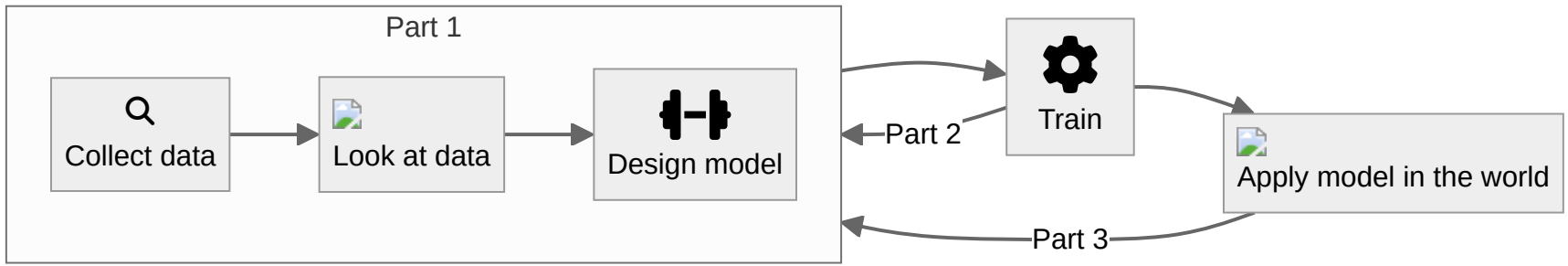
Data and Advanced Network Design

Recap: Developing a Model



Looking at Your Data

Images

- Randomly sample
- Smallest / largest file size
- Rare classes

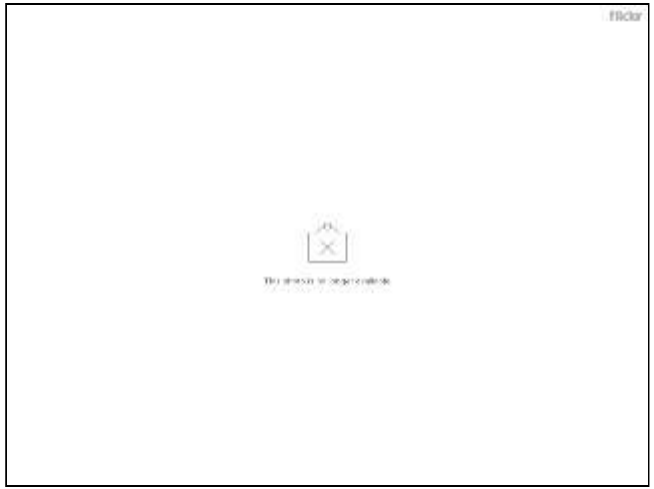
Try solving the task manually

Random Images

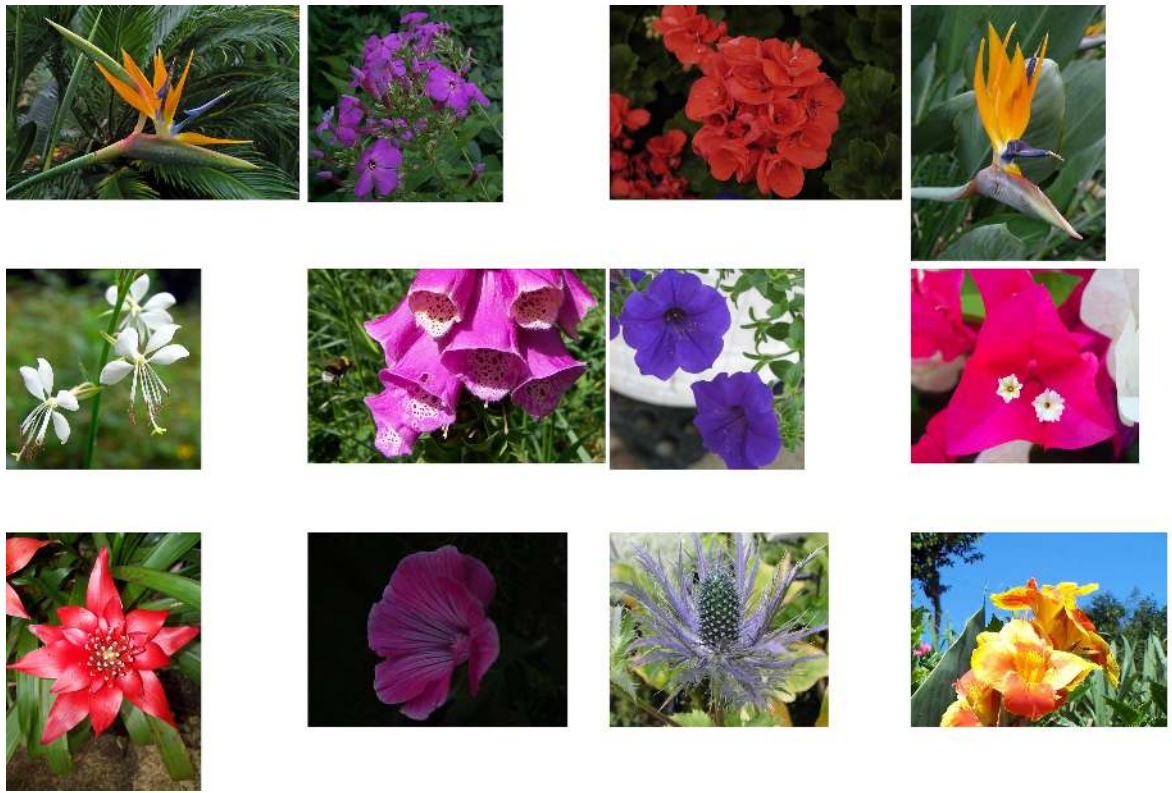
Largest File Size

Smallest File Size

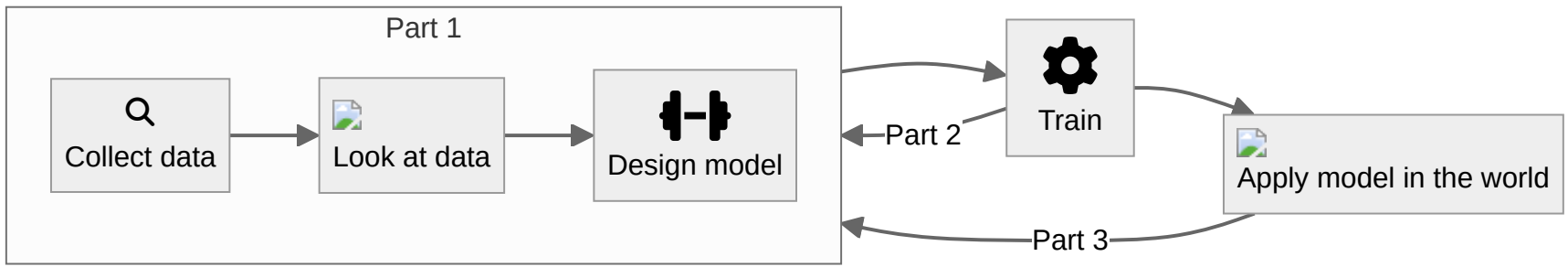
Smallest File Size



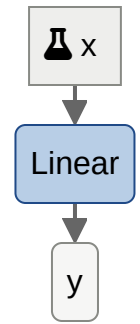
Solving the Task Manually



Advanced Network Design



How to Feed Data Into the Network?



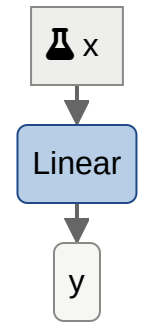
Example: Gradients of Strictly Positive Inputs

Input: $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2] \in \mathbb{R}^2$

Output: $y \in \mathbb{R}$

Gradient: $\frac{\partial l(\theta, \mathbf{x}, y)}{\partial \mathbf{W}} = e\mathbf{x}^T$

- L2 loss: $e = \hat{y} - y$
- L1 loss: $e = \text{sign}(\hat{y} - y)$



Consider: $\mathbf{x}_1 > 0$ and $\mathbf{x}_2 > 0$

- All gradients are all positive or negative
- Highly correlated updates to the weights

$\Longrightarrow$ slow training!

How to Feed Data Into the Network?

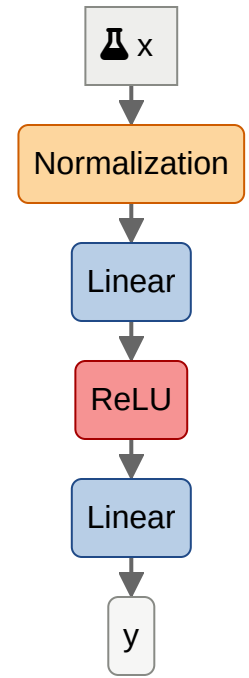
Solution: Mean subtraction

Input: $\mathbf{x}_i$

Apply affine transformation

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

Result: sign of gradients are no longer correlated



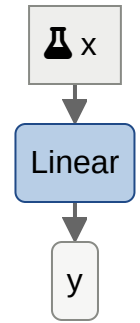
Example: Gradients of Unnormalized Inputs

Input: $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2] \in \mathbb{R}^2$

Output: $y \in \mathbb{R}$

Gradient: $\frac{\partial l(\theta, \mathbf{x}, y)}{\partial \mathbf{W}} = e\mathbf{x}^T$

- L2 loss: $e = \hat{y} - y$
- L1 loss: $e = \text{sign}(\hat{y} - y)$



Consider: $|\mathbf{x}_1| \ll |\mathbf{x}_2|$

$\implies |\frac{\partial l(\theta, \mathbf{x}, y)}{\partial \mathbf{W}_1}| \ll |\frac{\partial l(\theta, \mathbf{x}, y)}{\partial \mathbf{W}_2}|$

$\implies |\mathbf{W}_1| \ll |\mathbf{W}_2|$ [after some training]

$\implies$ model looks at $\mathbf{x}_2$ only

$\implies$ slow training

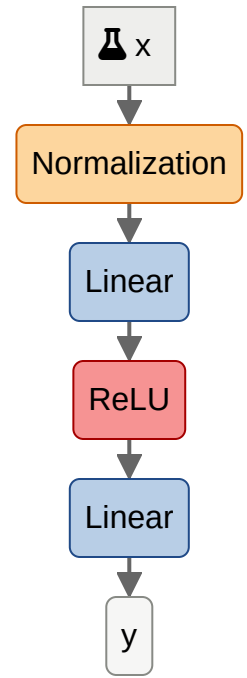
How to Feed Data Into the Network?

Solution: Input normalization

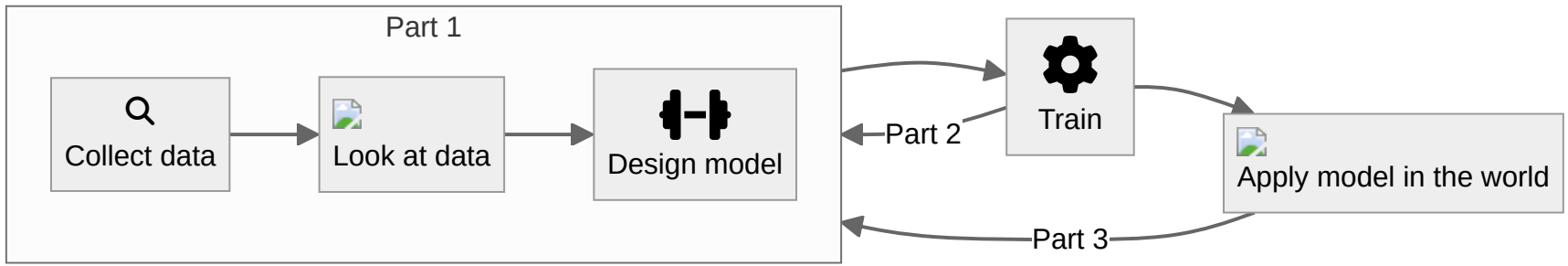
Input: $\mathbf{x}_i$

Apply affine transformation

$$\hat{\mathbf{x}}_i = (\mathbf{x}_i - \boldsymbol{\mu}_\mathbf{x})/\boldsymbol{\sigma}_\mathbf{x}$$



Advanced Network Design

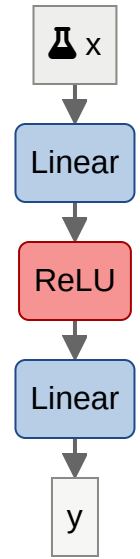


Design Your Model

Pick Your Favorite Model

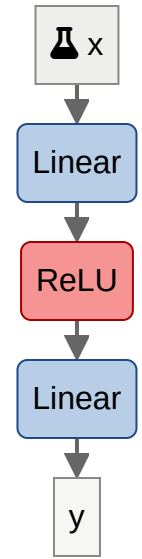
- CNN vs. Transformer
- Performance / speed / accuracy tradeoff

Now What?



How to Initialize the Network?

How should we choose the initial parameters θ?

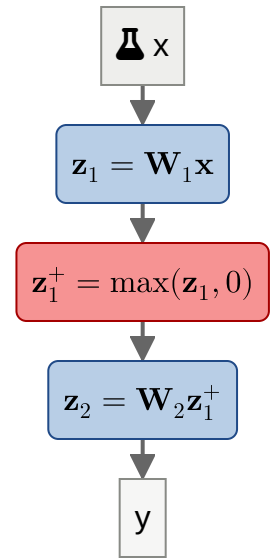


How to Initialize the Network?

Solution 1: All zeros

✗ Does not work - no gradient

✗ Saddle point

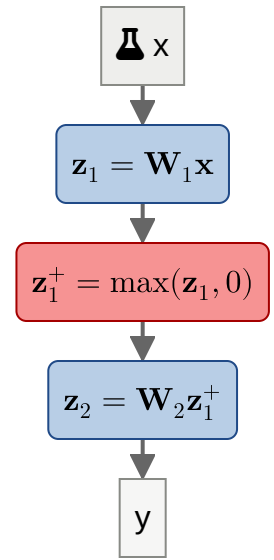


How to Initialize the Network?

Solution 2: Constant

✗ Does not break symmetries

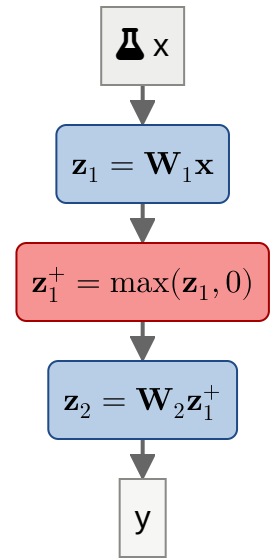
✗ Saddle point



How to Initialize the Network?

Solution 3: Random initialization

$$\mathbf{W}_l \sim \mathcal{N}(\mu_l, \sigma_l^2 \mathbf{I})$$



Random Initialization

Weight Initialization

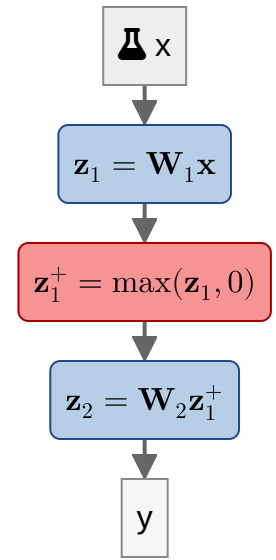
- Normal distribution $\mathcal{N}(\mu, \sigma^2\mathbf{I})$
- Uniform distribution $U(\mu - \sigma, \mu + \sigma)$

What should μ_l and σ_l be?

- For simplicity, $\mu_l = 0$ and bias = 0

$$\mathbf{W}_1 \sim \mathcal{N}(\mu_1, \sigma_1^2\mathbf{I})$$

$$\mathbf{W}_2 \sim \mathcal{N}(\mu_2, \sigma_2^2\mathbf{I})$$



Recall: Vanishing and Exploding Activations

Case 1: $w < 1$

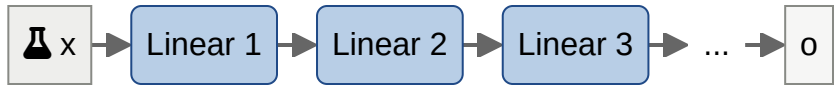
- $w^n x \to 0$ for large n
- Vanishing inputs: $a_n \approx \frac{b}{1-w}$

Case 2: $w = 1$

- $a_n = x + nb$

Case 3: $w > 1$

- Activation n: $w^n \to \infty$
- Exploding activations



Layer	Activation
$l = 1$	$a_1 \approx wx + b$
$l = 2$	$a_2 \approx w^2 x + (w + 1)b$
$l = 3$	$a_3 \approx w^3 x + (w^2 + w + 1)b$
$l = n$	$a_n \approx w^n x + b\underbrace{\sum_{k=0}^{n-1} w^k}_{\frac{1-w^n}{1-w}}$

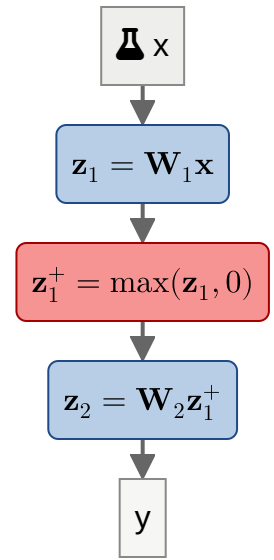
How Do We Scale the Initialization?

By hand

- A lot of tuning

Automatically

- A lot of math
- Xavier Initialization
- Kaiming Initialization



Kaiming Initialization

Strategy to set variance σ^2 of normal distribution

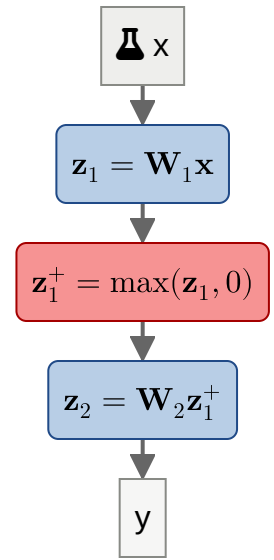
Make all activations of similar scale $\sigma^2_{\mathbf{z}_l} \approx \sigma^2_{\mathbf{z}_{l+1}}$

$\mathbf{W}_1 \sim \mathcal{N}(\mu_{\mathbf{W}_1}, \sigma^2_{\mathbf{W}_1}\mathbf{I})$

$\mathbf{W}_2 \sim \mathcal{N}(\mu_{\mathbf{W}_2}, \sigma^2_{\mathbf{W}_2}\mathbf{I})$

How do we choose $\sigma^2_{\mathbf{W}_l}$?

How is $\sigma^2_{\mathbf{z}_{l+1}}$ related to $\sigma^2_{\mathbf{z}_l}$?



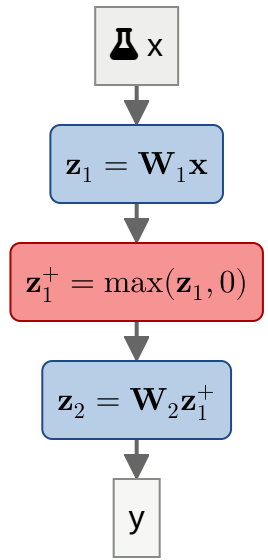
How Does Scale Change Through a Layer?

$\mathbf{z}_l = \mathbf{W}_l \mathrm{ReLU}(\mathbf{z}_{l-1}) = \mathbf{W}_l \mathbf{z}_{l-1}^+$

weights $\mathbf{W}_l \sim \mathcal{N}(0, \sigma^2_{\mathbf{W}_l} \mathbf{I})$

previous $\mathbf{z}_{l-1} \sim \mathcal{N}(0, \sigma^2_{\mathbf{z}_{l-1}} \mathbf{I})$ of length n_{l-1}

current $\mathbf{z}_l \sim \mathcal{N}(0, \underbrace{\frac{1}{2} \cdot n_{l-1} \cdot \sigma^2_{\mathbf{W}_l} \cdot \sigma^2_{\mathbf{z}_{l-1}}}_{\sigma^2_{\mathbf{z}_l}} \mathbf{I})$



Proof Setup

Let $z' = \sum_{i=1}^n w_i z_i^+ = \sum_{i=1}^n w_i \cdot \max(z_i, 0)$
where

- i.i.d $w_i \sim \mathcal{N}(0, \sigma_w^2)$
- i.i.d. $z_i \sim \mathcal{N}(0, \sigma_z^2)$

Show $\mathrm{Var}(z') = \frac{1}{2} \cdot n \cdot \sigma_w^2 \cdot \sigma_z^2$

How Does Scale Change Through a Layer?

$\mathbf{z}_l = \mathbf{W}_l \mathrm{ReLU}(\mathbf{z}_{l-1}) = \mathbf{W}_l \mathbf{z}_{l-1}^+$

weights $\mathbf{W}_l \sim \mathcal{N}(0, \sigma^2_{\mathbf{W}_l} \mathbf{I})$

previous $\mathbf{z}_{l-1} \sim \mathcal{N}(0, \sigma^2_{\mathbf{z}_{l-1}} \mathbf{I})$ of length n_{l-1}

current $\mathbf{z}_l \sim \mathcal{N}(0, \underbrace{\frac{1}{2} \cdot n_{l-1} \cdot \sigma^2_{\mathbf{W}_l} \cdot \sigma^2_{\mathbf{z}_{l-1}}}_{\sigma^2_{\mathbf{z}_l}} \mathbf{I})$

Proof Setup

Let $z' = \sum_{i=1}^n w_i z_i^+ = \sum_{i=1}^n w_i \cdot \max(z_i, 0)$ where

- i.i.d $w_i \sim \mathcal{N}(0, \sigma_w^2)$
- i.i.d. $z_i \sim \mathcal{N}(0, \sigma_z^2)$

Show $\mathrm{Var}(z') = \frac{1}{2} \cdot n \cdot \sigma_w^2 \cdot \sigma_z^2$

Proof (Part 1/2) $\mathrm{Var}(z') = n \cdot \sigma_w^2 \cdot \mathbb{E}[(z_1^+)^2]$

$$
\begin{aligned}
\mathrm{Var}(z') &= \mathrm{Var}(\sum_{i=1}^n w_i z_i^+) \\
&= \sum_{i=1}^n \mathrm{Var}(w_i z_i^+) \\
&= \sum_{i=1}^n \mathbb{E}[(w_i z_i^+)^2] - \mathbb{E}[w_i z_i^+]^2 \\
&= \sum_{i=1}^n \mathbb{E}[w_i^2] \cdot \mathbb{E}[(z_i^+)^2] \\
&= \sigma_w^2 \sum_{i=1}^n \mathbb{E}[(z_i^+)^2] \\
&= n \cdot \sigma_w^2 \cdot \mathbb{E}[(z_1^+)^2]
\end{aligned}
$$

How Does Scale Change Through a Layer?

$\mathbf{z}_l = \mathbf{W}_l \mathrm{ReLU}(\mathbf{z}_{l-1}) = \mathbf{W}_l \mathbf{z}_{l-1}^+$

weights $\mathbf{W}_l \sim \mathcal{N}(0, \sigma^2_{\mathbf{W}_l} \mathbf{I})$

previous $\mathbf{z}_{l-1} \sim \mathcal{N}(0, \sigma^2_{\mathbf{z}_{l-1}} \mathbf{I})$ of length n_{l-1}

current $\mathbf{z}_l \sim \mathcal{N}(0, \underbrace{\frac{1}{2} \cdot n_{l-1} \cdot \sigma^2_{\mathbf{W}_l} \cdot \sigma^2_{\mathbf{z}_{l-1}}}_{\sigma^2_{\mathbf{z}_l}} \mathbf{I})$

Proof Setup

Let $z' = \sum_{i=1}^n w_i z_i^+ = \sum_{i=1}^n w_i \cdot \max(z_i, 0)$
where

- i.i.d $w_i \sim \mathcal{N}(0, \sigma_w^2)$
- i.i.d. $z_i \sim \mathcal{N}(0, \sigma_z^2)$

Show $\mathrm{Var}(z') = \frac{1}{2} \cdot n \cdot \sigma_w^2 \cdot \sigma_z^2$

Proof (Part 2/2): $\mathbb{E}[(z_1^+)^2] = \frac{1}{2} \cdot \sigma_z^2$

$$
\begin{aligned}
\mathbb{E}[(z_1^+)^2] &= \mathbb{E}[\max(z_1, 0)^2] \\
&= \int_{-\infty}^{\infty} \max(x, 0)^2 f(x) \mathrm{d}x \\
&= \int_{0}^{\infty} x^2 f(x) \mathrm{d}x \\
&= \frac{1}{2} \int_{-\infty}^{\infty} x^2 f(x) \mathrm{d}x \\
&= \frac{1}{2} \cdot \mathbb{E}[z_1^2] \\
&= \frac{1}{2} \cdot \sigma_z^2
\end{aligned}
$$

How Does Scale Change Through a Network?

$\mathbf{z}_l = \mathbf{W}_l \mathrm{ReLU}(\mathbf{z}_{l-1}) = \mathbf{W}_l \mathbf{z}_{l-1}^+$

weights $\mathbf{W}_l \sim \mathcal{N}(0, \sigma_{\mathbf{W}_l}^2 \mathbf{I})$

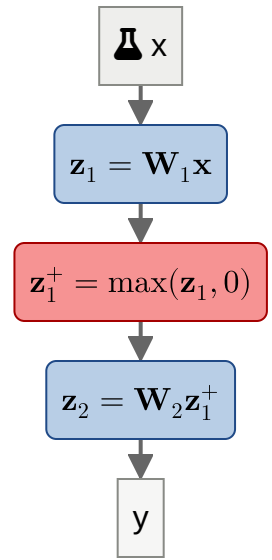
previous $\mathbf{z}_{l-1} \sim \mathcal{N}(0, \sigma_{\mathbf{z}_{l-1}}^2 \mathbf{I})$ of length n_{l-1}

current $\mathbf{z}_l \sim \mathcal{N}(0, \underbrace{\frac{1}{2} \cdot n_{l-1} \cdot \sigma_{\mathbf{W}_l}^2 \cdot \sigma_{\mathbf{z}_{l-1}}^2}_{\sigma_{\mathbf{z}_l}^2} \mathbf{I})$

Scale of Activations

$$\sigma_{\mathbf{z}_l}^2 = (\frac{1}{2} \cdot n_{l-1} \cdot \sigma_{\mathbf{W}_l}^2) \cdot \sigma_{\mathbf{z}_{l-1}}^2$$

$$\sigma_{\mathbf{z}_l}^2 = (\prod_{k=0}^{l-1} \frac{1}{2} \cdot n_k \cdot \sigma_{\mathbf{W}_{k+1}}^2) \cdot \sigma_x^2$$



Kaiming Initialization

Scale of Activations

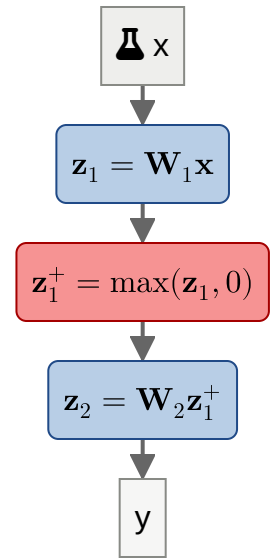
$$\sigma^2_{\mathbf{z}_l} = (\frac{1}{2} \cdot n_{l-1} \cdot \sigma^2_{\mathbf{W}_l}) \cdot \sigma^2_{\mathbf{z}_{l-1}}$$

$$\sigma^2_{\mathbf{z}_l} = (\prod_{k=0}^{l-1} \frac{1}{2} \cdot n_k \cdot \sigma^2_{\mathbf{W}_{k+1}}) \cdot \sigma^2_x$$

Make all activations of similar scale $\sigma^2_{\mathbf{z}_{l+1}} \approx \sigma^2_{\mathbf{z}_l}$

$$\boxed{\sigma^2_{\mathbf{W}_l} = \frac{2}{n_{l-1}}}$$

$$\mathbf{W}_l \sim \mathcal{N}(0, \frac{2}{n_{l-1}}\mathbf{I})$$

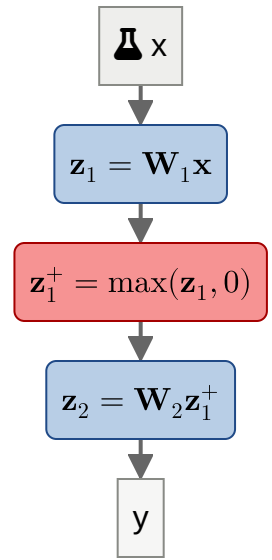


Variance of Backpropagation

Scale of activations/gradients

$$\sigma_{\mathbf{z}_l}^2 = (\prod_{k=0}^{l-1} \frac{1}{2} \cdot n_k \cdot \sigma_{\mathbf{W}_{k+1}}^2) \cdot \sigma_x^2$$

$$\hat{\sigma}_{\mathbf{z}_l}^2 = (\prod_{k=l+1}^{L} \frac{1}{2} \cdot n_k \cdot \sigma_{\mathbf{W}_k}^2) \cdot \hat{\sigma}_{z_L}^2$$



Kaiming Initialization

Scale of activations/gradients

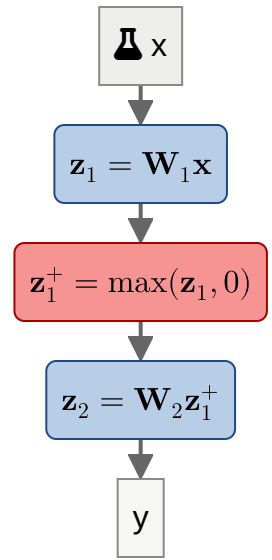
$$\sigma^2_{\mathbf{z}_l} = (\prod_{k=0}^{l-1} \frac{1}{2} \cdot n_k \cdot \sigma^2_{\mathbf{W}_{k+1}}) \cdot \sigma^2_x$$

$$\hat{\sigma}^2_{\mathbf{z}_l} = (\prod_{k=l+1}^{L} \frac{1}{2} \cdot n_k \cdot \sigma^2_{\mathbf{W}_k}) \cdot \hat{\sigma}^2_{z_L}$$

Try keep activation **or** gradient magnitude constant

$$\boxed{\sigma^2_{\mathbf{W}_l} = \frac{2}{n_{l-1}} \text{ (Option 1 - Activations)}}$$

$$\boxed{\sigma^2_{\mathbf{W}_l} = \frac{2}{n_l} \text{ (Option 2 - Gradients)}}$$



Xavier Initialization

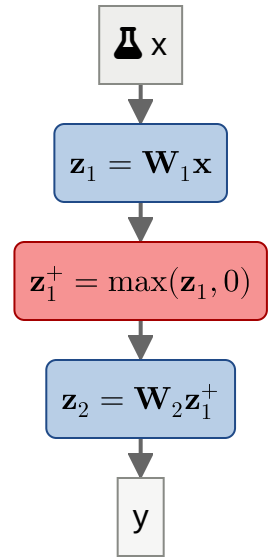
Scale of activations/gradients

$$\sigma^2_{\mathbf{z}_l} = (\prod_{k=0}^{l-1} \frac{1}{2} \cdot n_k \cdot \sigma^2_{\mathbf{W}_{k+1}}) \cdot \sigma^2_x$$

$$\hat{\sigma}^2_{\mathbf{z}_l} = (\prod_{k=l+1}^{L} \frac{1}{2} \cdot n_k \cdot \sigma^2_{\mathbf{W}_k}) \cdot \hat{\sigma}^2_{z_L}$$

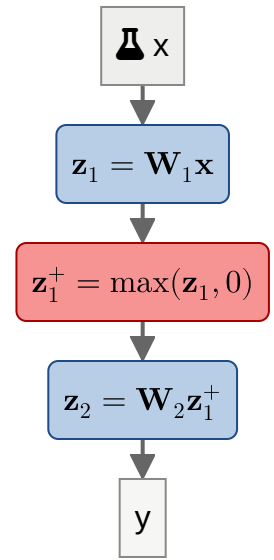
Try keep activation **and** gradient magnitude constant

$$\boxed{\sigma^2_{\mathbf{W}_l} = 2 \cdot \frac{2}{n_{l-1} + n_l}}$$

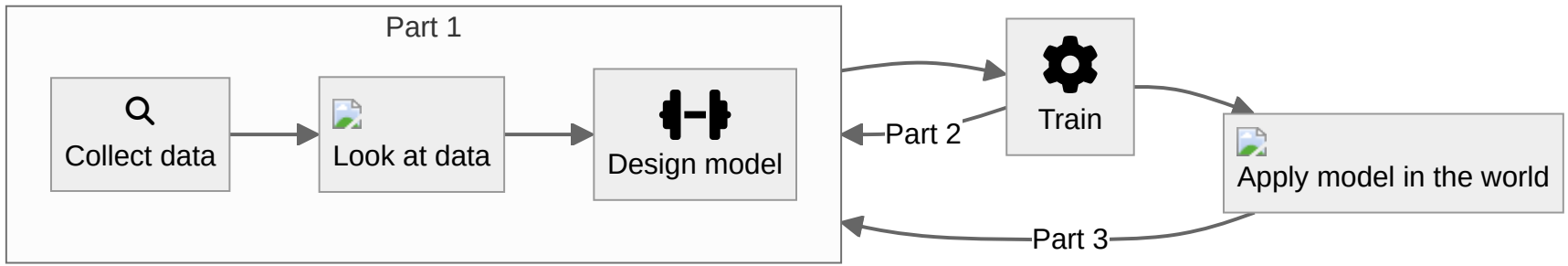


Final Layer

Initialize weights to 0 and disable bias with `bias=False`



Making It Work



Data and Advanced Network Design - TL;DR

Look at your data!

Normalize your input data $\hat{\mathbf{x}}_i = (\mathbf{x}_i - \boldsymbol{\mu}_\mathbf{x})/\boldsymbol{\sigma}_\mathbf{x}$

Default PyTorch initialization (Kaiming init) is usually good enough