EMPIRICAL STUDY OF THE TOPOLOGY AND GEOMETRY OF DEEP NETWORKS

Fawzi, Moosavi-Dezfooli, Frossard, Soatto @ CVPR 2018

UC San Diego

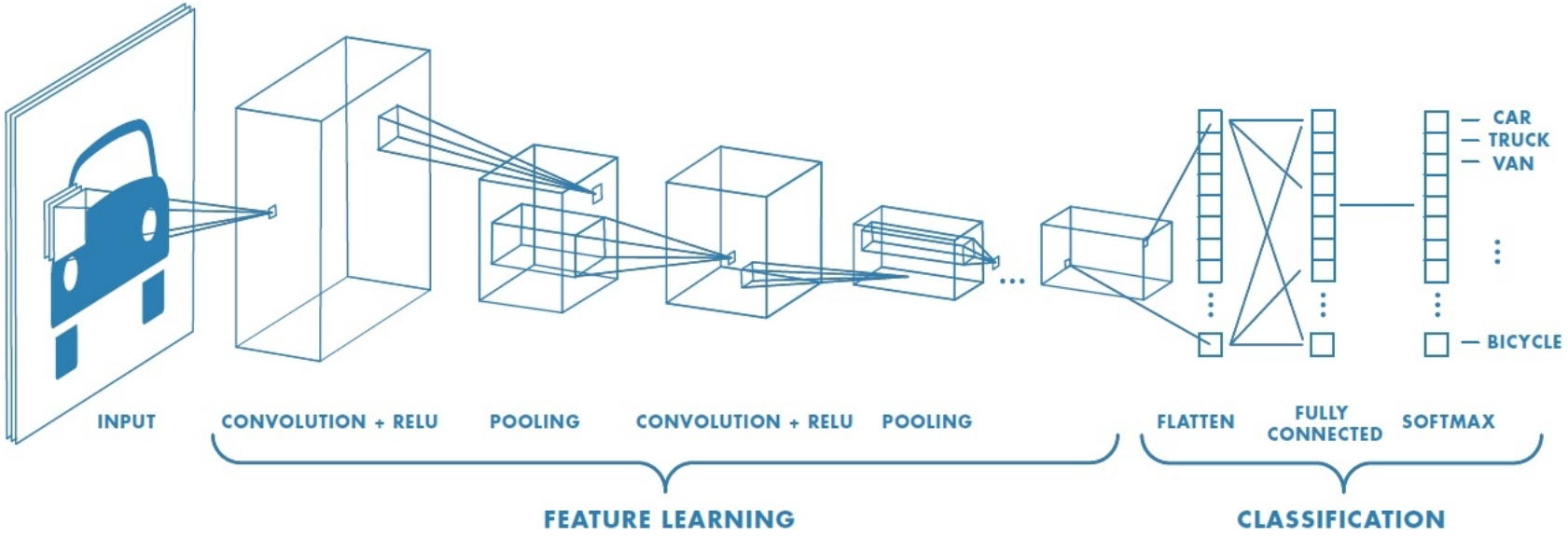
An Empirical Look into the Input-Space Geometry of Deep Image Classifiers

- A study in three parts
  1. Topology of classification regions
     - **a. Insight:** deep classifiers learn big, connected classification regions
  2. Curvature of decision boundaries
     - **a. Insight:** decision boundary flat in most directions near natural images
     - **b. Insight:** vulnerable to perturbation in the directions of high curvature
  3. Applying insights to adversarial example detection
     - a. Can detect adversarial examples with small additive perturbations
     - b. Can recover the classification labels of the images pre-perturbation

Setting the Scene

- L-class image classification network

$$f : \mathbb{R}^d \rightarrow \mathbb{R}^L$$



(Authors run experiments using **LeNet**, **NiN**, **GoogLeNet**, **CaffeNet**, **VGG-19,** and **ResNet-152** architectures)

Image source: MathWorks

Setting the Scene

- Send image $\mathbf{x}_i \in \mathbb{R}^d$ through classifier, get $f(\mathbf{x}_i) \in \mathbb{R}^L$
- Then $\hat{k}(\mathbf{x}_i) = \arg\max_k f_k(\mathbf{x}_i)$ is the estimated label
- There are L classification regions $\mathcal{R}_1, ..., \mathcal{R}_L$ in $\mathbb{R}^d$

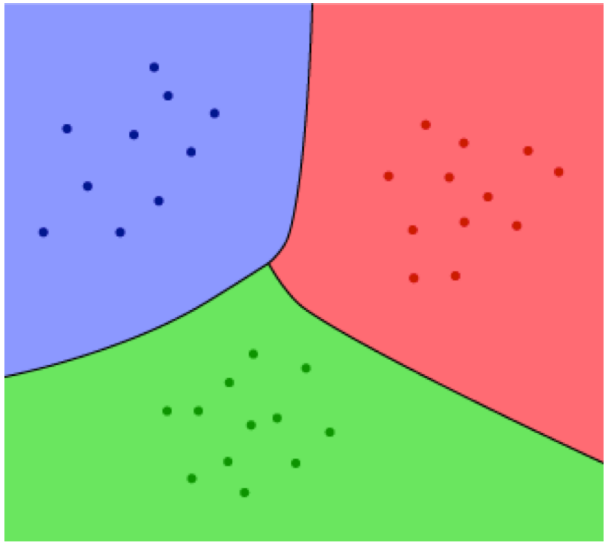
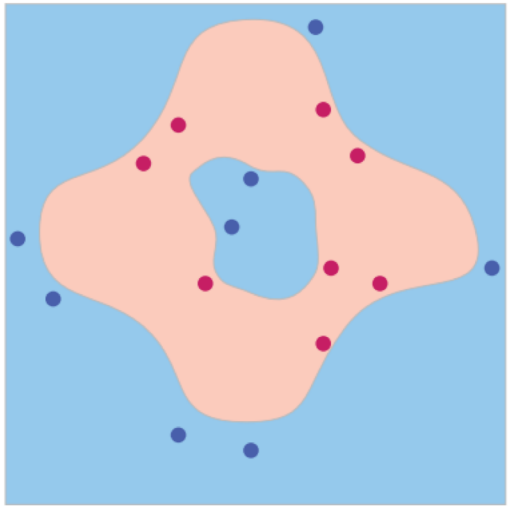
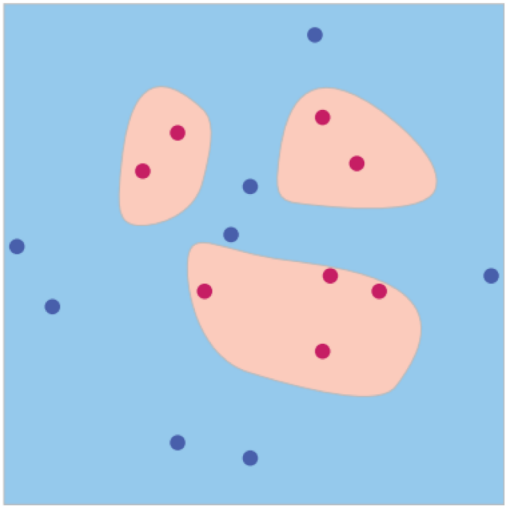


Image source: The Shape of Data

Topology of Classification Regions

- **Question:** are f's classification regions connected?





Characterizing topology of classification regions

Topology of Classification Regions

- **Question:** are f's classification regions connected?
  - **Let's write an algorithm to find out.**

Idea:
If we can take any two points in any one classification region $\mathcal{R}_i$ and connect them with a path that doesn't leave $\mathcal{R}_i$, then all $\mathcal{R}_i$s must be connected.

Characterizing topology of classification regions

Topology of Classification Regions

- **Question:** are f's classification regions connected?
  - **Let's write an algorithm to find out.**

Idea:
If we can take any two points in any one classification region $\mathcal{R}_i$ and connect them with a path that doesn't leave $\mathcal{R}_i$, then all $\mathcal{R}_i$s must be connected.

Validate empirically!

Characterizing topology of classification regions

Path Finding

- Given two data points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ (same estimated label $\hat{k}$), construct a piecewise-linear connecting path within $\mathcal{R}_{\hat{k}}$

$$\mathcal{P} = (\mathbf{p}_0 = \mathbf{x}_1, \mathbf{p}_1, \ldots, \mathbf{p}_n, \mathbf{p}_{n+1} = \mathbf{x}_2)$$

1. Try to go directly from x_1 to x_2.
2. If straight path stays within classification region,
   a. done.
3. Otherwise,
   a. project midpoint to classification region.
   b. repeat recursively on both resulting segments.

Characterizing topology of classification regions

Path Finding

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.
  $\boldsymbol{x}_\mathrm{m} \leftarrow (\boldsymbol{x}_1 + \boldsymbol{x}_2)/2$
  if $\hat{k}(\boldsymbol{x}_\mathrm{m}) \neq \hat{k}(\boldsymbol{x}_1)$ **then**
    $\boldsymbol{r} \leftarrow \mathrm{argmin}_{\boldsymbol{r}} \|\boldsymbol{r}\|_2$ s.t. $\hat{k}(\boldsymbol{x}_\mathrm{m} + \boldsymbol{r}) = \hat{k}(\boldsymbol{x}_1)$
    $\boldsymbol{x}_\mathrm{m} \leftarrow \boldsymbol{x}_\mathrm{m} + \boldsymbol{r}$
  end if
  $\mathcal{P} \leftarrow (\boldsymbol{x}_1, \boldsymbol{x}_\mathrm{m}, \boldsymbol{x}_2)$
  // 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($\boldsymbol{x}_1$,$\boldsymbol{x}_\mathrm{m}$)
  $\mathcal{P}_2 \leftarrow$FINDPATH($\boldsymbol{x}_\mathrm{m}$,$\boldsymbol{x}_2$)
  $\mathcal{P} \leftarrow$ **concat**$(\mathcal{P}_1, \mathcal{P}_2)$
  return $\mathcal{P}$
end function

Notes

- Algorithm doesn't seem to take no for an answer
- Check validity by sampling "lines" between anchor points
  - "Is *this* image in the right region?"
  - Distance between sampled data points is set to $10^{-4}\|\mathbf{x}_1 - \mathbf{x}_2\|$

Characterizing topology of classification regions

Path Finding

- Three investigations using CaffeNet classifier (*though others apparently yield similar results*) and ImageNet validation data
- The question is always
  - **"Is there a path between two points with the same estimated label?"**

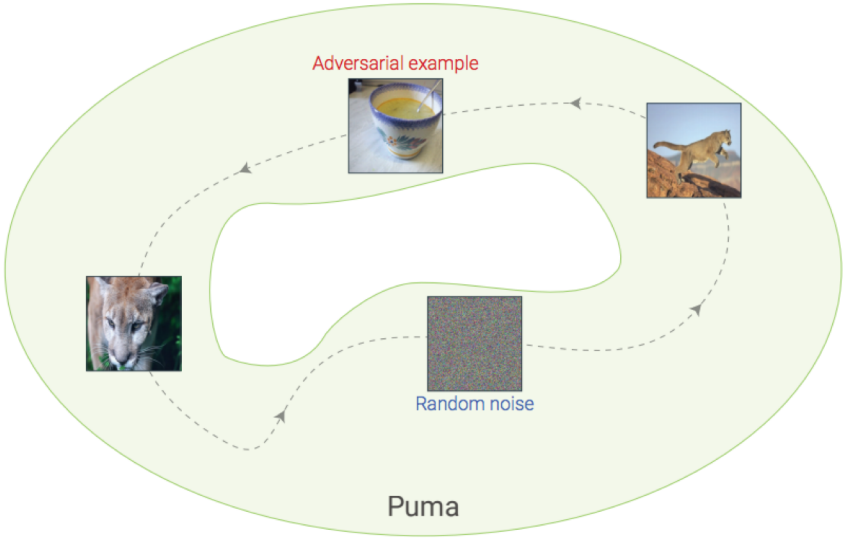
  where the two identically-labeled points are

  1. (a) random from data, (b) random from data
  2. (a) random from data, (b) adversarially perturbed from a different class
  3. (a) random from data, (b) adversarially perturbed from random noise

Characterizing topology of classification regions

Path Finding

- Note: analysis doesn't depend on visual characteristics
  - In scenarios 2 and 3, one of the images won't look like true label
- Might find a path between these images, all classified as *puma*:



Characterizing topology of classification regions

Path Finding

- For each scenario, attempt to find paths for 1000 pairs of points

Findings:

1. A connecting path within the region always exists.
2. This connecting path is approximately straight.

Obviously these are *empirical* conclusions.

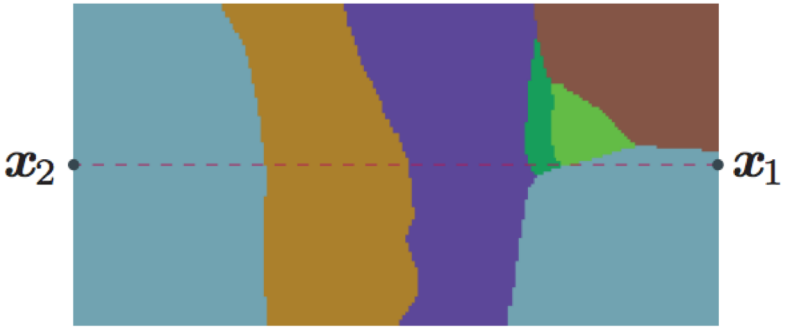
Characterizing topology of classification regions

A connecting path within the region always exists.

(on average 10 anchor points needed)

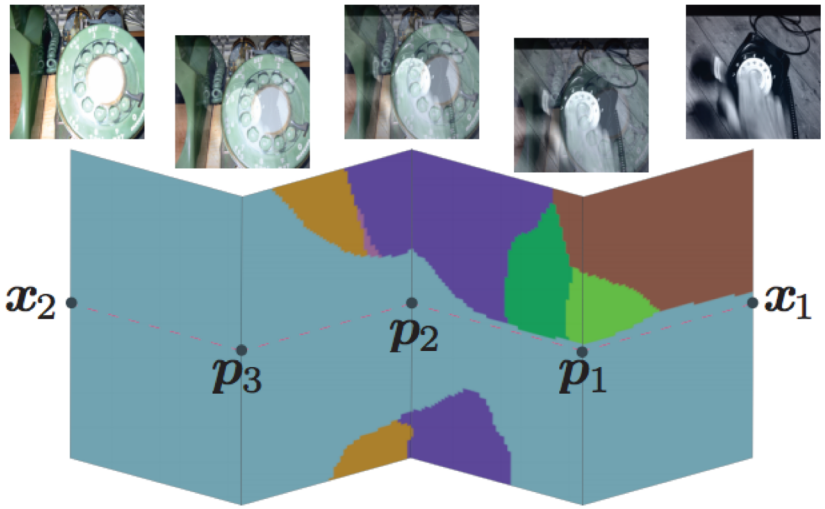
- **Conclusion:** input-space classification regions induced by deep networks are *connected*.

[Scenario 1
Path Finding Example**]**



A convex path which crosses classification regions. Visualization is the cross-section of $\mathbb{R}^d$ spanned by $\mathbf{r}(\mathbf{x}_1)$ (v. axis) and $\mathbf{x}_1 - \mathbf{x}_2$ (h. axis).

[$\mathbf{r}(\mathbf{x}_1)$ is smallest perturbation necessary to misclassify $\mathbf{x}_1$]
i.e. vector from $\mathbf{x}_1$ to decision boundary, orthogonal to boundary



A nonconvex path that sticks to a classification region. Visualization consists of 2D subspaces spanned by $\mathbf{r}(\mathbf{x}_1)$ (v. ax.) and $\mathbf{p}_i - \mathbf{p}_{i+1}$ (h. ax.).

Characterizing topology of classification regions

Straight, you say?

- Calculate path length relative to straight path length

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

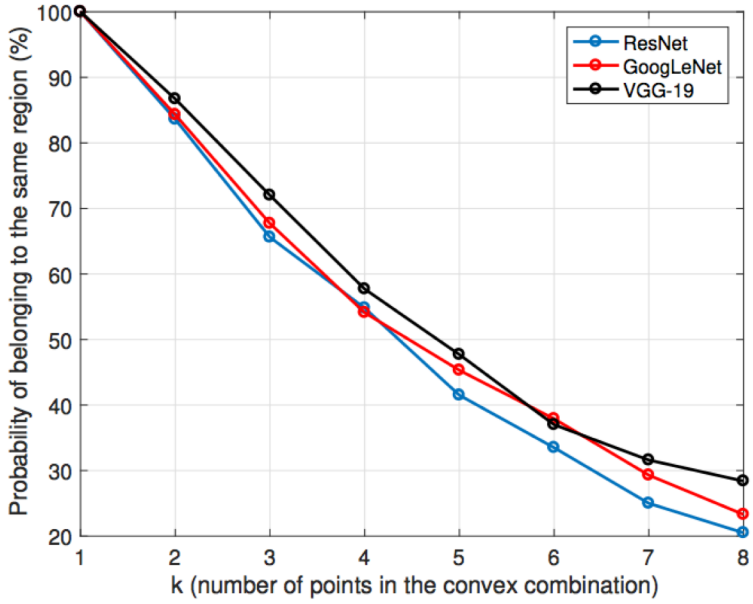
- Across all three scenarios, average length relative to straight path is $1 + 10^{-4}$, meaning paths are close to straight
- Suggests that classification regions are set up s.t. arbitrary points within them can be connected with almost-straight paths

→ decision boundaries not too complex?

Characterizing topology of classification regions

Convex classification regions?

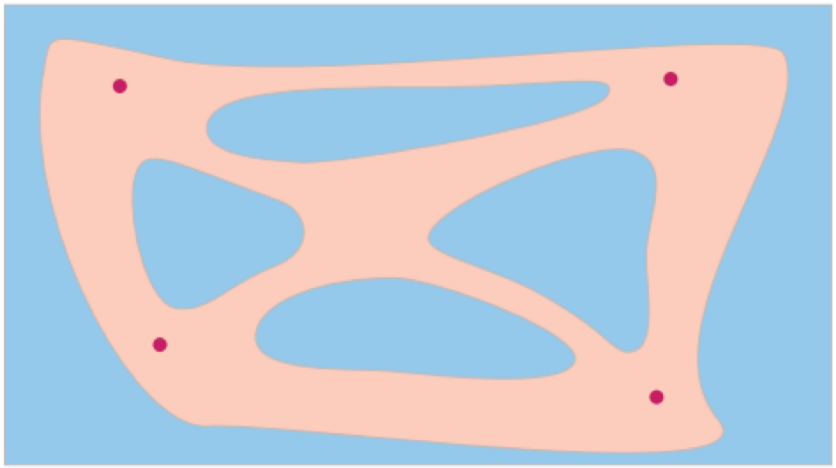
- Are classification regions convex sets?
- **Experiment:** take k images from same estimated class, check whether random convex combination is labeled in that class too



Characterizing topology of classification regions

~~Convex classification regions~~

- Classification regions are not convex sets
- Probably only near-convex in the regions between **pairs** of data points which are close to the real space of images



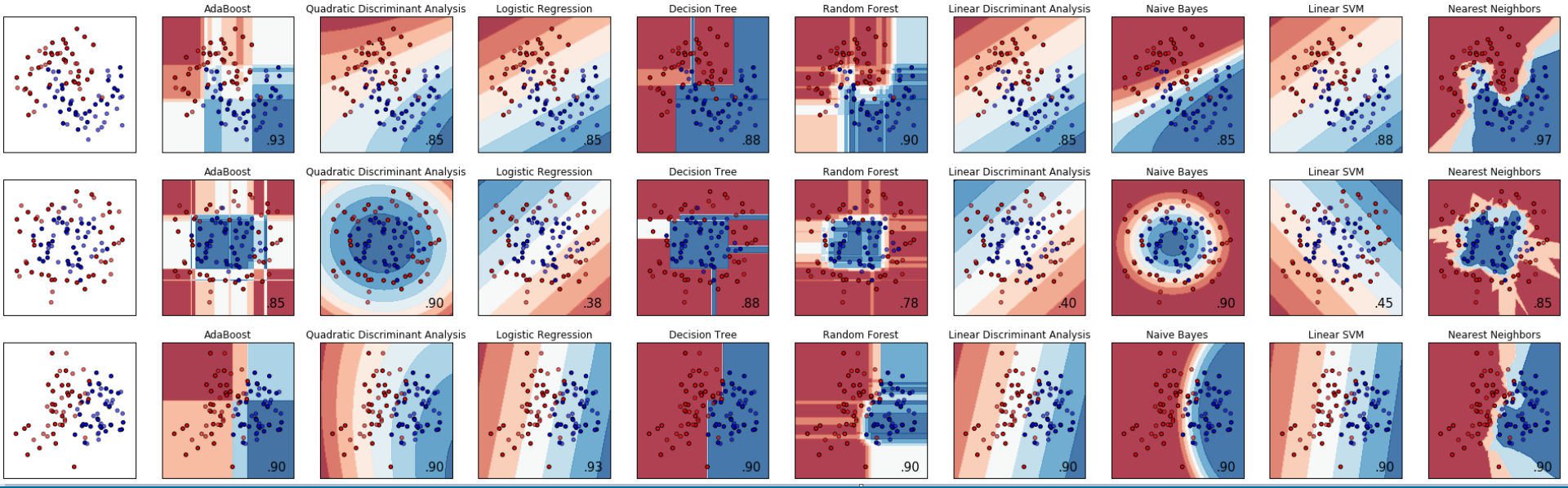
Note that this is a hypothetical **2D** illustration.

Characterizing topology of classification regions

Decision Boundaries

- How complex are the functions that our networks are learning?
- What do their **decision boundaries** look like?

Image source: [Aayush Agrawal]



Decision Boundaries

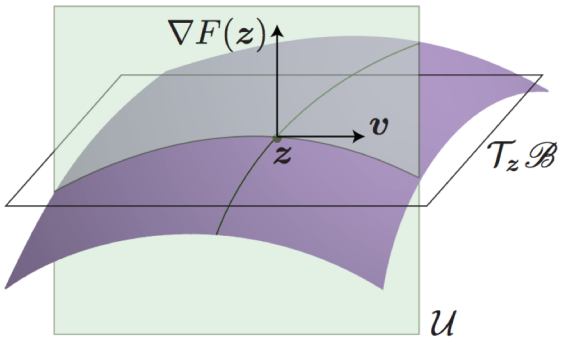
- Pairwise decision boundary between classes i and j

$$\mathcal{B} = \{\mathbf{z} : f_i(\mathbf{z}) = f_j(\mathbf{z})\}$$

- Hypersurface in $\mathbb{R}^d$
- At any point $\mathbf{z}$ on the decision boundary, the gradient of $F(\mathbf{z}) = f_i(\mathbf{z}) - f_j(\mathbf{z})$ is orthogonal to the tangent space $\mathcal{T}_{\mathbf{z}}(\mathcal{B})$

i.e. $\nabla F(\mathbf{z})$ is a normal vector to the tangent space

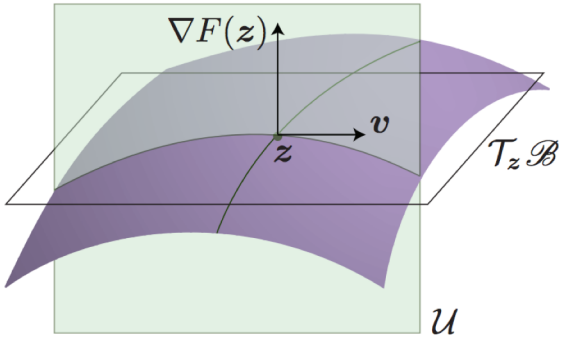
gradient is the direction in input space of most rapid **F(z)** increase



normal section of decision boundary

Review of Normal Curvature

- **Question:** what is the curvature of the decision boundary near natural images?



Normal curvature at point z along tangent direction v

$$\kappa(\mathbf{z}, \mathbf{v}) = \frac{\mathbf{v}^T \mathbf{H}_F \mathbf{v}}{\|\mathbf{v}\|_2^2 \ \|\nabla F(\mathbf{z})\|_2}$$

how quickly the normal is changing along tangent direction v

Characterizing decision boundary curvature

Aside: What's That Formula?

- We've learned that normal curvature is defined

$$\kappa_N(X) = \frac{\langle df(X), dN(X) \rangle}{\langle df(X), df(X) \rangle}$$

- How does the paper's formulation equate?

$$\begin{aligned}
\kappa(\mathbf{z}, \mathbf{v}) &= \frac{\mathbf{v}^T \mathbf{H}_F \mathbf{v}}{\|\mathbf{v}\|_2^2 \ \|\nabla F(\mathbf{z})\|_2} \\
&= \frac{\left\langle \mathbf{v}, \mathbf{J} \left(\frac{\nabla F(\mathbf{z})}{\|\nabla F(\mathbf{z})\|_2} \right)^T \mathbf{v} \right\rangle}{\langle \mathbf{v}, \mathbf{v} \rangle} \\
&= \frac{\langle \mathbf{v}, \mathbf{J}(\mathbf{n})^T \mathbf{v} \rangle}{\langle \mathbf{v}, \mathbf{v} \rangle}
\end{aligned}$$

where **n** is the normal vector

Characterizing decision boundary curvature

Review of Normal Curvature

- d - 1 **principal directions** with associated **principal curvatures**

principal curvatures are the eigenvalues of the shape operator

$$\frac{1}{\|\nabla F(\mathbf{z})\|_2}\mathbf{P}\mathbf{H}_F\mathbf{P} \quad \text{for} \quad \mathbf{P} = \mathbf{I} - \nabla F(\mathbf{z})\nabla F(\mathbf{z})^T$$

projection operator on tangent space

$\mathbf{nn}^T$ projects onto $\mathbf{n}$,
$\mathbf{v} - \mathbf{nn}^T\mathbf{v}$ orthogonal to $\mathbf{n}$

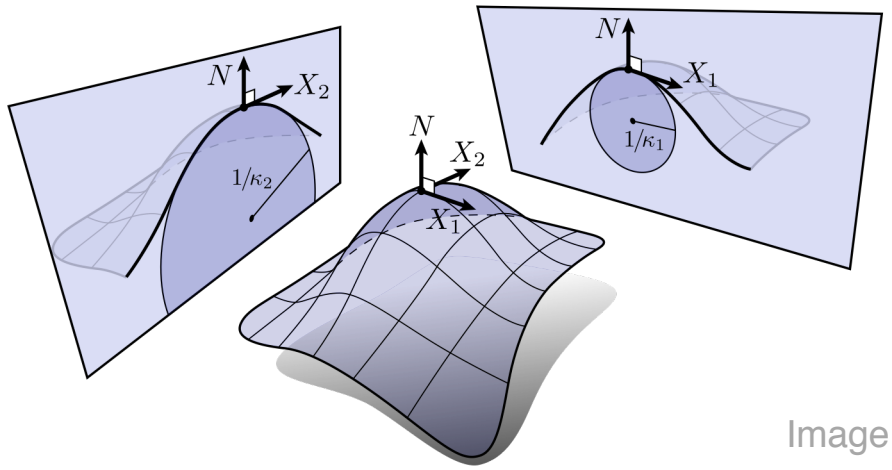


Image source: Keenan Crane

Characterizing decision boundary curvature

Review of Normal Curvature

- d - 1 **principal directions** with associated **principal curvatures**

each principal direction is the tangent direction which **maximizes curvature** subject to the constraint of being **orthogonal** to all previous principal directions

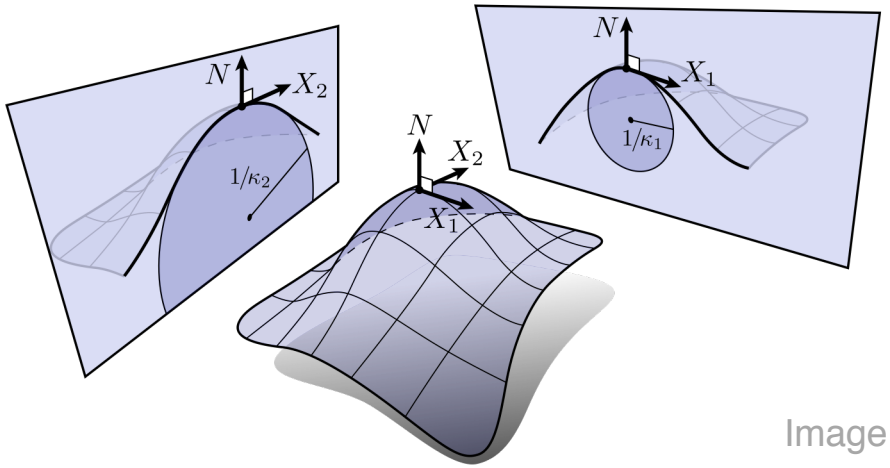


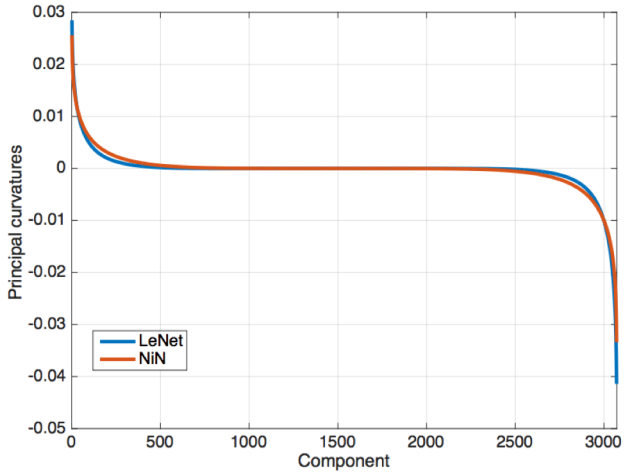
Image source: Keenan Crane

Characterizing decision boundary curvature

Curvature of the Decision Boundaries

- Choose 1000 natural images from CIFAR-10 validation set
  - For CIFAR-10, **d** = 32 * 32 * 3 = 3072
- Compute nearest image on decision boundary **z**
  - Recall: $\mathbf{z} = \mathbf{x} + \mathbf{r}(\mathbf{x})$, where $\mathbf{r}(\mathbf{x})$ is the shortest vector from $\mathbf{x}$ to the decision boundary (the "adversarial perturbation")
- Compute principal curvatures at **z**

average for each principal curvature
over all 1000 randomly sampled images



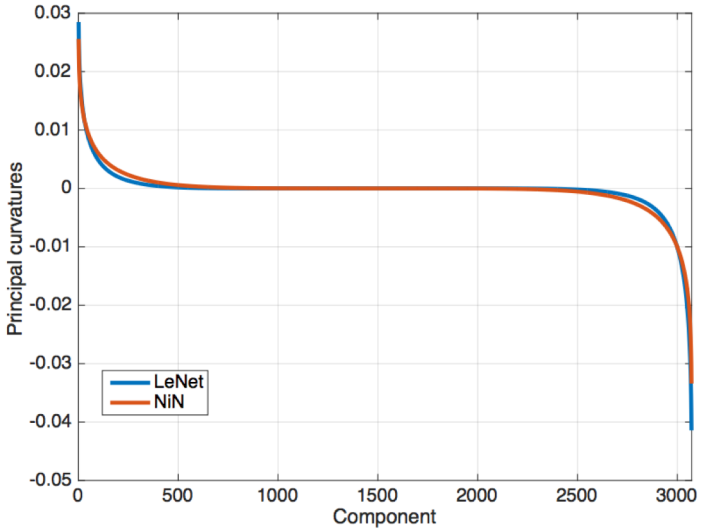
Characterizing decision boundary curvature

Curvature of the Decision Boundaries

Observations

In the vicinity of **real** images:

- Most principal curvatures are ~0 (decision boundary is essentially flat in most tangent directions)
- But *some* directions of high positive/negative curvature
  - Positive curvature → convex
  - Negative curvature → concave
- Asymmetric toward negative curvature
  - The negative curvatures are of slightly greater magnitude than positive curvatures
  - **Average curvature near real images is negative**
    - **Important later!**
  - Consistent across different data, networks, etc.



Characterizing decision boundary curvature

Curvature of the Decision Boundaries

- **Question:** are directions of high curvature shared across different data points (images)?
- To determine:
  1. Compute largest principal directions for 100 random training samples
     a. Put directions into a matrix **M**
  2. Most common curved directions ← m largest singular vectors of **M**
  3. Check curvature in these directions for boundary near unseen samples

$$\mathbf{z} = \mathbf{x} + \mathbf{r}(\mathbf{x})$$

note similarity to previous form; numerator is just $(\mathbf{P}\mathbf{u}_i)^T \mathbf{H}_F (\mathbf{P}\mathbf{u}_i)$

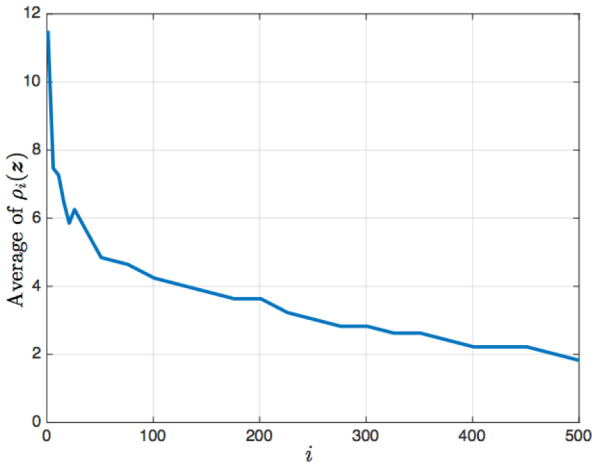
$$\rho_i(\mathbf{z}) = \frac{|\mathbf{u}_i^T \mathbf{P} \mathbf{H}_F \mathbf{P} \mathbf{u}_i|}{\mathbb{E}_{\mathbf{v} \sim \mathbb{S}^{d-1}} \left(|\mathbf{v}^T \mathbf{P} \mathbf{H}_F \mathbf{P} \mathbf{v}| \right)}$$

how curved is decision boundary in direction $\mathbf{u}_i$, relative to random directions

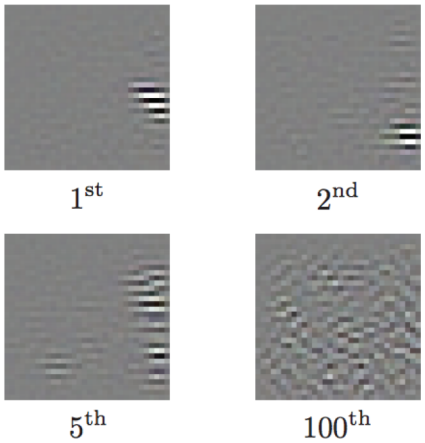
Characterizing decision boundary curvature

Curvature of the Decision Boundaries

- **Results:** $\rho_i(\mathbf{z})$ average for 1000 different images



highly curved along the first directions, independent of the data!



directions of universal curvature

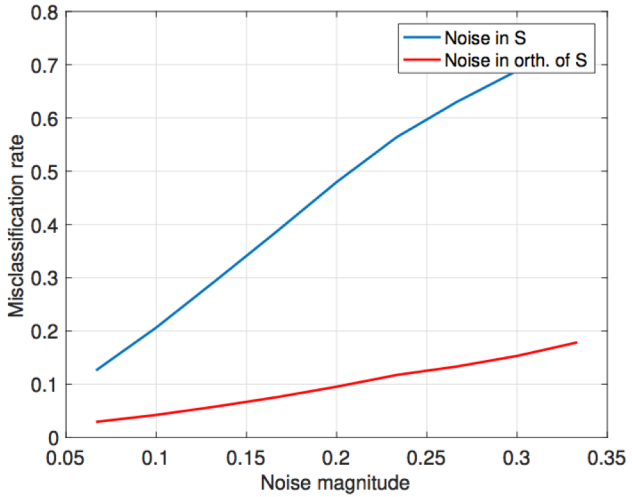
$$\mathbf{z} = \mathbf{x} + \mathbf{r}(\mathbf{x})$$

$$\rho_i(\mathbf{z}) = \frac{|\mathbf{u}_i^T \mathbf{P}\mathbf{H}_F\mathbf{P}\mathbf{u}_i|}{\mathbb{E}_{\mathbf{v}\sim\mathbb{S}^{d-1}}(|\mathbf{v}^T\mathbf{P}\mathbf{H}_F\mathbf{P}\mathbf{v}|)}$$

Characterizing decision boundary curvature

Curvature of the Decision Boundaries

- Empirically, classifier much less robust to perturbations in directions where the decision boundary is highly curved



Construct subspace S of first 200 shared directions of high curvature. Try perturbing in directions from S or from $S^{\perp}$.

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).

Characterizing decision boundary curvature

Curvature of the Decision Boundaries

- Adversarial perturbations have much larger components in the directions of high curvature than random perturbations do

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 $\mathcal{S}$.

Curvature plays a big role in characterizing network sensitivities.

Characterizing decision boundary curvature

Adversarial Examples

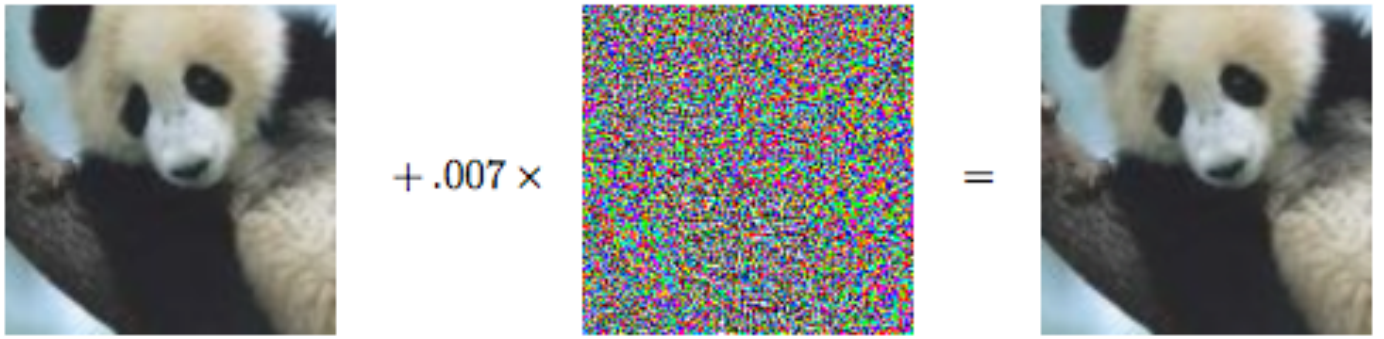
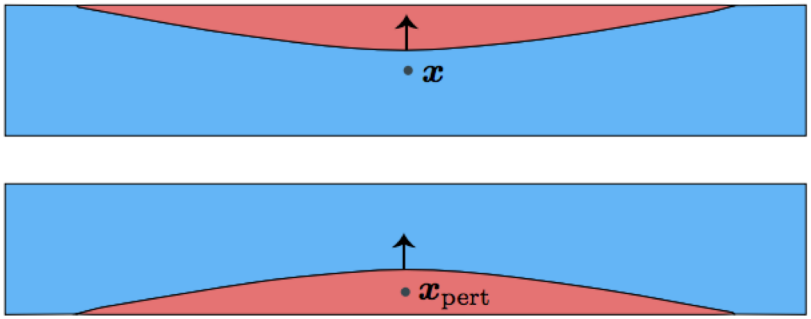


Image source: Goodfellow et al.

Detecting Adversarial Examples

- Average curvature of boundary near natural images is negative
- Average curvature of boundary near adversarial images is positive



normal section in vicinity of
natural image/perturbed image

- **Look at average curvature of boundary near data point to decide whether original or adversarially-perturbed**

Applying geometric insights to defend against adversarial examples

Detecting Adversarial Examples

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 $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T$ 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* $\leftarrow$ $\operatorname{argmax}_{\substack{i \in \{1,\ldots,L\} \\ i \neq \hat{k}(\boldsymbol{x})}} \sum_{j=1}^{T} \boldsymbol{v}_j^T G_{F_i} \boldsymbol{v}_j$.
8: **end if**

Use threshold t (instead of 0) to control the tradeoff between true/false positives

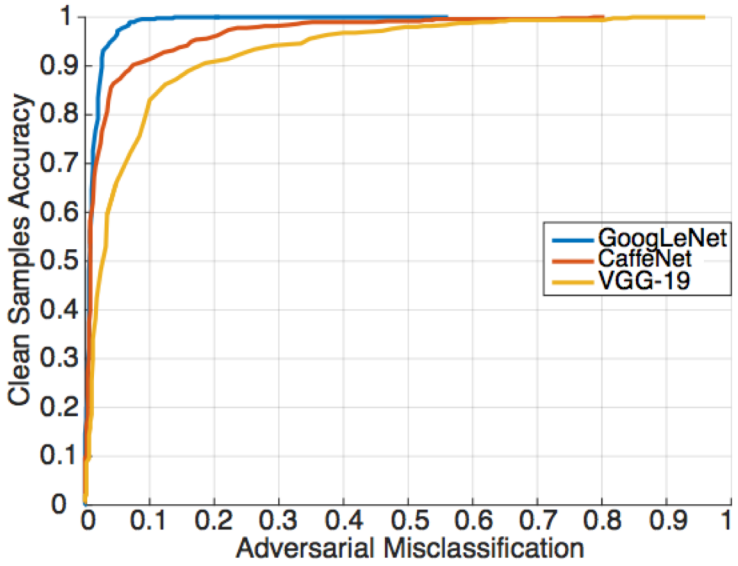
Estimate avg. curvature over decision boundaries between predicted class and *all* of the other classes

Recover correct label as class corresponding to the decision boundary with highest positive curvature

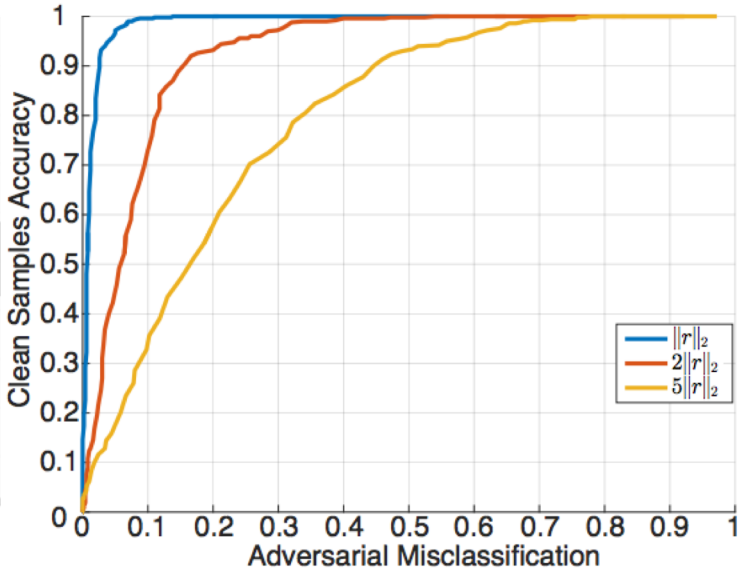
from perspective of perturbed point

Applying geometric insights to defend against adversarial examples

Results: True vs False Positives



Detection accuracy on original images versus failure rate on perturbed images

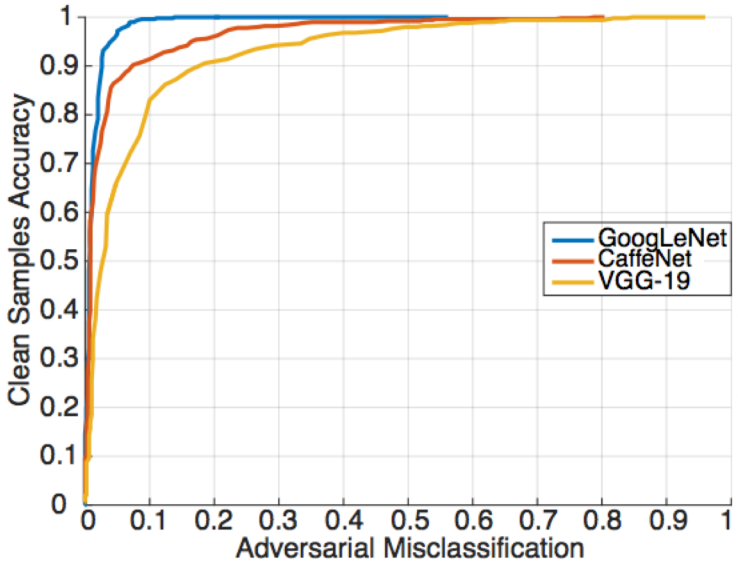


Same thing, but perturbations now multiplied by some constant factor

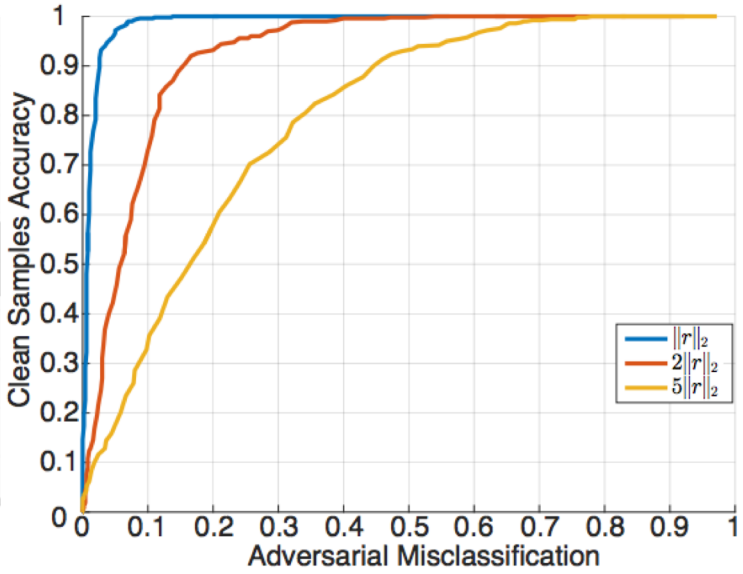
Correctly recovers labels with accuracy of 92%, 88%, and 74% for GoogLeNet, CaffeNet, and VGG-19 respectively (with t = 0).

Applying geometric insights to defend against adversarial examples

Results: True vs False Positives



Detection accuracy on original images versus failure rate on perturbed images

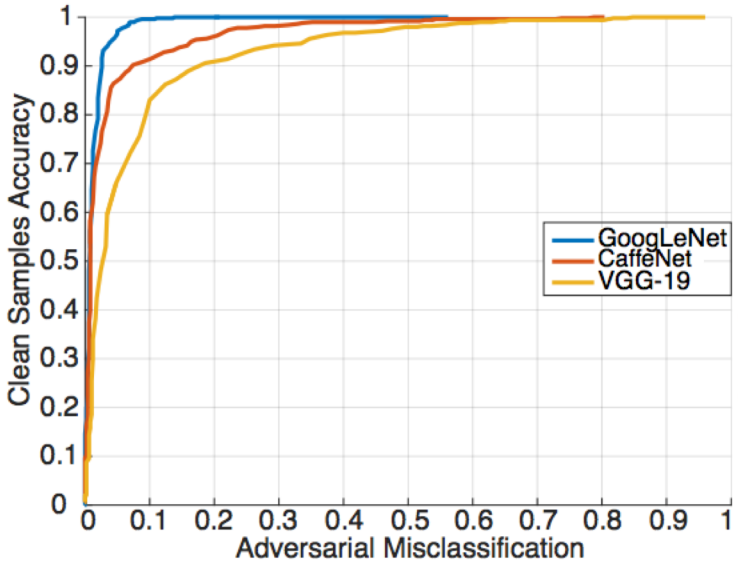


Same thing, but perturbations now multiplied by some constant factor

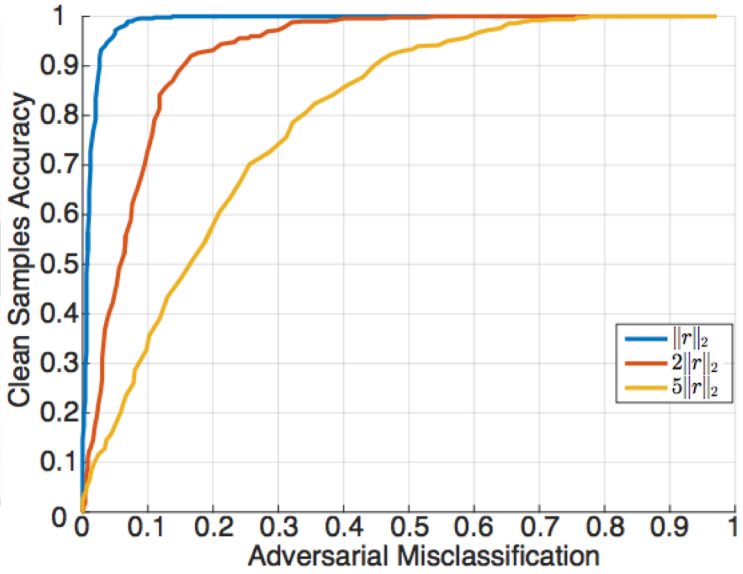
Does not require any extra training!

Applying geometric insights to defend against adversarial examples

Results: True vs False Positives



Detection accuracy on original images versus failure rate on perturbed images



Same thing, but perturbations now multiplied by some constant factor

Good at detecting small perturbations, bad at detecting large perturbations (opposite is true for traditional methods).

Applying geometric insights to defend against adversarial examples

Takeaways

- A study in three parts
  1. Topology of classification regions
     a. **Insight:** deep classifiers learn big, connected classification regions
  2. Curvature of decision boundaries
     a. **Insight:** decision boundary flat in most directions near natural images
     b. **Insight:** vulnerable to perturbation in the directions of high curvature
  3. Applying insights to adversarial example detection
     a. Can detect adversarial examples with small additive perturbations
     b. Can recover the classification labels of the images pre-perturbation