Structural Robustness for Deep Learning Architectures

Carlos Lassance, Vincent Gripon, Jian Tang, Antonio Ortega

Mila (UdeM and HEC), IMT Atlantique and USC

Data Science Workshop, 2019

June 4th, 2019

Outline

Context

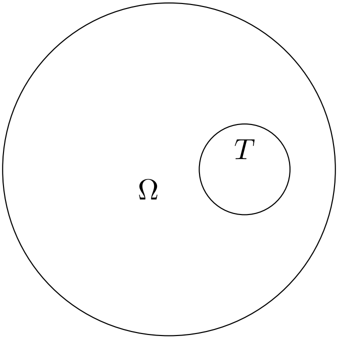
- Deep Nets are easily fooled;
- Methods to prevent this:
    - Enrich the training set:
    - **However:** How to enrich? Implicit control.
    - Impose structural properties on network functions:
    - **However:** Often too restrictive.

Our work

- **Our Proposal**: localized lipschitz constraint around the examples;
- **Main contributions**:
    - Why proposed structural properties fail;
    - Relation between: proposed criterion and existing methods;
    - Robustness prediction using training set only.

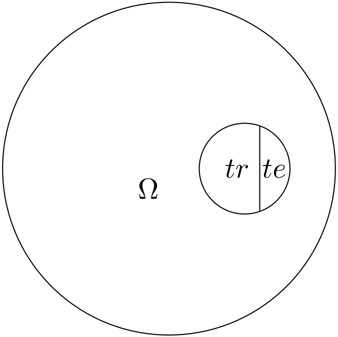
Classification

- Regression with finite output;
- **Objective:** Generalization;
  - We have a training (restrict) set T of the domain Ω;
  - How does the classifier work on images outside the training set?
- **Problem:** How to define generalization performance?



Cross-Validation

- Randomly divide the restrict set T in train (tr) and test (te);
- Proxy to unseen images;
- **Problem:** te and tr follow the same distribution!

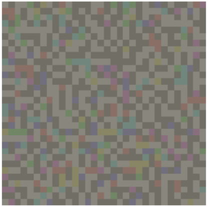


Robustness

Worst case scenario

Adversarial attacks

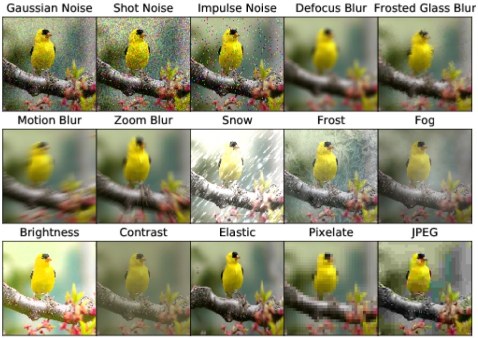
Noise generated to specifically fool the network.

Original	Noise	Adversarial Image
		
Deer 99.96%	Cat 36.63%	Cat 90.66%

" Limitations of adversarial robustness: strong No Free Lunch Theorem "

Random corruptions

Noise generated due to hardware problems, weather, noise, etc. [Heynckes & Dietterich 2019]



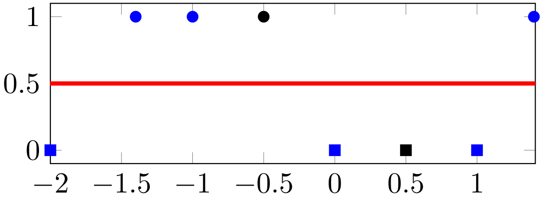
Related Work

We analyze works based in two directions:

1. Increase the size of the domain T:
   - Bigger T -> Smaller $|\Omega| - |T|$;
   - However $|\Omega| \approx \infty$;
   - Example: Adversarial Training (PGD, FGSM ...).
2. Design network architectures with robust properties:
   - A: Control the Lipschitz constant of the network;
     - α-Lipschitz: $\forall x, \forall \epsilon, ||f(x+\epsilon) - f(x)|| \leqslant \alpha||\epsilon||$;
   - B: Control the deformation of the boundary;
   - **Prior:** Small changes in the input -> Small changes in the output;
   - Examples: Parseval, Laplacian and L2NonExpansive networks.

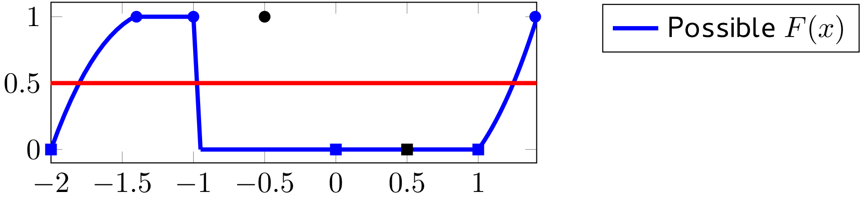
Scenario

- Classify data, two classes (circles and squares);
- tr: blue;
- te: black.

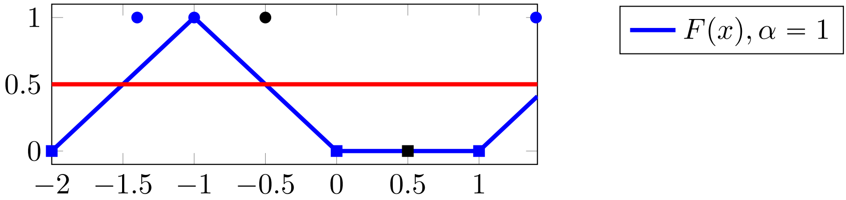


Scenario

- Train a network F(x);
- How to make it respect the **prior**?



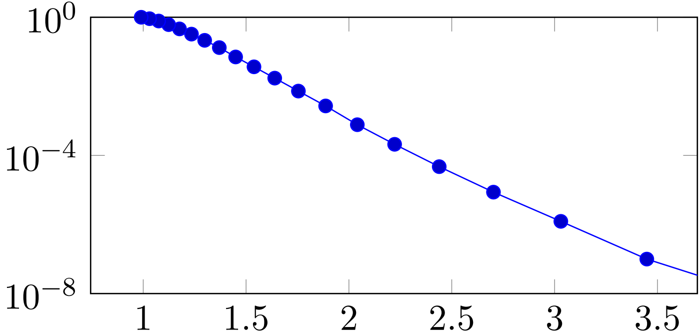
- Bound the network variation;
- Bound $\alpha \leqslant 1 \rightarrow$ respect prior;
- **However:** sometimes incompatible with dataset.



Lipschitz constant vs CIFAR-10

- Test α incompatibility on CIFAR-10 tr;
- Metric: L_∞;
- Output: One-hot embeddings.

Fraction of pairs incompatible with the constraint:



Proposed Solution

- Recall α-Lipschitz: $\forall x, \forall \epsilon, ||f(x+\epsilon) - f(x)|| \leqslant \alpha||\epsilon||$;

Proposed Solution

- Recall α-Lipschitz: $\forall x \in T, \forall \epsilon, ||f(x+\epsilon) - f(x)|| \leqslant \alpha ||\epsilon||$;

Proposed Solution

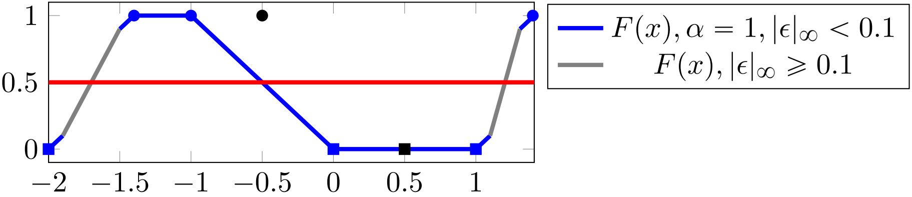
- Recall α-Lipschitz: $\forall x \in T, \forall ||\epsilon|| \leqslant ||r||, ||f(x+\epsilon) - f(x)|| \leqslant \alpha||\epsilon||$;

Proposed Solution

- Local α-Lipschitz: $\forall x \in T, \forall ||\epsilon|| \leqslant ||r||, ||f(x+\epsilon) - f(x)|| \leqslant \alpha ||\epsilon||$;

Proposed Solution

- Local α-Lipschitz: $\forall x \in T, \forall ||\epsilon|| \leqslant ||r||, ||f(x+\epsilon) - f(x)|| \leqslant \alpha||\epsilon||$;

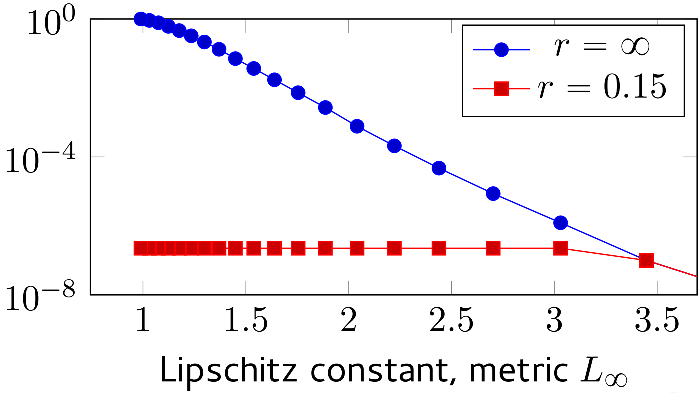


Locality and domain-restricted

Lipschitz constant vs CIFAR-10

- Test α incompatibility on CIFAR-10 tr;
- Metric: L_∞;
- Output: One-hot embeddings.

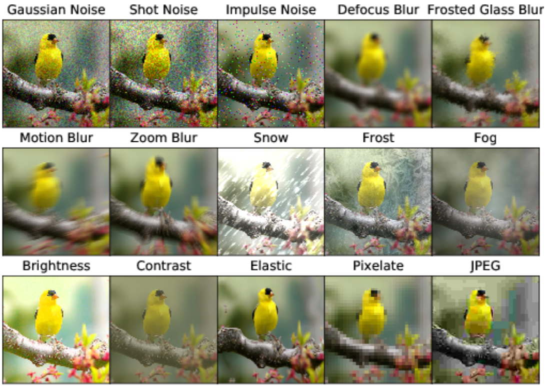
Fraction of pairs incompatible with the constraint:



Existing methods under our definitions

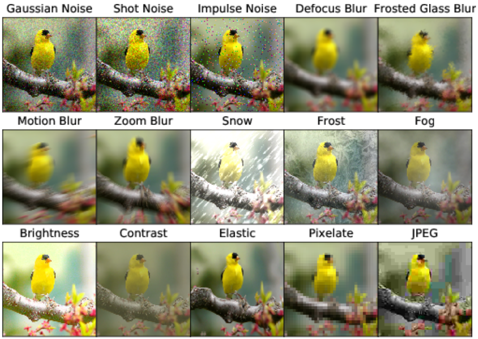
1. Vanilla (V)
2. Parseval Networks (P) [Cisse et al 2017]:
   - Regularizer to enforce $\alpha_{lim} = 1$;
   - Soft constraint, everywhere on the space.
3. L2 Non Expansive (L2NN) [Qian and Wegman 2019]:
   - Change network structure to enforce $\alpha_{lim} = 1$;
   - Hard constraint, everywhere on the space.
4. Laplacian Networks (L) [Ours 2019]:
   - Regularizer to enforce smooth transitions;
   - Soft constraint, around the boundary region.
5. PGD Training (PGD) [Madry et al 2018]:
   - Add adversarial examples to tr;
   - Increases the domain tr in a localized way.

- Robustness benchmark [Heynckes & Dietterich 2019];
- Generates $\hat{te}$.



Results

Clean images (Acc_{te}): P > V > PGD > L > L2NN;
Relative performance $(Acc_{te} - Acc_{\hat{te}})$: PGD > L2NN > L > P > V.



- Test α_{lim} and r around examples in tr;
- Robustness comes from:
  - Small $r \rightarrow$ Small α.

- Test α_{lim} and r around examples in tr;
- Robustness comes from:
  - Small $r \rightarrow$ Small α.

Recall

Relative performance: PGD > L2NN > L > P > V.

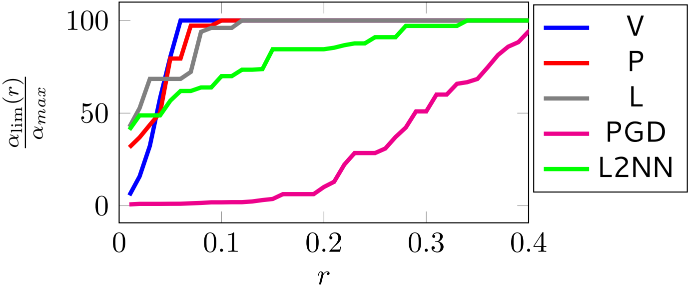
Experiments

Proposed measure

- Test α_{lim} and r around examples in tr;
- Robustness comes from:
  - Small $r \rightarrow$ Small α.

Recall

Relative performance: PGD > L2NN > L > P > V.



Conclusion

- Introduced a formal definition of robustness:
  - Based on a slope α defined on a radius r around T.
- Analyzed existing methods in the literature;
- Demonstrated an empirical link between proposal and robustness.