Deep Learning for advanced practitioners

01. Optimization

Introduction

- Use the ***latest theory*** to provide ***practical insights*** on the three ***pillars of deep learning***

  - **[*Optimization*]** Analysis of the root causes of training inefficiencies + tricks to improve convergence speed.

  - **[*Generalization*]** What is the state of our understanding on generalization in DL, and how to use these insights?

  - **[*Computation*]** Introduction to High Performance Computing for Deep Learning. How to troubleshoot compute bottlenecks?

Introduction

- This tutorial is meant for advanced practitioners. I assume a good understanding of the following topics:

  - ***Building blocks of DL***: Convolution, Batch Normalization, Residual connections, Pooling, etc.

  - ***Math notations*** for gradients, loss function, etc. and a working knowledge of ***Python***, Numpy, PyTorch (opt.)

  - ***Taylor Series expansion***

Introduction

- **[Slides]** Give an overview of the presentation *[10 min.]*
    - Clear picture of the structure of the argument

- **[Notebook]** Detail every step of the argument *[1h20]*
    - Use math, programming and visualization
    - Interactive environment

- **[Google Form]** I'd be grateful if you take a minute to give me feedback.

Introduction

- Main question: Why is it hard to train a neural network?
  - Identify ***the main source of instability*** of optimization
- Side questions:
  - Why does ***Batch Normalization*** work?
  - Why do ***Residual Connections*** work?
  - Why do ***Meta-Init*** and ***Data-Driven*** initialization work?

Deep Learning for advanced practitioners

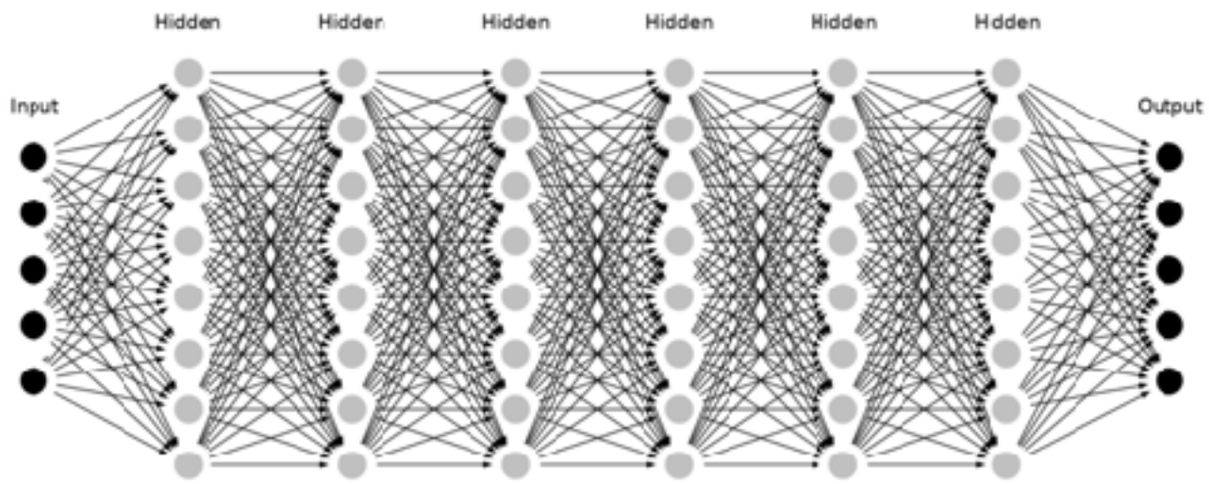
01. Optimization

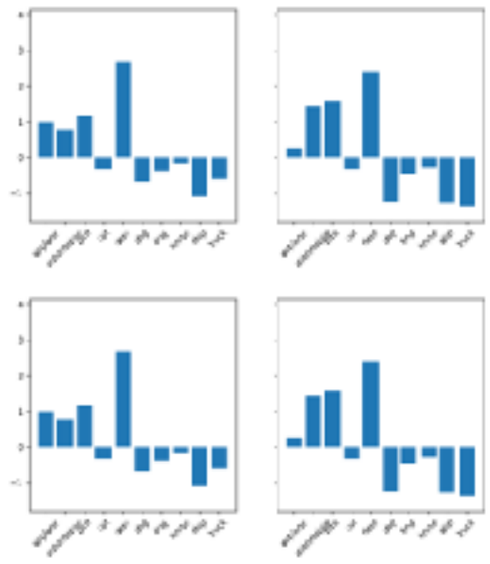
Structure

1. **[Observation]** ReLU networks collapse to constant functions
2. **[Reasons]** Gradual loss of intra-neuron variance
3. **[Consequences]** Optimization is hard
    1. Optimization is a fight between first and second order effects
    2. First order (Gradients) vanish with the loss of variance
    3. Second order (Hessian) explode with the loss of variance
4. **[Solutions]** Every breakthrough actually fight this phenomenon
    1. Batch normalization
    2. Residual Networks

Observation

At initialization, deep ReLU nets collapse to constant functions





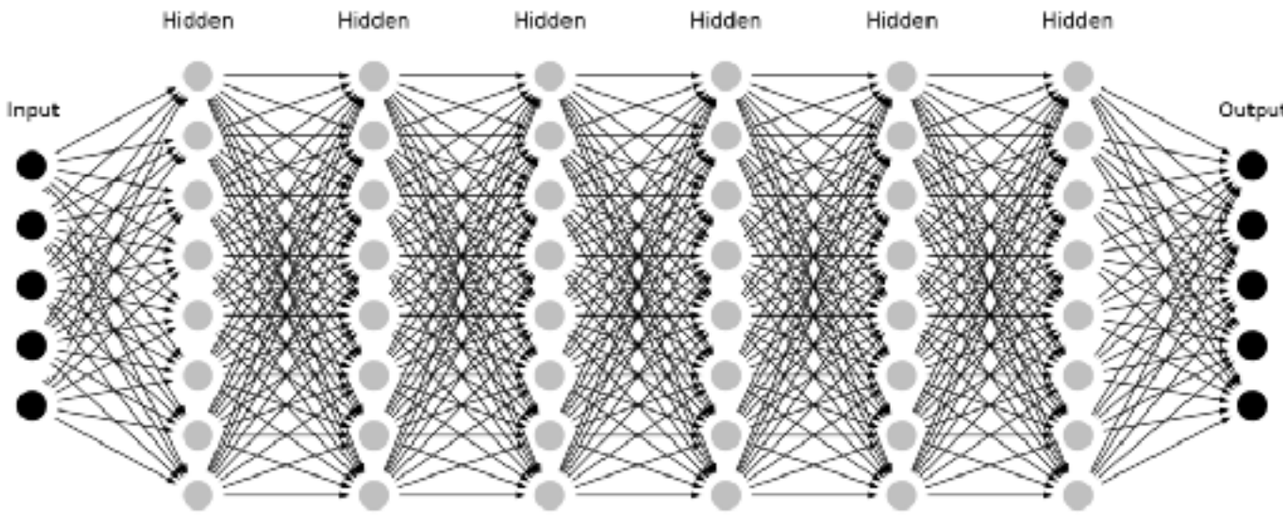
For all input x

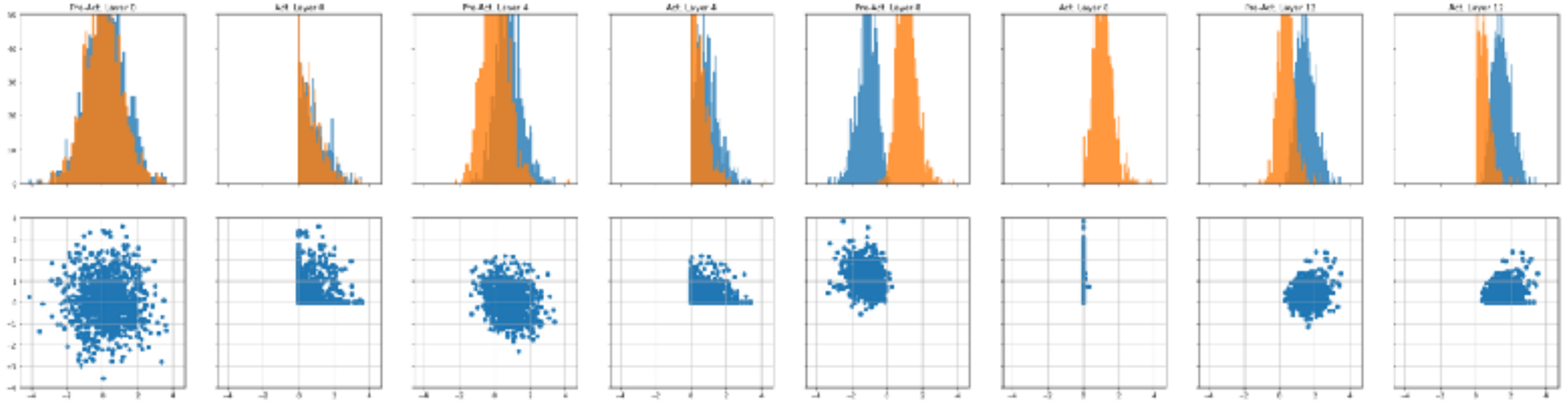
$f(x)=C$

the model output y is the same

Observation

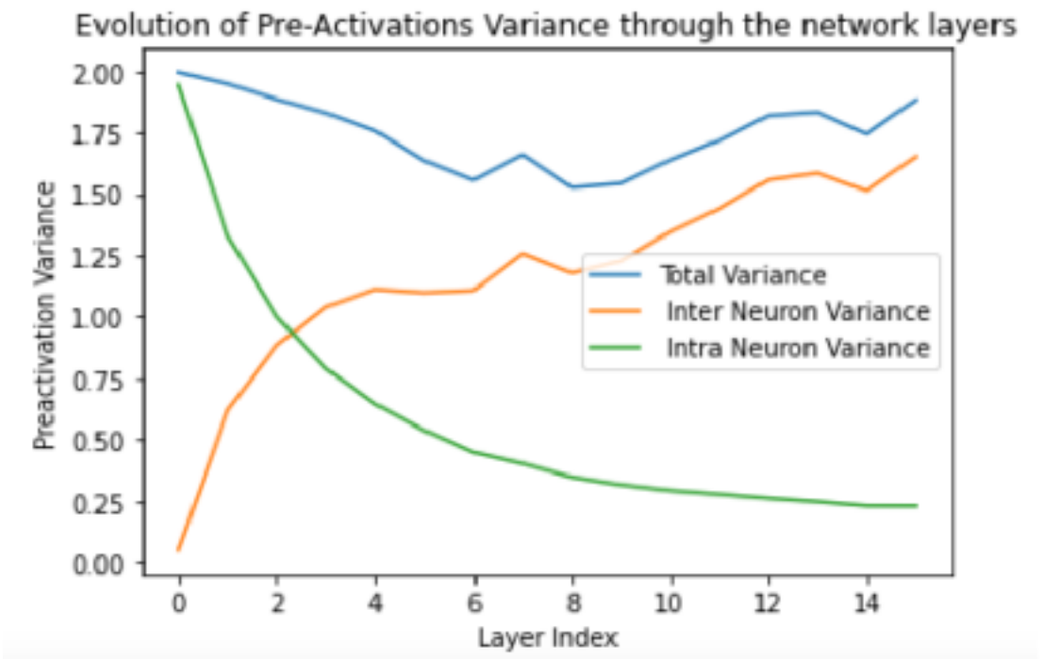
This phenomenon happens gradually through the layers





Reason

This happens because “fake variance” is injected at each layer in order to compensate the loss of variance from ReLU



The decrease in “meaningful variance” can be theoretically computed for infinite width networks (mean field theory)

Consequence

Optimization is a fight between first order and second order effects

$$\Delta\mathcal{L} = \frac{\delta\mathcal{L}(\theta_t)}{\delta\theta_t}\Delta\theta + \Delta\theta H \Delta\theta$$

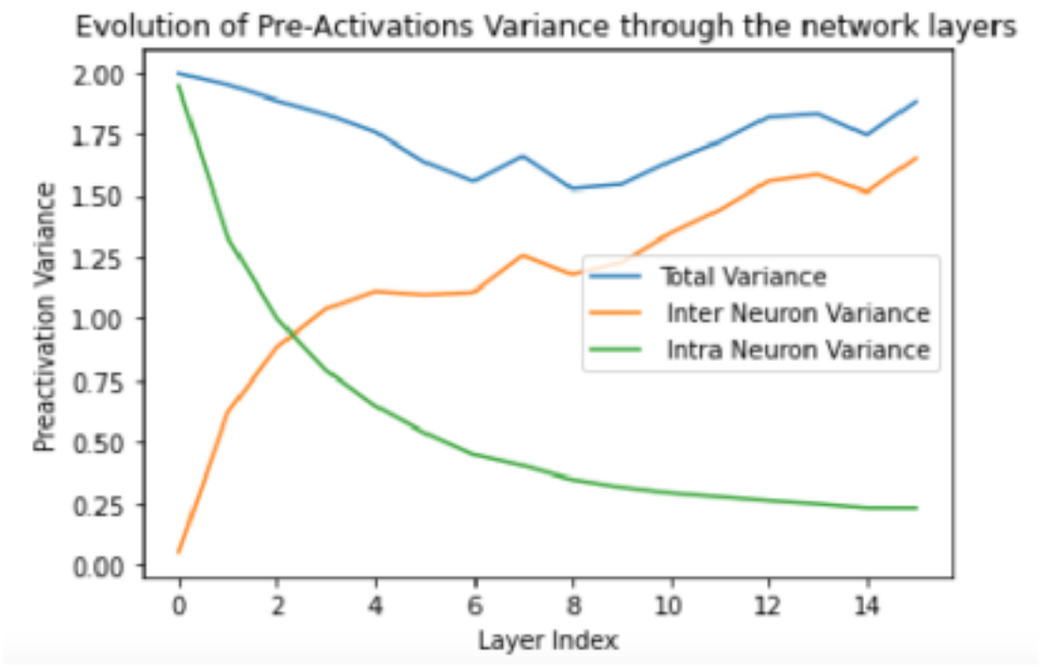
Decreases the loss

Increases the loss

Consequence

The “meaningful variance” loss results in vanishing gradients

The “fake variance” increase results in exploding Hessian



$$\Delta\mathcal{L} = \frac{\delta\mathcal{L}(\theta_t)}{\delta\theta_t}\Delta\theta + \Delta\theta H \Delta\theta$$

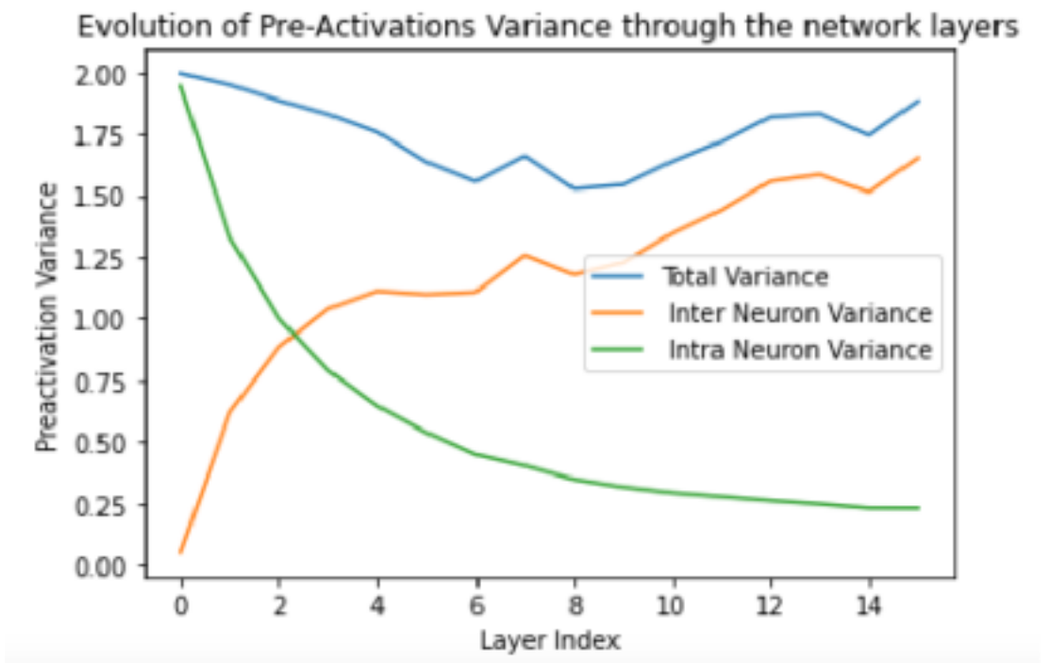
Vanishes ***Explodes***

Hence, convergence becomes both slow or suboptimal

Consequence

We will focus on understanding how the left figure

Creates the situation on the right equation

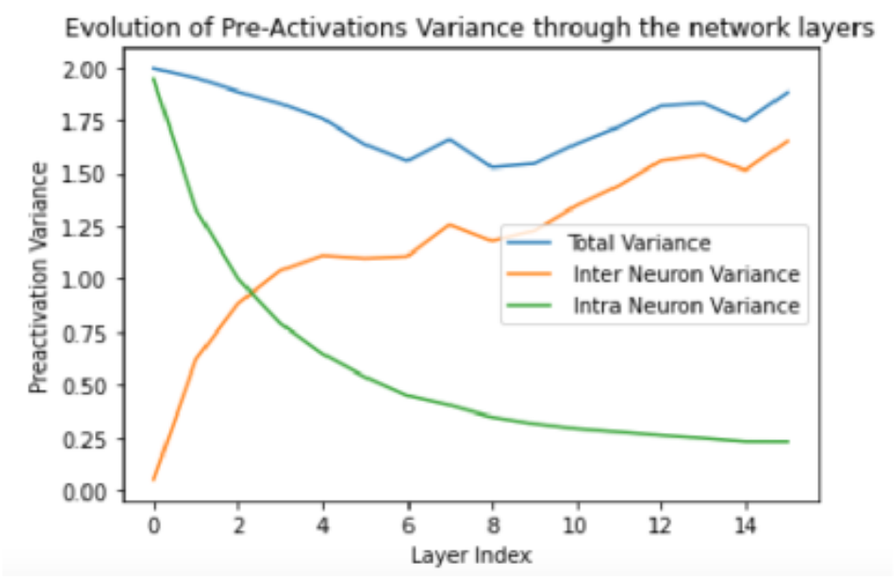


$$\Delta\mathcal{L} = \frac{\delta\mathcal{L}(\theta_t)}{\delta\theta_t}\Delta\theta + \Delta\theta H \Delta\theta$$

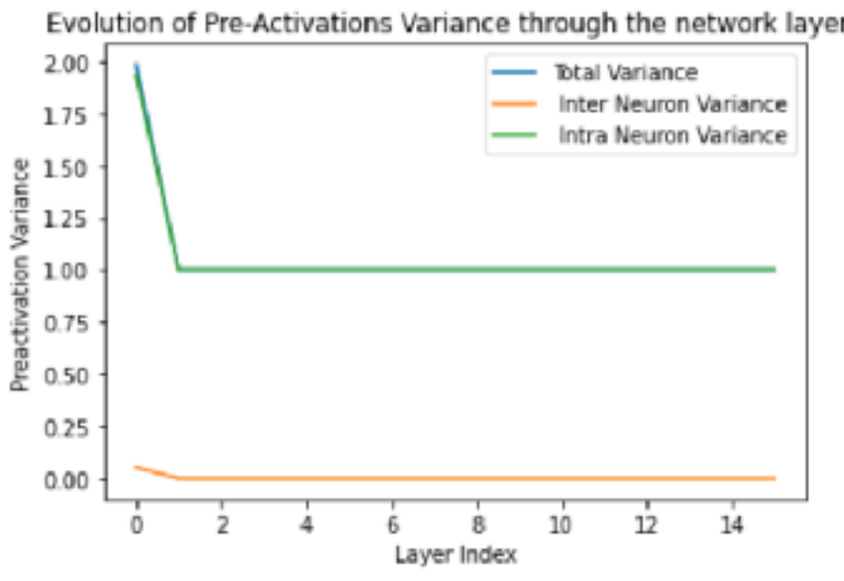
Vanishes ***Explodes***

Solutions

Batch Normalization prevents the collapse to constant function



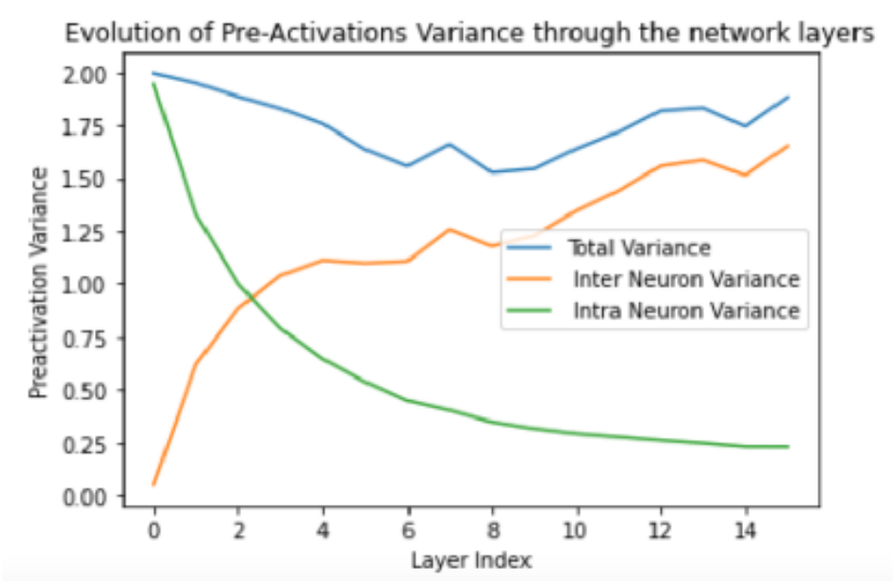
Vanilla ReLU MLP



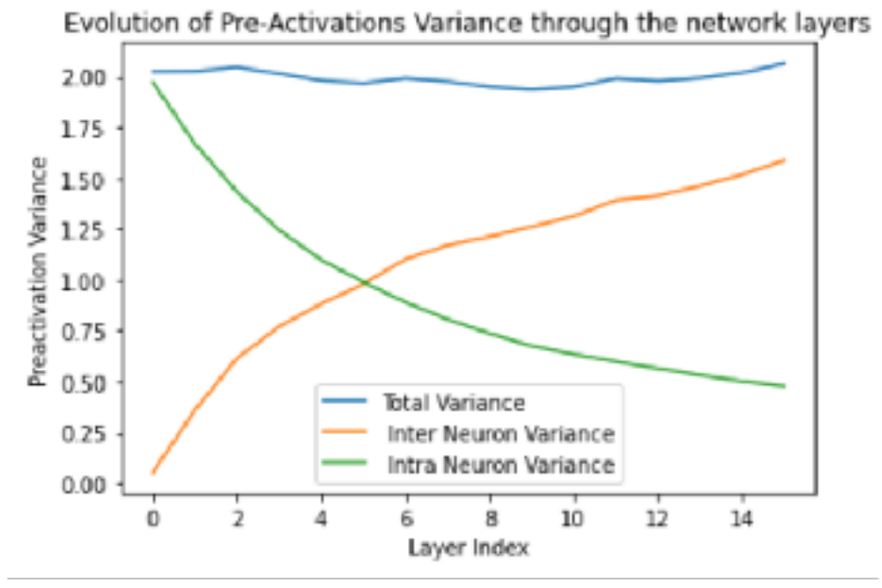
ReLU MLP with BatchNorm

Solutions

Residual connections slow down the collapse to constant function



Vanilla ReLU MLP



ReLU MLP with Residual Connections

Overview

1. **[Observation]** ReLU MLP collapse to constant functions
2. **[Reasons]** Gradual loss of intra-neuron variance
3. **[Consequences]** Optimization is hard
    1. Optimization is a fight between first and second order effects
    2. First order (Gradients) vanish with the loss of variance
    3. Second order (Hessian) explode with the loss of variance
4. **[Solutions]** Every breakthrough actually fights this phenomenon
    1. Batch normalization
    2. Residual Networks

Let’s use a Notebook to study this phenomenon