Paper Review: Sensitivity and Generalization in Neural Networks: an Empirical Study

Introduction

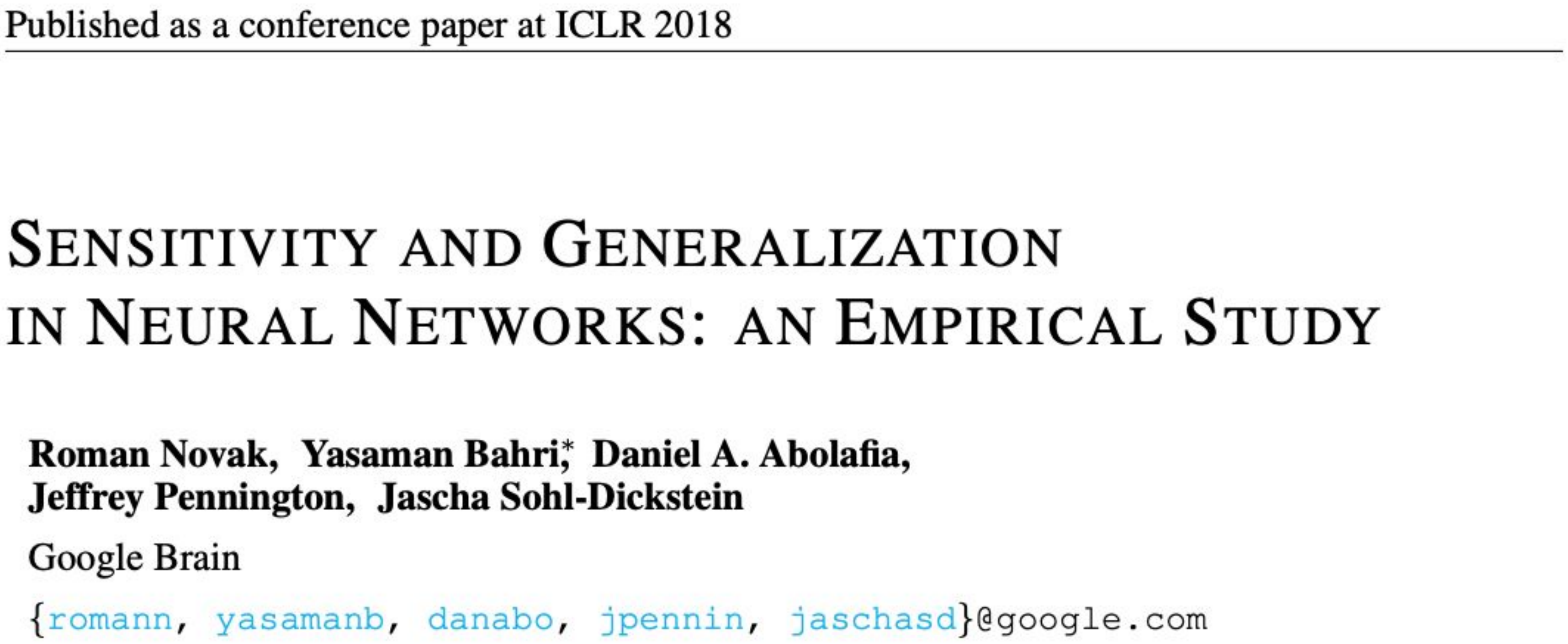
Sensitivity Analysis

- You have found the minimum of function f: $x^* = \operatorname{argmin} f(x)$

- How much your solution changes if you perturb your function?

$$x_\varepsilon = \operatorname{argmin}\ f(x) + \varepsilon g(x)$$
$$\|x_\varepsilon - x^*\| = \ ?$$

Sensitivity Analysis of Deep Neural Nets



Published as a conference paper at ICLR 2018

SENSITIVITY AND GENERALIZATION IN NEURAL NETWORKS: AN EMPIRICAL STUDY

Roman Novak, Yasaman Bahri,[*] Daniel A. Abolafia,
Jeffrey Pennington, Jascha Sohl-Dickstein

Google Brain

{romann, yasamanb, danabo, jpennin, jaschasd}@google.com

Review: [Modern] Generalization

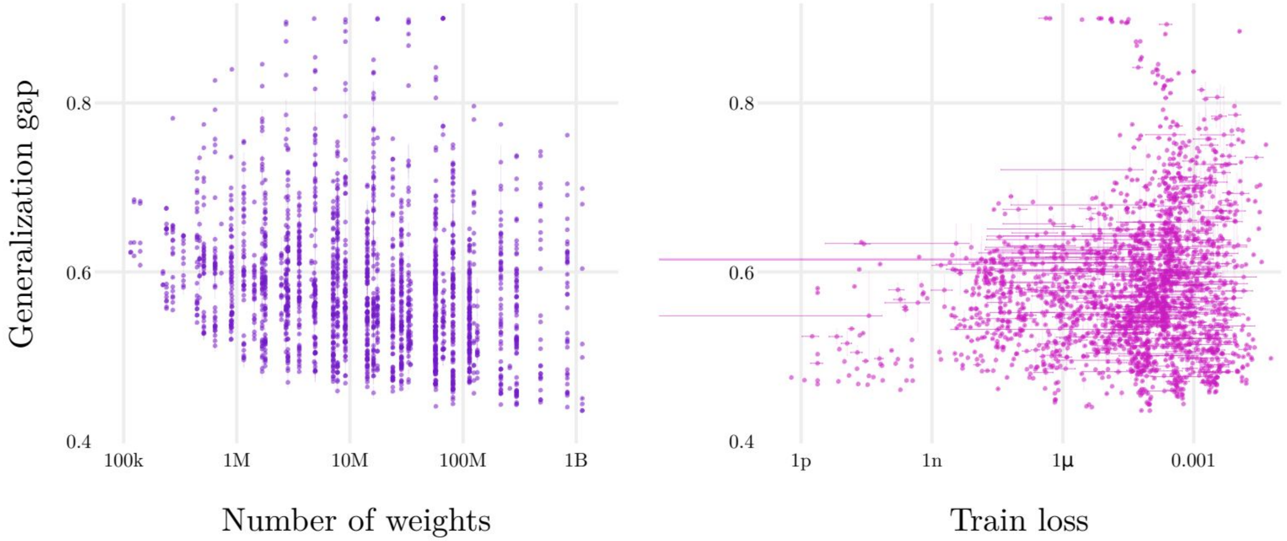


Figure 1: 2160 networks trained to 100% training accuracy on CIFAR10 (see §A.5.5 for experimental details). **Left**: while increasing capacity of the model allows for overfitting (top), very few models do, and a model with the maximum parameter count yields the best generalization (bottom right). **Right**: train loss does not correlate well with generalization, and the best model (minimum along the y-axis) has training loss many orders of magnitude higher than models that generalize worse (left). This observation rules out underfitting as the reason for poor generalization in low-capacity models. See (Neyshabur et al., 2015) for similar findings in the case of achievable 0 training loss.

Review: ReLU Nets

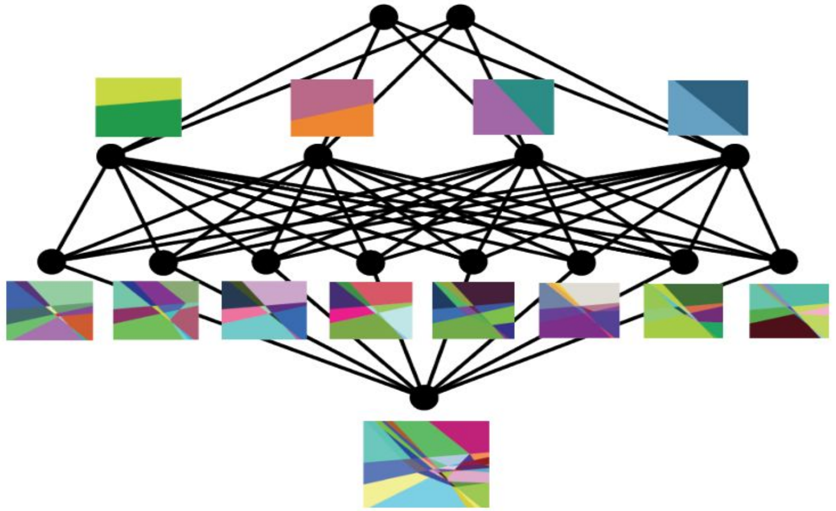


Figure 2. Evolution of linear regions within a ReLU network for 2-dimensional input. Each neuron in the first layer defines a linear boundary that partitions the input space into two regions. Neurons in the second layer combine and split these linear boundaries into higher level patterns of regions, and so on. Ultimately, the input

- Image from Hanin et. al. https://arxiv.org/pdf/1901.09021.pdf

Sensitivity Metrics (1)

1. How does the output of the network change as the input is perturbed within the linear region?
   - *Frobenius norm of the class probabilities Jacobian:*

     $$\mathbf{J}(\mathbf{x}) = \partial \mathbf{f}_\sigma(\mathbf{x}) / \partial \mathbf{x}^\mathbf{T} \text{ (with } J_{ij}(\mathbf{x}) = \partial \left[\mathbf{f}_\sigma(\mathbf{x})\right]_i / \partial x_j)$$

   - *Given test points, we calculate the expected value:*

     $$\mathbb{E}_{\mathbf{x}_\text{test}}\left[\left\|\mathbf{J}\left(\mathbf{x}_\text{test}\right)\right\|_F\right] \qquad \|\mathbf{J}(\mathbf{x})\|_F = \sqrt{\sum_{ij} J_{ij}(\mathbf{x})^2}$$

Sensitivity Metrics (2)

2. How likely is the linear region to change in response to change in the input?

- Brilliant idea!!! Count transitions!
- For a network with piecewise linear activations, we can, given an input x, assign a code to each neuron in the network f, that identifies the linear region of the pre-activation of that neuron. E.g. each ReLU unit will have 0 or 1 assigned to it if the pre-activation value is less or greater than 0 respectively
- Then, a concatenation of codes of all neurons in the network (denoted by c(x)) uniquely identifies the linear region of the input x
- Given x , create k samples near that and measure the change in the code.

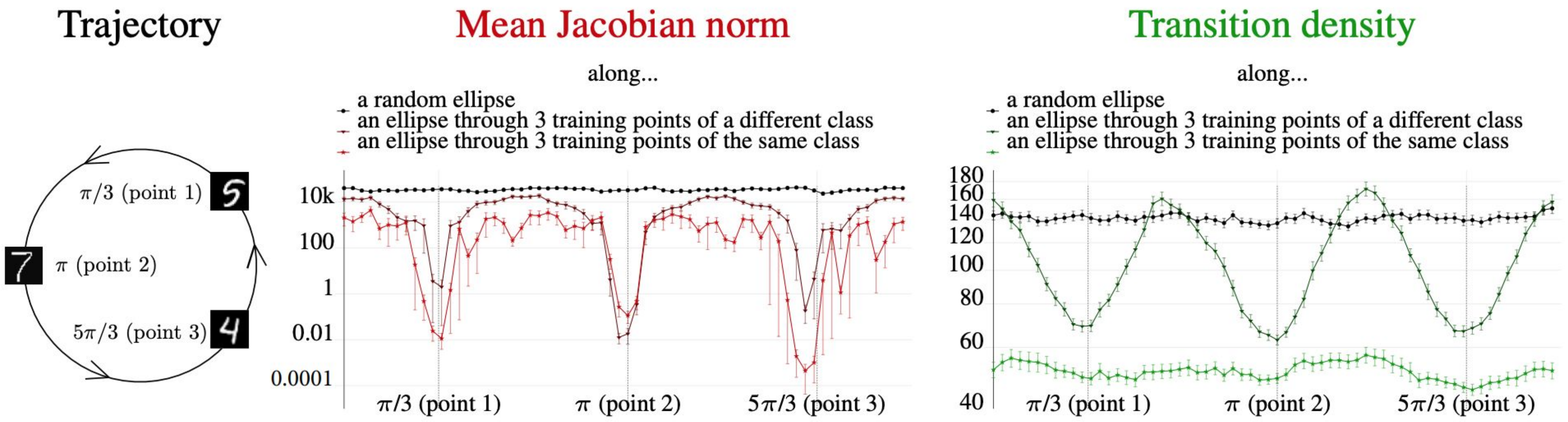
$$t(\mathbf{x}) := \sum_{i=0}^{k-1} \left\| \mathbf{c}\left(\mathbf{z}_i\right) - \mathbf{c}\left(\mathbf{z}_{(i+1)\%k}\right) \right\|_1$$

Sensitivity & Data Manifold

We analyze the behavior of a trained neural network near and away from training data. We do this by comparing sensitivity of the function along 3 types of trajectories:

1. A random ellipse. This trajectory is extremely unlikely to pass anywhere near the real data, and indicates how the function behaves in random locations of the input space that it never encountered during training.
2. An ellipse passing through three training points of different class (Figure 2, left). This trajectory does pass through the three data points, but in between it traverses images that are linear combinations of different-class images, and are expected to lie outside of the natural image space. Sensitivity of the function along this trajectory allows comparison of its behavior on and off the data manifold, as it approaches and moves away from the three anchor points.
3. An ellipse through three training points of the same class. This trajectory is similar to the previous one, but, given the dataset used in the experiment (MNIST), is expected to traverse overall closer to the data manifold, since linear combinations of the same digit are more likely to resemble a realistic image. Comparing transition density along this trajectory to the one through points of different classes allows further assessment of how sensitivity changes in response to approaching the data manifold.

Sensitivity & Data Manifold (2)



According to both metrics, as the input moves between points of different classes, the function becomes less stable than when it moves between points of the same class.

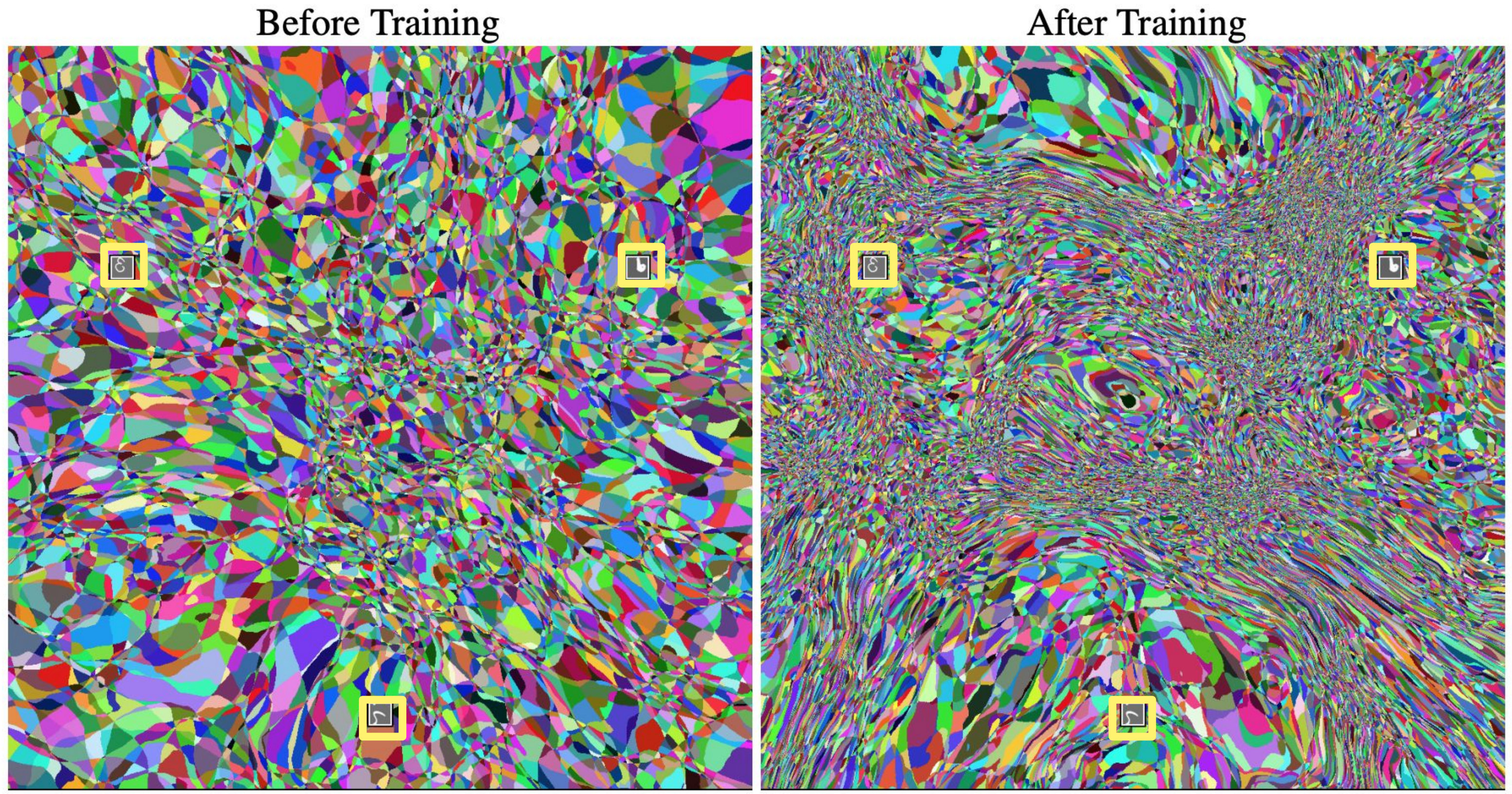
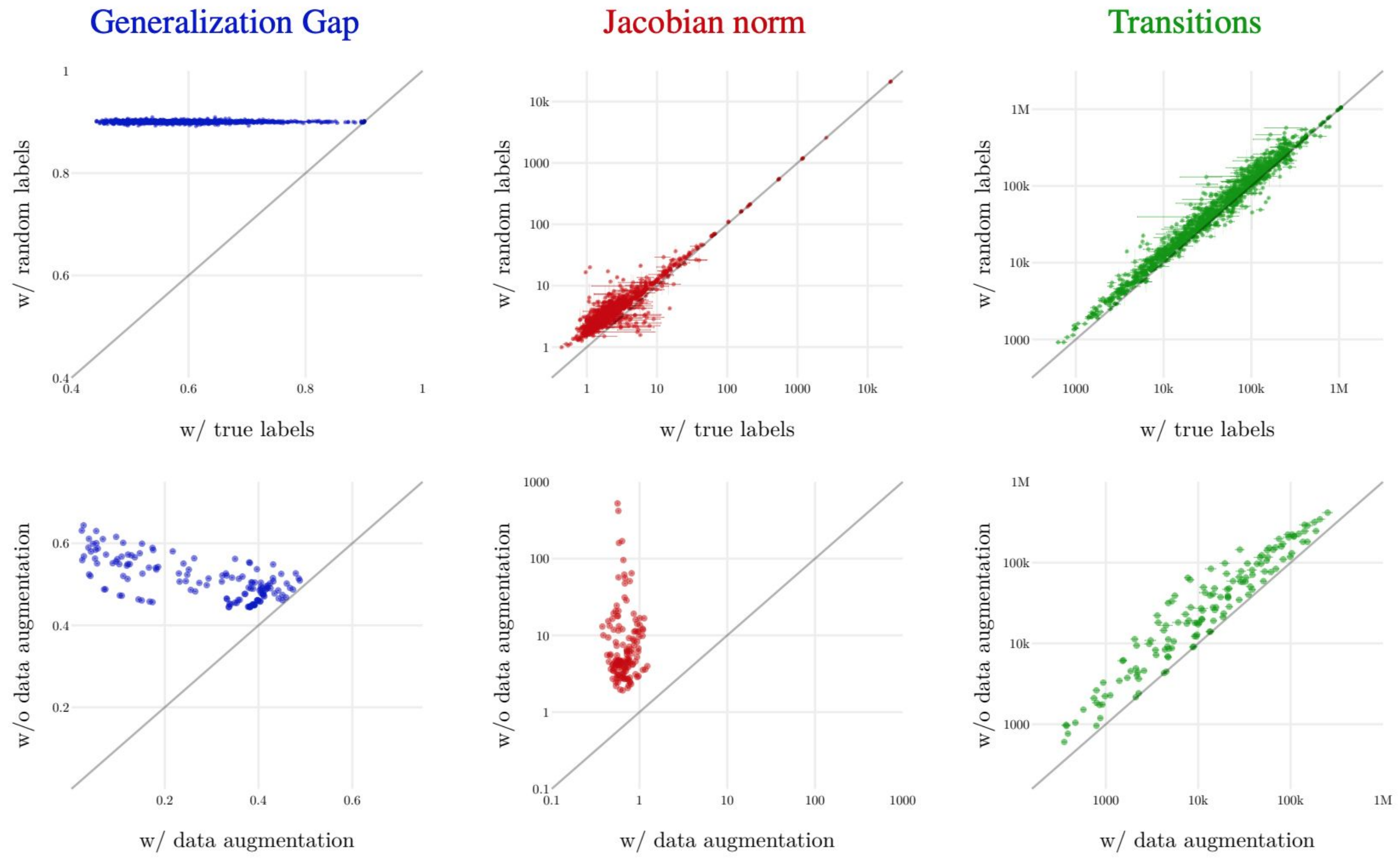


Figure 3: Transition boundaries of the last (pre-logits) layer over a 2-dimensional slice through the input space defined by 3 training points (indicated by inset squares). **Left**: boundaries before training. **Right**: after training, transition boundaries become highly non-isotropic, with training points lying in regions of lower transition density. See §A.5.3 for experimental details.

Sensitivity & Generalization



Conclusion

- Sensitivity of trained neural networks through the input-output Jacobian norm and linear regions counting in the context of image classification tasks.

- Experimental evidence indicating that the local geometry of the trained function as captured by the input-output Jacobian can be predictive of generalization in many different contexts, and that it varies drastically depending on how close to the training data manifold the function is evaluated.

Thank You!

Contact: seyediman.mirzadeh@wsu.edu