Bounding Indicator Function Smoothness for Neural Networks Robustness

Carlos Eduardo Rosar Kos Lassance, Vincent Gripon
IMT Atlantique/Lab-STICC and Université de Montréal/Mila

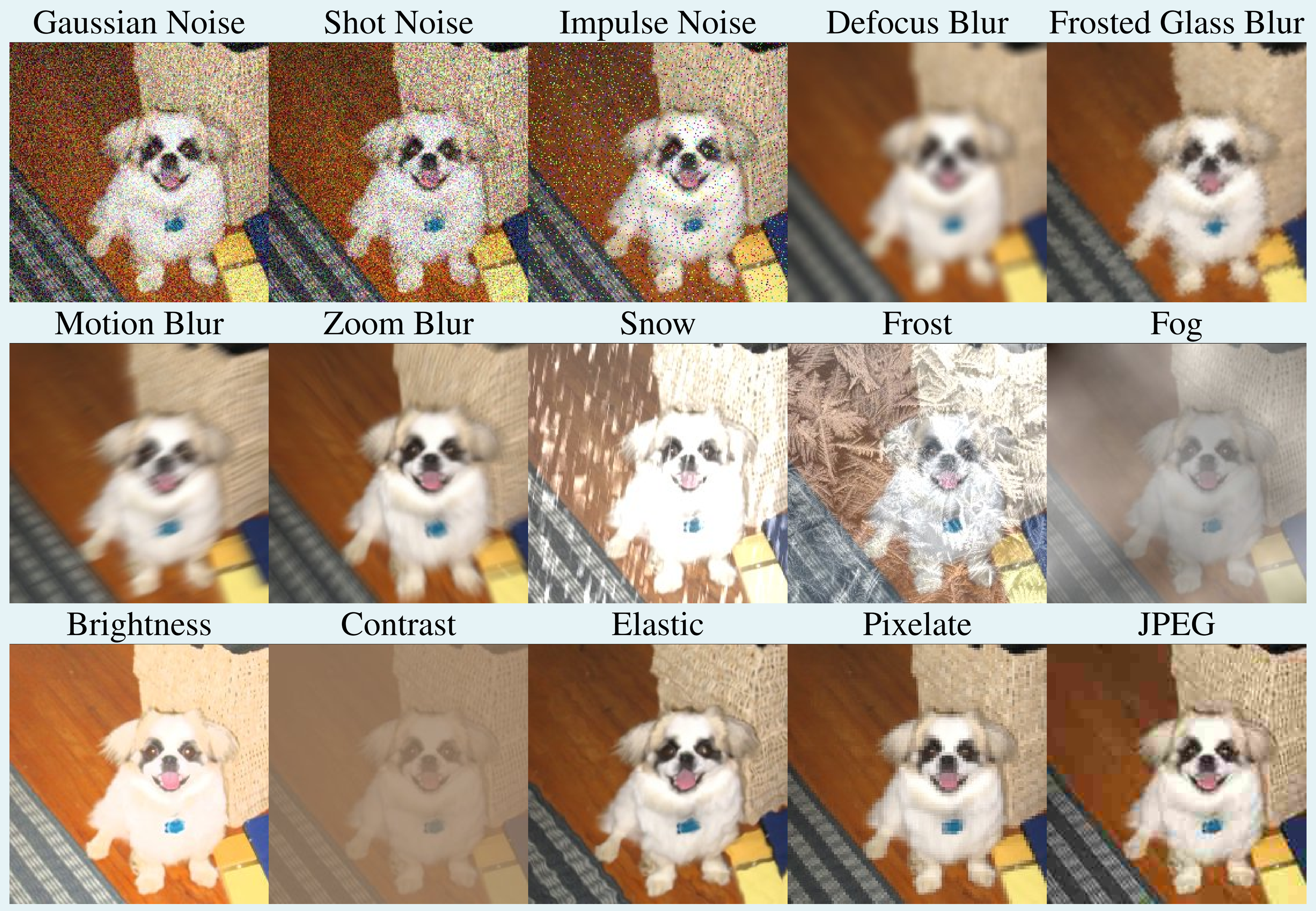
Antonio Ortega
University of Southern California

1. Context

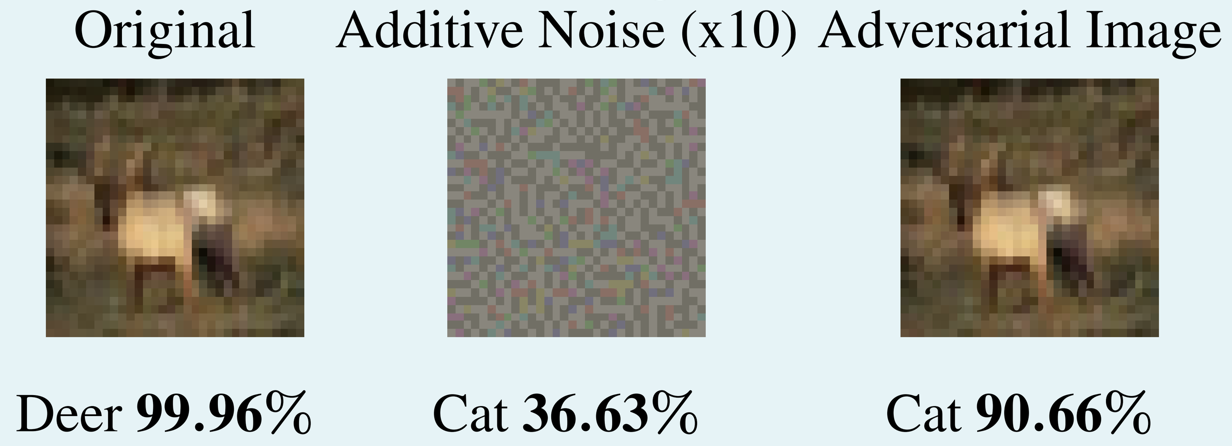
- **DNNs** achieve **state of the art performance** on various tasks [1];
- **How?:** they absorb stats of data through millions of parameters;
- **Drawback:** they are easily fooled (adversarially or randomly) [2,3,4];
- **Reasoning:** exagerated deformation of space around class boundaries;
- **Proposal:** enforce smooth deformations via regularization;
- Three steps:
  1. Generate similarity graphs of the intermediate representations of the network;
  2. Compute class indicator vector signal smoothness [6] over the graphs;
  3. Penalize unsmooth transitions over successive layers.

2. Perturbations

Random perturbations (from [2])



Adversarial attacks (using the method from [3])



3. Illustration

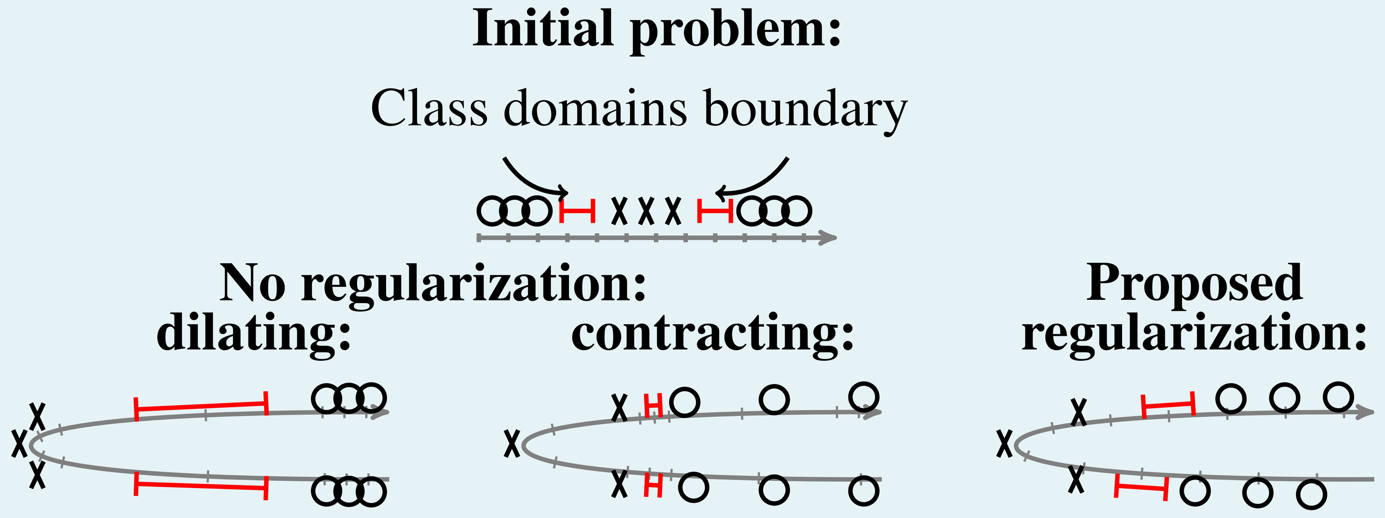


Figure: Consider the problem of classifying circles and crosses (top). Without use of regularizers, one may considerably stretch the boundary regions (bottom left), or push the inputs closer (bottom center). The first case may lead to sharp transitions while the second reduces the classification margin. Forcing small variations (bottom right), we ensure the topology is not dramatically changed in the boundary regions.

4. Methodology

Add a **regularization** term to the **loss function**:

procedure SMOOTHNESS($\mathbf{activations}^\ell, \mathbf{s}, \boldsymbol{m}$)
  $\mathbf{M}^\ell \leftarrow$ Pairwise cosine similarity of $\mathbf{activations}^\ell$
  $\mathbf{D}^\ell \leftarrow$ Diagonal degree matrix of $\mathbf{M}^\ell$
  $\mathbf{L}^\ell \leftarrow \mathbf{D}^\ell - \mathbf{M}^\ell$
  $\sigma^\ell \leftarrow \text{Trace}(\mathbf{s}^\top (L^\ell)^m \mathbf{s})$
  return σ^ℓ
end procedure
procedure LOSS($list_{activations}, \mathbf{y}, \mathbf{s}, \boldsymbol{m}, \gamma$)
  for $\mathbf{activations}^\ell \in list_{activations}$ **do**
    $\sigma^\ell \leftarrow \text{Smoothness}(\mathbf{activations}^\ell, \mathbf{s}, \boldsymbol{m})$
  end for
  $\Delta \leftarrow \frac{\sum_{i=1}^{\ell_{max}} |\sigma^i - \sigma^{i-1}|}{\ell_{max} - 1}$
  return $\text{CategoricalCrossEntropy}(\mathbf{s}, \mathbf{y}) + \gamma^m \Delta$
end procedure

5. What power of the Laplacian to use?

Analysis: effect of Laplacian powers on similarity graphs.
Conclusion: higher powers of the Laplacian $\longrightarrow$ better visualization.

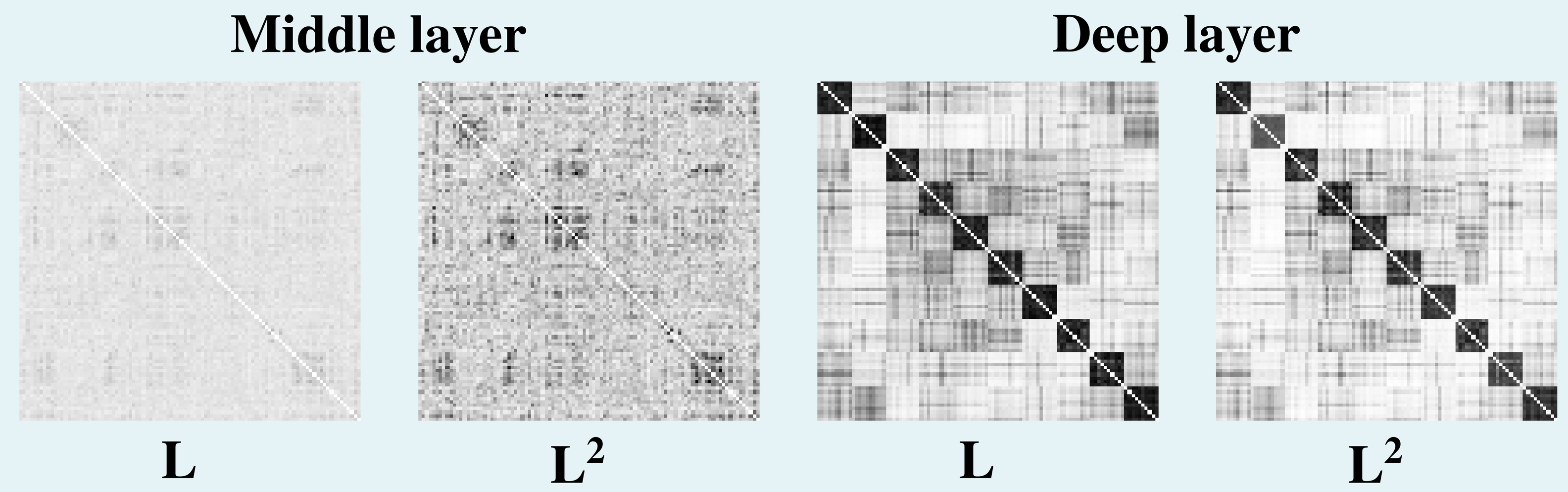


Figure: Comparison of Laplacian and squared Laplacian of similarity graphs.

6. Smoothness evolution

Analysis: smoothness evolution under different training methods.
Conclusion: square laplacian leads to a finer control of the evolution.

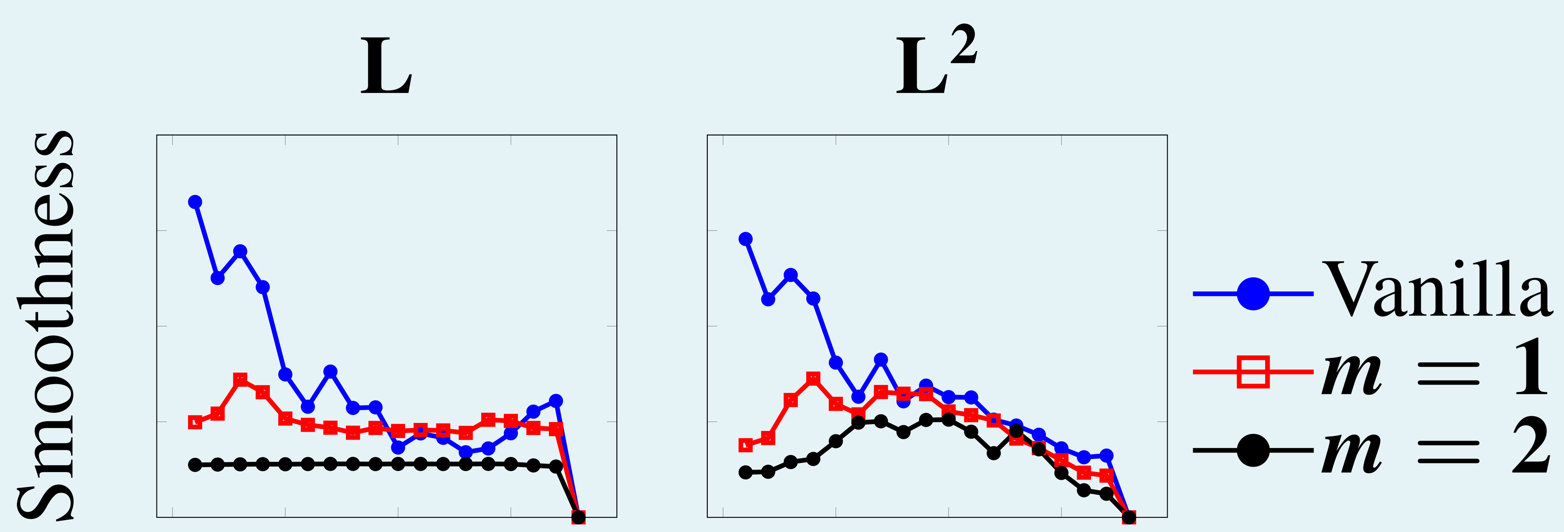


Figure: Evolution of smoothness of label signals. Regularized networks yield flatter curves, therefore changes in smoothness are smaller. This implies that average distances between examples in distinct classes remain almost constant.

Full article available at https://arxiv.org/abs/1805.10133

7. Experiments

We compare our method against vanilla CNNs and [5] on three datasets:

	CIFAR-10				CIFAR-100			Imagenet32	
Model	C	[2]	[3]	GN	C	GN	[4]	C	GN
Vanilla	12%	100%	99.3%	48%	21%	87%	80%	47.9%	63.2%
Parseval [5]	**10%**	104%	99.2%	50%	**20%**	85%	78%	51.9%	65.9%
Ours	13%	**98%**	**89.6%**	**39%**	21%	**84%**	**77%**	**47.6%**	**62.6%**

Table: Median test set error rate on various dataset states. [2] is a relative measure against the baseline. GN is the test set perturbed with Gaussian Noise, C is the clean test set. Smaller values are better.

7. Conclusion

- We proposed a **regularizer** to increase neural network robustness,
- Our method enforces **signal smoothness** over **intermediate representation similarity graphs** to restrict **boundary deformations**,
- The proposed method **outperforms vanilla** CNNs and [5].

8. Future work

- We would like to **better understand** the robustness with a more **theoretical analysis**,
- Instead of **training** the network from **zero**, use the **regularizer** as an **ad-hoc post training step**,
- **Extend** the evaluations to **other domains in machine learning**.

9. References

[1] "Identity mappings in deep residual networks", 2016.
[2] "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations", 2019.
[3] "Towards Deep Learning Models Resistant to Adversarial Attacks", 2018.
[4] "Explaining and harnessing adversarial examples", 2014.
[5] "Parseval Networks: Improving Robustness to Adversarial Examples", 2017
[6] "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains", 2013.