Empirical study of the topology and geometry of deep networks

Alhussein Fawzi*† fawzi@cs.ucla.edu
Seyed-Mohsen Moosavi-Dezfooli*‡ seyed.moosavi@epfl.ch
Pascal Frossard‡ pascal.frossard@epfl.ch
Stefano Soatto† soatto@cs.ucla.edu

Abstract

The goal of this paper is to analyze the geometric properties of deep neural network image classifiers in the input space. We specifically study the topology of classification regions created by deep networks, as well as their associated decision boundary. Through a systematic empirical study, we show that state-of-the-art deep nets learn connected classification regions, and that the decision boundary in the vicinity of datapoints is flat along most directions. We further draw an essential connection between two seemingly unrelated properties of deep networks: their sensitivity to additive perturbations of the inputs, and the curvature of their decision boundary. The directions where the decision boundary is curved in fact characterize the directions to which the classifier is the most vulnerable. We finally leverage a fundamental asymmetry in the curvature of the decision boundary of deep nets, and propose a method to discriminate between original images, and images perturbed with small adversarial examples. We show the effectiveness of this purely geometric approach for detecting small adversarial perturbations in images, and for recovering the labels of perturbed images.

1. Introduction

While the geometry of classification regions and decision functions induced by traditional classifiers (such as linear and kernel SVM) is fairly well understood, these fundamental geometric properties are to a large extent unknown for state-of-the-art deep neural networks. Yet, to understand the recent success of deep neural networks and potentially address their weaknesses (such as their instability to perturbations [1]), an understanding of these geometric properties remains primordial. While many fundamental properties of deep networks have recently been studied, such as their *optimization landscape* in [2, 3], their *generalization* in [4, 5], and their *expressivity* in [6, 7], the geometric properties of the decision boundary and classification regions of deep networks have comparatively received little attention. The goal of this paper is to analyze these properties, and leverage them to improve the robustness of such classifiers to perturbations.

In this paper, we specifically view classification regions as topological spaces and decision boundaries as hypersurfaces, and we examine their geometric properties. We first study the classification regions induced by state-of-the-art deep networks, and provide empirical evidence suggesting that these classification regions are *connected*; that is, there exists a continuous path that remains in the region between any two points of the same label. Up to our knowledge, this represents the first instance where the connectivity of classification regions is empirically shown. Then, to study the complexity of the functions learned by the deep network, we analyze the curvature of their decision boundary. We empirically show that

- The decision boundary in the vicinity of natural images is flat in most directions, with only a very few directions that are significantly curved.
- We reveal the existence of a fundamental asymmetry in the decision boundary of deep networks, whereby the decision boundary (near natural images x) is biased towards negative curvatures.[1]
- Directions with significantly curved decision boundaries are shared between different datapoints.
- We demonstrate the existence of a relation between the sensitivity of a classifier to perturbations of the inputs, and these shared directions: a deep net is vulnerable to perturbations along these directions, and is insensitive to perturbations along the remaining directions.

We finally leverage the fundamental asymmetry of deep networks revealed in our analysis, and propose an algorithm

*The first two authors contributed equally to this work.
†University of California, Los Angeles, US
‡École polytechnique fédérale de Lausanne, Switzerland

[1]Throughout the paper, the sign of the curvature is chosen according to the normal vector, and the data point x, as illustrated in Fig. 8 (top).

to detect natural images from imperceptibly similar images with very small adversarial perturbations [1], as well as estimate the correct label of these perturbed samples. We show that our purely geometric characterization of (small) adversarial examples, which does not involve any re-training, is very effective to recognize perturbed samples.

Related works. In [8], the authors employ tools from Riemannian geometry to study the expressivity of random deep neural networks. In particular, the largest principal curvatures are shown to increase exponentially with the depth; the decision boundaries hence become more complex with depth. We provide in this paper a complementary analysis of the decision boundary, where the curvature of the decision boundary along *all* directions are analyzed (and not only in the direction of largest curvature). The authors of [9] show that the number of linear regions (in the input space) of deep networks grows exponentially with the number of layers. In [2, 3, 10, 11], the geometry of the optimization function in the *weight space* is studied; in particular, generalization of deep networks is shown to be intimately related to geometric properties of the optimization landscape (e.g., width of a minima). Closer to our work, in [12], the authors study the optimization landscape of deep neural networks, where the connectedness of solutions with low error is shown in the weight space. An algorithm is provided to assess the nature of this connection in the weight space; empirical evidence supports the existence of "easy" paths between trained models. We follow here a similar goal to that of [12], but are interested instead in the connectivity of deep networks in the *input space* (and not weight space). Finally, we note that graph-based techniques have been proposed in [13, 14] to analyze the classification regions of shallow neural networks; we rather focus here on the new generation of deep neural networks, which have shown remarkable performance.

2. Definitions and notations

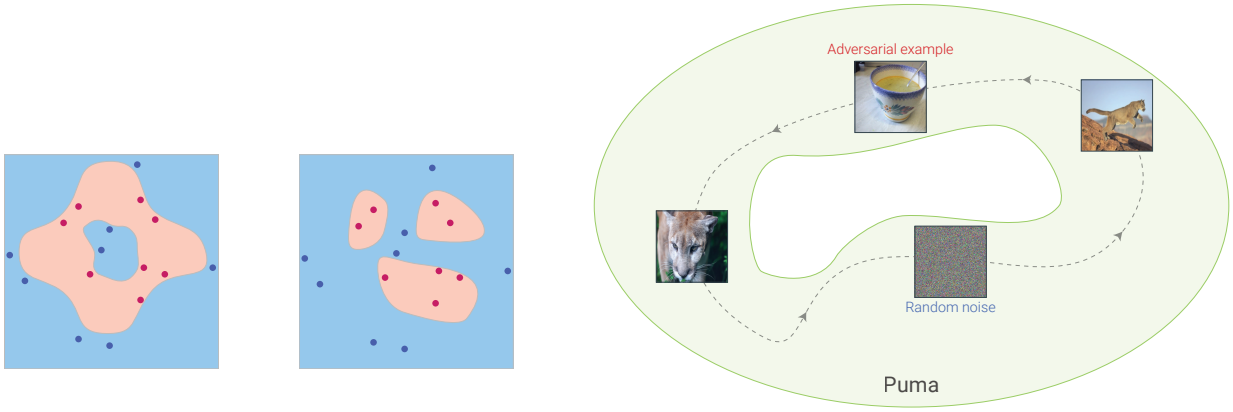
Let $f : \mathbb{R}^d \to \mathbb{R}^L$ denote a L class classifier. Given a datapoint $\boldsymbol{x}_0 \in \mathbb{R}^d$, the estimated label is obtained by $\hat{k}(\boldsymbol{x}_0) = \operatorname{argmax}_k f_k(\boldsymbol{x}_0)$, where $f_k(\boldsymbol{x})$ is the k^{th} component of $f(\boldsymbol{x})$ that corresponds to the k^{th} class. The classifier f partitions the space $\mathbb{R}^d$ into *classification regions* $\mathcal{R}_1, \ldots, \mathcal{R}_L$ of constant label. That is, for any $\boldsymbol{x} \in \mathcal{R}_i$, $\hat{k}(\boldsymbol{x}) = i$. For a neighboring class j, the pairwise decision boundary of the classifier (between these two classes i and j) is defined as the set $\mathscr{B} = \{ \boldsymbol{z} : F(\boldsymbol{z}) = 0\}$, where $F(\boldsymbol{z}) = f_i(\boldsymbol{z}) - f_j(\boldsymbol{z})$ (we omit dependence on i,j for simplicity). The decision boundary defines a hypersurface (of dimension $d-1$) in $\mathbb{R}^d$. Note that for any point on the decision boundary $\boldsymbol{z} \in \mathscr{B}$, the gradient $\nabla F(\boldsymbol{z})$ is orthogonal to the tangent space $\mathcal{T}_{\boldsymbol{z}}(\mathscr{B})$ of $\mathscr{B}$ at $\boldsymbol{z}$ (see Fig. 5 (a) for an illustration).

In this paper, we are interested in studying the decision boundary of a deep neural network in the vicinity of natural images. To do so, for a given point $\boldsymbol{x}$, we define the mapping $\boldsymbol{r}(\boldsymbol{x})$, given by $\boldsymbol{r}(\boldsymbol{x}) = \operatorname{argmin}_{\boldsymbol{r} \in \mathbb{R}^d} \|\boldsymbol{r}\|_2$ subject to $\hat{k}(\boldsymbol{x} + \boldsymbol{r}) \neq \hat{k}(\boldsymbol{x})$, which corresponds to the smallest perturbation required to misclassify image $\boldsymbol{x}$. Note that $\boldsymbol{r}(\boldsymbol{x})$ corresponds geometrically to the vector of minimal norm required to reach the decision boundary of the classifier, and is often dubbed an *adversarial perturbation* [1]. It should further be noted that, due to simple optimality conditions, $\boldsymbol{r}(\boldsymbol{x})$ is orthogonal to the decision boundary at $\boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$.

In the remainder of this paper, our goal is to analyze the geometric properties of classification regions $\{\mathcal{R}_i\}$ and decision boundaries $\mathscr{B}$ of deep networks. In particular, we study the connectedness of classification regions in Sec. 3, and the curvature of decision boundaries in Sec. 4, and draw a connection with the robustness of classifiers. We then use the developed geometric insights, and propose a method in Sec. 5 to detect artificially perturbed data points, and improve the robustness of classifiers.

3. Topology of classification regions

Do deep networks create shattered and disconnected classification regions, or on the contrary, one large connected region per label (see Fig. 1a)? While deep neural networks have an exponential number of linear regions (with respect to the number of layers) in the input space [9], it remains unclear whether deep nets create one connected region per class, or shatters a classification region around a large number of small connected sets. In the following, we treat the regions $\mathcal{R}_i$ as topological spaces, and study their path connectedness. We formally cast the problem of connectivity of classification regions as follows: given any two data points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{R}_i$, does a continuous curve $\gamma : [0,1] \to \mathcal{R}_i$ exist, such that $\gamma(0) = \boldsymbol{x}_1, \gamma(1) = \boldsymbol{x}_2$? The problem is complex to address theoretically; we therefore propose a heuristic method to study this question. To assess the connectivity of regions, we propose a path finding algorithm between two points belonging to the same classification region. That is, given two points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$, our proposed approach attempts to construct a piecewise linear path $\mathcal{P}$ that remains in the classification region. The path $\mathcal{P}$ is represented as a finite set of anchor points $(\boldsymbol{p}_0 = \boldsymbol{x}_1, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_n, \boldsymbol{p}_{n+1} = \boldsymbol{x}_2)$, where a convex path is taken between two consecutive points. To find the path (i.e., the anchor points), the algorithm first attempts to take a convex path between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$; when the path is not entirely included in the classification region, the path is modified by projecting the midpoint $\boldsymbol{p} = (\boldsymbol{x}_1 + \boldsymbol{x}_2)/2$ onto the target classification region. The same procedure is applied recursively on the two segments of the path $(\boldsymbol{x}_1, \boldsymbol{p})$ and $(\boldsymbol{x}_2, \boldsymbol{p})$ till the whole path is entirely in the region. The algorithm is summarized in Algorithm 1. In practice, the validity of a path $\mathcal{P}$ is checked empirically through a fine sampling of the convex combinations of the consecutive anchor points. Specifically, we set in practice the distance



(a) (b)

Figure 1: **(a)** Disconnected versus connected yet complex classification regions. **(b)** All four images are classified as puma. There exists a path between two images classified with the same label.

between sampled points to four orders of magnitude smaller than the distance between the original two images.

Algorithm 1 Finding a path between two data points.

function FINDPATH($\boldsymbol{x}_1$,$\boldsymbol{x}_2$)
  // input: Datapoints $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$.
  // output: Path $\mathcal{P}$ represented by anchor points.
  $\boldsymbol{x}_\text{m} \leftarrow (\boldsymbol{x}_1 + \boldsymbol{x}_2)/2$
  if $\hat{k}(\boldsymbol{x}_\text{m}) \neq \hat{k}(\boldsymbol{x}_1)$ **then**
    $\boldsymbol{r} \leftarrow \text{argmin}_{\boldsymbol{r}} \|\boldsymbol{r}\|_2$ s.t. $\hat{k}(\boldsymbol{x}_\text{m} + \boldsymbol{r}) = \hat{k}(\boldsymbol{x}_1)$
    $\boldsymbol{x}_\text{m} \leftarrow \boldsymbol{x}_\text{m} + \boldsymbol{r}$
  end if
  $\mathcal{P} \leftarrow (\boldsymbol{x}_1, \boldsymbol{x}_\text{m}, \boldsymbol{x}_2)$
  // Check the validity of the path by sampling in the convex combinations of consecutive anchor points
  if $\mathcal{P}$ is a valid path **then**
    return $\mathcal{P}$
  end if
  $\mathcal{P}_1 \leftarrow$FINDPATH($\boldsymbol{x}_1$,$\boldsymbol{x}_\text{m}$)
  $\mathcal{P}_2 \leftarrow$FINDPATH($\boldsymbol{x}_\text{m}$,$\boldsymbol{x}_2$)
  $\mathcal{P} \leftarrow$ **concat**$(\mathcal{P}_1, \mathcal{P}_2)$
  return $\mathcal{P}$
end function

The proposed approach is used to assess the connectivity of the CaffeNet architecture[2] [15] on the ImageNet classification. To do so, we examine the existence of paths between

1. Two randomly sampled points from the validation set with the same estimated label,

2. A randomly sampled point from the validation set, and an adversarially perturbed image [1]. That is, we consider $\boldsymbol{x}_1$ to be an image from the validation set, and $\boldsymbol{x}_2 = \tilde{\boldsymbol{x}}_2 + \boldsymbol{r}$, where $\tilde{\boldsymbol{x}}_2$ corresponds to an image classified differently than $\boldsymbol{x}_1$. $\boldsymbol{x}_2$ is however classified similarly as $\boldsymbol{x}_1$, due to the targeted perturbation $\boldsymbol{r}$.

3. A randomly sampled point from the validation set, and a perturbed random point. This is similar to scenario 2, but $\tilde{\boldsymbol{x}}_2$ is set to be a random image (i.e., an image sampled uniformly at random from the sphere $\rho\mathbb{S}^{d-1}$, where ρ denotes the typical norm of images).

Note that in all scenarios, we check the connectivity between two images that have the same *estimated* label by the classifier (but not necessarily the same true label). In particular, in scenario 2 and 3, $\boldsymbol{x}_2$ does not even visually correspond to an image of the same class as $\boldsymbol{x}_1$ (but has the same estimated label as $\boldsymbol{x}_1$ by the classifier). With this setting, the geometric properties of the classification regions are analyzed independently of the visual properties of the images. These scenarios are illustrated in Fig. 1b. For each scenario, 1,000 pairs of points are considered, and the approach described above is used to find the connecting path. Our results can be stated as follows:

1. In all three scenarios, evidence hints that **a continuous path included in the region always exists** between points sampled from the same classification region.[3]

2. Moreover, the continuous path connecting the points is **approximately a straight path**.

[2]We tested other architectures (GoogLeNet, VGG-19, ResNet-152), and the results were similar to CaffeNet. We therefore report only results on CaffeNet in this section.

[3]While not providing a formal certificate that the *continuous* path is entirely included in the classification region (as boundary regions can meander between neighbouring points in the continuum), we believe the sampling procedure used to verify the connectedness of a region is conservative, especially in the presence of regularizers that bound the curvature of the decision boundary (e.g., weight decay).

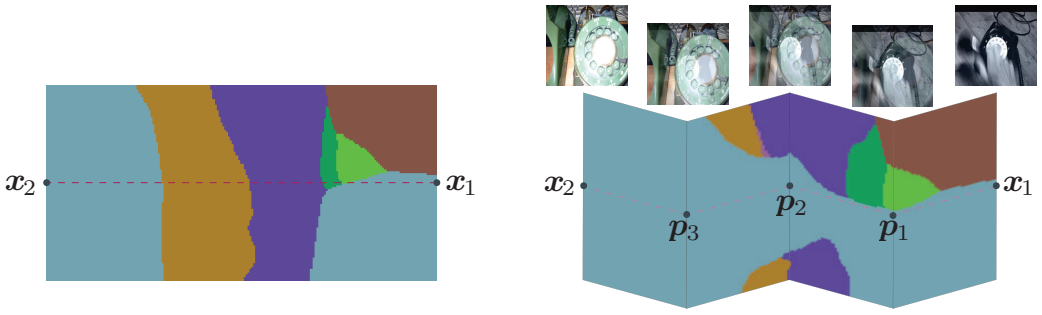


Figure 2: Classification regions (shown with different colors), and illustration of different paths between images $\boldsymbol{x}_1$, $\boldsymbol{x}_2$. **Left:** Example where the convex path between two datapoints is not entirely included in the classification region (note that the linear path traverses 4 other regions, each depicted with a different color). The image is the cross-section spanned by $\boldsymbol{r}(\boldsymbol{x}_1)$ (adversarial perturbation of $\boldsymbol{x}_1$) and $\boldsymbol{x}_1 - \boldsymbol{x}_2$. Images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are natural images from the ILSVRC 12 validation set, and the CaffeNet deep network is used. **Right:** Illustration of the classification regions along a nonconvex path; observe that the path entirely remains in the same classification region. The illustration is obtained by stitching cross-sections spanned by $\boldsymbol{r}(\boldsymbol{x}_1)$ (vertical axis) and $\boldsymbol{p}_i - \boldsymbol{p}_{i+1}$ (two consecutive anchor points in the path $\mathcal{P}$) (horizontal axis). It is shown broken to emphasize that the horizontal axes are different. Angles between stitched cross-sections are purely illustrative. On top of each anchor point (as well as $\boldsymbol{x}_1, \boldsymbol{x}_2$), image on the path is visualized.

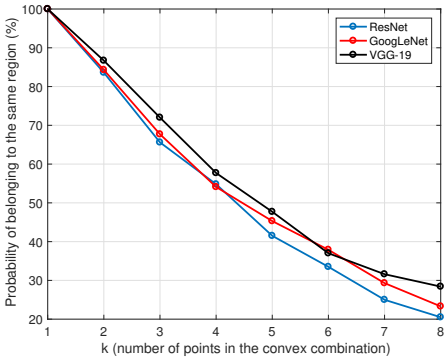


Figure 3: Empirical probability (y axis) that a convex combination of k samples (x axis) from the same classification region stays in the region, for networks trained on ImageNet. Samples are randomly chosen from the validation set.

The first result suggests that the classification regions created by deep neural networks are *connected* in $\mathbb{R}^d$: deep nets create single large regions containing all points of the same label. This goes against common belief, whereby classification regions are thought to be disconnected, and to concentrate around data points. To further understand the paths found by Algorithm 1, we show in Fig. 2 (right) an illustrative example of a path connecting two data points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ from the validation set (i.e., scenario 1).[4] While this nonstraight path connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is entirely included in the classification region, observe that the convex path illustrated in Fig. 2 (left) is *not* a valid path. In practice, the paths found by Algorithm 1 have, in average, 10 anchor points for the three scenarios.

Our second result provides an answer to the next natural question: how do the paths connecting data points (and staying inside a classification region) "look like"? Specifically, we show that two points in a classification region can be connected by an *approximately straight path*. To quantify how the paths of Algorithm 1 deviate from the straight path, we report the quantity

$$D(\boldsymbol{p}) = \frac{\sum_{i=0}^{n} \|\boldsymbol{p}_i - \boldsymbol{p}_{i+1}\|_2}{\|\boldsymbol{p}_0 - \boldsymbol{p}_{n+1}\|_2}.$$

Values of $D(\boldsymbol{p}) \approx 1$ indicate that the path $\boldsymbol{p}$ is close to a straight line. For the three scenarios, we have an average deviation $D(\boldsymbol{p}) = 1 + 10^{-4}$, which indicates that the paths found in Algorithm 1 are slight deviations from the straight path. With this very small deviation from the straight path, it is possible to connect arbitrary points in classification regions.[5] This observation is intriguingly similar to that of [12], where it is shown that solutions (in the *weight space*) achieving small error can be connected with an approximately straight path. This suggests that the data space and weight space have common properties; the specifics of this duality between these spaces is outside the scope of this paper and will be subject of future work.

Despite the existence of approximately straight paths connecting any pairs of points in the classification regions, it is important to note that classification regions are *not* convex bodies. In fact, Fig. 3 illustrates the estimated

[4] Illustrations for the other scenarios can be found in the supplementary material.

[5] Straight paths might not be entirely inside the classification region; tiny deviations are crucial to guarantee that complete paths are inside the region.

probability that random convex combinations of k images $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \in \mathcal{R}_i$ belong to $\mathcal{R}_i$. Observe that for the different tested networks, random convex combinations of two images (i.e., case where $k = 2$) belong with probability ≈ 0.8 to the classification region. However, for larger k, this probability gets much smaller, which implies that classification regions are *not* convex bodies in $\mathbb{R}^d$. These results suggest that the classification regions of deep networks extrapolate their classification regions in an approximately flat way between different images (i.e., there exist near-convex paths between pairs of images of the same class), but that the classification region is *not* a convex body. In a simplistic two-dimensional world, a classification region satisfying these two constraints would look like Fig. 4.

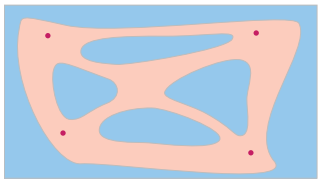


Figure 4: Schematic illustration in 2d of the properties of a classification region of a deep net. A classification region is connected by an almost convex path, despite classification regions being *non-convex* sets.

In the next section, we explore the *complexity* of the boundaries of these classification regions learned by deep networks, through their curvature property.

4. Curvature of the decision boundaries

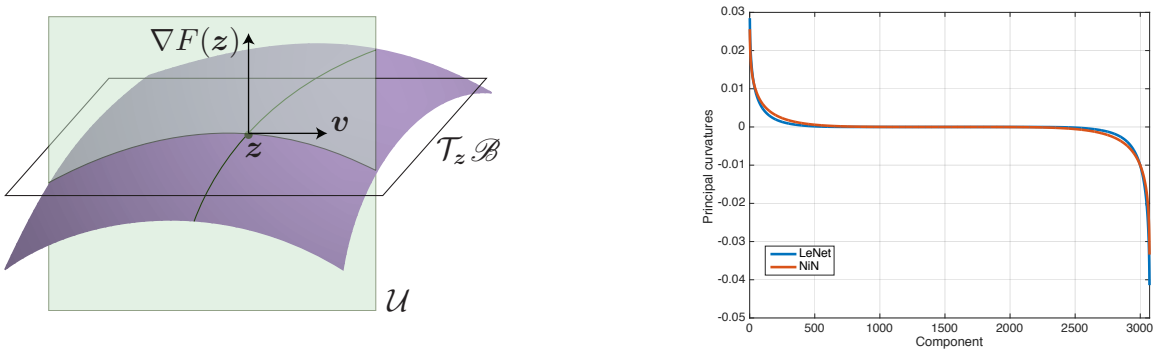
We start with basic definitions of curvature. The *normal* curvature $\kappa(\boldsymbol{z}, \boldsymbol{v})$ along a tangent direction $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{z}}(\mathscr{B})$ is defined as the curvature of the planar curve resulting from the cross-section of $\mathscr{B}$ along the two-dimensional normal plane spanning $(\nabla F(\boldsymbol{z}), \boldsymbol{v})$ (see Fig. 5a for details). The curvature along a tangent vector $\boldsymbol{v}$ can be expressed in terms of the Hessian matrix H_F of F [16]:

$$\kappa(\boldsymbol{z}, \boldsymbol{v}) = \frac{\boldsymbol{v}^T H_F \boldsymbol{v}}{\|\boldsymbol{v}\|_2^2 \|\nabla F(\boldsymbol{z})\|_2}. \quad (1)$$

Principal directions correspond to the orthogonal directions in the tangent space maximizing the curvature $\kappa(\boldsymbol{z}, \boldsymbol{v})$. Specifically, the l-th principal direction $\boldsymbol{v}_l$ (and the corresponding principal curvature κ_l) is obtained by maximizing $\kappa(\boldsymbol{z}, \boldsymbol{v})$ with the constraint $\boldsymbol{v}_l \perp \boldsymbol{v}_1 \ldots \boldsymbol{v}_{l-1}$. Alternatively, the principal curvatures correspond to the nonzero eigenvalues of the matrix $\frac{1}{\|\nabla F(\boldsymbol{z})\|_2} P H_F P$, where P is the projection operator on the tangent space; i.e., $P = I - \nabla F(\boldsymbol{z}) \nabla F(\boldsymbol{z})^T$.

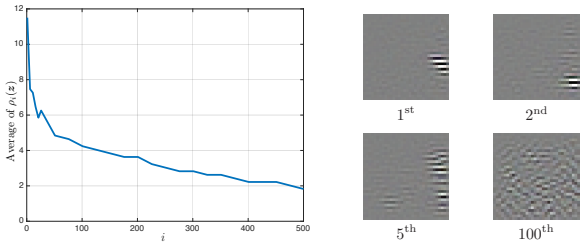
We now analyze the curvature of the decision boundary of deep neural networks in the vicinity of natural images. We consider the LeNet and NiN [17] architectures trained on the CIFAR-10 task, and show the principal curvatures of the decision boundary, in the vicinity of 1,000 randomly chosen images from the validation set. Specifically, for a given image $\boldsymbol{x}$, the perturbed sample $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ corresponds to the closest point to $\boldsymbol{x}$ on the decision boundary. We then compute the principal curvatures at point $\boldsymbol{z}$ with Eq. 1. The average profile of the principal curvatures (over $1,000$ data points) is illustrated in Fig. 5b. Observe that, for both networks, the large majority of principal curvatures are approximately zero: along these principal directions, the decision boundary is almost flat. Along the remaining principal directions, the decision boundary has (non-negligible) positive or negative curvature. Interestingly, the principal curvature profile is asymmetric towards *negatively curved* directions. We have consistently observed this asymmetry in different settings: different datapoints, different networks (e.g., LeNet and NiN), and even different datasets (CIFAR-10 and ImageNet, see Section 5 for more details), which suggests that this property (negatively curved decision boundary) is not an artifact of the experimental setting. In the next section, we leverage this characteristic asymmetry of the decision boundaries of deep neural networks (in the vicinity of natural images) to detect adversarial examples from clean examples.

While the above local analysis shows the existence of few directions along which the decision boundary is curved, we now examine whether these directions are *shared* across different datapoints, and relate these directions with the robustness of deep nets. To estimate the *shared* common curved directions, we compute the largest *principal directions* for a randomly chosen batch of 100 training samples and merge these directions into a matrix M. We then estimate the common curved directions as the m largest singular vectors of M that we denote by $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$. To assess whether the decision boundary is curved in such directions, we then evaluate the curvature of the decision boundary in such directions for points $\boldsymbol{z}$ in the vicinity of *unseen* samples from the *validation* set. That is, for $\boldsymbol{x}$ in the validation set, and $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$, we compute $\rho_i(\boldsymbol{z}) = \frac{|\boldsymbol{u}_i^T P H_F P \boldsymbol{u}_i|}{\mathbb{E}_{\boldsymbol{v} \sim \mathbb{S}^{d-1}}(|\boldsymbol{v}^T P H_F P \boldsymbol{v}|)}$, which measures how relatively curved is the decision boundary in direction $\boldsymbol{u}_i$, compared to random directions sampled from the unit sphere in $\mathbb{R}^d$. When $\rho_i(\boldsymbol{z}) \gg 1$, this indicates that $\boldsymbol{u}_i$ constitutes a direction that significantly curves the decision boundary at $\boldsymbol{z}$. Fig. 6a shows the average of $\rho_i(\boldsymbol{z})$ over 1,000 points $\boldsymbol{z}$ on the decision boundary in the vicinity of *unseen* natural images, for the LeNet architecture on CIFAR-10. Note that the directions $\boldsymbol{u}_i$ (with i sufficiently small) lead to universally curved directions across *unseen* points. That is, the decision boundary is highly curved along such data-independent directions. Note that, despite using a relatively small number of samples (i.e., 100 samples) to compute the shared directions, these generalize well to unseen points. We illustrate in Fig. 6b these directions $\boldsymbol{u}_i$,



(a) (b)

Figure 5: **(a)** Normal section $\mathcal{U}$ of the decision boundary, along the plane spanned by the normal vector $\nabla F(\boldsymbol{z})$ and $\boldsymbol{v}$. **(b)** Principal curvatures for NiN and LeNet, computed at a point $\boldsymbol{z}$ on the decision boundary in the vicinity of a natural image.



(a) (b)

Figure 6: **(a)** Average of $\rho_i(\boldsymbol{z})$ as a function of i for different points $\boldsymbol{z}$ in the vicinity of natural images. **(b)** Basis of $\mathcal{S}$.

along which decision boundary is universally curved in the vicinity of natural images; interestingly, the first principal directions (i.e., directions along which the decision boundary is highly curved) are very localized Gabor-like filters. Through discriminative training, the deep neural network has implicitly learned to curve the decision boundary along such directions, and preserve a flat decision boundary along the orthogonal subspace.

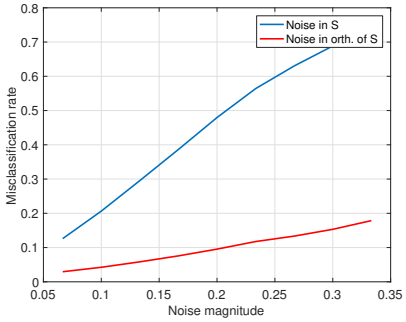


Figure 7: Misclassification rate (% of images that change labels) on the noisy validation set, w.r.t. the noise magnitude (ℓ_2 norm of noise divided by the typical norm of images).

Interestingly, the data-independent directions $\boldsymbol{u}_i$ (where

Type of perturbation $\boldsymbol{v}$	LeNet	NiN
Random	0.25	0.25
Adversarial	0.64	0.60
$\boldsymbol{x}_2 - \boldsymbol{x}_1$	0.10	0.09
$\nabla \boldsymbol{x}$	0.22	0.24

Table 1: Norm of projected perturbation on $\mathcal{S}$, normalized by norm of perturbation: $\frac{\|P_{\mathcal{S}}\boldsymbol{v}\|_2}{\|\boldsymbol{v}\|_2}$, with $\boldsymbol{v}$ the perturbation. Larger values (i.e., closer to 1) indicate that the perturbation has a larger component on subspace $\mathcal{S}$.

the decision boundary is highly curved) are also tightly connected with the *sensitivity* of the classifier to perturbations. To elucidate this relation, we construct a subspace $\mathcal{S} = \text{span}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{200})$ containing the first 200 shared curved directions. Then, we show in Fig. 7 the accuracy of the CIFAR-10 LeNet model on a noisy validation set, where the noise either belongs to $\mathcal{S}$, or to $\mathcal{S}^\perp$ (i.e., orthogonal of $\mathcal{S}$). It can be observed that the deep network is much more robust to noise orthogonal to $\mathcal{S}$, than to noise in $\mathcal{S}$. Hence, $\mathcal{S}$ also represents the subspace of perturbations to which the classifier is highly vulnerable, while the classifier has learned to be invariant to perturbations in $\mathcal{S}^\perp$. To support this claim, we report in Table 1, the norm of the projection of adversarial perturbations (computed using the method in [18]) on the subspace $\mathcal{S}$, and compare it to that of the projection of random noise onto $\mathcal{S}$. Note that for both networks under study, adversarial perturbations project well onto the subspace $\mathcal{S}$ comparatively to random perturbations, which have a significant component in $\mathcal{S}^\perp$. In contrast, the perturbations obtained by taking the difference of two random images belong overwhelmingly to $\mathcal{S}^\perp$, which agrees with the observation drawn in Sec. 3 whereby straight paths are likely to belong to the classification region. Finally, note that the gradient of the image $\nabla \boldsymbol{x}$ also does not have an important component in $\mathcal{S}$, as the robustness to such directions is fundamental to achieve invariance to small geometric

deformations.[6]

The importance of the shared directions $\{\boldsymbol{u}_i\}$, where the decision boundary is curved, hence goes beyond our curvature analysis, and capture the modes of sensitivity learned by the deep network.

5. Exploiting the asymmetry to detect perturbed samples

State-of-the-art image classifiers are highly vulnerable to imperceptible adversarial perturbations [1, 19]. That is, adding a well-sought small perturbation to an image causes state-of-the-art classifiers to misclassify. In this section, we leverage the asymmetry of the principal curvatures (illustrated in Fig. 5b), and propose a method to distinguish between original images, and images perturbed with small adversarial perturbations, as well as improve the robustness of classifiers. For an element $\boldsymbol{z}$ on the decision boundary, denote by $\overline{\kappa}(\boldsymbol{z}) = \frac{1}{d-1}\sum_{i=1}^{d-1}\kappa_i(\boldsymbol{z})$ the average of the principal curvatures. For points $\boldsymbol{z}$ sampled in the vicinity of natural images, the profile of the principal curvature is asymmetric (see Fig. 5b), leading to a negative average curvature; i.e., $\overline{\kappa}(\boldsymbol{z}) < 0$. In contrast, if $\boldsymbol{x}$ is now perturbed with an adversarial example (that is, we observe $\boldsymbol{x}_{\text{pert}} = \boldsymbol{x} + \boldsymbol{r}(\boldsymbol{x})$ instead of $\boldsymbol{x}$), the average curvature at the vicinity of $\boldsymbol{x}_{\text{pert}}$ is instead *positive*, as schematically illustrated in Fig. 8. Table 2 supports this observation empirically with adversarial examples computed with the method in [18]. Note that for both networks, the asymmetry of the principal curvatures allows to distinguish very accurately original samples from perturbed samples using the *sign* of the curvature.[7] Based on this simple idea, we now derive an algorithm for detecting adversarial perturbations.

Since the computation of all the principal curvatures is intractable for large-scale datasets, we derive a tractable estimate of the average curvature. Note that the average curvature $\overline{\kappa}$ can be equivalently written as $\mathop{\mathbb{E}}_{v\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right)$, where $G(\boldsymbol{z}) = \|\nabla F(\boldsymbol{z})\|_2^{-1}(I - \nabla F(\boldsymbol{z})\nabla F(\boldsymbol{z})^T)H_F(\boldsymbol{z})(I - \nabla F(\boldsymbol{z})\nabla F(\boldsymbol{z})^T)$. In fact, we have

$$\mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right) = \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T\left(\sum_{i=1}^{d-1}\kappa_i\boldsymbol{v}_i\boldsymbol{v}_i^T\right)\boldsymbol{v}\right)$$
$$= \frac{1}{d-1}\sum_{i=1}^{d-1}\kappa_i,$$

where $\boldsymbol{v}_i$ denote the principal directions. It therefore follows that the average curvature $\overline{\kappa}$ can be efficiently estimated using a sample estimate of $\mathop{\mathbb{E}}_{v\sim\mathbb{S}^{d-1}}\left(\boldsymbol{v}^T G(\boldsymbol{z})\boldsymbol{v}\right)$ (and without requiring the full eigen-decomposition of G). To further make the approach of detecting perturbed samples more practical, we approximate $G(\boldsymbol{z})$ (for $\boldsymbol{z}$ on the decision boundary) with $G(\boldsymbol{x})$, assuming that $\boldsymbol{x}$ is sufficiently close to the decision boundary.[8] This approximation avoids the computation of the closest point on the decision boundary $\boldsymbol{z}$, for each $\boldsymbol{x}$.

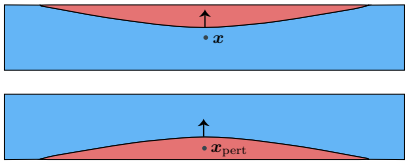


Figure 8: Schematic representation of normal sections in the vicinity of a natural image (top), and perturbed image (bottom). The normal vector to the decision boundary is indicated with an arrow.

	LeNet	NiN
$\%\ \overline{\kappa} > 0$ for original samples	97%	94%
$\%\ \overline{\kappa} < 0$ for perturbed samples	96%	93%

Table 2: Percentage of points on the boundary with positive (resp. negative) average curvature, when sampled in the vicinity of natural images (resp. perturbed images). CIFAR-10 dataset is used; results are computed on the test set.

We provide the details in Algorithm 2. Note that, in order to extend this approach to multiclass classification, an empirical average is taken over the decision boundaries with respect to all other classes. Moreover, while we have used a threshold of 0 to detect adversarial examples from original data in the above explanation, a threshold parameter t is used in practice (which controls the true positive vs. false positive tradeoff). Finally, it should be noted that in addition to detecting whether an image is perturbed, the algorithm also provides an estimate of the original label when a perturbed sample is detected (the class leading to the highest positive curvature is returned).

We now test the proposed approach on different networks trained on the ImageNet dataset [20], with adversarial examples computed using the approach in [18]. The latter

[6] In fact, a first order Taylor approximation of a translated image $\boldsymbol{x}(\cdot + \tau_1, \cdot + \tau_2) \approx \boldsymbol{x} + \tau_1\nabla_x\boldsymbol{x} + \tau_2\nabla_y\boldsymbol{x}$. To achieve robustness to translations, a deep neural network hence needs to be locally invariant to perturbations along the gradient directions.

[7] This idea might first appear counter-intuitive: if curvature is negative at the vicinity of data points, then the curvature has to be positive for data points lying on the other side of the boundary! However, natural data points are very "sparse" in $\mathbb{R}^d$; hence, two natural images never lie exactly opposite to each other (from the two sides of the boundary). Instead, different data points lie at the vicinity of very distinct parts of the decision boundary, which makes it possible to have negatively curved decision boundary at the vicinity of all data points. See Fig. 1a (left) for an illustration of such a decision boundary, with negative curvature at the vicinity of all points.

[8] The matrix G is never computed in practice, since only matrix vector multiplications of G are needed.

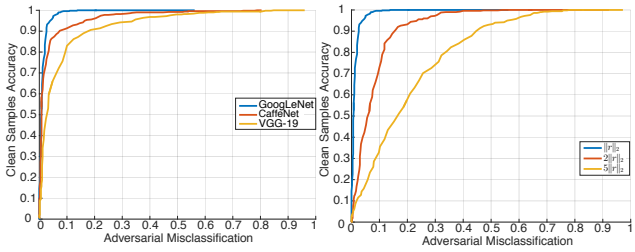


Figure 9: True positives (i.e., detection accuracy on *clean samples*) vs. False positives (i.e., detection error on *perturbed samples*) on the ImageNet classification task. **Left:** Results reported for GoogLeNet, CaffeNet and VGG-19 architectures, with perturbations computed using the approach in [18]. **Right:** Results reported for GoogLeNet, where perturbations are scaled by a constant factor $\alpha = 1, 2, 5$.

Algorithm 2 Detecting and denoising perturbed samples.

1: **input:** classifier f, sample $\boldsymbol{x}$, threshold t.
2: **output:** boolean *perturbed*, recovered label *label*.
3: Set $F_i \leftarrow f_i - f_{\hat{k}}$ for $i \in [L]$.
4: Draw iid samples $\boldsymbol{v}_1, \dots, \boldsymbol{v}_T$ from the uniform distribution on $\mathbb{S}^{d-1}$.
5: Compute $\rho \leftarrow \frac{1}{LT} \sum_{\substack{i=1 \\ i \neq \hat{k}(\boldsymbol{x})}}^{L} \sum_{j=1}^{T} \boldsymbol{v}_j^T G_{F_i} \boldsymbol{v}_j$, where G_{F_i} denotes the Hessian of F_i projected on the tangent space; i.e., $G_{F_i}(\boldsymbol{x}) = \|\nabla F(\boldsymbol{x})\|_2^{-1} (I - \nabla F(\boldsymbol{x}) \nabla F(\boldsymbol{x})^T) H_{F_i}(\boldsymbol{x}) (I - \nabla F(\boldsymbol{x}) \nabla F(\boldsymbol{x})^T)$.
6: **if** $\rho < t$ **then** *perturbed* $\leftarrow$ *false*.
7: **else** *perturbed* $\leftarrow$ *false* and *label* $\leftarrow \underset{\substack{i \in \{1,\dots,L\} \\ i \neq \hat{k}(\boldsymbol{x})}}{\operatorname{argmax}} \sum_{j=1}^{T} \boldsymbol{v}_j^T G_{F_i} \boldsymbol{v}_j$.
8: **end if**

approach is used as it provides small and difficult to detect adversarial examples, as mentioned in [21, 22]. Fig. 9 (left) shows the accuracy of the detection of Algorithm 2 on *original* images with respect to the detection error on *perturbed* images, for varying values of the threshold t. For the three networks under test, the approach achieves very accurate detection of adversarial examples (e.g., more than 95% accuracy on GoogLeNet with an optimal threshold). Note first that the success of this strategy *confirms the asymmetry* of the curvature of the decision boundary on the more complex setting of large-scale networks trained on ImageNet. Moreover, this simple curvature-based detection strategy outperforms the detection approach recently proposed in [22]. In addition, unlike other approaches of detecting perturbed samples (or improving the robustness), our approach only uses the characteristic geometry of the decision boundary of deep neural networks (i.e., the curvature *asymmetry*), and does not involve any training/fine-tuning with perturbed samples, as commonly done.

The proposed approach not only distinguishes original from perturbed samples, but it also provides an estimate of the correct label, in the case a perturbed sample is detected. Algorithm 2 correctly recovers the labels of perturbed samples with an accuracy of 92%, 88% and 74% respectively for GoogLeNet, CaffeNet and VGG-19, with $t = 0$. This shows that the proposed approach can be effectively used to denoise the perturbed samples, in addition to their detection.

Finally, Fig. 9 (right) reports a similar graph to that of Fig. 9 (left) for the GoogLeNet architecture, where the perturbations are now multiplied by a factor $\alpha \geq 1$. Note that, as α increases, the detection accuracy of our method decreases, as it heavily relies on *local* geometric properties of the classifier (i.e., the curvature). Interestingly enough, [22, 21] report that the regime where perturbations are very small (like those produced by [18]) are the hardest to detect; we therefore foresee that this geometric approach will be used along with other detection approaches, as it provides very accurate detection in a distinct regime where traditional detectors do not work well (i.e., when the perturbations are very small).

6. Conclusion

We analyzed in this paper the geometry induced by deep neural network classifiers in the input space. Specifically, we provided empirical evidence showing that classification regions are connected. Next, to analyze the complexity of the functions learned by deep networks, we provided an empirical analysis of the curvature of the decision boundaries. We showed in particular that, in the vicinity of natural images, the decision boundaries learned by deep networks are flat along most (but not all) directions, and that some curved directions are *shared* across datapoints. We finally leveraged a fundamental observation on the *asymmetry* in the curvature of deep nets, and proposed an algorithm for detecting adversarially perturbed samples from original samples. This geometric approach was shown to be very effective, when the perturbations are sufficiently small, and that recovering the label was further possible using this algorithm. This shows that the study of the geometry of state-of-the-art deep networks is not only key from an analysis perspective, but it can also lead to classifiers with better properties.

Acknowledgments

S.M and P.F gratefully acknowledge the support of NVIDIA with the donation of the Titan X Pascal GPU used for this research. This work has been partly supported by the Hasler Foundation, Switzerland. A.F was supported by the SNSF under grant P2ELP2-168511. S.S. was supported by ONR N00014-17-1-2072 and ARO W911NF-15-1-0564.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations (ICLR)*, 2014.

[2] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2014.

[3] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 2933–2941, 2014.

[4] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[5] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," *arXiv preprint arXiv:1509.01240*, 2015.

[6] O. Delalleau and Y. Bengio, "Shallow vs. deep sum-product networks," in *Advances in Neural Information Processing Systems*, pp. 666–674, 2011.

[7] N. Cohen and A. Shashua, "Convolutional rectifier networks as generalized tensor decompositions," in *International Conference on Machine Learning (ICML)*, 2016.

[8] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, "Exponential expressivity in deep neural networks through transient chaos," in *Advances In Neural Information Processing Systems*, pp. 3360–3368, 2016.

[9] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Advances In Neural Information Processing Systems*, pp. 2924–2932, 2014.

[10] P. Chaudhari, A. Choromanska, S. Soatto, and Y. LeCun, "Entropy-sgd: Biasing gradient descent into wide valleys," in *International Conference on Learning Representations*, 2017.

[11] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio, "Sharp minima can generalize for deep nets," in *International Conference on Machine Learning (ICML)*, 2017.

[12] C. D. Freeman and J. Bruna, "Topology and geometry of half-rectified network optimization," *arXiv preprint arXiv:1611.01540*, 2016.

[13] O. Melnik and J. Pollack, "Using graphs to analyze high-dimensional classifiers," in *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, vol. 3, pp. 425–430, IEEE, 2000.

[14] M. Aupetit and T. Catz, "High-dimensional labeled data analysis with topology representing graphs," *Neurocomputing*, vol. 63, pp. 139–169, 2005.

[15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM International Conference on Multimedia (MM)*, pp. 675–678, 2014.

[16] J. M. Lee, *Manifolds and differential geometry*, vol. 107. American Mathematical Society Providence, 2009.

[17] M. Lin, Q. Chen, and S. Yan, "Network in network," in *International Conference on Learning Representations (ICLR)*, 2014.

[18] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 387–402, 2013.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[21] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," in *International Conference on Learning Representations (ICLR)*, 2017.

[22] J. Lu, T. Issaranon, and D. Forsyth, "Safetynet: Detecting and rejecting adversarial examples robustly," in *International Conference on Computer Vision (ICCV)*, 2017.