Follow Up: Deep Learning Basics

Tobias Jülg

March 9, 2022

Technical University of Munich

Outline

1. Weight Initialization

2. Optimizer

Weight Initialization

How to initialize learnable parameters?

Ideas:

- Set all weights to zero: $w = 0$

How to initialize learnable parameters?

Ideas:

- Set all weights to zero: $w = 0$

$\rightarrow$ all hidden units compute the same function, no symmetric breaking

How to initialize learnable parameters?

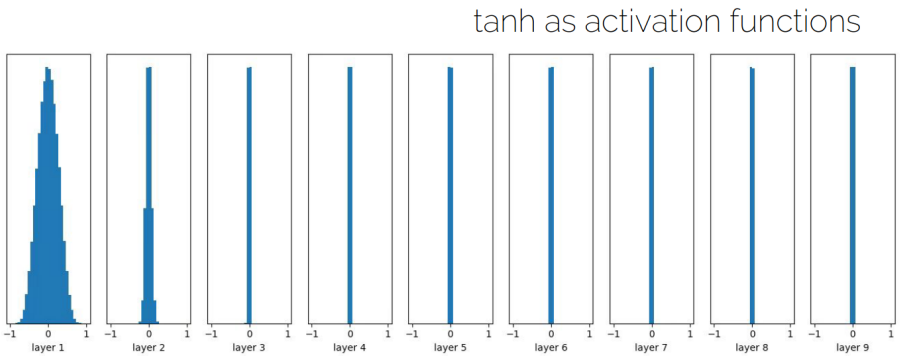
Ideas:

- Set all weights to zero: $w = 0$

$\rightarrow$ all hidden units compute the same function, no symmetric breaking

- Small random numbers, e.g. Gaussian $\mu = 0, \sigma^2 = 0.01$

How to initialize learnable parameters?



Ideas:

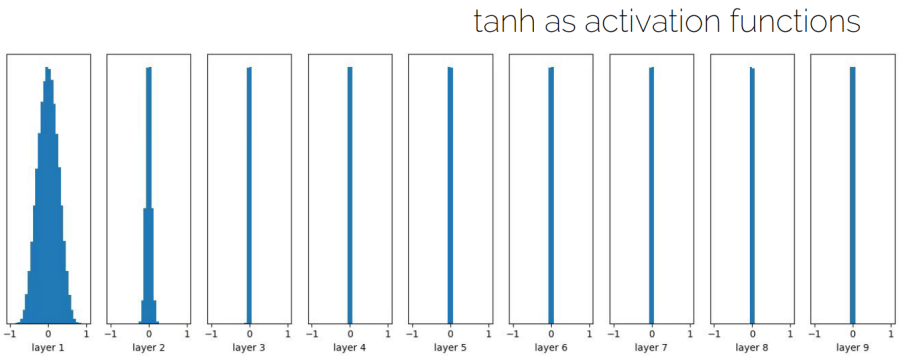
- Set all weights to zero: $w = 0$

$\rightarrow$ all hidden units compute the same function, no symmetric breaking

- Small random numbers, e.g. Gaussian $\mu = 0, \sigma^2 = 0.01$

$\rightarrow$ Output becomes very small, vanishing gradient

How to initialize learnable parameters?



Ideas:

- Set all weights to zero: $w = 0$

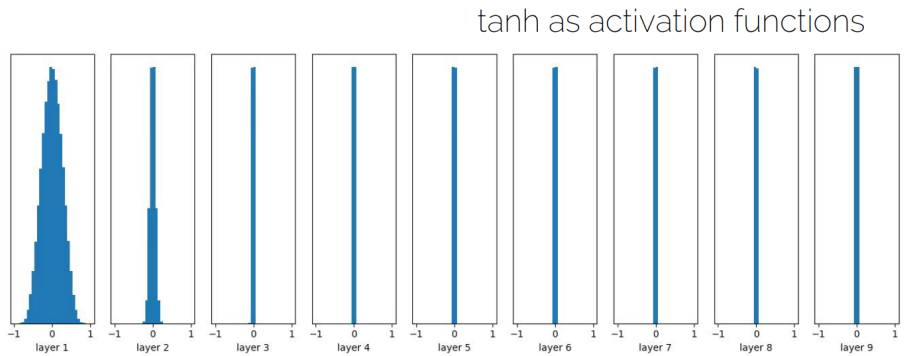
$\rightarrow$ all hidden units compute the same function, no symmetric breaking

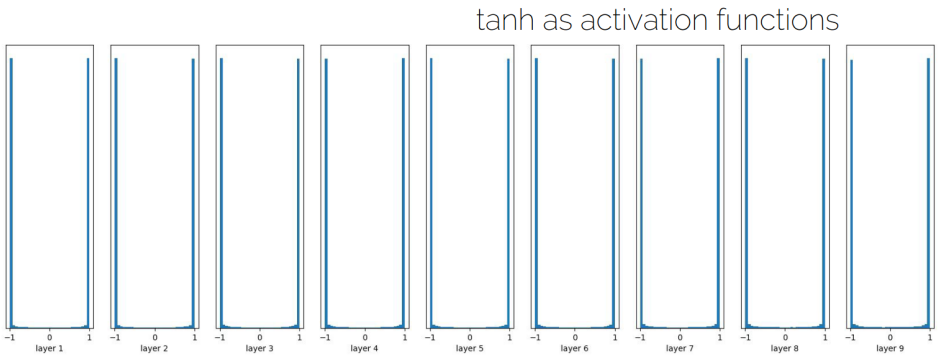
- Small random numbers, e.g. Gaussian $\mu = 0, \sigma^2 = 0.01$

$\rightarrow$ Output becomes very small, vanishing gradient

- Large random numbers e.g. $\mu = 0, \sigma^2 = 1$

How to initialize learnable parameters?





Ideas:

- Set all weights to zero: $w = 0$

$\rightarrow$ all hidden units compute the same function, no symmetric breaking

- Small random numbers, e.g. Gaussian $\mu = 0, \sigma^2 = 0.01$

$\rightarrow$ Output becomes very small, vanishing gradient

- Large random numbers e.g. $\mu = 0, \sigma^2 = 1$

$\rightarrow$ Large outputs, outputs are regions of small gradients for some activation functions

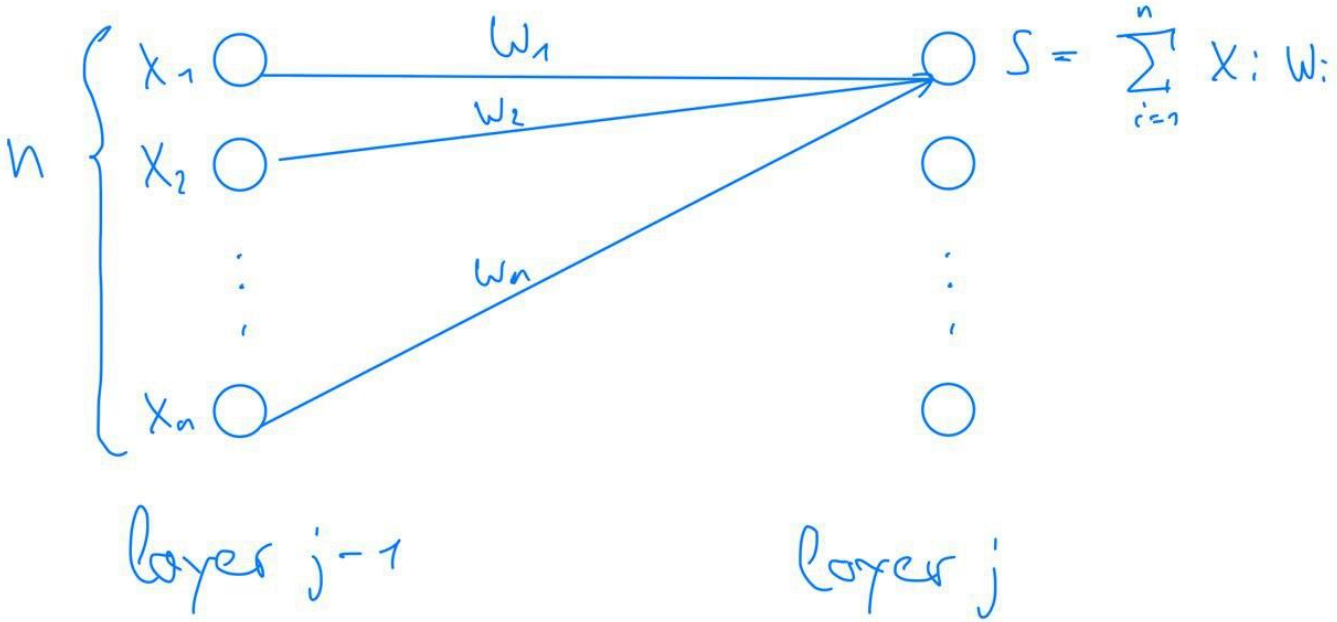
Xavier Initialization [1]: Idea

Distribution of each output neuron S should match the distribution X_i of each input neuron

Xavier Initialization [1]: Idea

Distribution of each output neuron S should match the distribution X_i of each input neuron

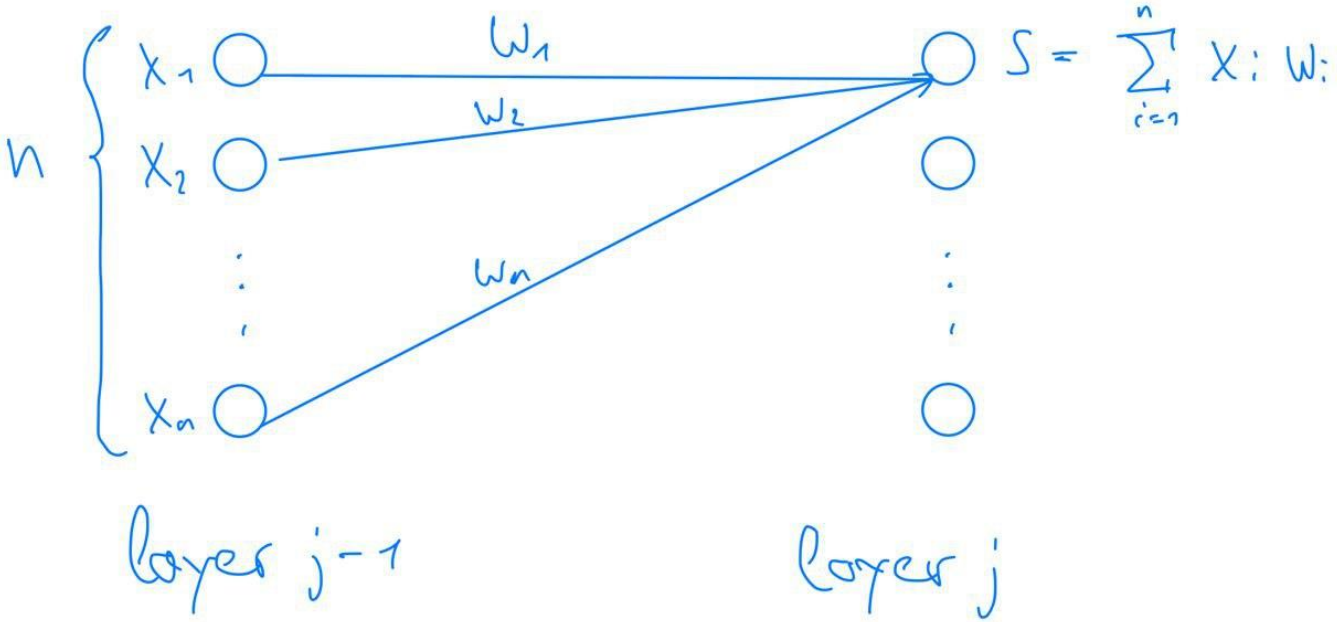
- Assume we have n input neurons, so $i \in \{0, ..., n\}$
- Given: $\mathbb{E}[X_i] = \mathbb{E}[W_i] = 0$, $\text{Var}(X_i)$
- Condition: $\text{Var}(X_i) \stackrel{!}{=} \text{Var}(S)$



Xavier Initialization [1]: Idea

Distribution of each output neuron S should match the distribution X_i of each input neuron

- Assume we have n input neurons, so $i \in \{0, ..., n\}$
- Given: $\mathbb{E}[X_i] = \mathbb{E}[W_i] = 0$, $\text{Var}(X_i)$
- Condition: $\text{Var}(X_i) \overset{!}{=} \text{Var}(S)$



How to choose $\text{Var}(W_i)$?

Xavier Initialization: Derivation

$$\mathrm{Var}(S) \tag{1}$$

$$\tag{2}$$

$$\tag{5}$$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

(2)

(5)

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \tag{2}$$

$$\tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \sum_{i=1}^{n}\mathrm{Var}(W_i X_i) \tag{2}$$

$$\tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \sum_{i=1}^{n}\mathrm{Var}(W_i X_i) \tag{2}$$

$$\mathrm{Var}(W_i X_i) \overset{2.}{=}$$

$$\tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \sum_{i=1}^{n}\mathrm{Var}(W_i X_i) \tag{2}$$

$$\mathrm{Var}(W_i X_i) \overset{2.}{=} \mathbb{E}[W_i^2 X_i^2] - \mathbb{E}[W_i X_i]^2 \tag{3}$$

$$\overset{3.}{=} \tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X+Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \sum_{i=1}^{n}\mathrm{Var}(W_i X_i) \tag{2}$$

$$\mathrm{Var}(W_i X_i) \overset{2.}{=} \mathbb{E}[W_i^2 X_i^2] - \mathbb{E}[W_i X_i]^2 \tag{3}$$

$$\overset{3.}{=} \mathbb{E}[W_i^2 X_i^2] - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \tag{4}$$

$$\overset{1.}{=} \tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation

$$\mathrm{Var}(S) = \mathrm{Var}\left(\sum_{i=1}^{n}(W_i X_i)\right) \tag{1}$$

$$\overset{4.}{=} \sum_{i=1}^{n}\mathrm{Var}(W_i X_i) \tag{2}$$

$$\mathrm{Var}(W_i X_i) \overset{2.}{=} \mathbb{E}[W_i^2 X_i^2] - \mathbb{E}[W_i X_i]^2 \tag{3}$$

$$\overset{3.}{=} \mathbb{E}[W_i^2 X_i^2] - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \tag{4}$$

$$\overset{1.}{=} (\mathrm{Var}(W_i) + \mathbb{E}[W_i]^2)(\mathrm{Var}(X_i) + \mathbb{E}[X_i]^2) - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \tag{5}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation II

$$\mathrm{Var}(W_i X_i) = (\mathrm{Var}(W_i) + \mathbb{E}[W_i]^2)(\mathrm{Var}(X_i) + \mathbb{E}[X_i]^2) - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \quad (6)$$

(8)

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation II

$$\mathrm{Var}(W_i X_i) = (\mathrm{Var}(W_i) + \mathbb{E}[W_i]^2)(\mathrm{Var}(X_i) + \mathbb{E}[X_i]^2) - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \tag{6}$$

$$= \mathrm{Var}(W_i)\mathrm{Var}(X_i) + \mathrm{Var}(W_i)\mathbb{E}[X_i]^2 + \mathbb{E}[W_i]^2\mathrm{Var}(X_i) + \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \tag{7}$$

$$\tag{8}$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation II

$$\mathrm{Var}(W_i X_i) = (\mathrm{Var}(W_i) + \mathbb{E}[W_i]^2)(\mathrm{Var}(X_i) + \mathbb{E}[X_i]^2) - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \quad (6)$$

$$= \mathrm{Var}(W_i)\mathrm{Var}(X_i) + \underbrace{\mathrm{Var}(W_i)\mathbb{E}[X_i]^2}_{\mathbb{E}[X_i]=0} + \underbrace{\mathbb{E}[W_i]^2\mathrm{Var}(X_i)}_{\mathbb{E}[W_i]=0} \quad (7)$$

$$+ \cancel{\mathbb{E}[W_i]^2\mathbb{E}[X_i]^2} - \cancel{\mathbb{E}[W_i]^2\mathbb{E}[X_i]^2} \quad (8)$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation II

$$\mathrm{Var}(W_i X_i) = (\mathrm{Var}(W_i) + \mathbb{E}[W_i]^2)(\mathrm{Var}(X_i) + \mathbb{E}[X_i]^2) - \mathbb{E}[W_i]^2\mathbb{E}[X_i]^2 \quad (6)$$

$$= \mathrm{Var}(W_i)\mathrm{Var}(X_i) + \underbrace{\mathrm{Var}(W_i)\mathbb{E}[X_i]^2}_{\mathbb{E}[X_i]=0} + \underbrace{\mathbb{E}[W_i]^2\mathrm{Var}(X_i)}_{\mathbb{E}[W_i]=0} \quad (7)$$

$$+ \cancel{\mathbb{E}[W_i]^2\mathbb{E}[X_i]^2} - \cancel{\mathbb{E}[W_i]^2\mathbb{E}[X_i]^2}$$

$$= \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad (8)$$

If X and Y are independent:

1. $\mathbb{E}[X^2] = \mathrm{Var}(X) + \mathbb{E}[X]^2$
2. $\mathrm{Var}(XY) = \mathbb{E}[X^2 Y^2] - \mathbb{E}[XY]^2$
3. $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$
4. $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y)$

Xavier Initialization: Derivation III

$$\text{Var}(S) = \sum_{i=1}^{n} \text{Var}(W_i X_i) = \sum_{i=1}^{n} \text{Var}(W_i)\text{Var}(X_i) \tag{9}$$

$$\tag{10}$$

[5] $\text{Var}(X_i) = \text{Var}(X_j)$ and $\text{Var}(W_i) = \text{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Xavier Initialization: Derivation III

$$\mathrm{Var}(S) = \sum_{i=1}^{n} \mathrm{Var}(W_i X_i) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\mathrm{Var}(X_i) \tag{9}$$

$$\overset{5}{=} n \cdot \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad \forall i \in \{0, ..., n\} \tag{10}$$

[5] $\mathrm{Var}(X_i) = \mathrm{Var}(X_j)$ and $\mathrm{Var}(W_i) = \mathrm{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Xavier Initialization: Derivation III

$$\mathrm{Var}(S) = \sum_{i=1}^{n} \mathrm{Var}(W_i X_i) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\mathrm{Var}(X_i) \tag{9}$$

$$\stackrel{5}{=} n \cdot \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad \forall i \in \{0, ..., n\} \tag{10}$$

$$\Rightarrow \ \mathrm{Var}(W_i) = \frac{\mathrm{Var}(S)}{n \cdot \mathrm{Var}(X_i)}$$

[5] $\mathrm{Var}(X_i) = \mathrm{Var}(X_j)$ and $\mathrm{Var}(W_i) = \mathrm{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Xavier Initialization: Derivation III

$$\mathrm{Var}(S) = \sum_{i=1}^{n} \mathrm{Var}(W_i X_i) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\mathrm{Var}(X_i) \tag{9}$$

$$\overset{5}{=} n \cdot \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad \forall i \in \{0, ..., n\} \tag{10}$$

$$\Rightarrow \ \mathrm{Var}(W_i) = \frac{\mathrm{Var}(S)}{n \cdot \mathrm{Var}(X_i)} \overset{\mathrm{Var}(S)=\mathrm{Var}(X_i)}{=} \frac{1}{n}$$

[5] $\mathrm{Var}(X_i) = \mathrm{Var}(X_j)$ and $\mathrm{Var}(W_i) = \mathrm{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Xavier Initialization: Derivation III

$$\mathrm{Var}(S) = \sum_{i=1}^{n} \mathrm{Var}(W_i X_i) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\mathrm{Var}(X_i) \tag{9}$$

$$\overset{5}{=} n \cdot \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad \forall i \in \{0, ..., n\} \tag{10}$$

$\Rightarrow \mathrm{Var}(W_i) = \frac{\mathrm{Var}(S)}{n \cdot \mathrm{Var}(X_i)} \overset{\mathrm{Var}(S)=\mathrm{Var}(X_i)}{=} \frac{1}{n}$

$\Rightarrow$ Weights for the jth layer should be initialized with $\mathbb{E}[W] = 0$ and $\mathrm{Var}(W) = \frac{1}{n}$, where n is the number of output neurons from the layer $j - 1$.

[5] $\mathrm{Var}(X_i) = \mathrm{Var}(X_j)$ and $\mathrm{Var}(W_i) = \mathrm{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Xavier Initialization: Derivation III

$$\mathrm{Var}(S) = \sum_{i=1}^{n} \mathrm{Var}(W_i X_i) = \sum_{i=1}^{n} \mathrm{Var}(W_i)\mathrm{Var}(X_i) \tag{9}$$

$$\overset{5}{=} n \cdot \mathrm{Var}(W_i)\mathrm{Var}(X_i) \quad \forall i \in \{0, ..., n\} \tag{10}$$

$\Rightarrow$ $\mathrm{Var}(W_i) = \frac{\mathrm{Var}(S)}{n \cdot \mathrm{Var}(X_i)} \overset{\mathrm{Var}(S)=\mathrm{Var}(X_i)}{=} \frac{1}{n}$

$\Rightarrow$ Weights for the jth layer should be initialized with $\mathbb{E}[W] = 0$ and $\mathrm{Var}(W) = \frac{1}{n}$, where n is the number of output neurons from the layer $j-1$.

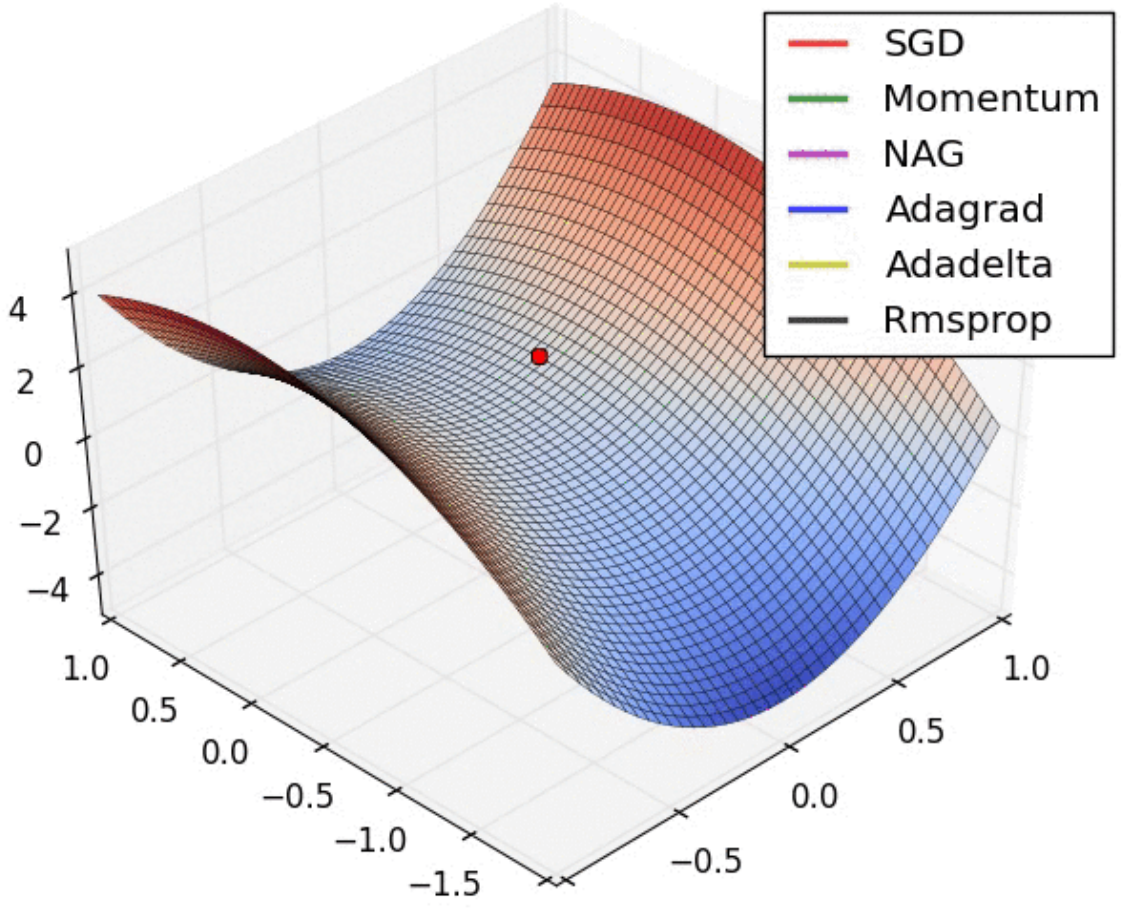
Note: Plain Xavier cannot be used with ReLU activation function as it removes half of the data, use $Var(W_i) = \frac{2}{n}$ [2]

[5] $\mathrm{Var}(X_i) = \mathrm{Var}(X_j)$ and $\mathrm{Var}(W_i) = \mathrm{Var}(W_j)$ for $\forall i, j \in \{0, ..., n\}$

Optimizer

Optimizer Repitition

Strategy to update the weights θ given the loss functions gradients with respect to the weights $\nabla_\theta \mathcal{L}$



Gradient Descent

Weight update rule for

- Single sample $x_i \in X$ with target y_i

$$\theta^{k+1} = \theta^k - \alpha \nabla_\theta \underbrace{\mathcal{L}(\overbrace{\text{net}(x_i, \theta^k)}^{\hat{y}_i}, y_i)}_{\mathcal{L}_i} \tag{11}$$

  - α: Learning rate
  - θ^k: weights at step k

Gradient Descent

Weight update rule for

- Single sample $x_i \in X$ with target y_i

$$\theta^{k+1} = \theta^k - \alpha \nabla_\theta \underbrace{\mathcal{L}(\overbrace{\text{net}(x_i, \theta^k)}^{\hat{y}_i}, y_i)}_{\mathcal{L}_i} = \theta^k - \alpha \nabla_\theta \mathcal{L}_i \tag{11}$$

  - α: Learning rate
  - θ^k: weights at step k

Gradient Descent II

Weight update rule for

- Muli sample / batch $B \subset \{1, ..., n\}$ (n is the size of the dataset)

$$\theta^{k+1} = \theta^k - \alpha \frac{1}{|B|} \sum_{i \in B} \nabla_\theta \mathcal{L}_i \tag{12}$$

Gradient Descent II

Weight update rule for

- Muli sample / batch $B \subset \{1, ..., n\}$ (n is the size of the dataset)

$$\theta^{k+1} = \theta^k - \alpha \frac{1}{|B|} \sum_{i \in B} \nabla_\theta \mathcal{L}_i = \theta^k - \alpha \nabla_\theta \underbrace{\frac{1}{|S|} \sum_{i \in B} \mathcal{L}_i}_{\mathcal{L}_B} \tag{12}$$

Gradient Descent II

Weight update rule for

- Muli sample / batch $B \subset \{1, ..., n\}$ (n is the size of the dataset)

$$\theta^{k+1} = \theta^k - \alpha \frac{1}{|B|} \sum_{i \in B} \nabla_\theta \mathcal{L}_i = \theta^k - \alpha \nabla_\theta \underbrace{\frac{1}{|S|} \sum_{i \in B} \mathcal{L}_i}_{\mathcal{L}_B} \tag{12}$$

  $\rightarrow$ Define the mean operation already in the loss function

L2 Regularization / Weight Decay

- Large weights are likely a result of overfitting i.e. "remembering" the trainings dataset
- No generalization

$\rightarrow$ punish high weights in the loss

$$\mathcal{L}'_B = \mathcal{L}_B + \frac{\lambda}{2} \sum_{w \in \theta} w^2 \tag{13}$$

Momentum

- Accumulate momentum in repeated gradient directions

$\rightarrow$ Overcome local minima

Momentum

- Accumulate momentum in repeated gradient directions

$\rightarrow$ Overcome local minima

- Use exponentially weighted average: Gradients pointing in main downward direction accumulate

$$v^{k+1} = \beta \cdot v^k - \alpha \nabla_\theta \mathcal{L}_B \tag{14}$$

$\rightarrow$ v is defined per weight

Momentum

- Accumulate momentum in repeated gradient directions

$\rightarrow$ Overcome local minima

- Use exponentially weighted average: Gradients pointing in main downward direction accumulate

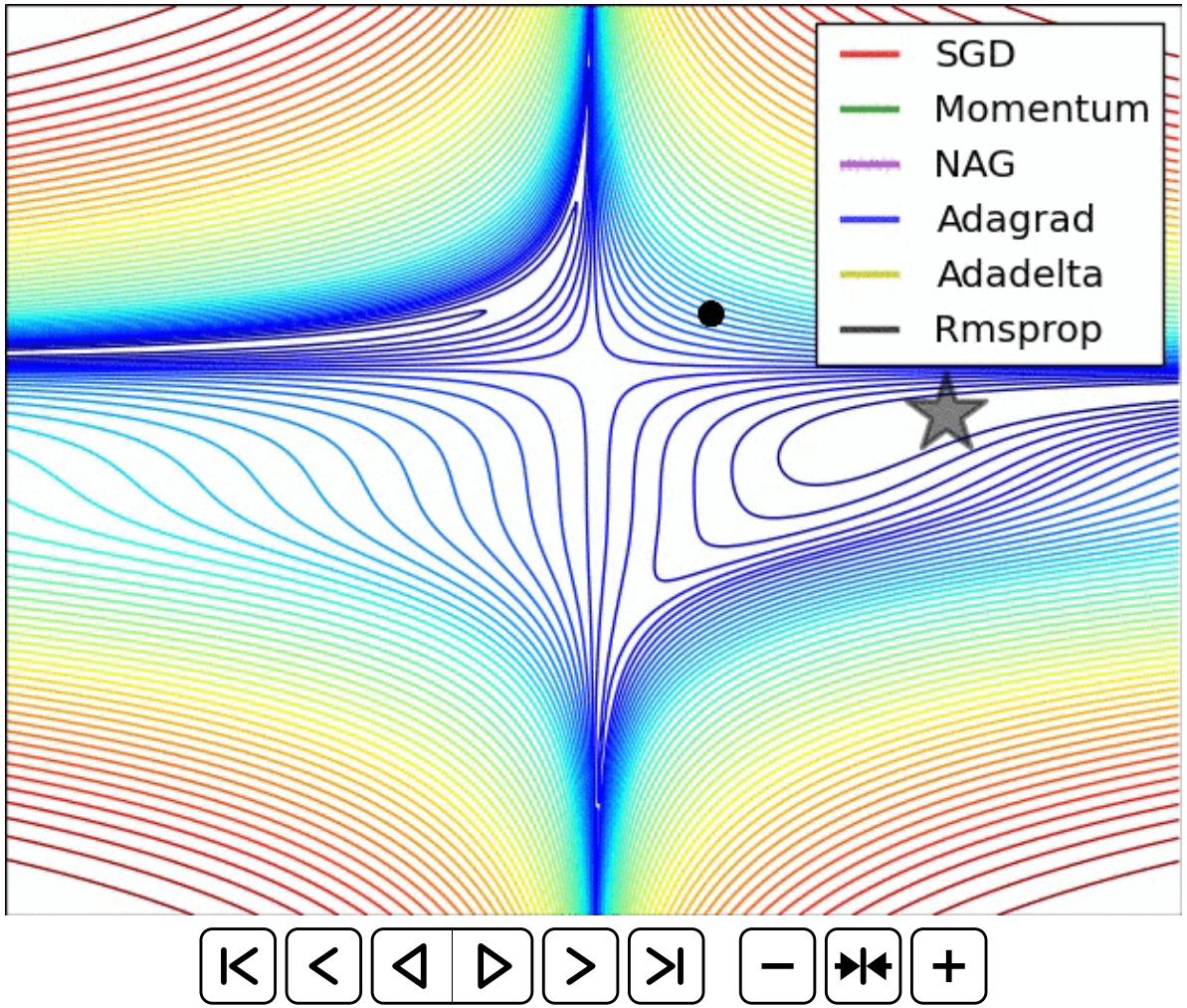
$$v^{k+1} = \beta \cdot v^k - \alpha \nabla_\theta \mathcal{L}_B \tag{14}$$

$\rightarrow$ v is defined per weight

- Weight update:

$$\theta^{k+1} = \theta^k + v^{k+1} \tag{15}$$

Visualization



RMS Prop

- Root Mean Squared **Prop** remove high oscillations resulting from high gradients

$\rightarrow$ Dampen high steps by normalizing the step size by their square

$$s^{k+1} = \beta \cdot s^k + (1 - \beta)(\nabla_\theta \mathcal{L}_B)^2 \tag{16}$$

RMS Prop

- **R**oot **M**ean **S**quared **Prop** remove high oscillations resulting from high gradients

$\rightarrow$ Dampen high steps by normalizing the step size by their square

$$s^{k+1} = \beta \cdot s^k + (1 - \beta)(\nabla_\theta \mathcal{L}_B)^2 \tag{16}$$

- Weight update:

$$\theta^{k+1} = \theta^k - \frac{\alpha}{\sqrt{s^{k+1}} + \epsilon} \nabla_\theta \mathcal{L}_B \tag{17}$$

$\rightarrow$ Dynamic learning rate: Scales the learning rate per parameter

Adam [3]

- **Ada**ptive **M**oment Estimation combines the ideas of Momentum and RMS Prop

$$m^{k+1} = \beta_1 m^k + (1 - \beta_1)\nabla_\theta \mathcal{L}_B \tag{18}$$

$$v^{k+1} = \beta_2 v^k + (1 - \beta_2)(\nabla_\theta \mathcal{L}_B)^2 \tag{19}$$

Adam [3]

- **Ada**ptive **M**oment Estimation combines the ideas of Momentum and RMS Prop

$$m^{k+1} = \beta_1 m^k + (1 - \beta_1)\nabla_\theta \mathcal{L}_B \tag{18}$$

$$v^{k+1} = \beta_2 v^k + (1 - \beta_2)(\nabla_\theta \mathcal{L}_B)^2 \tag{19}$$

- Initialize at zero: $m^0 = v^0 = 0$
- Problem: Choice of β_1 and β_2 influence the training in the beginning
- Bias correction:

$$\hat{m}^{k+1} = \frac{m^{k+1}}{1 - \beta_1} \tag{20}$$

$$\hat{v}^{k+1} = \frac{v^{k+1}}{1 - \beta_2} \tag{21}$$

Adam [3]

- **Ada**ptive **M**oment Estimation combines the ideas of Momentum and RMS Prop

$$m^{k+1} = \beta_1 m^k + (1 - \beta_1)\nabla_\theta \mathcal{L}_B \tag{18}$$

$$v^{k+1} = \beta_2 v^k + (1 - \beta_2)(\nabla_\theta \mathcal{L}_B)^2 \tag{19}$$

- Initialize at zero: $m^0 = v^0 = 0$
- Problem: Choice of β_1 and β_2 influence the training in the beginning
- Bias correction:

$$\hat{m}^{k+1} = \frac{m^{k+1}}{1 - \beta_1} \tag{20}$$

$$\hat{v}^{k+1} = \frac{v^{k+1}}{1 - \beta_2} \tag{21}$$

- Weight update:

$$\theta^{k+1} = \theta^k - \frac{\alpha}{\sqrt{\hat{v}^{k+1}} + \epsilon}\hat{m}^{k+1} \tag{22}$$

Adam [3]

- **Ada**ptive **M**oment Estimation combines the ideas of Momentum and RMS Prop

$$m^{k+1} = \beta_1 m^k + (1 - \beta_1) \nabla_\theta \mathcal{L}_B \tag{18}$$

$$v^{k+1} = \beta_2 v^k + (1 - \beta_2)(\nabla_\theta \mathcal{L}_B)^2 \tag{19}$$

- Initialize at zero: $m^0 = v^0 = 0$
- Problem: Choice of β_1 and β_2 influence the training in the beginning
- Bias correction:

$$\hat{m}^{k+1} = \frac{m^{k+1}}{1 - \beta_1} \tag{20}$$

$$\hat{v}^{k+1} = \frac{v^{k+1}}{1 - \beta_2} \tag{21}$$

- Weight update:

$$\theta^{k+1} = \theta^k - \frac{\alpha}{\sqrt{\hat{v}^{k+1}} + \epsilon} \hat{m}^{k+1} \tag{22}$$

$\rightarrow$ Adaptive learning rate with momentum

Questions?

References i

[1] X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Y. W. Teh and M. Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[2] S. K. Kumar. "On weight initialization in deep neural networks". en. In: *arXiv:1704.08863 [cs]* (May 2017). arXiv: 1704.08863. URL: http://arxiv.org/abs/1704.08863 (visited on 03/09/2022).

[3] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv:1412.6980 [cs]* (Jan. 2017). arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980 (visited on 03/09/2022).

[4] S. Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv:1609.04747 [cs]* (June 2017). arXiv: 1609.04747. URL: http://arxiv.org/abs/1609.04747 (visited on 03/09/2022).

Backup

Train Loss Higher Than Validation Loss?!

There are legitimate reasons why in some cases this might happen even though everything is setup correctly:

- Average is usually calculated online while training (loss is still changing) to avoid a separate forward pass just for the loss metric. This can however lead to discrepancies between validation and training loss.
- Some layers behave differently when in training mode leading to different loss values e.g. Dropout layers.
- Different distributions in validation and train data can lead to different "difficulties" and thus different loss values. The likelihood for this to happen is low when performing random splits, however it is not zero.