Correlation of the Jacobian and Probability Flow

John Y. Shin
Department of Computer Science
New York University
New York, NY 10033
jys308@nyu.edu

Abstract

1 Introduction

There has been past work on showing that the smoothness of the Hessian of the loss function as a function of the weights of the final trained model is correlated with good generalization [1, 2]. Recently, there has been work on leveraging stochastic methods for the computation of the empirical spectral density of the Hessian [3, 4], and further studies of the spectrum of the Hessian have brought some contention to this claim [5, 6, 7].

[3, 5, 6, 2, 1, 7, 4]

The Jacobian of the neural network, or the derivative of the neural network function with respect to either the data (input/output map) or the weights, has also been studied extensively, covering a wide range of topics, such as it's initialization [8, 9], as a measure of generalization [10, 11, 12, 13, 13, 14] as a way to regularize the network [15, 16], it's spectrum [8, 9, 17, 18, 19, 20], as a learning objective itself [21], as a measure of robustness [22], as well as it's connections to information geometry [23].

[10, 8, 11, 9, 15, 17, 12, 22, 16, 13, 23, 21, 13, 18, 14, 19, 20]

2 Computing the Jacobian

Suppose we have a Neural Network classifier $f : \mathbb{R}^d \to \mathbb{R}^k$, where d is the dimensionality of the input and k is the number of classes. We define the Jacobian matrix of the Neural Network as:

$$J(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x}) \in \mathbb{R}^k \times \mathbb{R}^d \tag{1}$$

Or the derivative of the Neural Network with respect to the input. Suppose we have a training dataset of size n, with examples given as $(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$, where $\mathbf{x}_i \in \mathbb{R}^d$. We can concatenate the $J(\mathbf{x})$ for each example into a vector:

$$\mathcal{J}_i = \text{flatten}(J(\mathbf{x}_i)) \in \mathbb{R}^{k \times d} \tag{2}$$

We can then form a matrix of these $\mathcal{J}_i$'s into a matrix $\mathcal{J} \in \mathbb{R}^n \times \mathbb{R}^{k \times d}$, which we will call the Jacobian matrix of the training dataset. Each row corresponds to a training example and the columns are a concatenation of the derivatives of each dimension of the output with each dimension of the input.

As a classifier trained with a cross-entropy loss and with a softmax as the final output of the network, we can interpret the output of the network as a vector of probabilities:

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

$$f(\mathbf{x}) = (p(y_1|\mathbf{x}), p(y_2|\mathbf{x}), \cdots, p(y_k|\mathbf{x})) \in \mathbb{R}^k \quad (3)$$

The Jacobian matrix of the neural network then becomes:

$$J(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x}) = (\nabla_{\mathbf{x}} p(y_1|\mathbf{x}), \nabla_{\mathbf{x}} p(y_2|x), \cdots, \nabla_{\mathbf{x}} p(y_k|\mathbf{x})) \in \mathbb{R}^k \times \mathbb{R}^d \quad (4)$$

For conciseness (and abuse) of notation, we can write the the flattened $\mathcal{J}_i$ in vector notation as:

$$\mathcal{J}_i = \nabla_{\mathbf{x}_i} p(\mathbf{y}|\mathbf{x}_i) \in \mathbb{R}^{k \times d} \quad (5)$$

We can then build the correlation matrix of the Jacobian over the training dataset as:

$$M_{ij} = (\mathcal{J}\mathcal{J}^T)_{ij} = \nabla_{\mathbf{x}_i} p(\mathbf{y}|\mathbf{x}_i) \cdot \nabla_{\mathbf{x}_j} p(\mathbf{y}|\mathbf{x}_j) \in \mathbb{R}^{n \times n} \quad (6)$$

Theorem 1. *testing*

References

[1] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[2] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[3] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. *arXiv preprint arXiv:1901.10159*, 2019.

[4] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael Mahoney. Pyhessian: Neural networks through the lens of the hessian. *arXiv preprint arXiv:1912.07145*, 2019.

[5] Micah Goldblum, Jonas Geiping, Avi Schwarzschild, Michael Moeller, and Tom Goldstein. Truth or backpropaganda? an empirical investigation of deep learning theory. *arXiv preprint arXiv:1910.00359*, 2019.

[6] Wesley J Maddox, Gregory Benton, and Andrew Gordon Wilson. Rethinking parameter counting in deep models: Effective dimensionality revisited. *arXiv preprint arXiv:2003.02139*, 2020.

[7] Diego Granziol. Flatness is a false friend. *arXiv preprint arXiv:2006.09091*, 2020.

[8] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Advances in neural information processing systems*, pages 4785–4795, 2017.

[9] Jeffrey Pennington, Samuel S Schoenholz, and Surya Ganguli. The emergence of spectral universality in deep networks. *arXiv preprint arXiv:1802.09979*, 2018.

[10] Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280, 2017.

[11] Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

[12] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

[13] Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. Generalization, adaptation and low-rank representation in neural networks. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 581–585. IEEE, 2019.

[14] Amartya Sanyal, Philip HS Torr, and Puneet K Dokania. Stable rank normalization for improved generalization in neural networks and gans. *arXiv preprint arXiv:1906.04659*, 2019.

[15] Fredrik Bagge Carlson, Rolf Johansson, and Anders Robertsson. Tangent-space regularization for neural-network models of dynamical systems. *arXiv*, pages arXiv–1806, 2018.

[16] Judy Hoffman, Daniel A Roberts, and Sho Yaida. Robust learning with jacobian regularization. *arXiv preprint arXiv:1908.02729*, 2019.

[17] Zenan Ling and Robert C Qiu. Spectrum concentration in deep residual learning: a free probability approach. *IEEE Access*, 7:105212–105223, 2019.

[18] Matthew Shunshi Zhang and Bradly Stadie. One-shot pruning of recurrent neural networks by jacobian spectrum evaluation. *arXiv preprint arXiv:1912.00120*, 2019.

[19] Wojciech Tarnowski, Piotr Warchoł, Stanisław Jastrzbski, Jacek Tabor, and Maciej Nowak. Dynamical isometry is achieved in residual networks in a universal way for any activation function. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2221–2230, 2019.

[20] Shengjie Wang, Abdel-rahman Mohamed, Rich Caruana, Jeff Bilmes, Matthai Plilipose, Matthew Richardson, Krzysztof Geras, Gregor Urban, and Ozlem Aslan. Analysis of deep neural networks with extended data jacobian matrix. In *International Conference on Machine Learning*, pages 718–726, 2016.

[21] Jonathan Lorraine and Safwan Hossain. Jacnet: Learning functions with structured jacobians.

[22] Fuxun Yu, Chenchen Liu, Yanzhi Wang, Liang Zhao, and Xiang Chen. Interpreting adversarial robustness: A view from decision surface in input space. *arXiv preprint arXiv:1810.00144*, 2018.

[23] Piotr A Sokol and Il Memming Park. Information geometry of orthogonal initializations and training. *arXiv preprint arXiv:1810.03785*, 2018.