Normalization Techniques in Training DNNs: Methodology, Analysis and Application

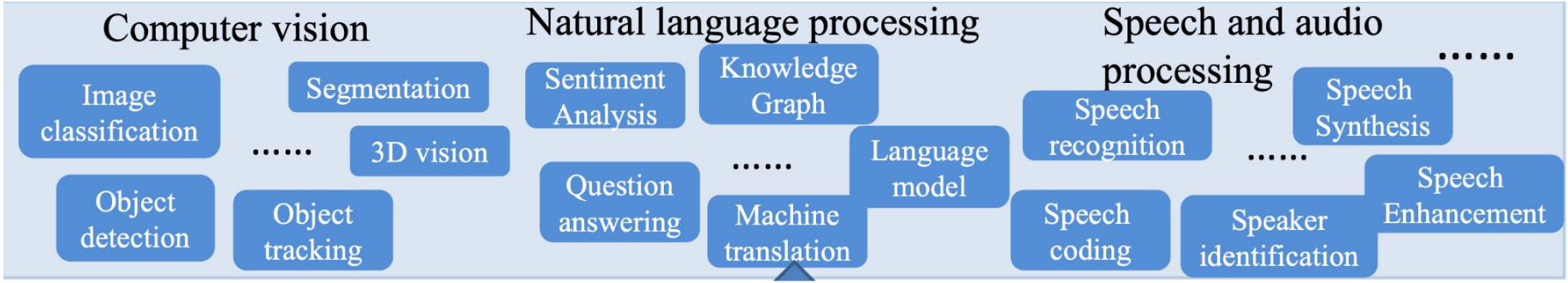
Lei Huang; Jie Qin; Yi Zhou; Fan Zhu; Li Liu; Ling Shao

Presenter: Yaorong Xiao

Based on materials from:: https://normalization-dnn.github.io/

Content

- Motivations of normalization techniques
- Review of normalization methods
- Analysis of normalization

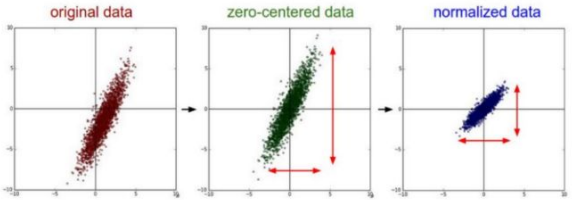


Motivation

- Definition of normalization
  - In statistics: adjustments of values or distributions in statistics
  - In image processing: changing the range of pixel intensity values
  - In data processing: general reduction of data to canonical form

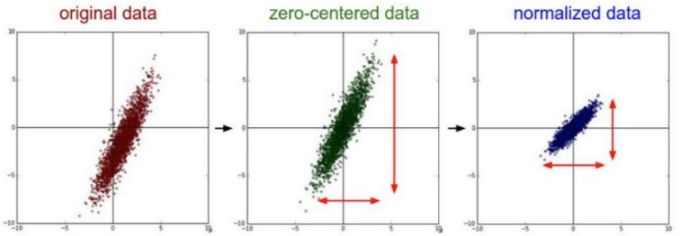
- Definition of normalization in this tutorial
  - Given a set of data $\mathbb{D} = \{\mathrm{x}^{(i)}\}_{i=1}^{N}$, the normalization operation is a function $\Phi: \mathrm{x} \longmapsto \hat{\mathrm{x}}$, which ensures that the transformed data $\widehat{\mathbb{D}} = \{\hat{\mathrm{x}}^{(i)}\}_{i=1}^{N}$ has certain statistical properties.

$$\hat{\mathbf{x}} = \frac{\mathbf{x}-\boldsymbol{\mu}}{\boldsymbol{\sigma}}$$



Motivation of normalizing input

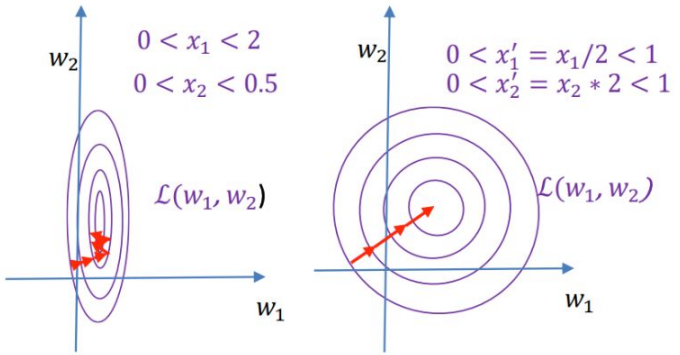
- Improve the effects of learning
  - Non-parameter models (KNN, Kernel SVM)
  - Distance/ Similarity



- Improve optimization efficiency
  - Parametric model (logistic regression)
  - Update parameters iteratively

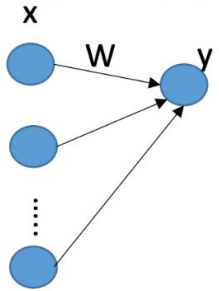
$$\theta_{t+1} \leftarrow \theta_t - \eta \frac{\partial \mathcal{L}}{\partial \theta_t}$$

$$y = w_1 x_1 + w_2 x_2 + b,\ \mathcal{L} = (y - \hat{y})^2$$
$$\theta = \{w_1, w_2\}$$

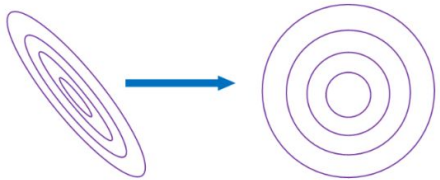


Motivation of optimization

- Learning dynamics are controlled by the spectrum of curvature matrix (Hessian $\boldsymbol{H}$)
  - $\lambda_{max}(\boldsymbol{H})$:
    - Optimal learning rate: $\eta = \frac{1}{\lambda_{max}(\boldsymbol{H})}$
    - Diverge if $\eta > \frac{2}{\lambda_{max}(\boldsymbol{H})}$:
  - Condition number $\kappa = \frac{\lambda_{max}(\boldsymbol{H})}{\lambda_{min}(\boldsymbol{H})}$ control the iterations required for convergence

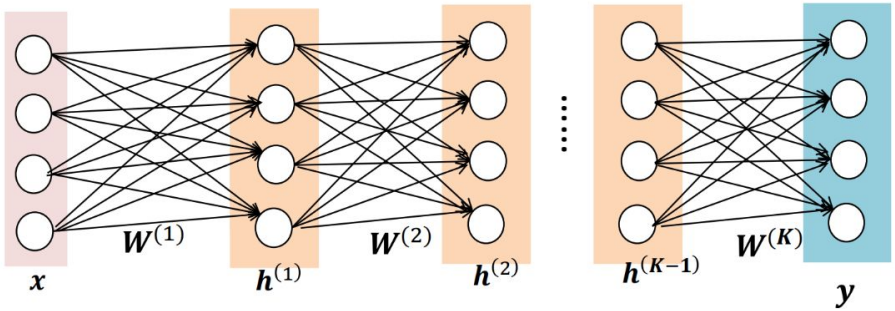


- Hessian of multiple output:
  $\boldsymbol{H} = \mathbb{E}_{\mathcal{D}}(\boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}}) \otimes \boldsymbol{I}$



Motivation of activation normalization

- Difficulty of analysis for DNNs
  - Nonlinear model
  - X is only linearly connected by $W^{(1)}$; Optimization is over θ, not $W^{(1)}$ only



$$\theta = \{W^{(1)}, W^{(2)}, \dots, W^{(K)}\}$$

- What we can exploit?
  - Layer-wise structure
  - $h^{(i)}$ is linearly connected by $W^{(i+1)}$

→ Normalizing Activations

Well-conditioned landscape

- Denoting $\Sigma_x = \mathbb{E}_{p(x)}\,(xx^T)$ and $\Sigma_{\nabla h} = \mathbb{E}_{p(x),q(y|x)}(\frac{\partial \ell}{\partial h}^T \frac{\partial \ell}{\partial h})$
- Criteria
  - 1. The statistics of the layer input (e.g., Σ_x) and output-gradient ($\Sigma_{\nabla h}$) across different layers are equal (across layer)
  - 2. Σ_x and $\Sigma_{\nabla h}$ are well conditioned (in layer)
- Initialization techniques: designed to satisfy Criteria 1 and/or 2 during initialization
  - Arxiv-Init [Glorot and Bengio, 2010], He-Init [He et al, 2015]: for Criteria 1
  - Orthogonal Initialization [Saxe et al, 2014] : for Criteria 1 and 2
- General goals of "normalization" in DNNs: Controlling the distribution of the activations/output-gradients during training.

Review of Normalization Methods

- Normalizing activations
- Normalizing weights
- Normalizing gradients

Population Normalizing

- Centering the activation
  - Montavon et al, 2014; Wiesler et al, 2014

$$\hat{\mathbf{x}} = \mathbf{x} - \hat{\boldsymbol{\mu}}$$

$\hat{\boldsymbol{\mu}}$ is the mean of activation over the training dataset;
Parameter to be estimated

- Standardizing the activation: centering + scaling
  - Wiesler et al, 2014

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \hat{\boldsymbol{\mu}}}{\hat{\sigma}}$$

$\hat{\sigma}$ is the standard deviation of activation over the training dataset;
Parameter to be estimated

- Whitening the activations
  - Desjardins et al 2015; Luo, 2017

$$\hat{x_I} = \hat{\Sigma}^{-\frac{1}{2}}(\mathbf{x} - \hat{\boldsymbol{\mu}})$$

$\hat{\Sigma}^{-\frac{1}{2}}$ is the whitening matrix of activation over the training dataset;
Parameter to be estimated

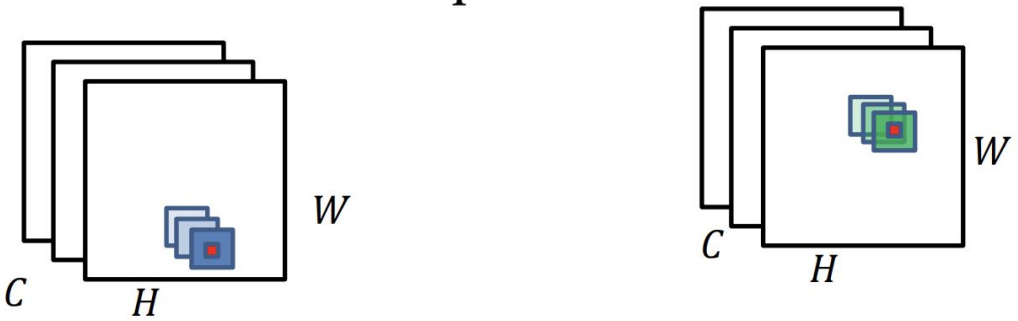
- Advantages
  - Well exploit the beneficial property of normalization in optimization
- Drawbacks
  - Training instability
    - The estimation is not accurate (sampled data)
    - Internal covariant shift (the distribution of activation varying with training progressing)
  - Can not be used to large networks
    - An inaccurate estimation of population statistics will be amplified as the layers increase

As function

Local normalization

- Local contrast normalization [Jarrett et al, ICCV 2009]
- Local response normalization [Krizhevsky et al, NeurIPS 2012]
- Divisive normalization [Ren et al, ICLR 2017]
- Local context normalization [Ortiz et al, CVPR 2020]

Given an example $X \in \mathbb{R}^{C \times H \times M}$



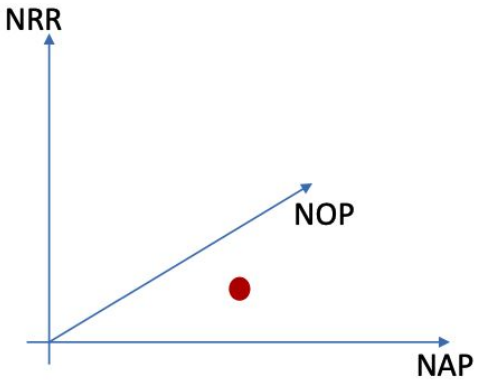
As function

- Advantage
  - Training is somewhat stable due to back-propagating through normalization
  - The visual contrast invariant property may benefit generalization

- Limits
  - Specific to visual data (feature maps)
  - May change the representation ability and reduce the discriminative information
  - It is not clear whether benefits optimization

A framework for decomposing normalization

- The framework
  - Normalization Area Partitioning (NAP): which area to calculate the 'statistics'
  - Normalization Operation (NOP): what kind of normalization operation?
  - Normalization Representation Recovery (NRR)

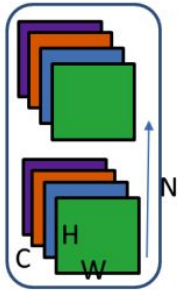


Algorithm 1 Framework of algorithms normalizing activations as functions.

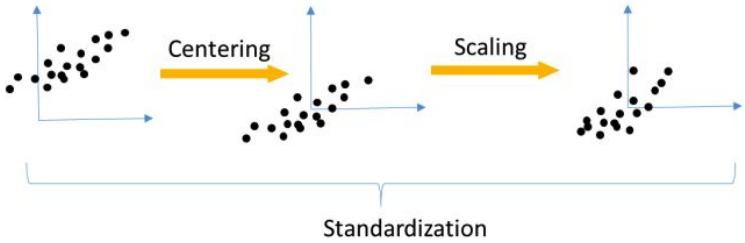
1: **Input**: mini-batch inputs $\mathbf{X} \in \mathbb{R}^{d \times m \times h \times w}$.
2: **Output**: $\widetilde{\mathbf{X}} \in \mathbb{R}^{d \times m \times h \times w}$.
3: Normalization area partitioning: $\boldsymbol{X} = \Pi(\mathbf{X})$.
4: Normalization operation: $\widehat{\boldsymbol{X}} = \Phi(\boldsymbol{X})$.
5: Normalization representation recovery: $\widetilde{\boldsymbol{X}} = \Psi(\widehat{\boldsymbol{X}})$.
6: Reshape back: $\widetilde{\mathbf{X}} = \Pi^{-1}(\widetilde{\boldsymbol{X}})$.

- Batch Normalization
  - NAP: $X = \Pi_{BN}(\mathbf{X}) \in \mathbb{R}^{d \times mhw}$.

$\mathbf{X} \in \mathcal{R}^{N \times C \times H \times W}$



  - NOP: $\widehat{X} = \Phi_{SD}(X) = \Lambda^{-\frac{1}{2}}(X - \mathbf{u}\mathbf{1}^T)$.



  - NRR: $\widetilde{X} = \Psi_{AF}(\widehat{X}) = \widehat{X} \odot (\gamma \mathbf{1}^T) + (\beta \mathbf{1}^T)$

NAP

Normalization Method	Computation (How statistics are calculated)	Main Advantage / Typical Use Case
Batch Normalization (BN)	Mean and variance are computed **across the mini-batch** dimension	Stabilizes training and allows a larger learning rate, but depends on batch size
Layer Normalization (LN)	Mean and variance are computed **across all channels within a single sample**	Works well for RNNs and Transformers
Instance Normalization (IN)	Mean and variance are computed **independently for each channel of each sample**	Used in image style transfer and generative tasks
Group Normalization (GN)	Channels in each sample are **divided into groups**, and statistics are computed per group	Alternative to BN when batch size is small
Batch Group Normalization (BGN)	A **combination of BN and GN**	More robust for multi-domain or small-batch training
Position / Region Normalization	Normalizes **locally within spatial regions** of the image	Used in vision tasks such as segmentation and detection

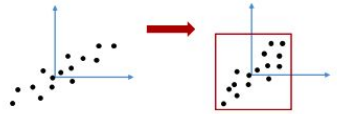
NOP

- Batch Whitening (BW)

Activation distribution

Standardization:

$$\hat{X} = \varphi(X) = \left(diag(\Sigma)\right)^{-\frac{1}{2}}(X - \mu 1^T)$$

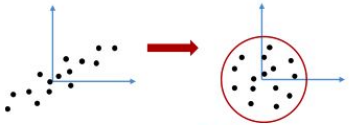


$$diag(\hat{X}\hat{X}^T) = I$$

Covariance matrix

Whitening:

$$\hat{X} = \varphi(X) = \Sigma^{-\frac{1}{2}}(X - \mu 1^T)$$



$$\hat{X}\hat{X}^T = I$$

Standardization is a special case of whitening

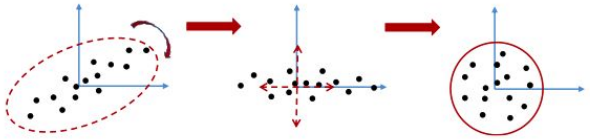
Whitening further improver conditioning over standardization

Decorrelated Batch Normalization [Huang et al, CVPR 2018]

NOP

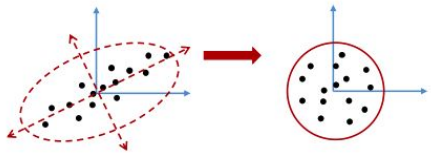
- Whitening
  - PCA whitening not work

$$G_{PCA} = \Lambda^{-\frac{1}{2}} D^T, \qquad D\Lambda D^T = \Sigma$$

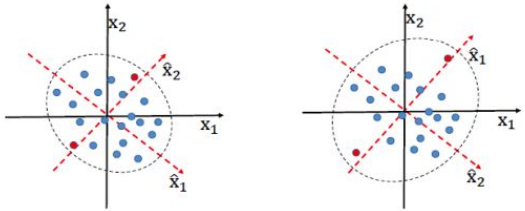


  - ZCA whitening work

$$G_{ZCA} = D\Lambda^{-\frac{1}{2}} D^T$$



- PCA cause stochastic axis swapping



Batch whitening

- Advantages over standardization
  - Better conditioning theoretically
  - Probably better generalization (the amplified stochasticity) by controlling the extent of whitening

- Disadvantages
  - Computational costs ⇒ Group based, Newtown's iteration
  - Numerical instability ⇒ Cholesky decomposition, Newtown's iteration
  - More difficulty in ensuring the training and inference consistency

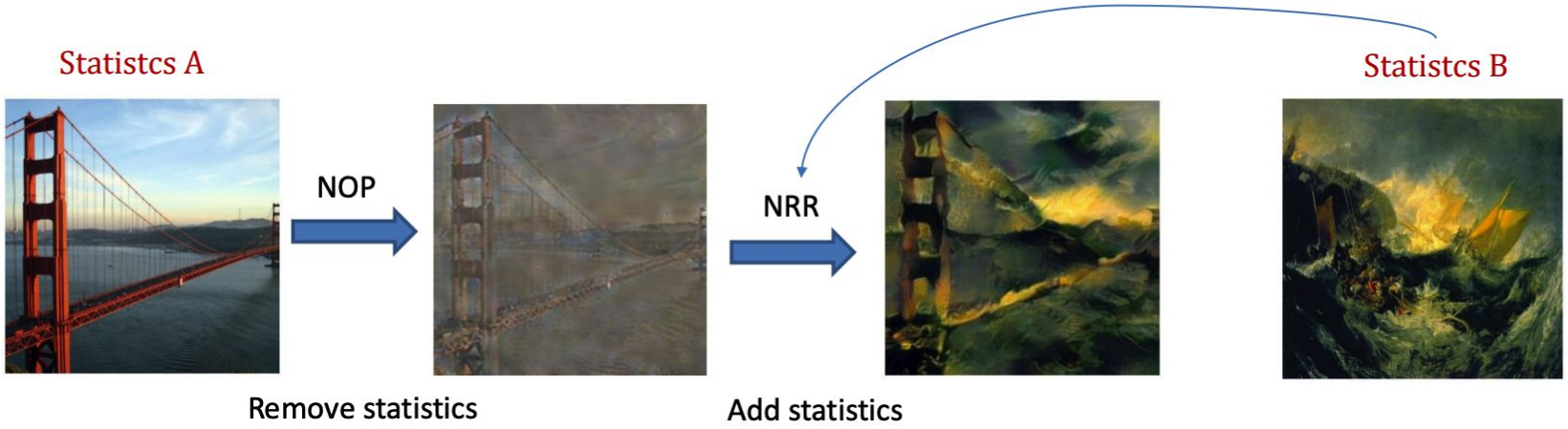
NRR (Normalization Representation Recovery)

Method	Core Idea	Application Scenario
Affine Transformation (γ, β)	Adds learnable **scaling (γ)** and **shifting (β)** parameters	Used in all standard BN / LN implementations
Adaptive Instance Normalization (AdaIN)	Transfers the **statistical features (mean & variance)** of a *style image* to a *content image*	Style transfer
SPADE (Spatially-Adaptive Denormalization)	Dynamically generates normalization parameters **based on semantic maps**	Semantic image synthesis
Attentive Normalization (AN)	Uses **attention mechanisms** to control normalization adaptively	Image classification / detection
Conditional / Dynamic Normalization	Dynamically generates normalization parameters conditioned on **text, task, or context information**	Multi-modal or language-conditioned vision tasks

- Why NRR
  - Recover the representation
  - Edit the statistical distribution

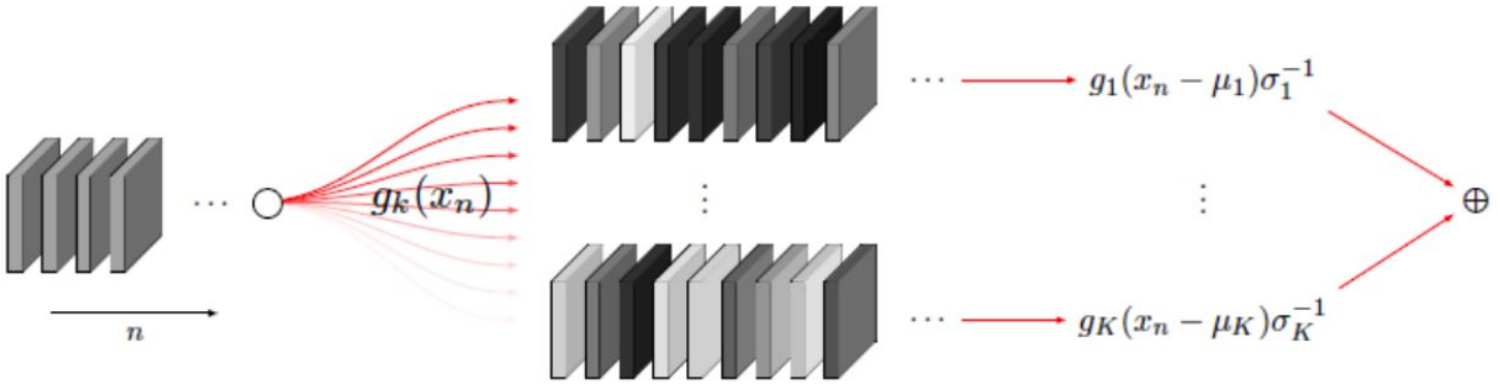
NOP: $\widehat{X} = \Phi_{SD}(X) = \Lambda^{-\frac{1}{2}}(X - \mathbf{u}\mathbf{1}^T)$

NRR: $\widetilde{X} = \Psi_{AF}(\widehat{X}) = \widehat{X} \odot (\gamma \mathbf{1}^T) + (\beta \mathbf{1}^T)$



Multi-mode normalization

$$\mathrm{MN}(x_n) \triangleq \alpha\left(\sum_{k=1}^{K} g_k(x_n)\frac{x_n - \mu_k}{\sigma_k}\right) + \beta$$



- Switchable Normalization (SN)
  - Combing BN, LN and IN

$$\hat{x}_{nchw} = \gamma \frac{x_{nchw} - (w_{IN}\mu_{IN} + w_{BN}\mu_{BN} + w_{LN}\mu_{LN})}{\sqrt{w'_{IN}\sigma^2_{IN} + w'_{BN}\sigma^2_{BN} + w'_{LN}\sigma^2_{LN}}} + \beta$$

$$w_k = \frac{e^{\lambda_k}}{\sum_{z\in\{IN,LN,BN\}} e^{\lambda_z}}, k \in \{IN, LN, BN\}$$

- Switchable Whitening (SW)
  - $k \in \{BW, IW\}$ or $k \in \{BW, IW, IN, LN, BN\}$

Normalizing weights

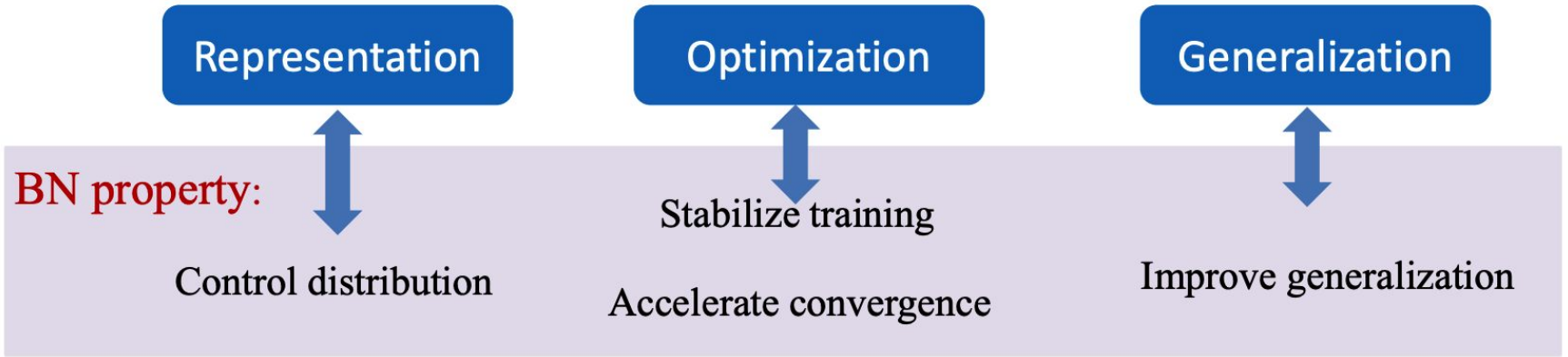
- The general idea: normalize the activation implicitly during training
- Normalization Propagation [Arpit et al 2016; Shekhovtsov et al, 2018]
  - Normalize input: 0-mean and unit variance
  - Assuming **W** is orthogonal

Normalizing gradients

Method	Core Idea	Application Scenario
Gradient Normalization (per layer)	Normalizes the gradient magnitude for each layer to prevent gradient explosion	Deep networks
LARS (Layer-wise Adaptive Rate Scaling)	Adjusts each layer's learning rate adaptively according to its gradient magnitude	Large-batch training (e.g., ImageNet)
LAMB	Integrates LARS into Adam, designed for large-batch training of models like BERT	NLP models
Gradient Centralization	Subtracts the mean from gradients to centralize them	Improves convergence speed
LANS	Adds Nesterov momentum to LAMB	Efficient Transformer training

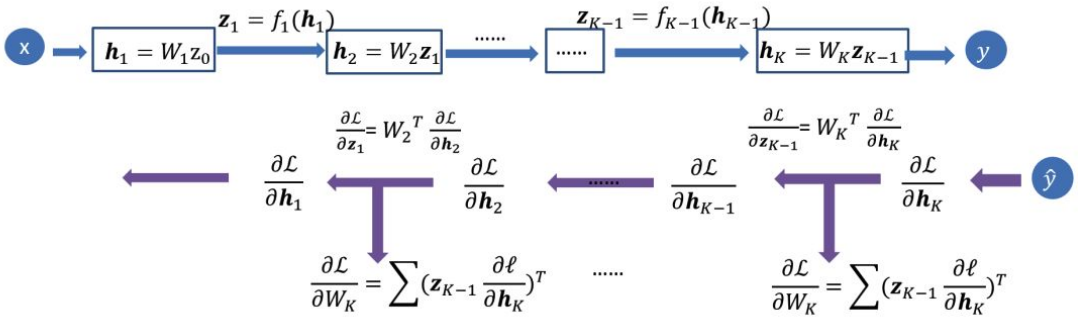
Analysis of normalization

- Topics of deep learning theory



Scale invariant

- Stabilize training in a network
  - Scale dynamics for a network



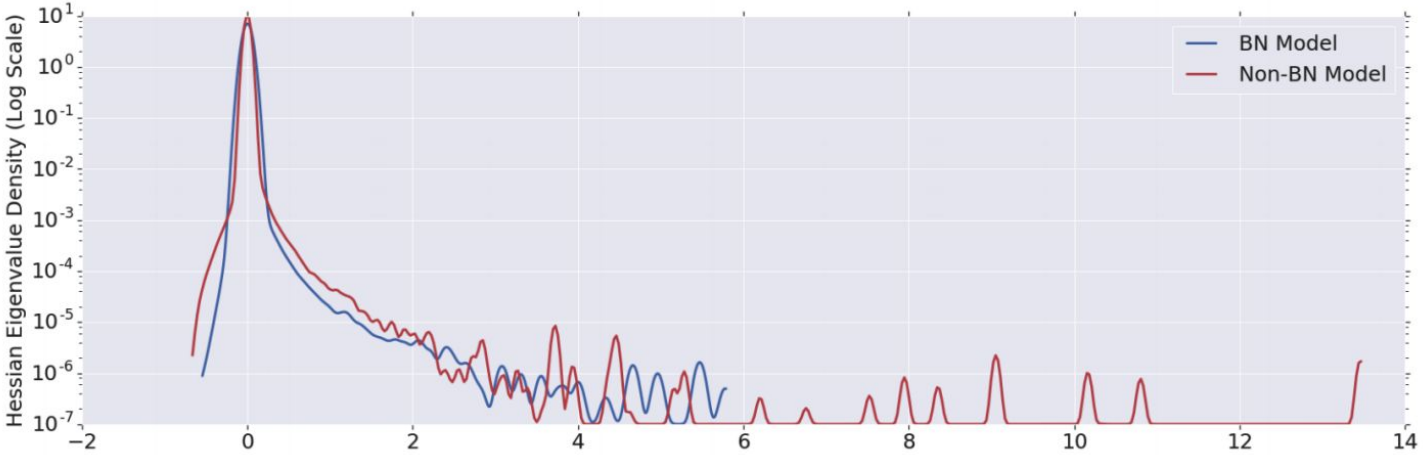
$$\widehat{W}_k = \alpha_k W_k$$

	Layer input	Output-gradient	Weight-gradient
Un-normalized	$\hat{x}_k = (\prod_{i=1}^{k} \alpha_i) x_k$	$\frac{\partial L}{\partial \hat{h}_k} = (\prod_{i=k+1}^{K} \alpha_i) \frac{\partial L}{\partial h_k}$	$\frac{\partial L}{\partial \widehat{W}_k} = (\prod_{i=1, i \neq k}^{K} \alpha_i) \frac{\partial L}{\partial W_k}$
normalized	$\hat{x}_k = x_k$	$\frac{\partial L}{\partial \hat{h}_k} = \frac{1}{\alpha_k} \frac{\partial L}{\partial h_k}$	$\frac{\partial L}{\partial \widehat{W}_k} = \frac{1}{\alpha_k} \frac{\partial L}{\partial W_k}$

Layer-wise Conditioning Analysis in Exploring the Learning Dynamics of DNNs [Huang et al, ECCV 2020]

Improved conditioning

- The full Hessian spectrum



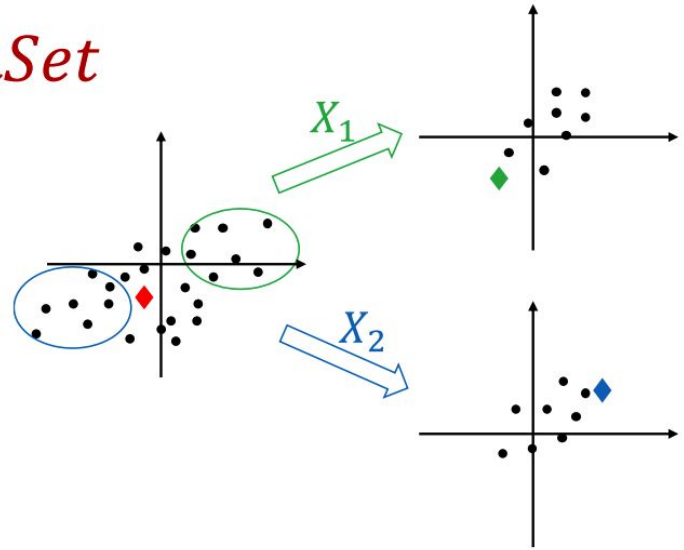
The eigenvalue comparison of the Hessian of the VGG network with BN (blue) and without BN (red)

Stochasticity

- Introduced Stochasticity
  - During training

$$\hat{X}_m = \frac{X_m - \mu_{X_m}}{\sigma_{X_m}}$$

$X_m \sim DataSet$



  - During inference

Population statistics $\{\hat{\mu}, \hat{\sigma}\}$:

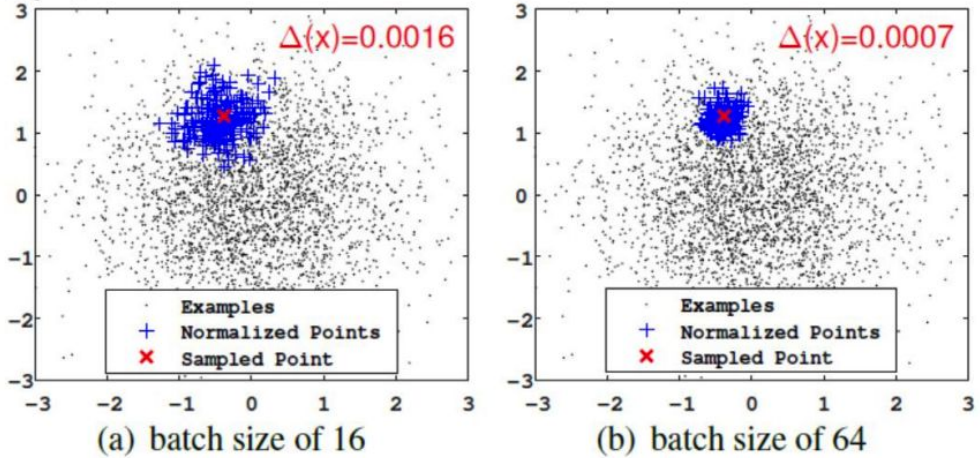
$$\hat{\mu} = (1 - \lambda)\hat{\mu} + \lambda\mu_{X_m}$$

$$\hat{\sigma} = (1 - \lambda)\hat{\sigma} + \lambda\sigma_{X_m}$$

- Stochastic Normalization Disturbance (SND)

Empirical SND: $\widehat{\boldsymbol{\Delta}}_{\mathbf{G}}(\mathbf{x}) = \frac{1}{s}\sum_{i=1}^{s} \|\mathbf{G}(\mathbf{X}_i^B;\mathbf{x}) - \frac{1}{s}\sum_{j=1}^{s}\mathbf{G}(\mathbf{X}_j^B;\mathbf{x})\|$

$\{\mathbf{X}_j^B\}_{j=1}^{s} \sim$ Dataset



(a) batch size of 16 (b) batch size of 64

Questions