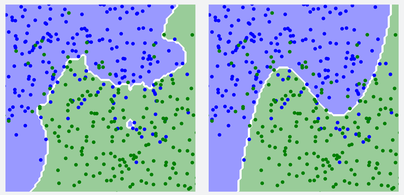
Efficient Training of Low-Curvature Neural Networks

Suraj Srinivas*, Kyle Matoba*, Hima Lakkaraju, François Fleuret
Harvard University, Idiap Research Institute & EPFL, Harvard University, University of Geneva
*Equal contribution

Neural networks can have excessive non-linearity, making them sensitive to input noise. This increases susceptibility to adversarial examples and gradient instability.



High Curvature Map Low Curvature Map

Research Question: How to efficiently train (low-curvature) neural networks without excessive non-linearity?

Our Approach:

- We find that max (global) curvature of neural networks can be upper bounded by curvatures and slopes of their constituent layers (Thm 1)
- Define novel architectural components: centered β-softplus, and γ-Batchnorm that enable well-defined penalization
- Penalize β and γ to penalize global curvature, without computing expensive point-wise Hessian norms!

How to Penalize Curvature

Given function $f : \mathbb{R}^n \to \mathbb{R}$, define its curvature at $\mathbf{x}$ by

$$\mathcal{C}_f(x) = \frac{\|\nabla_x^2 f(\mathbf{x})\|_2}{\|\nabla_x f(\mathbf{x})\|_2 + \epsilon} \quad \begin{matrix} \to \text{spectral norm of Hessian} \\ \to \text{gradient norm} \end{matrix}$$

There are two broad strategies for curvature penalization

1. Local Penalization: minimizes point-wise curvature, e.g.: [Dezfooli et al., 2019]
2. Global Penalization: minimizes upper-bound on curvature across data points, e.g.: [Dombrowski et al., 2022]

Local Penalization	Global Penalization
+ (Nearly) exact penalties	- Loose bounds can lead to inexact penalization
- Expensive	+ Efficient!
- Penalization only at data	+ Penalization everywhere

In this work, we choose to do global penalization, but find that using approximate local penalization also helps!

Upper Bounding Normalized Curvature

Given a function $f = f_L \circ f_{L-1} \circ \ldots \circ f_1$ with $f_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$, the curvature $\mathcal{C}_f$ can be bounded by the sum of curvatures of individual layers $\mathcal{C}_{f_i}(\mathbf{x})$, i.e.,

$$\max_x \mathcal{C}_f(\mathbf{x}) \leq \sum_{i=1}^{L} \max_x \mathcal{C}_{f_i}(\mathbf{x}) \prod_{j=1}^{i} \|W_j\|_2.$$

To minimize the bound, we

1. Penalize curvature of activation functions, $\max_x \mathcal{C}_{f_i}(\mathbf{x})$
2. Penalize Lipschitz constants of linear layers, $\|W_j\|_2$

Ingredients of LCNNs

(1) Centered β-softplus is defined as

$$s_c(\mathbf{x};\beta) = \frac{1}{\beta} \log\left(\frac{1+\exp(\beta \mathbf{x})}{2}\right)$$

This has several nice properties:

1. Approximates ReLU: $s_c(x;\beta \to \infty) = \text{ReLU}(x)$
2. Bounded curvature: $\mathcal{C}_{s_c}(x) \leq \beta$
3. Stable at low curvature: $s_c(x;\beta \to 0) = x/2$
4. Stable upon composition: $s_c^n(x=0;\beta) = 0$

Standard softplus diverges for both #3 and #4!

(2) γ-Batchnorm is defined as

$$\gamma\text{-BN}(x) \leftarrow \underbrace{\min(\gamma, ||\text{BN}(x)||_2)}_{\text{scaling factor} \leq \gamma} \times \frac{\text{BN}(x)}{||\text{BN}(x)||_2}$$

(3) "Real" spectral normalization of linear layers, which is defined in previous work [Ryu et al., 2019].

(4) Penalization of curvature by penalization of β and γ parameters, avoids the need to compute Hessian norms!

Advantages of LCNNs

Low curvature models have stable gradients,

$$\frac{\|\nabla_x f(\mathbf{x}+\epsilon) - \nabla_x f(\mathbf{x})\|_2}{\|\nabla_x f(\mathbf{x})\|_2} \sim \|\epsilon\|_2 \mathcal{C}_f(\mathbf{x})$$

...and low curvature is necessary for robustness!

$$\|f(\mathbf{x}+\epsilon) - f(\mathbf{x})\|_2 \sim \|\epsilon\|_2 \|\nabla_x f(\mathbf{x})\|_2 \left(1 + \|\epsilon\|_2 \mathcal{C}_f(\mathbf{x})\right)$$

Evaluating Point-wise Curvature

We present results on Resnet-18 models trained on CIFAR100 dataset.

We find that LCNN-based models reduce point-wise curvature without loss in accuracy, even though they are designed to penalize global curvature.

Model	$\mathbb{E}_X \mathcal{C}_f(X)$	Accuracy (%)
Standard	270.89 $\pm$ 75.04	77.42 $\pm$ 0.11
LCNNs	69.50 $\pm$ 2.41	77.30 $\pm$ 0.11
GradReg	89.47 $\pm$ 5.86	77.20 $\pm$ 0.26
LCNNs + GradReg	25.30 $\pm$ 0.09	77.29 $\pm$ 0.07
HessianReg [1]	116.31 $\pm$ 4.58	76.48 $\pm$ 0.07

Evaluating Adversarial Robustness

We find that LCNN models provide adversarial robustness (to PGD advex and Autoattack) competitive with adversarial training without sacrificing accuracy.

Model	Clean acc. (%)	Adv. acc. (%)
Standard	77.42 $\pm$ .10	16.11 $\pm$ .21
LCNN	77.16 $\pm$ .07	17.66 $\pm$ .18
GradReg	77.20 $\pm$ .26	38.09 $\pm$ .47
LCNNs + GradReg	77.29 $\pm$ .26	42.70 $\pm$ .77
Adv. Training	76.96 $\pm$ .26	44.98 $\pm$ .57

References

[1] Dezfooli, Fawzi, Juesato, and Frossard. "Robustness via curvature regularization, and vice versa", CVPR 2019
[2] Ryu, Liu, Wang, Chen, Wang, and Yin. "Plug-and-play methods provably converge with properly trained denoisers", ICML 2019
[3] Dombrowski, Anders, Muller, and Kessel. "Towards robust explanations for deep neural networks". Pattern Recognition 2022