tilde{\nabla}_X Y \overset{\text{def}}{=} \tilde{\nabla}_{\overline{\mathrm{d}\iota(X)}} \overline{\mathrm{d}\iota(Y)},$$

where the overlines denote arbitrary smooth extensions of $\mathrm{d}\iota(X)$ and $\mathrm{d}\iota(Y)$ to open neighbourhoods of M in $\tilde{M}$.

Second Fundamental Form Let $M \subseteq \tilde{M}$ be an embedded sumbanifold. The *second fundamental form* is the (bilinear) mapping $\mathrm{II} : \mathfrak{X}(M) \times \mathfrak{X}(M) \to \Gamma(\mathrm{N}M)$ with $\mathrm{II}(X,Y) \overset{\text{def}}{=} \left(\tilde{\nabla}_X Y\right)^\perp$. If we equip M with a normal field $N \in \Gamma(\mathrm{N}M)$, then the *scalar second fundamental form* is

$$\mathrm{II}_N(X,Y) \overset{\text{def}}{=} \tilde{g}(N, \mathrm{II}(X,Y)).$$

For the remainder we assume N is fixed. The *shape operator* $s : \mathfrak{X}(M) \to \mathfrak{X}(M)$ is the endomorphism obtained from II_N that satisfifes

$$\forall_{X,Y \in \mathfrak{X}(M)} : \; \mathrm{II}_N(X,Y) = g(sX, Y), \tag{6}$$

that is, by raising an index.

Curvature Let s be the shape operator associated to a normal field on an embedded submanifold M of dimension n. Then at $p \in M$, the (real) eigenvalues of s are called *principal curvatures* and the eigenvectors are called *principal directions* at p. The *Gaussian curvature* is the product of the principal curvatures, $\det(s)$, and the *mean curvature* is their average, $\frac{1}{n}\operatorname{tr}(s)$.

Assume (M, g) and $(\tilde{M}, \tilde{g})$ are Riemannian manifolds with an inclusion mapping $\iota : M \hookrightarrow \tilde{M}$ and M has the submanifold structure. Suppose $\tilde{M}$ posesses a shape operator (or any other linear endomorphism) $\tilde{s} : \mathrm{T}\tilde{M} \to \mathrm{T}\tilde{M}$. The submanifold structure then gives us an operator $\iota^* \tilde{s} : \mathrm{T}M \to \mathrm{T}M$, the pullback of s by ι, via

$$\forall_{X \in \mathrm{T}M} : \; \iota^* s X = \mathrm{d}\iota^{-1} s \, \mathrm{d}\iota(X).$$

Remark 1 (Euclidean subspace). We consider the special case of the Riemannian manifold $\mathbb{R}^m$. Suppose $P : \mathbb{R}^n \to \mathbb{R}^m$ is orthogonal ($n \leq m$) and let $s : \mathbb{R}^m \to \mathbb{R}^m$ be a linear operator. Then $P^\mathsf{T} : \mathbb{R}^n \hookrightarrow \mathbb{R}^m$ is an inclusion map and the *subspace pullback* of s to $\mathbb{R}^n$ is

$$\left(P^\mathsf{T}\right)^* s = P s P^\mathsf{T}.$$

We refer to the curvature quantities and directions of $\left(P^\mathsf{T}\right)^* s$ with the suspace prefix. That is, *subspace principal curvature*, *subspace Gaussian curvature*, and so on.

A.2 Hypersurfaces

Let (M, g) be an embedded submanifold of $(\tilde{M}, \tilde{g})$. Suppose $F : U \subseteq \tilde{M} \to \mathbb{R}$ is a *local defining function for* M, that is, $M \cap U = F^{-1}(\{0\})$ for an open $U \subseteq \tilde{M}$. Choose arbitrary $X, Y \in \mathfrak{X}(M)$. From the connection product rule (Lee, 2018, p. 89)

$$\tilde{\nabla}_X \Big(\tfrac{1}{|\operatorname{grad} F|} \cdot \operatorname{grad} F \Big) = \frac{1}{|\operatorname{grad} F|} \tilde{\nabla}_X \operatorname{grad} F + \Big(X \tfrac{1}{|\operatorname{grad} F|} \Big) \cdot \operatorname{grad} F.$$

Whence

$$\begin{aligned} \tilde{g}\big(\tilde{\nabla}_X N, Y\big) &= \frac{\tilde{g}\big(\tilde{\nabla}_X \operatorname{grad} F, Y\big)}{|\operatorname{grad} F|} + \left(X \frac{1}{|\operatorname{grad} F|} \right) \underbrace{\tilde{g}(\operatorname{grad} F, Y)}_{0} \\ &= \frac{\tilde{g}\big(\tilde{\nabla}_X \operatorname{grad} F, Y\big)}{|\operatorname{grad} F|}. \end{aligned} \tag{7}$$

The zero in the underbrace follows because $\operatorname{grad} F \in \mathrm{N}M$ and $Y \in \mathrm{T}M$. The Weingarten equation (Lee, 2018, Thm. 8.13(c)) yields $sX = -\tilde{\nabla}_X N$, giving us

$$-\mathrm{II}_N(X, Y) \overset{(6)}{=} -g(sX, Y) = \tilde{g}\big(\tilde{\nabla}_X N, Y\big) \overset{(7)}{=} \frac{\tilde{g}\big(\tilde{\nabla}_X \operatorname{grad} F, Y\big)}{|\operatorname{grad} F|} \overset{(5)}{=} \frac{\tilde{\nabla}^2 F(X, Y)}{|\operatorname{grad} F|}. \tag{8}$$

A.2.1 The Curvature of a Graph Manifold

The notation and conventions of this section closely follow (Lee, 2018, §8). Let M be the graph of a smooth function $f : \mathbb{R}^n \to \mathbb{R}$, that is, $M = \{(x, f(x)) \mid x \in \mathbb{R}^n\}$. Since f is smooth M is indeed a manifold (Lee, 2013, Ex. 1.30, p. 20). We regard (M, g) as a Riemannian submanifold of $(\mathbb{R}^{n+1}, \langle \cdot, \cdot \rangle)$, with the inclusion map $\iota : M \hookrightarrow \mathbb{R}^{n+1}$.

The function $F : \mathbb{R}^{n+1} \to \mathbb{R}$ is a global defining function for M with $F(x, t) \overset{\text{def}}{=} f(x) - t$. Denote by ∇f the vector of partial derivatives of f, and by H_f and H_F, the matrices of second order partial derivatives of F and f, all with respect to usual coordinate frames, (x^i), on $\mathbb{R}^n$ and $\mathbb{R}^{n+1}$. Then

$$\forall_{X, Y \in \mathrm{T}\tilde{M}} : \ \nabla^2 F(X, Y) = \langle \mathrm{H}_F X, Y \rangle.$$

At $p \in M$ the differential $\mathrm{d}\iota_p$ has the Jacobian matrix

$$\big(\mathrm{Id} \quad \nabla f(x^1(p), \ldots, x^n(p)) \big)^{\mathsf{T}}.$$

We commit the mild sin of identifying $\mathrm{d}\iota_p$ with its Jacobian and $X, Y \in \mathrm{T}\tilde{M}$ with their parameterizations under the coordinate frame, so that we may use the notation of matrix multiplication.

In dropping the p-subscripts, the left inverse, $(\mathrm{d}\iota)^{-1}$, is has the parameterization

$$\big((\mathrm{d}\iota)^{\mathsf{T}} (\mathrm{d}\iota) \big)^{-1} (\mathrm{d}\iota)^{\mathsf{T}} = \big(\mathrm{Id} + \nabla f \, \nabla f^{\mathsf{T}} \big)^{-1} (\mathrm{d}\iota)^{\mathsf{T}}.$$

Thus

$$(\mathrm{d}\iota)^{\mathsf{T}} \, \mathrm{H}_F (\mathrm{d}\iota) = \begin{pmatrix} \mathrm{Id} \\ \nabla f^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \mathrm{H}_f & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{Id} \\ \nabla f^{\mathsf{T}} \end{pmatrix} = \mathrm{H}_f,$$

whence

$$\begin{aligned} (\mathrm{d}\iota)^{-1} \, \mathrm{H}_F (\mathrm{d}\iota) &= \big(\mathrm{Id} + \nabla f \, \nabla f^{\mathsf{T}} \big)^{-1} (\mathrm{d}\iota)^{\mathsf{T}} \, \mathrm{H}_F (\mathrm{d}\iota) \\ &= \big(\mathrm{Id} + \nabla f \, \nabla f^{\mathsf{T}} \big)^{-1} \mathrm{H}_f \, . \end{aligned} \tag{9}$$

We can now compute the shape operator of M associated to the unit normal field $\operatorname{grad} F / |\operatorname{grad} F|$. Pick arbitrary $X, Y \in \mathrm{T}M$. From (8) there is

$$\begin{aligned} -\mathrm{II}_N(X, Y) \cdot |\operatorname{grad} F| &\overset{(8)}{=} \tilde{g}(\tilde{\nabla}_X \operatorname{grad} F, \mathrm{d}\iota(Y)) \\ &= g\big(\mathrm{d}\iota^{-1} \circ \mathrm{H}_F \circ \mathrm{d}\iota(X), Y \big), \end{aligned} \tag{10}$$

and, using the bilinearity of of g we have

$$g(sX, Y) \overset{(6)}{=} \mathrm{II}_N(X, Y) \overset{(10)}{=} -\frac{g(\mathrm{d}\iota^{-1}\,\mathrm{H}_F\,\mathrm{d}\iota X, Y)}{|\mathrm{grad}\,F|}.$$

This, together with (9) yields the shape operator:

$$s = -\frac{\left(\mathrm{Id} + \nabla f\,\nabla f^{\mathsf{T}}\right)^{-1}\mathrm{H}_f}{\sqrt{|\nabla f|^2 + 1}}. \tag{11}$$

Whence the Gaussian and mean curvature are the scalar fields

$$-\frac{\det\left(\left(\mathrm{Id} + \nabla f\,\nabla f^{\mathsf{T}}\right)^{-1}\mathrm{H}_f\right)}{\left(|\nabla f|^2 + 1\right)^{\frac{n}{2}}} \quad \text{and} \quad -\frac{\mathrm{tr}\left(\left(\mathrm{Id} + \nabla f\,\nabla f^{\mathsf{T}}\right)^{-1}\mathrm{H}_f\right)}{n\left(|\nabla f|^2 + 1\right)^{\frac{1}{2}}} \tag{12}$$

respectively.

A.2.2 The Curvature of a Level-Set Manifold

We now consider the case where M is the level-set of a smooth function $f : \mathbb{R}^n \to \mathbb{R}$ so that for some $c \in \mathbb{R}$ we have $M = \{x \in \mathbb{R}^n \mid f(x) = c\}$. The following theorem can be used to ensure the level-sets of f do indeed posses a manifold structure, which is a simplification of the constant rank level-set theorem (Lee, 2013, Thm. 5.12) for our setting.

Theorem 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a smooth function having constant rank r. Then the level-sets of f are all smooth manifolds of codimension r.*

We use the same construction as Lee (2013, Ex. 1.32, §5) to calculate the Gaussian and mean curvature at a point $(x_0, y_0) \in (\mathbb{R}^{n-1} \times \mathbb{R}) \cap M$.

From the implicit function theorem (Lee, 2013, Thm. C.40) there are open connected neighbourhoods $U_0 \times V_0 \subseteq \mathbb{R}^{n-1} \times \mathbb{R}$ with $(x_0, y_0) \in U_0 \times V_0$ and a smooth function $g : U_0 \to V_0$ so that $(U_0 \times V_0) \cap M$ is the graph of g. That is, $(U_0 \times V_0) \cap M = \left\{(x, g(x)) \;\middle|\; x \in U_0 \subseteq \mathbb{R}^{n-1}\right\}$. Consequentially the gradient of g has the structure

$$\begin{aligned} \forall_{x \in U_0} :\; f(x, g(x)) = c \implies & \mathrm{D}_1 f(x, g(x)) + \mathrm{D}_2 f(x, g(x)) \nabla g(x) = 0 \\ \iff & \nabla g(x) = -[\mathrm{D}_2 f]^{-1}(x, g(x))\,\mathrm{D}_1 f(x, g(x)). \end{aligned} \tag{13}$$

Let $\partial_i \overset{\text{def}}{=} \partial/\partial x^j$ and $\partial^2_{ij} \overset{\text{def}}{=} \partial^2/\partial x^i\,\partial x^j$. Then for $i, j \in [n-1]$ and $x \in U_0$

$$\partial_i g(x) = -(\partial_n f(x, g(x)))^{-1} \partial_i f(x, g(x)), \tag{14}$$

and

$$\begin{aligned} \partial^2_{ij} g(x) &= \frac{1}{(\partial_n f)^2}\left[\partial^2_{nj} f + \partial^2_{nn} f \partial_j g\right](\partial_i f) - \frac{1}{\partial_n f}\left[\partial^2_{ij} f + \partial^2_{nj} f \partial_j g\right] \\ &= -\frac{1}{\partial_n f}\left[\partial^2_{nj} f + \partial^2_{nn} f \partial_j g\right](\partial_i g) - \frac{1}{\partial_n f}\left[\partial^2_{ij} f + \partial^2_{nj} f \partial_j g\right] \\ &= -\frac{1}{\partial_n f}\left[\partial^2_{nj} f \partial_i g + \partial^2_{nn} f \partial_i g \partial_j g + \partial^2_{ij} f + \partial^2_{nj} f \partial_j g\right] \\ &= -\frac{1}{\partial_n f}\left[\partial^2_{nj} f(\partial_i g + \partial_j g) + \partial^2_{nn} f \partial_i g \partial_j g + \partial^2_{ij} f\right], \end{aligned} \tag{15}$$

where every function on the right hand side is evaluated at either the point x or $(x, g(x))$ (depending on the function).[5] If we let $\bar{\nabla} \overset{\text{def}}{=} (\partial/\partial x^1, \ldots, \partial/\partial x^{n-1})^{\mathsf{T}}$, and $\bar{\mathrm{H}}$ be, similarly, the $(n-1) \times (n-1)$ matrix of cross-partial derivatives, there are the more compact expressions for (14) and (15):

$$\nabla g = -\frac{1}{\partial_n f}\bar{\nabla} f,$$

[5] This is easy to do at x_0 since $(x_0, g(x_0)) = (x_0, y_0)$.

Table 2: Accuracies of CIFAR models at different perturbation lengths

	Dimension									
ϵ	1	2	3	4	5	6	7	8	9	10
0	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%	95.25% / 90.83%
0.1	51.43% / 86.29%	62.86% / 90.83%	70.48% / 90.83%	76.2% / 90.83%	79.06% / 90.83%	80.01% / 90.83%	81.92% / 90.83%	81.92% / 90.83%	82.87% / 90.83%	83.82% / 90.83%
0.25	1.91% / 79.93%	4.76% / 89.01%	5.72% / 89.92%	6.67% / 90.83%	7.62% / 90.83%	8.57% / 90.83%	9.52% / 90.83%	9.52% / 90.83%	9.52% / 90.83%	10.48% / 90.83%
0.5	0.0% / 65.4%	0.0% / 84.47%	0.0% / 86.29%	0.0% / 87.2%	0.0% / 87.2%	0.0% / 87.2%	0.0% / 87.2%	0.0% / 87.2%	0.0% / 87.2%	0.0% / 87.2%
1	0.0% / 36.33%	0.0% / 54.5%	0.0% / 59.04%	0.0% / 61.76%	0.0% / 65.4%	0.0% / 69.03%	0.0% / 69.94%	0.0% / 70.85%	0.0% / 72.66%	0.0% / 73.57%
2	0.0% / 2.72%	0.0% / 5.45%	0.0% / 10.9%	0.0% / 12.72%	0.0% / 13.62%	0.0% / 14.53%	0.0% / 15.44%	0.0% / 15.44%	0.0% / 16.35%	0.0% / 17.26%

and

$$\mathrm{H}_g = -\frac{1}{\partial_n f}\Big[\big(\nabla g + \nabla g^\mathsf{T}\big)\,\mathrm{diag}\big(\bar{\nabla}\partial_n f\big) + \partial^2_{nn} f \cdot \big(\nabla g \nabla g^\mathsf{T}\big) + \bar{\mathrm{H}}_f\Big],$$

where the sum $\nabla g + \nabla g^\mathsf{T}$ is to be interpreted as the matrix of pairwise sums of the elements of ∇g: $\big(\nabla g + \nabla g^\mathsf{T}\big)_{ij} \stackrel{\text{def}}{=} \partial g/\partial x^i + \partial g/\partial x^j$. Then, as in Section A.2.1, $F(x) \stackrel{\text{def}}{=} g(x^1, \ldots, x^{n-1}) - x^n$ is a local defining function for M on $(U_0 \times V_0) \cap M$. The local graph parameterization provided by g is $\phi : U_0 \to M$ with $\phi(x) \stackrel{\text{def}}{=} (x, g(x))$, and its differential has the parameterization

$$\mathrm{d}\phi = (\mathrm{Id} \quad \nabla g)^\mathsf{T}.$$

Finally, *mutatis mutandis*, the shape operator is given by (11) and the Gaussian and mean curvature can be computed as in (12).

Remark 2 (Numerical considerations). In order to compute the level-set curvature we needed to apply the implicit function theorem. However, it may be difficult numerically to stabilize the inverse operation in (13). To remedy this problem, one may apply a change of basis before computing the quantities (14) and (15). Let $B \in \mathbb{R}^{n\times n}$ be an orthogonal change of basis operator. Then we apply all our operations to $B^\mathsf{T} M \stackrel{\text{def}}{=} \big\{B^\mathsf{T} x \;\big|\; x \in M\big\}$.

Differentiating $\tilde{f} \stackrel{\text{def}}{=} f \circ B^\mathsf{T}$ yields the revised quantities

$$\forall_{x\in\mathbb{R}^n} : \; \nabla \tilde{f}(x) = B \cdot \nabla f(B^\mathsf{T} x) \quad \text{and} \quad \mathrm{H}_{\tilde{f}}\,|_x = B\,\mathrm{H}_f\,|_{(B^\mathsf{T} x)} B^\mathsf{T}.$$

B Dataset and models

For the adversarial subspace analysis we use two identical ResNet-50 (He et al., 2016) architectures. The first model was trained on the unmodified CIFAR-10 dataset (Krizhevsky & Hinton, 2009). For the second model we additionally incorporated adversarial examples in the training process, widely known as adversarial training. A training perturbation size ϵ of 0.5 was used. To perform the curvature analysis, we used twice-differentiable ELU activation functions (Clevert et al., 2015) for both models, which had little impact on the clean or adversarial accuracies.

In the analysis we include 100 CIFAR-10 test images, class-balanced, all unseen during training, and correctly classified by both models. The optimization algorithm stopped after finding 50 orthogonal adversarial vectors. For all test image we found 50 dimensions on both models. For the curvature analysis we used a subset of 50 class-balanced images and 10 adversarial directions per image.

Additionally, we only considered one seed per CIFAR-10 model because we found that the analysis on the MNIST trained models had little variation across seeds (see Appendix D).

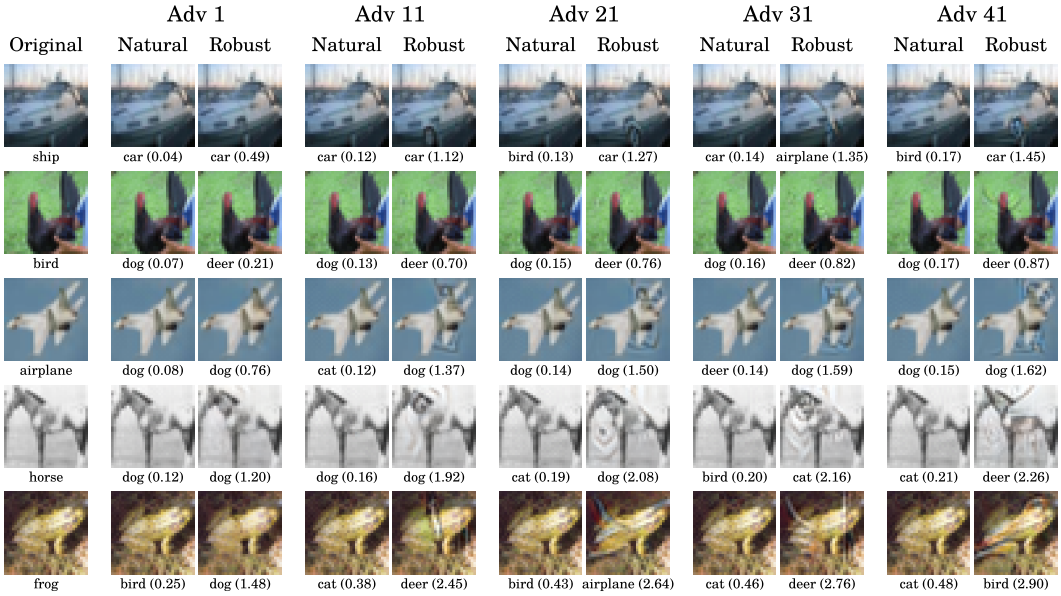

Figure 6: **CIFAR-10 example adversarials**. The panels show adversarial examples for a given input image. Column pairs are sorted by adversarial dimension. Within a pair, each column includes examples for the natural and robust models, respectively. The labels below the images indicate the class assigned by the model. The numbers in brackets are the ℓ_2-norms of the respective images.

The accuracy of both models at different perturbation lengths ϵ and dimensions is shown in Table 2. Here, the first line of each row corresponds to the accuracy of the naturally trained model, whereas the second line is referring to the adversarially trained model.

We repeated many of the experiments on the MNIST dataset (LeCun et al., 1998) and found largely consistent results (see Appendix G).

C Additional CIFAR analysis and supplementary figures

Here we show additional analysis to complement the above described findings on CIFAR-10 data. As was described earlier, we uniformly randomly sampled directions within adversarial subspaces to measure the distance to the decision boundary. Figure 7 shows how many of the samples pointed to decision boundaries that were outside of the allowed input pixel bounds.

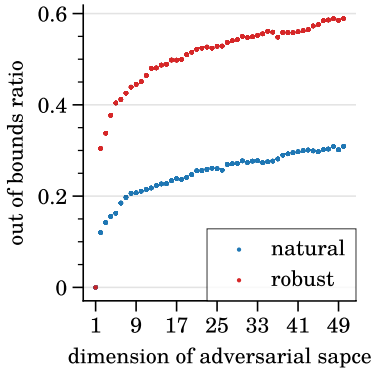

Figure 7: **Ratio of out of bounds samples**. We uniformly sample the distance to the decision boundary with increasing dimensionality per input image and adversarial subspace dimension. We report the fraction of directions pointing to out-of-bounds boundaries, averaged across 100 images.

Figure 6 shows five adversarially perturbed images and their corresponding original image found for the two investigated models. We can see that that the perturbation length is increasing with dimension number. Interestingly, the adversarially trained model seems to have more semantically meaningful adversarials, e.g. the images of boats in the top row, misclassified as cars, are perturbed to have elements looking like wheels. We can also see that there is no clear, explicit ℓ_2-length of an adversarial attack that makes an image ambiguous for a human observer. For example, the robust model example in the first (ship) row of the adversarial 11 column pair has a

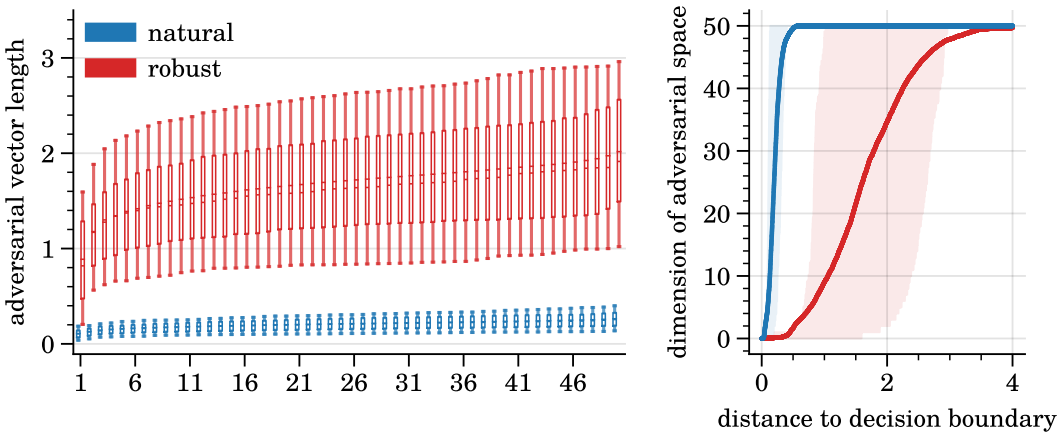

Figure 8: **Perturbation lengths and dimensionality**. (left) The subfigure shows the distribution of the n-th found adversarial perturbations across images. (right) We tested at distance increments of 0.001 how many adversarial dimensions were found at that distance per image. The lines represent the means over all test images and the color shaded areas show the 10th and 90th percentiles.

perturbation length of 1.12 and could be mistaken as a car on a first glance. In contrast, the robust model example in the fourth (horse) row of the adversarial 1 column pair is still clearly identifiable as a horse with a larger perturbation length of 1.2. This illustrates that the direction of the perturbation is as important as the perturbation magnitude itself, as was previously reported by (Goodfellow et al., 2014; Ilyas et al., 2019), although assessing the "perceptual distance" is an open area of research.

Figure 8 shows that perturbation lengths are increasing sublinearly and monotonically with increasing number of adversarial dimensions. For the naturally trained model there is also a much larger increase of adversarial dimension with increasing distance to distance to decision boundary. Note that in the right panel of this figure both models plateau at 50 dimensions because we stopped the optimizer at this point.

We also provide additional, and more detailed information on the distributions of distances to the decision boundary of randomly sampled vectors in adversarial subspaces in Figure 9. We observe that the variance of the distances to the decision boundary of randomly sampled vectors is smoothly increasing with dimensionality.

D Seed consistency

We trained the natural and robust MNIST models with five different seeds and extracted adversarial subspaces for each model to examine the influence of training seeds on our results. Figure 10 shows that there is slight variation in number of adversarial subspace directions found by the optimizer. However, the overall trend of bar heights is similar across seeds and there are no significant outliers.

To further validate our results, we repeated the experiment for Figure 14 (measuring the length of the adversarial vector as one increases dimensionality on the MNIST trained networks) across five different seeds per training type. The results are shown in figure 11 The variation across seeds for a single image is smaller than the variation across images for a single seed, which can be seen by comparing Figure 11 to Figure 14. In general, we also found that the result trends were consistent between MNIST and CIFAR networks (with differences noted in Appendix G). Thus, we considered one seed per CIFAR-10 model as sufficiently representative.

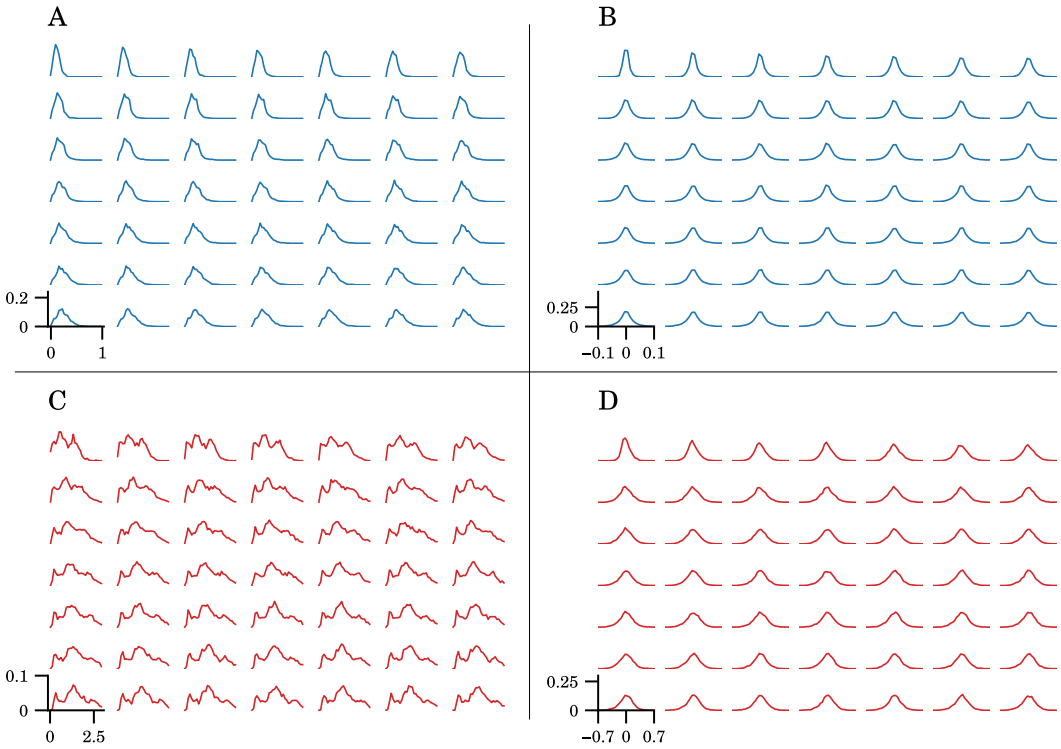

Figure 9: **Distributions of distances to decision boundary**. (A,B) Distributions for natural model. (C,D) Distributions for adversarially trained model. (A-C) We show the complete set histograms corresponding to Figure 2 dimensions. (C,D) Additionally, we provide the mean-normalized histograms to corresponding to the histograms in A and C to emphasize the smooth increase in variance of distances with increasing dimensionality.

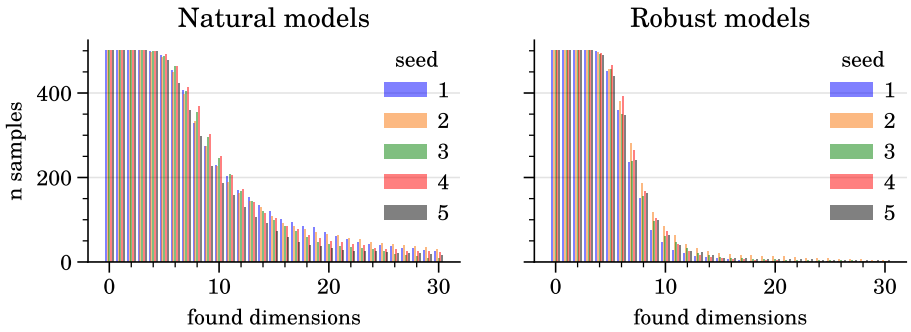

Figure 10: **Optimization consistency on MNIST**. The bars show number of samples for which at least n dimensions were found during optimization. Different colors indicate distinct seeds for model training.

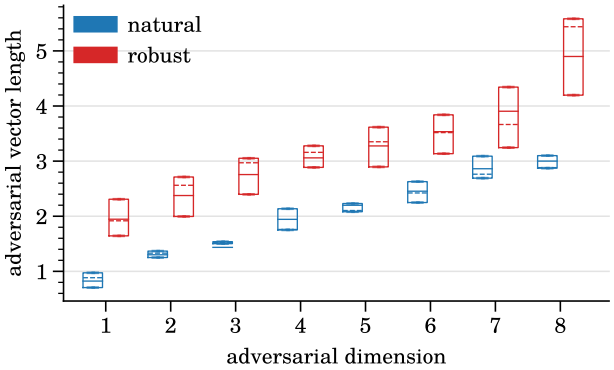

Figure 11: **Distance consistency on MNIST**. Box plots show the length of the adversarial vector as one increases dimensionality. The data is computed for a single image, where the variation comes from five different seeds used to initialize the networks before training. Note that the variation is much smaller than the variation across images for a single seed, as can be seen in Figure 14.

E Orthogonality and clipping

One obstacle in finding proper adversarial subspaces spanned by *orthogonal* vectors is assuring that perturbations do not cause pixel values to be out of the valid input boundaries $[0, 1]$. We noticed that many studies solve this problem by element wise pixel clipping. Here, we show that pixel clipping is problematic when talking about the dimensionality of subspaces. Element wise pixel clipping is a non-linear operation performed on a vector and can therefore change its direction in the input space and its relative angle to other, fixed vectors.

To show that clipping has a non-negligible effect, we investigated relative angles of basis vectors that span (sub-) spaces for 100 MNIST test images relevant for two studies.

First, in their work, Tramèr et al. (2017) propose the GAAS method. GAAS estimates the dimensionality of adversarial subspaces by defining an orthogonal set of vectors aligned with an attack vector to an input image. However, before passing the orthogonal basis vectors to the model, a clipping operation is performed. Figure 12 (left) shows that this clipping operation causes the input vectors to be non-orthogonal - orthogonality would constitute an angle of 90° or $\frac{\pi}{2}$. In fact, for the 100 sample images we tested, there is not a single vector pair that is completely orthogonal. The author's conclusions regarding the dimensionality of adversarial subspaces and transferable subspaces are therefore grounded on ill-defined subspaces, spanned by non-orthogonal vectors.

In another study, He et al. (2018) estimate the distance to the decision boundary for the complete input space of MNIST. For that, a basis of 784 vectors, spanning the whole input space, is randomly generated. Again, the inputs are clipped before they are passed to the model leading to the same problem as described earlier. And as before, there are no

orthogonal vector pairs, and angels are accumulated at much smaller values than $\frac{\pi}{2}$as can be seen in Figure 12 (right). We therefore note that it is not guaranteed that the input space is evenly covered and that edges of the 784-dimensional hypercube (in the case of MNIST) are arguably overrepresented.

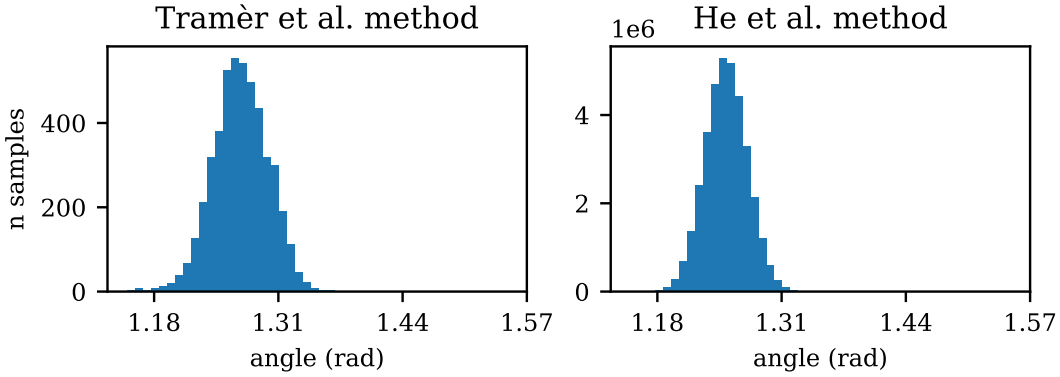

Figure 12: **Clipping causes non-orthogonality**. The histograms show angles between vectors that define "subspaces" in the GAAS method proposed by Tramèr et al. (2017) (left) and distance to decision boundary estimations by He et al. (2018).

F Additional related work on decision boundary curvature

Several works from Moosavi-Dezfooli and colleagues explore decision boundary curvature and its relationship to adversarial robustness (Fawzi et al., 2016; Moosavi-Dezfooli et al., 2017b; 2018; 2019; Moosavi-Dezfooli, 2019). Within these works they provide an inspired perspective on explaining adversarial robustness in terms of the curvature of decision boundaries. Almost all of the work is with respect to *universal* adversarial perturbations, which share a common subspace (Moosavi-Dezfooli et al., 2017a), although Fawzi et al. (2016) alternatively look at random subspaces. The universal perturbation subspaces differ from our adversarial subspaces in that theirs are defined by orthogonal directions that transfer well across inputs (in terms of proximity to the decision boundary) instead of those that point to the nearest decision boundary for a given input. Despite these differences, the stated quantity of interest, that is, the principle curvatures at a point on the decision boundary is the same in their work as well as ours.

However, each of these papers have a slightly different analytic formula for estimating the curvature of the boundary within a given subspace (Fawzi et al., 2016; Moosavi-Dezfooli et al., 2017b), or in the high-dimensional input space (Moosavi-Dezfooli et al., 2018; 2019). We found the most complete description to be given in Moosavi-Dezfooli (2019), which summarizes a majority of the earlier work and applies the decision boundary curvature analysis to both subspaces and the high-dimensional input space. Therein, Moosavi-Dezfooli (2019, Section. 7.4) state the following formula, without derivation, for the shape operator (in the notation of Section 2.2):

$$s = \frac{1}{|\nabla F_{ij}(x)|_2} P \, \mathrm{H}_{F_{ij}} \, P,$$

where $\mathrm{H}_{F_{ij}}$ is the Hessian of the boundary function, $F_{ij}(x) \stackrel{\text{def}}{=} f_i(x) - f_j(x)$, $P \stackrel{\text{def}}{=} \mathrm{Id} - \nabla F_{ij}(x) \nabla F_{ij}(x)^{\mathsf{T}}$ is described as a projection operator. This formulation is similar to that used by (Poole et al., 2016), with the only difference being that they use normalized gradients in the projection operator.

Our formulation differs from the above in several ways, the most important of which is that they claim to transform the Hessian by projecting it onto the tangent space (as written above), while we apply the implicit function theorem to parameterize the decision boundary as the graph of a function (see Appendix Section A.2.2, Equation 13). For computing the curvature of the decision boundary, exclusively projecting the Hessian is not going to result in

Table 3: Calculations from alternative attempts find incorrect principal curvature for a level-set on a sphere of varying dimensionality and radius. The quantity reported is the mean of the absolute difference between each computed principal curvature and the inverse radius.

dimensionality-radius	Moosavi et al.	Poole et al.	Ours
02 - 00.1	7.6e-01	5.0e+00	0.0e+00
02 - 01.0	2.4e+01	5.0e-01	0.0e+00
02 - 10.0	3.1e+04	5.0e-02	1.3e-17
05 - 00.1	7.2e-01	2.0e+00	2.2e-15
05 - 01.0	7.2e+01	2.0e-01	2.2e-16
05 - 10.0	7.9e+04	2.0e-02	3.8e-17
10 - 00.1	6.4e-01	1.0e+00	2.3e-15
10 - 01.0	1.5e+02	1.0e-01	2.2e-16
10 - 10.0	1.5e+05	1.0e-02	4.4e-17

the correct calculations, since for the decision boundary (as defined in Section 2.2, Equation 2) you need a different normal than for the graph of the response function. More specifically, the normal used for computing the shape operator of the graph surface is different from that for computing the shape operator of the decision boundary. The latter has to be exclusively in the pixel space, while the former always has a component in the direction of the function response. Projecting the Hessian can restrict the computed curvatures in a certain subspace of directions, but it cannot change the computed curvatures in said subspace. Another important difference is their projection operator is computed directly from the gradient, and thus has a dimensionality equal to the input dimensionality. To calculate the curvature of a decision boundary, one needs a level-set formulation, which should have one fewer dimensions than the input space.

The consequence of this difference can be made clear when looking at the curvature of an n-dimensional sphere, as defined by the quadratic function:

$$F(x) = x^T H x,$$

where x is the input point vector and H is the Hessian (i.e. an identity matrix in $\mathbb{R}^{n \times n}$). Given this function, the principal curvatures of the level-set at some radius, r, are all known to be (up to a sign change) $\frac{1}{r}$. However, we show in table 3 that the formulations provided by Moosavi-Dezfooli (2019) and Poole et al. (2016) produced curvatures that are often several orders of magnitude off from the correct values the case of high-dimensional inputs. Due to the dimensionality issue above, we computed the average error across each principal curvature, regardless of the number of curvatures reported, as they should all equal $\frac{1}{r}$.

We additionally tested a graph formulation (defined in Section A.2.1), using a one-sheeted hyperboloid. This formulation lives in the full space, so there is a principal curvature per input dimension. However, we still found similar errors 4. Note, that the formulation from (Golden et al., 2019) matched our own, although they did not provide an equation for mapping this to the level-set curvature. Our own implementations were correct with high precision on these and other experiments with known functions, all of which can be replicated from our provided code.

Finally, while the subspace projection methods proposed by (Fawzi et al., 2016; Moosavi-Dezfooli et al., 2017b) could potentially provide metrics related to curvature in the subspaces they defined, we chose to instead derive our pullback method described in Equation 3 from primary sources, as explained in Appendix A.

9 Poole et al. 02 0.1 10 Poole et al. 02 1.0 11 Poole et al. 02 10.0 12 Poole et al. 05 0.1 13 Poole et al. 05 1.0 14 Poole et al. 05 10.0 15 Poole et al. 10 0.1 16 Poole et al. 10 1.0 17 Poole et al. 10 10.0

G MNIST results

Table 4: Calculations from alternative attempts find incorrect Gaussian curvature (i.e. the product of the principal curvatures) for a Hyperboloid graph.

Analysis type	Absolute error
Moosavi et al.	6.8e+00
Poole et al.	1.4e-03
Golden et al.	4.1e-10
Ours	4.1e-10

Table 5: Accuracies of MNIST models (seed 0) at different perturbation lengths

	Dimension									
ϵ	1	2	3	4	5	6	7	8	9	10
0	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%	98.59% / 98.56%
0.5	90.51% / 97.77%	97.41% / 98.56%	98.0% / 98.56%	98.39% / 98.56%	98.39% / 98.56%	98.39% / 98.56%	98.37% / 98.56%	98.35% / 98.56%	98.29% / 98.56%	98.23% / 98.56%
1	60.53% / 94.03%	90.31% / 98.36%	95.83% / 98.56%	97.01% / 98.56%	97.4% / 98.56%	97.58% / 98.56%	97.94% / 98.56%	97.86% / 98.56%	97.69% / 98.56%	97.51% / 98.56%
2	1.77% / 68.99%	18.34% / 89.3%	47.91% / 95.01%	71.58% / 96.59%	83.54% / 97.57%	88.51% / 97.69%	91.63% / 97.73%	93.0% / 97.72%	93.8% / 98.56%	93.55% / 98.56%
4	0.0% / 0.59%	0.0% / 9.46%	0.2% / 27.99%	4.14% / 46.52%	15.05% / 65.77%	26.41% / 76.32%	32.43% / 82.59%	38.61% / 83.94%	37.46% / 85.42%	32.74% / 80.16%
6	0.0% / 0.0%	0.0% / 0.0%	0.0% / 0.2%	0.39% / 7.69%	1.98% / 24.69%	7.86% / 37.51%	10.45% / 47.63%	18.94% / 48.86%	16.48% / 54.54%	16.55% / 40.74%

We conducted the same experiments as described in Section 3 on networks trained with the MNIST dataset. As a test set we used 500 images, unseen in training and correctly labeled by all investigated models.

In Figure 16 we can see that perturbation lengths are much larger for MNIST than for CIFAR-10, although the relative increase in distances for the robust network is still present. The optimization was in general more difficult with MNIST, likely due to the fact that the images all live very close to the allowable input bounds. Specifically, the algorithm found less than 30 adversarial dimensions for more than 95% of the test images (Figure 10), to be compared with finding >50 dimensions for 100% of the CIFAR test images.

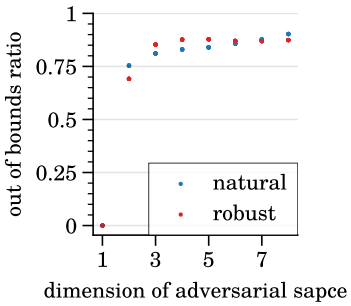

Figure 13: **Ratio of out of bounds samples**. The same data as in Figure 7 is shown for MNIST.

That it is harder to fool the MNIST model can also be seen in Table 5: the accuracies of both models are much higher for the same perturbation lengths than the accuracies of the CIFAR-10 models. For the curvature analysis, on the other hand, we found that the overall values to be larger on CIFAR. Additionally, the adversarially trained model had more decision boundary curvature within adversarial subspaces on MNIST than on CIFAR.

We additionally noticed that most of the MNIST adversarials lie in a thin regions between the decision boundary and the valid pixel box boundary (compare Figure 15). Figure 13 shows that this leads to large rates of out of bounds samples when measuring the distance to the decision boundary. For example 90% of the time the decision boundary is outside the valid pixel range in the 6 dimensional adversarial subspace of the naturally trained model.

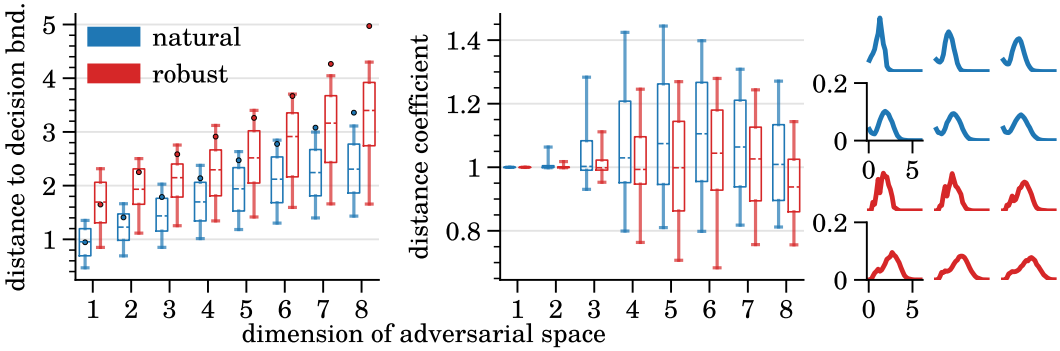

Figure 14: **MNIST distances to the Decision Boundary.** We repeated the distance analysis from Figure 2 on the MNIST dataset Only test images with at least 8 adversarial dimensions were included in this Figure

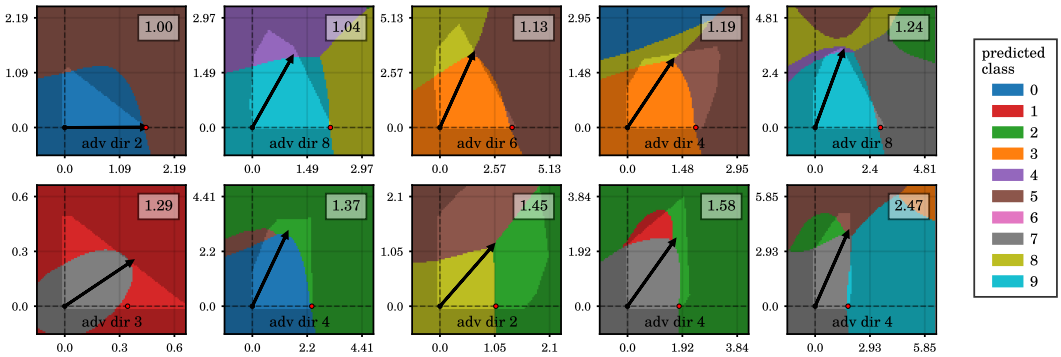

Figure 15: **MNIST boundary visualizations.** This figure is the MNIST version of Figure 3. Note that for MNIST, the decision boundary lies outside the valid pixel range more often than for CIFAR.

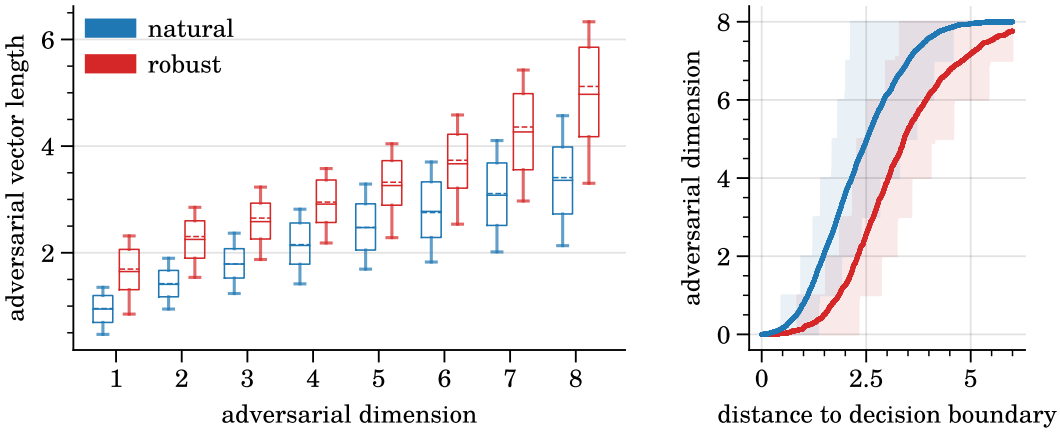

Figure 16: **MNIST perturbation lengths and dimensions.** Compare with Figure 2.

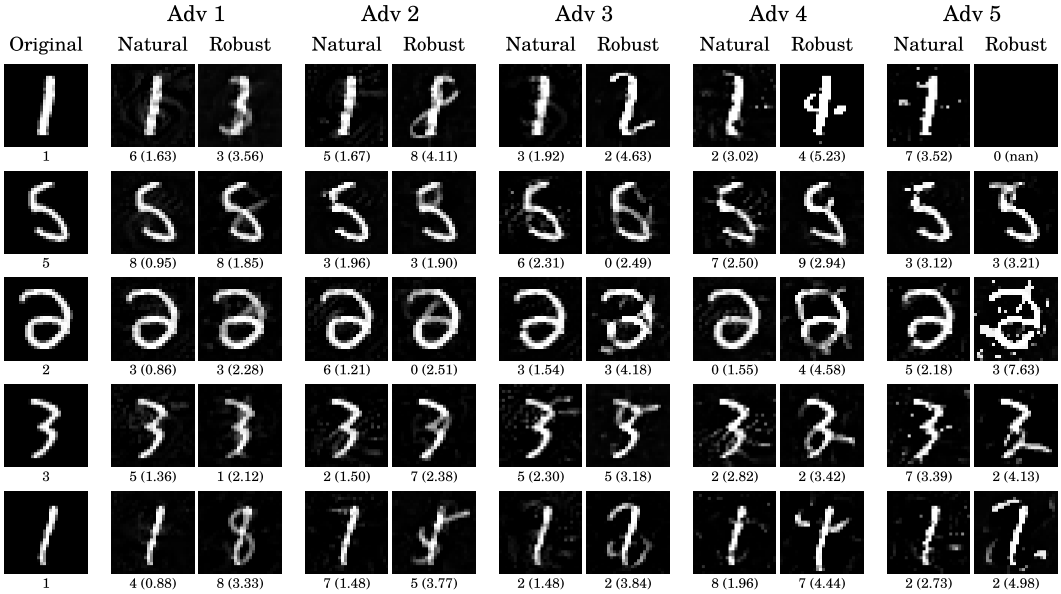

Figure 17: **MNIST example adversarials.** Compare to Figure 6.

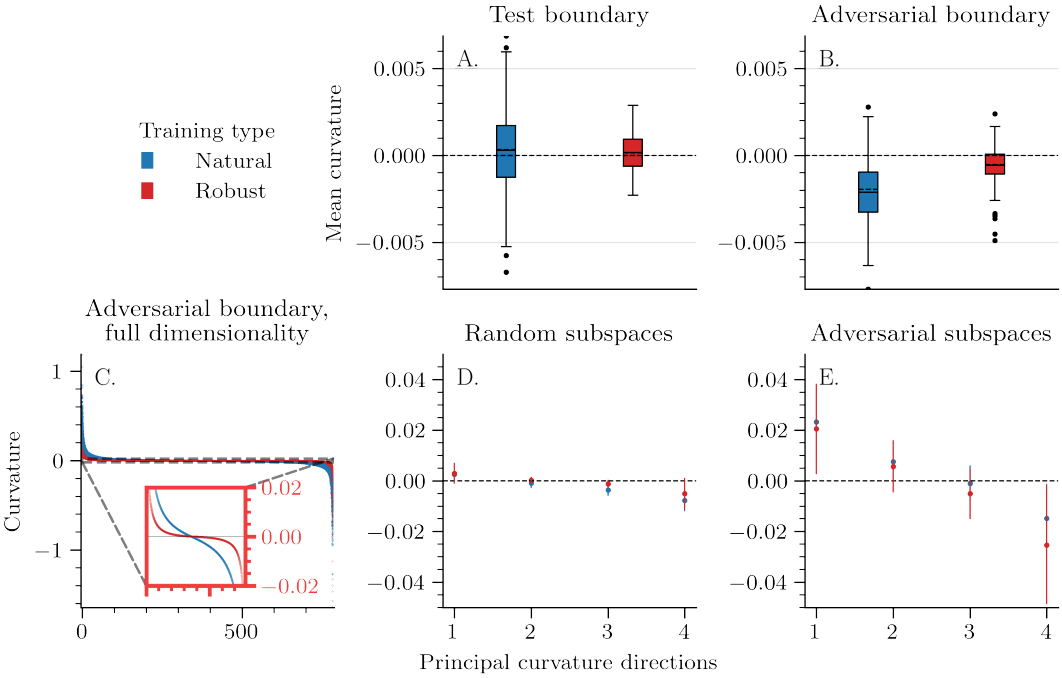

Figure 18: **MNIST decision boundary curvature.** Compare with Figure 4