Neural Networks and Deep Learning: Modern Convolutional Neural Network Modules

Nicolas Thome

Conservatoire National des Arts et Métiers (Cnam)
Département Informatique

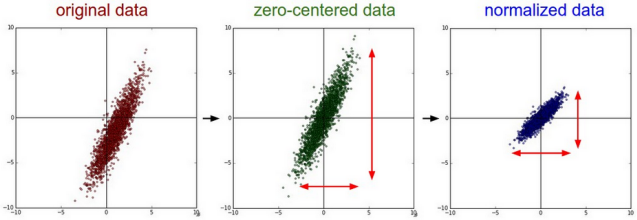
Pre-Processing Modules

- **Normalization bewteen input neurons known to help training**
- **<u>Idea:</u> enforcing fixed distribution**

Data set $\mathbf{X} = \{\mathbf{x}^j\}$, $j \in \{1; N\}$, $\mathbf{x}^j = \left\{x_i^j\right\} \in \mathbb{R}^m$

- Centering: mean subtraction μ_i across every individual feature x_i^j
  - $\mu_i = \frac{1}{N} \sum_{i=1}^{N} x_i^j \Rightarrow x_i^{N,j} = x_i^j - \mu_i$
- Normalization: centering + std division

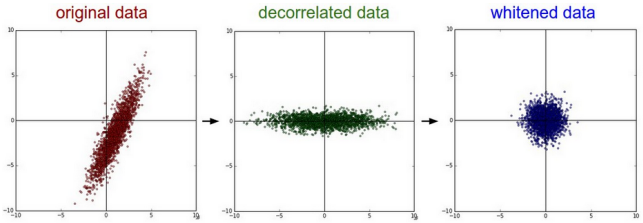
σ_i: $\sigma_i^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i^j - \mu_i)^2 \Rightarrow x_i^{N,j} = \frac{x_i^j - \mu_i}{\sigma_i}$



Pre-Processing Modules

More advanced processings:

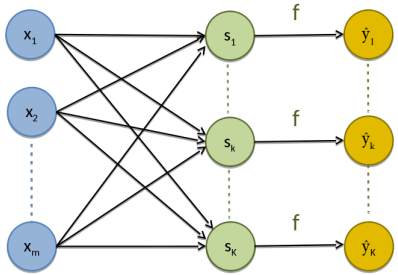
- De-correlation: centering + covariance matrix wrt principal axes alignment
- Whitening: divide by each std $\Rightarrow \mathcal{N}(0,1)$



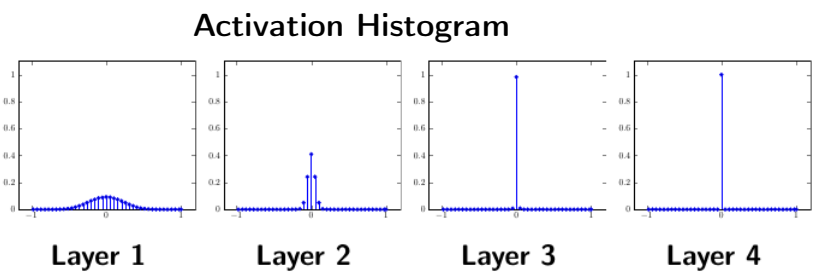
- In practice, not used with ConvNets

Weight Initialization

- Non-convex deep learning objective ⇒ param init important
- ~~**Zero-init:**~~ all neurons same output, thus same gradient
- **Random init will small numbers**, *e.g.* uniform or $\mathbf{W} \sim \mathcal{N}(0, \sigma^i)$
- Input layer $\mathbf{x}$ with m neurons of output s: $Var[s] = mVar[\mathbf{w}]Var[\mathbf{x}]$
  - Xavier init: $\mathbf{W} \sim \frac{1}{\sqrt{m}} \mathcal{N}(0, \sigma^i)$ [Glorot and Bengio, 2010]



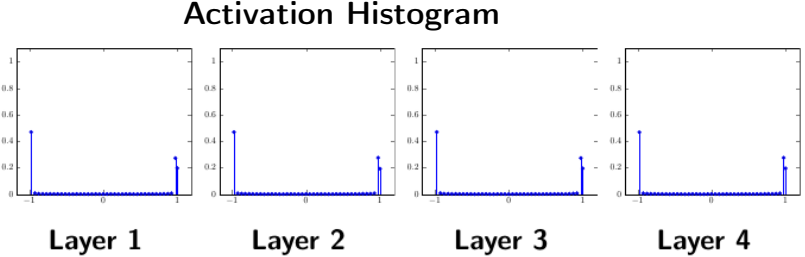
Weight Initialization Example



Credit: Sullivan

- 10-layer net, 500 nodes at each layer
- Tanh activation, parameters init $\mathbf{W} \sim \mathcal{N}(0, \sigma^i)$
- σ^i **small: activation may be** 0**, not good init**

Weight Initialization Example

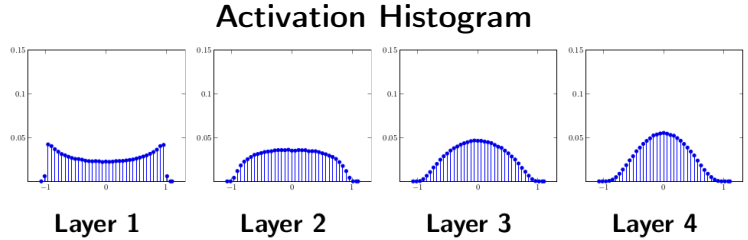


Credit: Sullivan

- 10-layer net, 500 nodes at each layer
- Tanh activation, parameters init $\mathbf{W} \sim \mathcal{N}(0, \sigma^i)$
- σ^i **large: activation may be** ± 1
  $\Rightarrow$ **vanishing gradient**

Weight Initialization Example

- Tanh activation, Xavier init $\mathbf{W} \sim \frac{1}{\sqrt{500}}\mathcal{N}(0,1)$



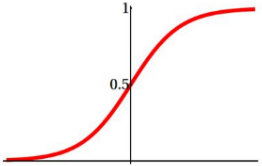
Credit: Sullivan

- **Helps to control activation variance through depth**

Modern Non-Linear Activation Modules

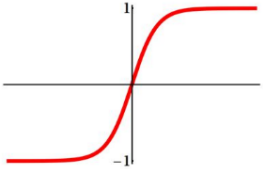
- Standard non-linear activation functions, *e.g.* sigmoid, tanh

$$\sigma(\Sigma) = \frac{1}{1+e^{-\Sigma}}$$



logistic (sigmoid, unipolar)

$$\tanh(\Sigma) = \frac{e^{\Sigma} - e^{-\Sigma}}{e^{\Sigma} + e^{-\Sigma}}$$

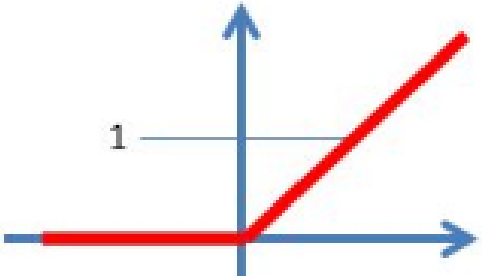


tanh (bipolar)

- **Saturating regime**
  ⇒ Vanishing gradient: no back-prop
  ⇒ Slow convergence

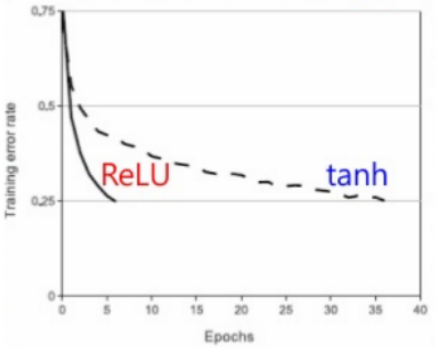
Rectified Linear Unit (ReLU)

$$ReLU(z) = \begin{cases} z \text{ si } z \geq 0 \\ 0 \text{ sinon} \end{cases} = max\{0, z\}$$



Rectified Linear Unit (ReLU)

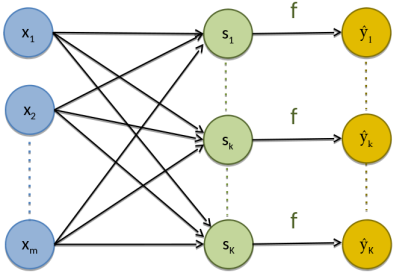
- **Reducing vanishing gradients problems ⇒ faster learning / convergence**
- Ex: 4-layer ConvNet, CIFAR-10
  ⇒ ReLU *vs* tanh: x6 speedup



From [Krizhevsky et al., 2012]

Weight Initialization with ReLU

- Input layer $\mathbf{x}$ with m neurons of output s:
  $Var[s] = 2mVar[\mathbf{w}]Var[\mathbf{x}]$ [He et al., 2015]

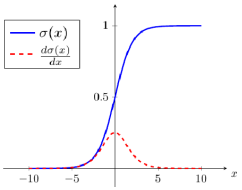


- $\mathbf{W} \sim \frac{1}{\sqrt{2n}}\,\mathcal{N}(0, \sigma)$

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. ICCV'15.

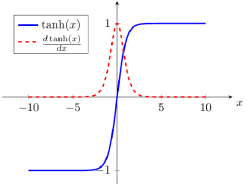
Non-Linear Activation Modules

Sigmoid



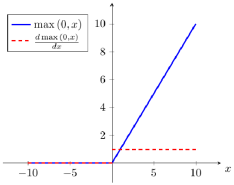
- Saturation
- Expensive
- Not zero-centered

Tanh



- Saturation
- Expensive
- Zero-centered

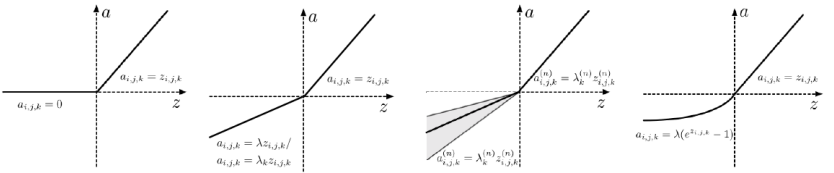
ReLU



- No saturation
- Very efficient
- Not zero-centered
- Negative activations ignored

Non-Linear Activation Modules

- ReLU: 0 for negative inputs ⇒ blocked gradient
- ReLU variants:



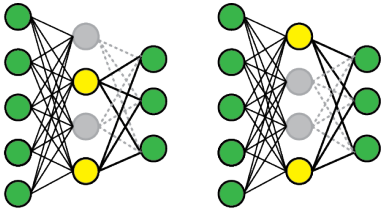
(a) ReLU (b) LReLU/PReLU (c) RReLU (d) ELU

From [Gu et al., 2015]

- Leaky ReLU (LReLU): λ empirically predefined
- Parametric ReLU (PReLU) : λ_k learned from data
- Randomized ReLU (RReLU): λ_k^n uniform sampling
- Exponential Linear Unit (ELU): λ fixed

Modern Deep Learning

- Pre-processing and weight init important for proper training
- **ReLU non-linearity: training speed-up**
  - **Used in AlexNet at ImageNet'12**
  - **Now vanilla activation for essentially every network**
- **Other training improvements**
  $\Rightarrow$ following!



References I

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).
Imagenet classification with deep convolutional neural networks.
In Advances in neural information processing systems, pages 1097–1105.