Regularization

Goodfellow et al. 2016 (sec. 7)

Regularization: linear regression

- Recall the linear regression problem

$$\hat{L}(\mathbf{w}) = \frac{1}{n}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 \qquad \mathbf{w}^* = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

- Inverse well-defined only when $\mathrm{rank}\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right) = d$

- Add **regularization term**

$$\hat{L}(\mathbf{w}) = \frac{1}{n}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \beta\|\mathbf{w}\|^2$$

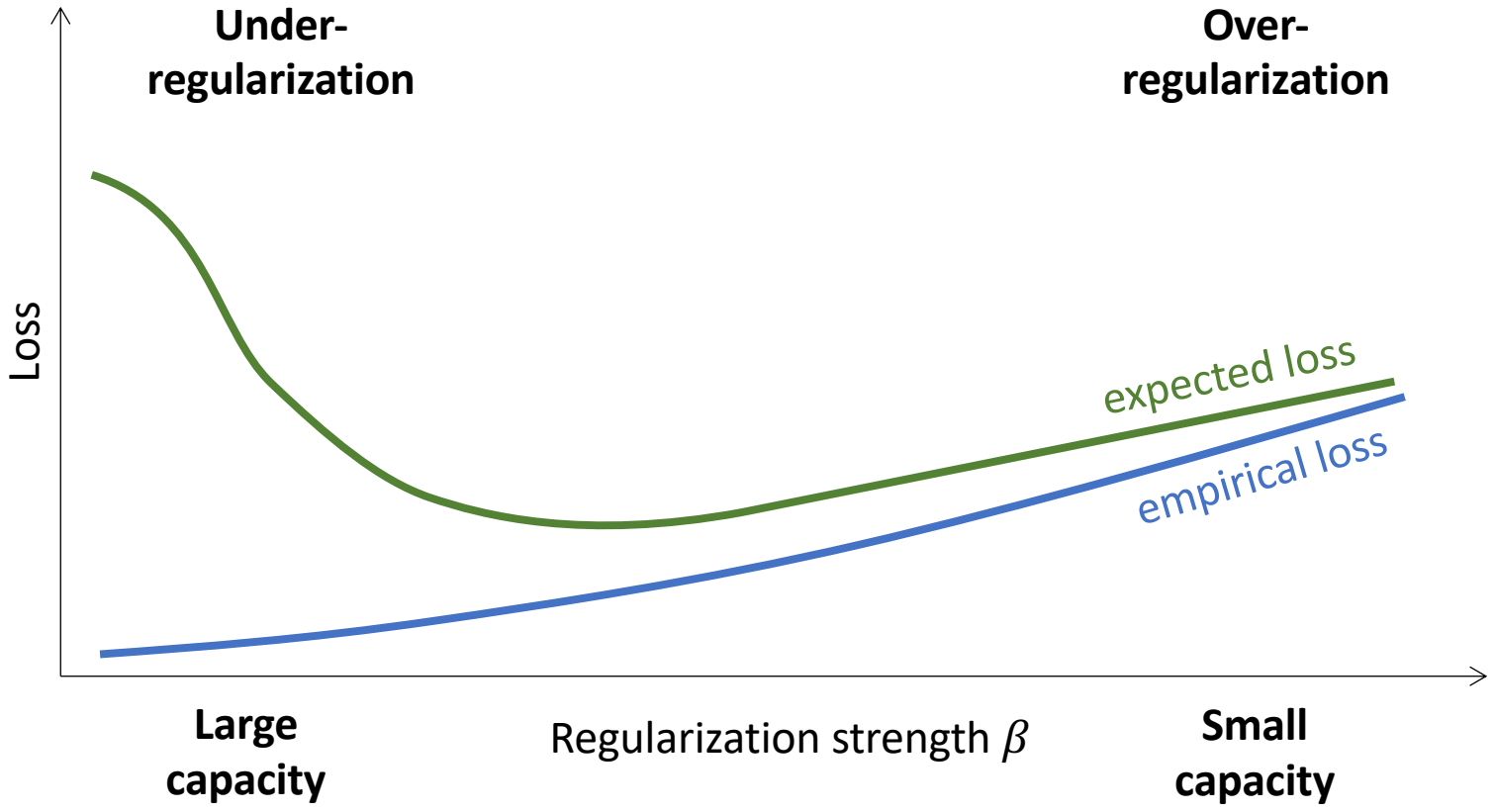
$$\mathbf{w}^* = \left(\frac{1}{n}\mathbf{X}^{\mathrm{T}}\mathbf{X} + \beta\mathbf{I}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

Regularization: general form

$$\min_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) + \beta R(\boldsymbol{\theta})$$

- Soft constraint ("penalty")
- Equivalent to hard constraint (Lagrangian)
- Maximum a posteriori interpretation: principled way of deriving R if we know the distribution of $\boldsymbol{\theta}$
- Restricts the hypothesis class
- Helps prevent overfitting

Regularization



Bayesian view

$$p(\mathbf{x}, y|\boldsymbol{\theta}) = \frac{p(\mathbf{x}, y)p(\boldsymbol{\theta}|\mathbf{x}, y)}{p(\boldsymbol{\theta})}$$

- Prior on hypotheses (parameters distribution): $p(\boldsymbol{\theta})$

- **Maximum a posteriori (MAP):**

$$\min_{\boldsymbol{\theta}} -\log p(\boldsymbol{\theta}|\mathbf{x}, y) = \min_{\boldsymbol{\theta}} \underbrace{-\log p(\boldsymbol{\theta})}_{\text{Regularization}} \underbrace{-\log p(\mathbf{x}, y|\boldsymbol{\theta})}_{\text{MLE}}$$

L_2 Regularization

$$\hat{L}_R(\boldsymbol{\theta}) = \hat{L}(\boldsymbol{\theta}) + \frac{\beta}{2}\|\boldsymbol{\theta}\|^2$$

- Gradient of regularized loss becomes

$$\nabla_{\boldsymbol{\theta}} \hat{L}_R(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) + \beta\boldsymbol{\theta}$$

- Effect on gradient descent update

$$\begin{aligned}\boldsymbol{\theta}^{(t+1)} &= \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}^{(t)}\right) - \alpha^{(t)}\beta\boldsymbol{\theta}^{(t)} \\ &= \left(1 - \alpha^{(t)}\beta\right)\boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}^{(t)}\right)\end{aligned}$$

"weight decay"

- Bayesian interpretation: $\boldsymbol{\theta} \sim \mathcal{N}\left(0, \frac{1}{\sqrt{2\beta}}\mathbf{I}\right)$

Geometric interpretation

- Consider second-order Taylor expansion around original problem optimum $\boldsymbol{\theta}^*$

$$\hat{L}(\boldsymbol{\theta}) \approx \hat{L}(\boldsymbol{\theta}^*) + \nabla_{\boldsymbol{\theta}}\hat{L}(\boldsymbol{\theta}^*)^{\mathrm{T}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^{\mathrm{T}}\nabla^2_{\boldsymbol{\theta}}\hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

Geometric interpretation

- Consider second-order Taylor expansion around original problem optimum $\boldsymbol{\theta}^*$

$$\hat{L}(\boldsymbol{\theta}) \approx \hat{L}(\boldsymbol{\theta}^*) + \tfrac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^{\mathrm{T}} \nabla^{\mathbf{2}}_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

- Gradient of the loss:

$$\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) \approx \nabla^{\mathbf{2}}_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

Geometric interpretation

- Consider second-order Taylor expansion around original problem optimum $\boldsymbol{\theta}^*$

$$\hat{L}(\boldsymbol{\theta}) \approx \hat{L}(\boldsymbol{\theta}^*) + \tfrac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^{\mathrm{T}} \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

- Gradient of the regularized loss:

$$\nabla_{\boldsymbol{\theta}} \hat{L}_R(\boldsymbol{\theta}) \approx \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*) + \beta\boldsymbol{\theta}$$

- Optimality condition:

$$\mathbf{0} = \nabla_{\boldsymbol{\theta}} \hat{L}_R(\boldsymbol{\theta}^*_R) \approx \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*) + \beta\boldsymbol{\theta}^*_R$$

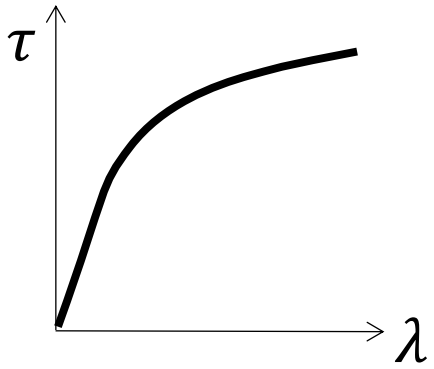
$$\boldsymbol{\theta}^*_R \approx \left(\nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*) + \beta\mathbf{I}\right)^{-1} \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\boldsymbol{\theta}^*$$

Geometric interpretation

- Assume Hessian has eigendecomposition $\nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*) = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}$

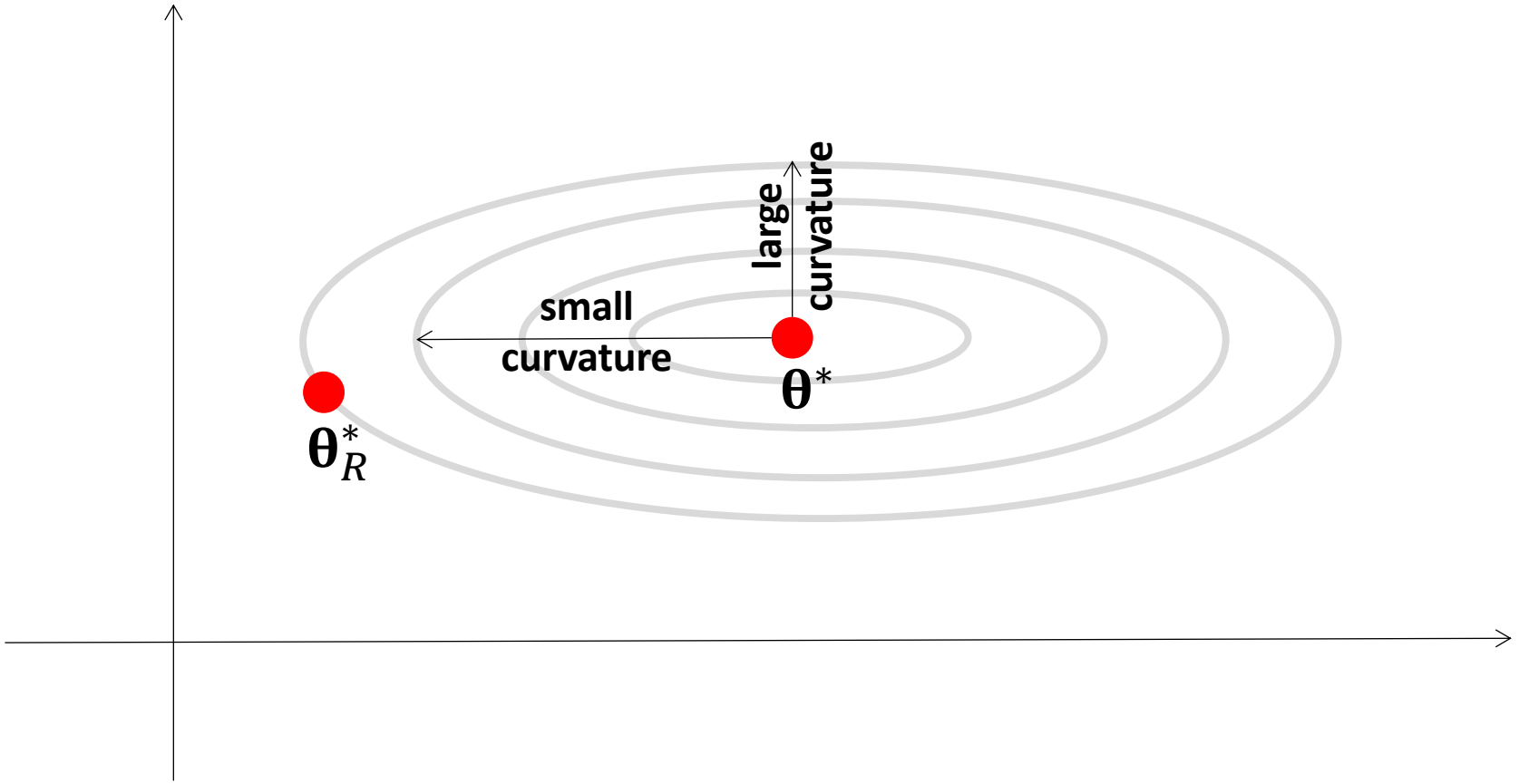
- Then

$$\begin{aligned}
\boldsymbol{\theta}_R^* &\approx \left(\nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*) + \beta\mathbf{I}\right)^{-1} \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\boldsymbol{\theta}^* \\
&\approx \mathbf{U}(\boldsymbol{\Lambda} + \beta\mathbf{I})^{-1}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}\boldsymbol{\theta}^* \\
&= \mathbf{U}\tau(\boldsymbol{\Lambda})\mathbf{U}^{\mathrm{T}}\boldsymbol{\theta}^* \\
&= \tau\left(\nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\right)\boldsymbol{\theta}^*
\end{aligned}$$



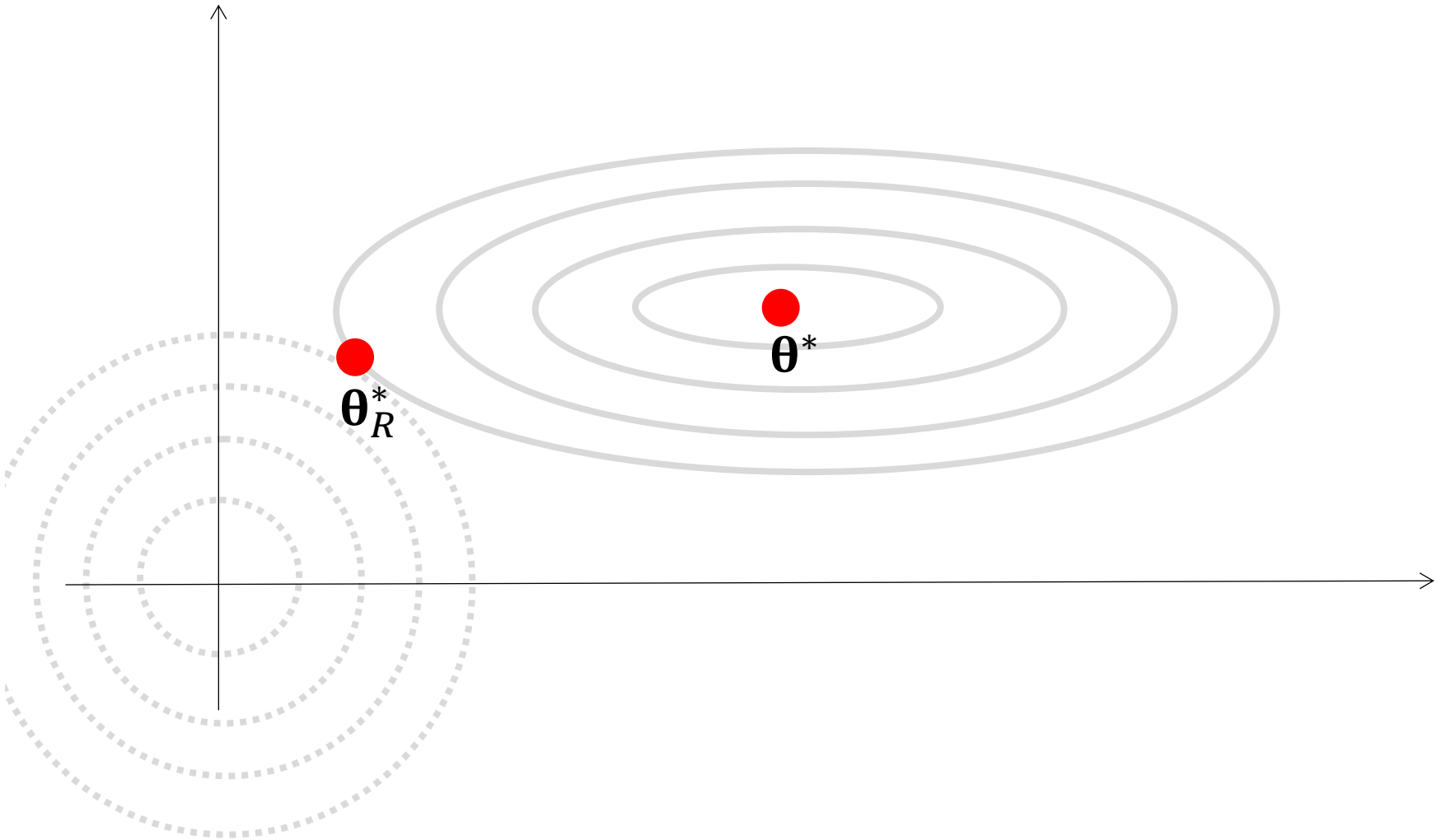
- **Filter of $\boldsymbol{\theta}^*$** (attenuates along small eigenvectors)

Geometric interpretation



Adapted from Goodfellow et al. 2016

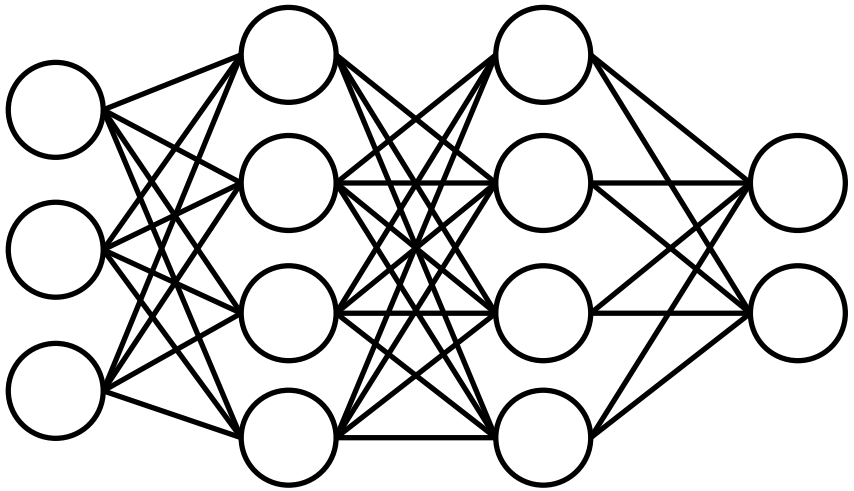
Geometric interpretation



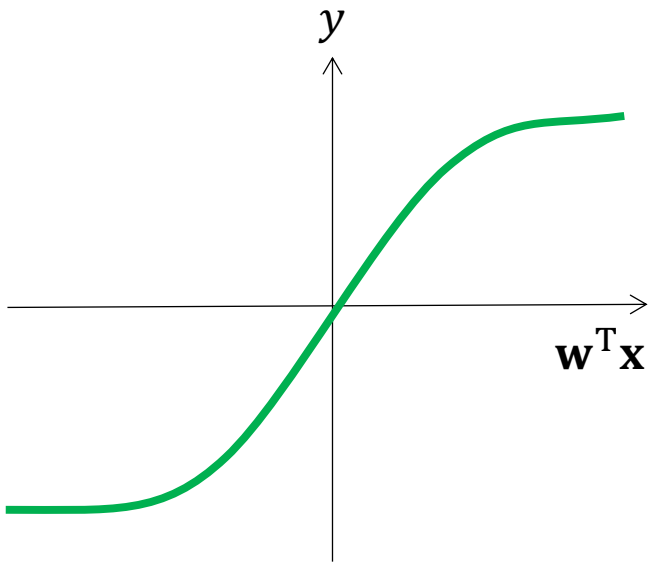
Adapted from Goodfellow et al. 2016

Intuition

Why small weights reduce capacity?



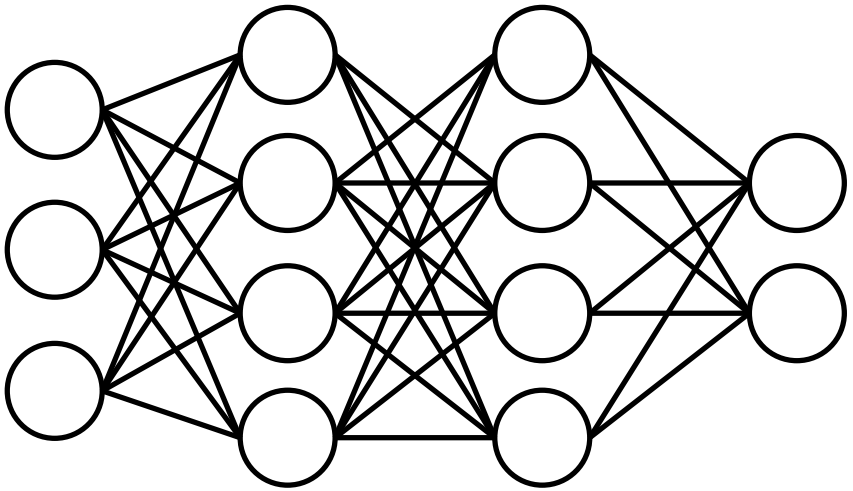
$$y = \mathbf{w}_L^{\mathrm{T}} \sigma\left(\ldots \mathbf{W}_2^{\mathrm{T}} \sigma\left(\mathbf{W}_1^{\mathrm{T}} \boldsymbol{x}\right)\right)$$



small weights = sigmoid is in the linear range

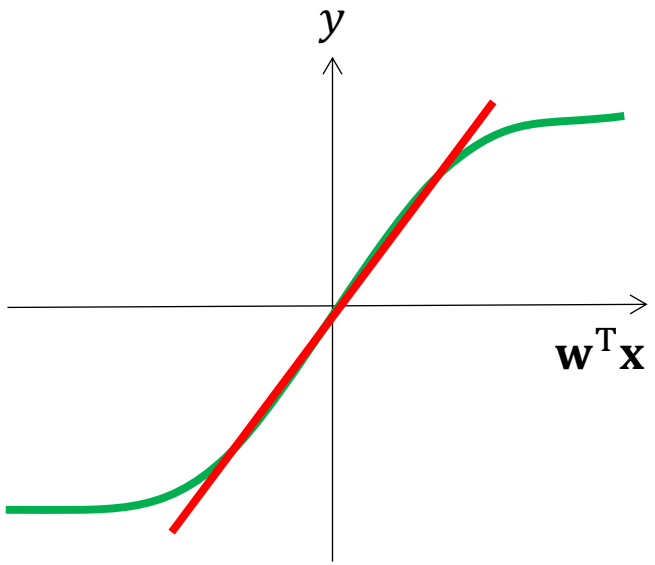
Intuition

Why small weights reduce capacity?



$$y = \mathbf{w}_L^{\mathrm{T}}\left(\ldots \mathbf{W}_2^{\mathrm{T}}\left(\mathbf{W}_1^{\mathrm{T}}\boldsymbol{x}\right)\right) = \mathbf{w}^{\mathrm{T}}\boldsymbol{x}$$

behaves like single layer neural network



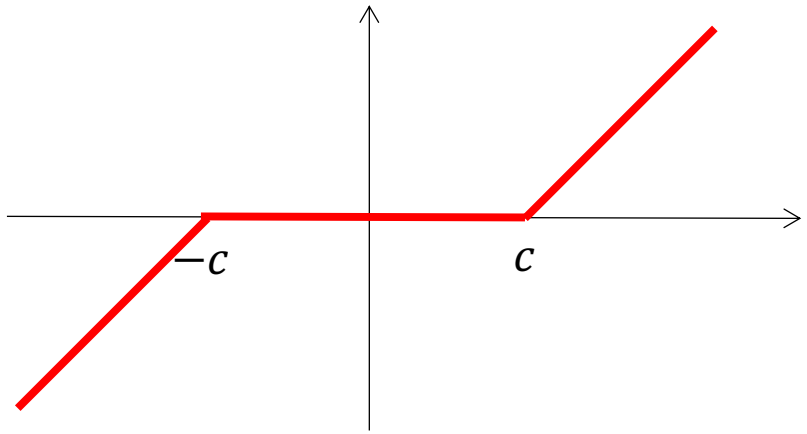
small weights = sigmoid is in the linear range

L_1 Regularization

$$\hat{L}_R(\boldsymbol{\theta}) = \hat{L}(\boldsymbol{\theta}) + \beta\|\boldsymbol{\theta}\|_1$$

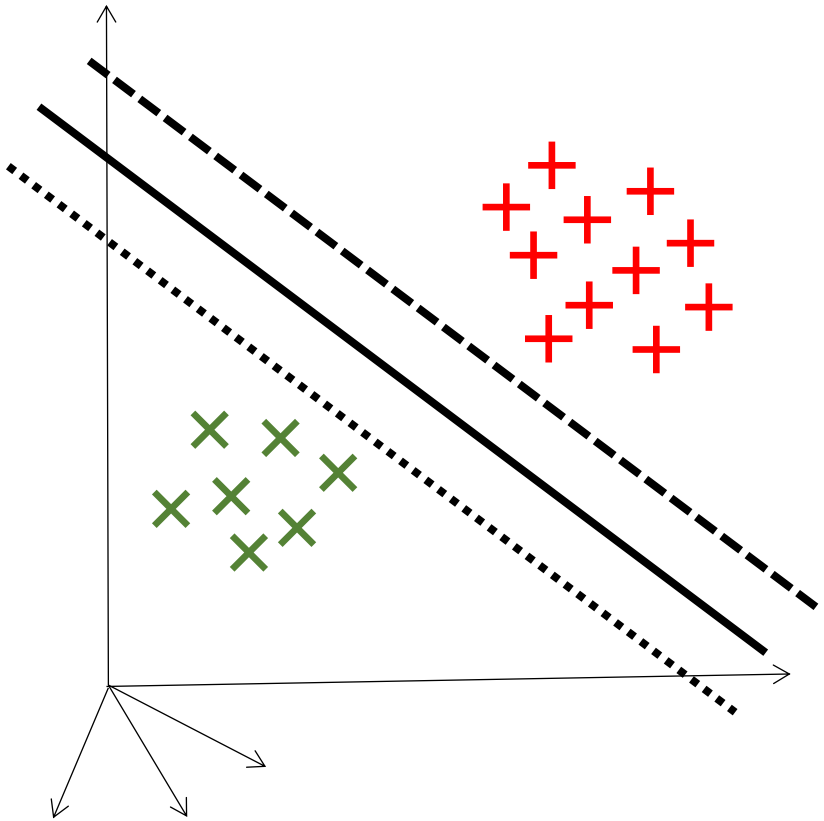
- Effect on solution: **shrinkage** (applied coordinate-wise) promoting sparsity

$$\boldsymbol{\theta}_R^* \approx \text{shrink}(\boldsymbol{\theta}^*) = \beta\text{sign}(\boldsymbol{\theta}^*)\max\{|\boldsymbol{\theta}^*| - c, 0\}$$

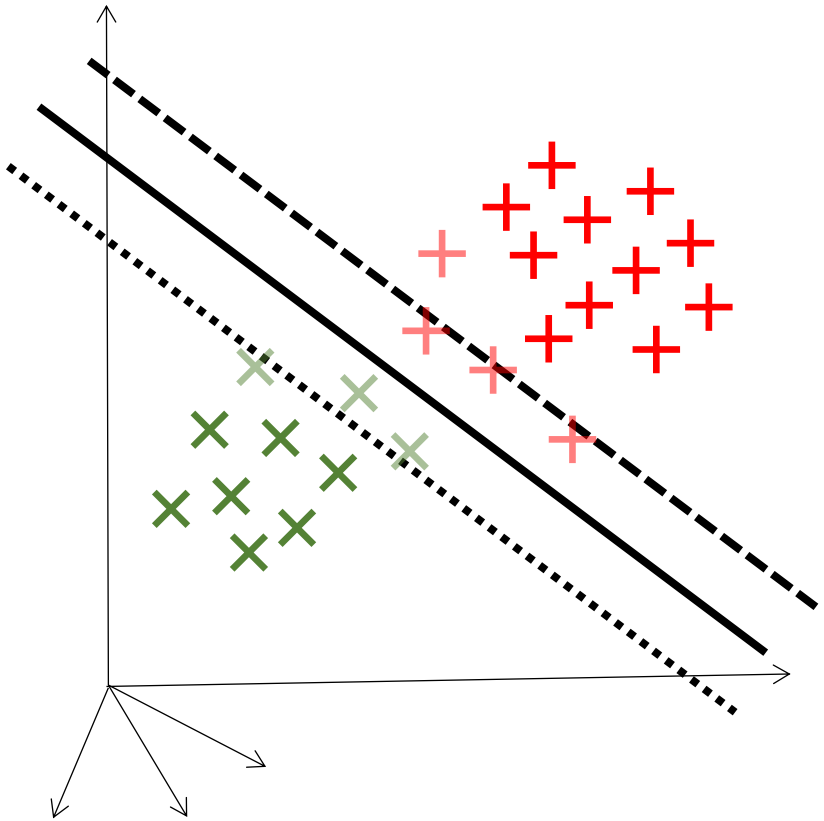


- Bayesian interpretation: Laplacian prior $p(\boldsymbol{\theta}) \propto e^{\beta\|\boldsymbol{\theta}\|_1}$

Noisy input



Noisy input



Add noise to data

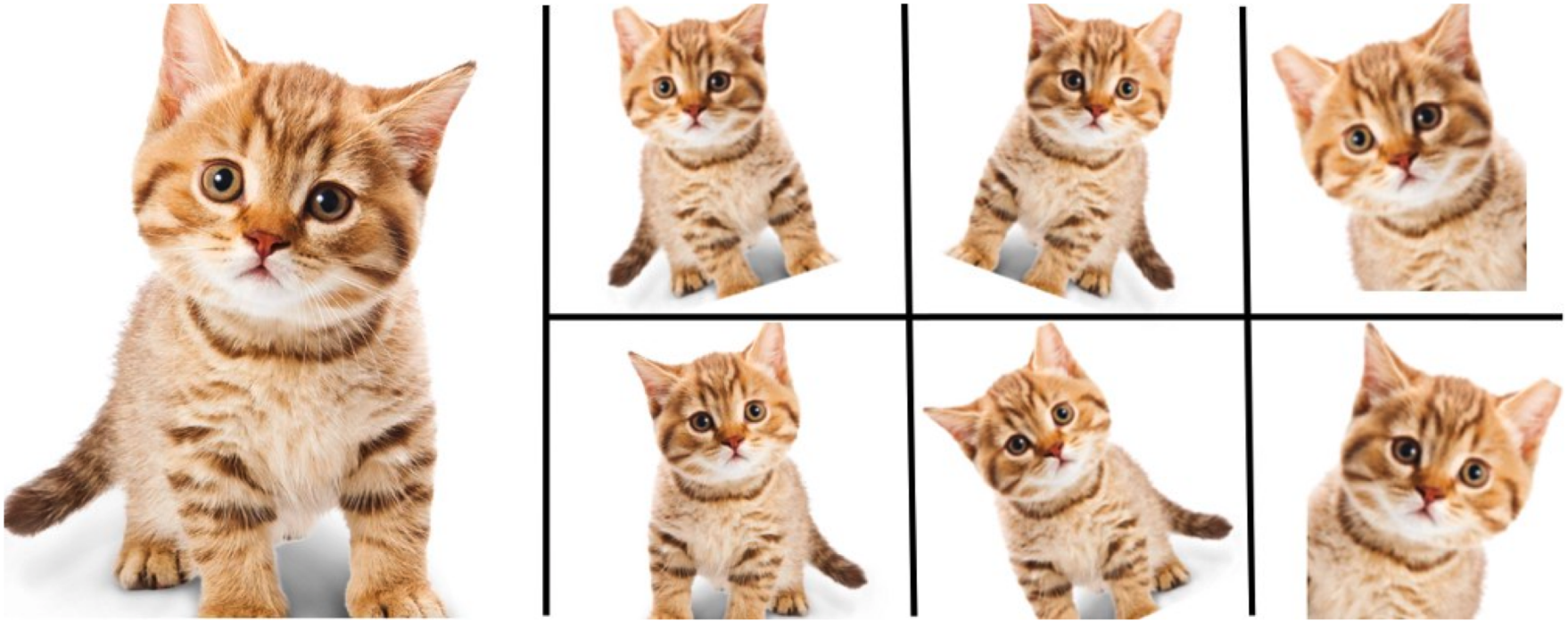
Noisy input

- Add i.i.d. noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \beta \mathbf{I})$ to input
- Loss becomes

$$
\begin{aligned}
L(\mathbf{w}) &= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}(\mathbf{x}+\boldsymbol{\epsilon}) - y\right)^2 \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x} - y\right)^2 + 2\mathbb{E}\,\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\epsilon}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x} - y\right) + \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\epsilon}\right)^2 \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x} - y\right)^2 + \mathbf{w}^{\mathrm{T}}\mathbb{E}\left(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}}\right)\mathbf{w} \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x} - y\right)^2 + \beta\|\mathbf{w}\|^2
\end{aligned}
$$

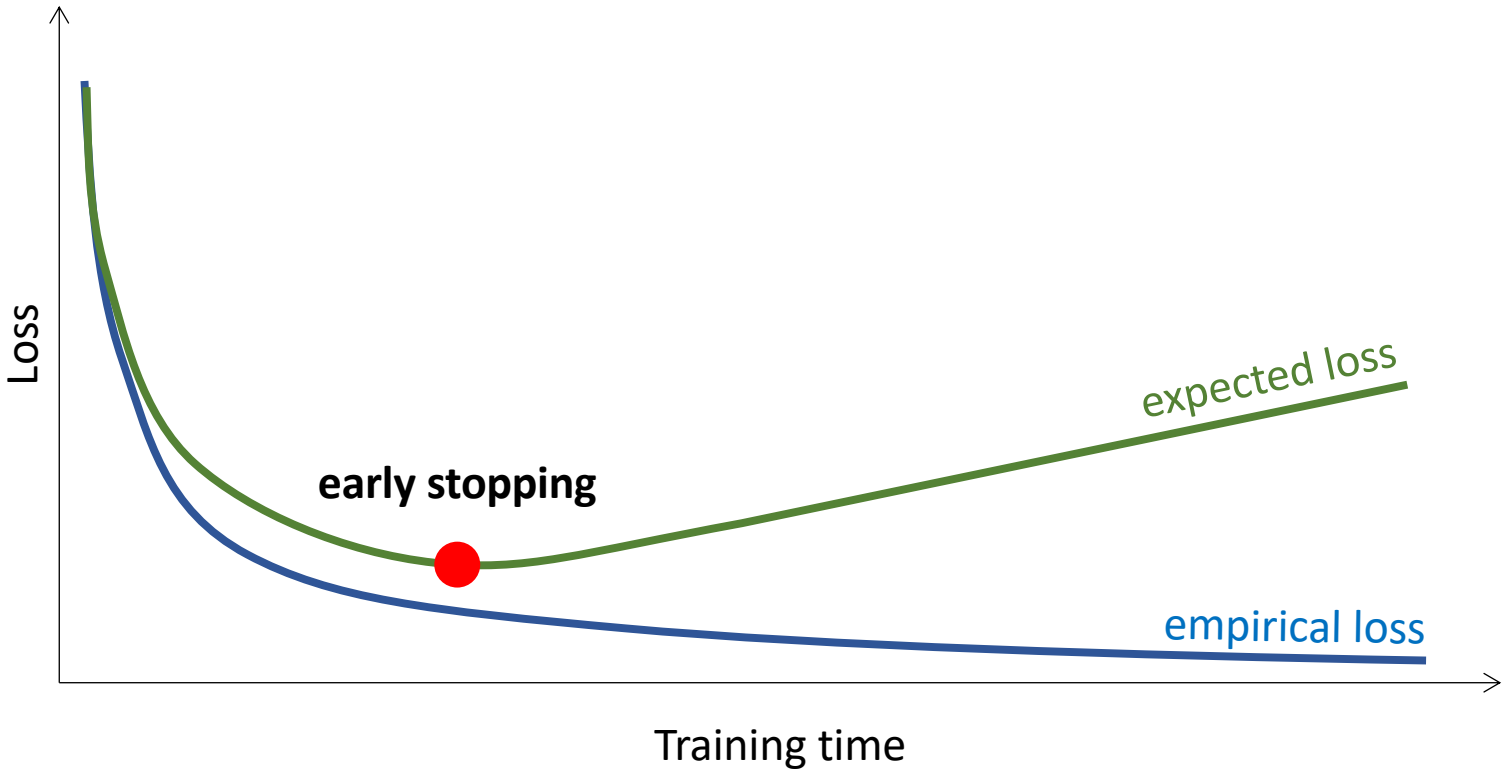
- Equivalent to weight decay!

Data augmentation



Very common in computer vision

Early stopping



Early stopping

- Terminate before reaching convergence
- Effect similar to weight decay regularization (eigenvalue filtering)

Early stopping as regularization

- Gradient of the loss around original problem optimum $\boldsymbol{\theta}^*$

$$\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) \approx \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

- For small gradient descent step α:

$$\begin{aligned} \boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)} - \alpha \nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}^{(k)}\right) \\ &\approx \boldsymbol{\theta}^{(k)} - \alpha \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\left(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^*\right) \end{aligned}$$

Early stopping as regularization

- Gradient of the loss around original problem optimum $\boldsymbol{\theta}^*$

$$\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) \approx \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$$

- For small gradient descent step α:

$$\boldsymbol{\theta}^{(k+1)} - \boldsymbol{\theta}^* \approx \left(\mathbf{I} - \alpha \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\right)\left(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^*\right)$$

- Applying recursively from $\boldsymbol{\theta}^{(k)} = \mathbf{0}$:

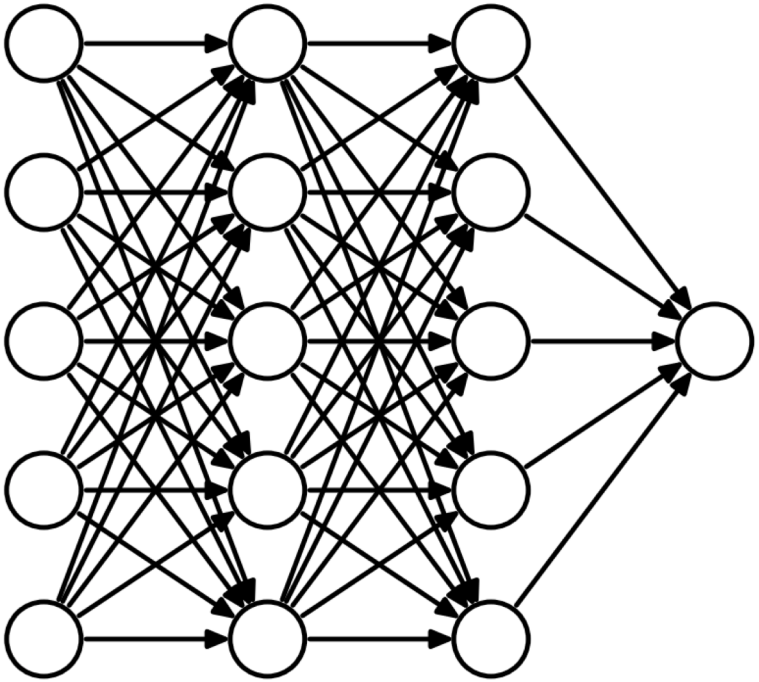
$$\boldsymbol{\theta}^{(k+1)} \approx \left(\mathbf{I} - \left(\mathbf{I} - \alpha \nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\right)^{k+1}\right)\boldsymbol{\theta}^*$$

$$= \tau\left(\nabla^2_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}^*)\right)\boldsymbol{\theta}^*$$

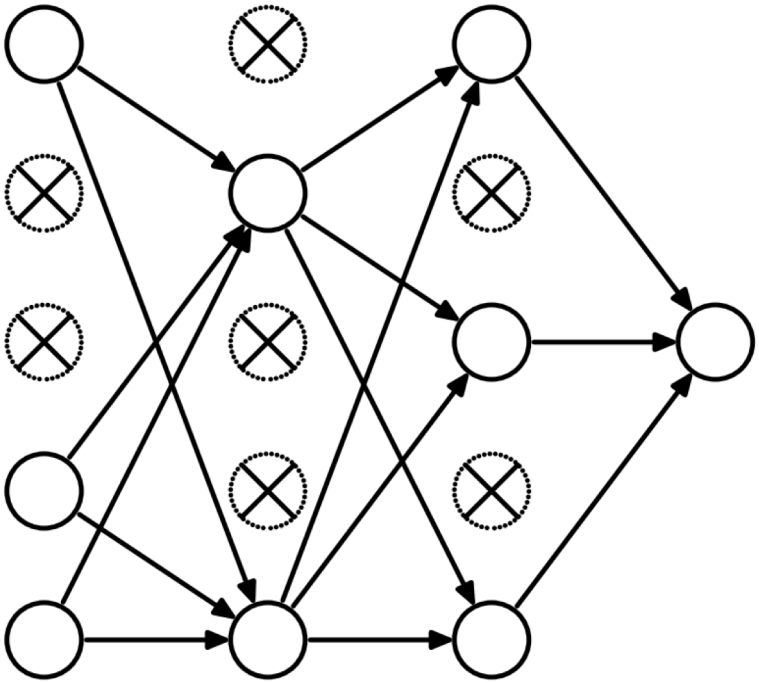
- **Filter of $\boldsymbol{\theta}^*$**
- Bounds on stopping time

Dropout

Dropout



Without dropout



With dropout

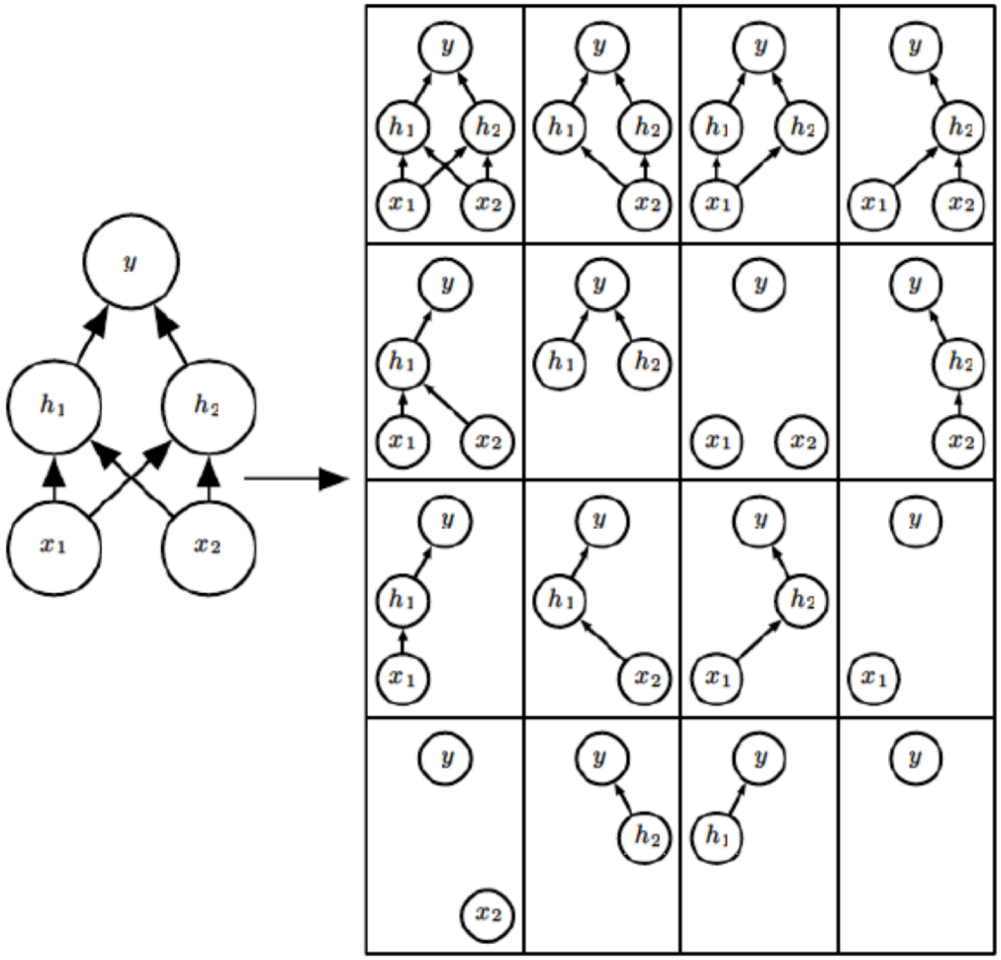
Srivastava et al. 2014

Dropout

- During training, multiply output y_i of neuron i by a mask $m_i \sim \text{Bernoulli}(p)$
- Explore various models with different connectivity
- “Ensemble learning”

Wan et al. 2013

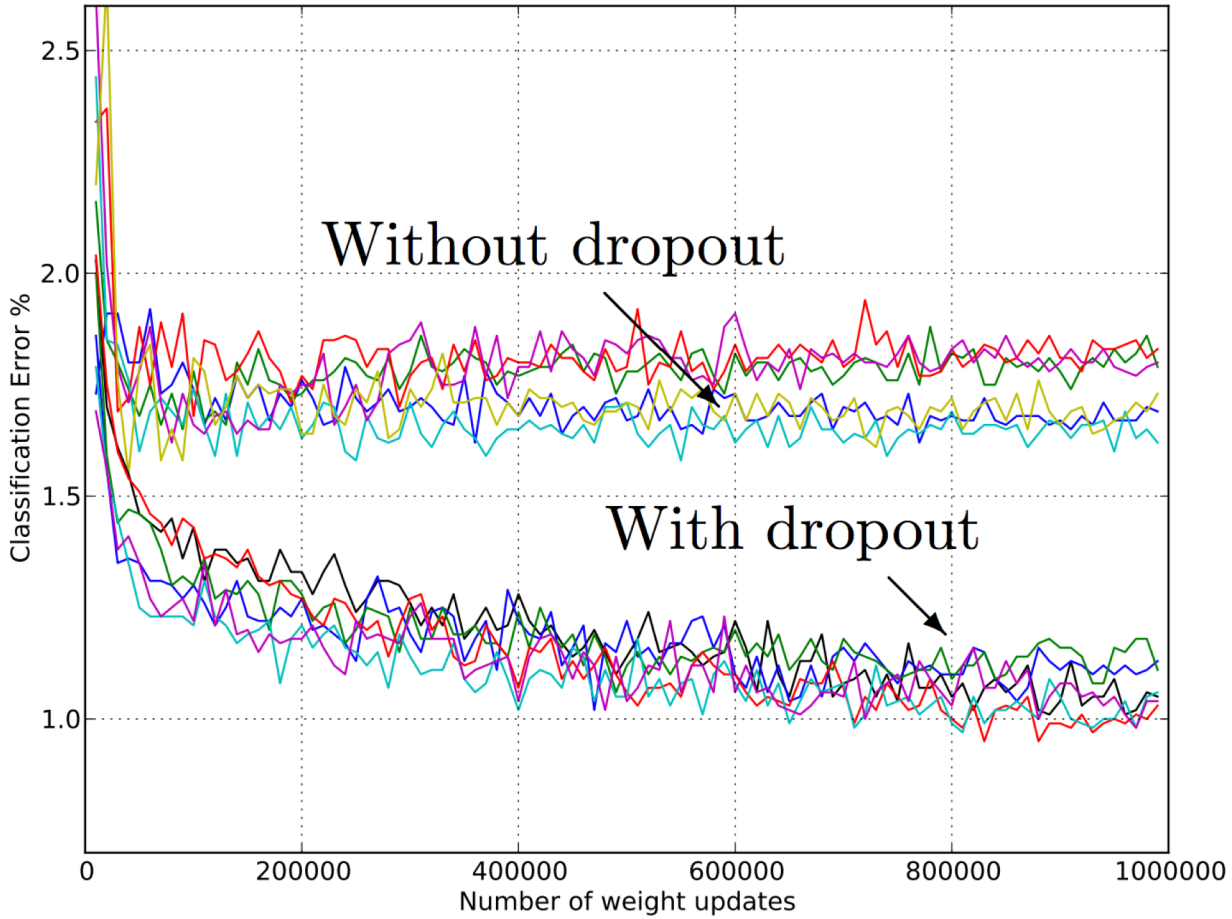
Dropout



"Average on ensemble of models"

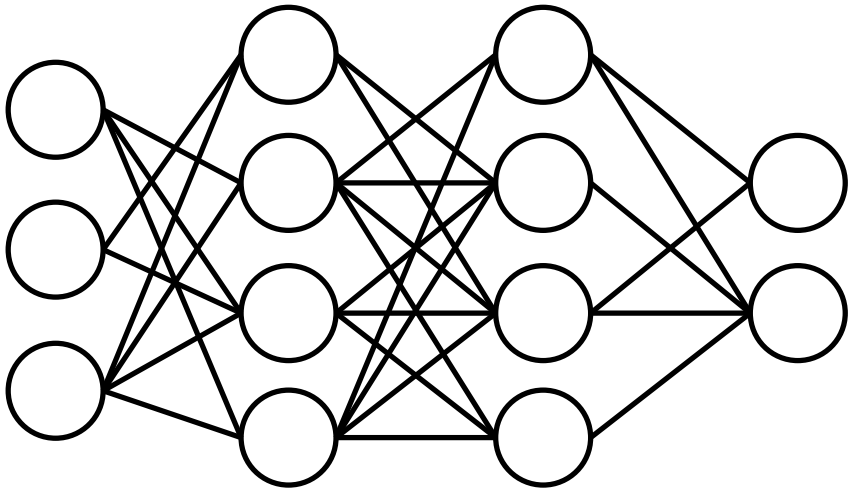
Srivastava et al. 2014

Dropout



Srivastava et al. 2014

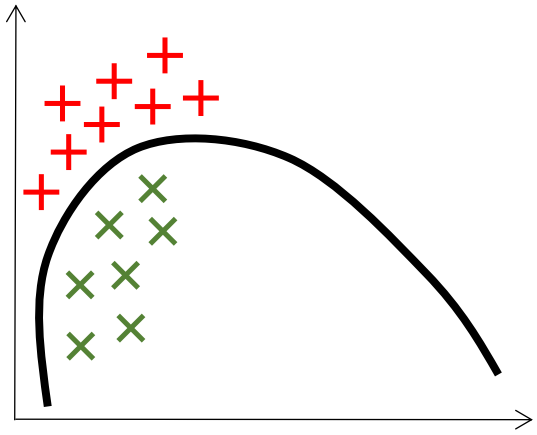
DropConnect



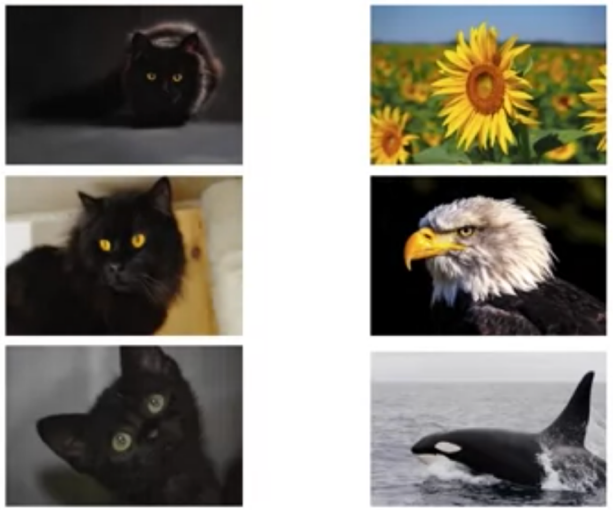
Turn off weights not neurons

Wan et al. 2013

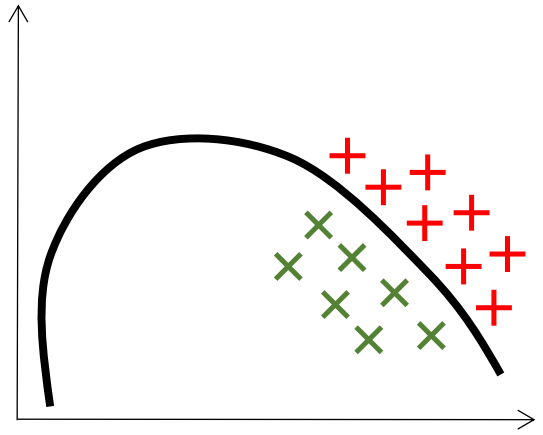
Covariate shift



Training



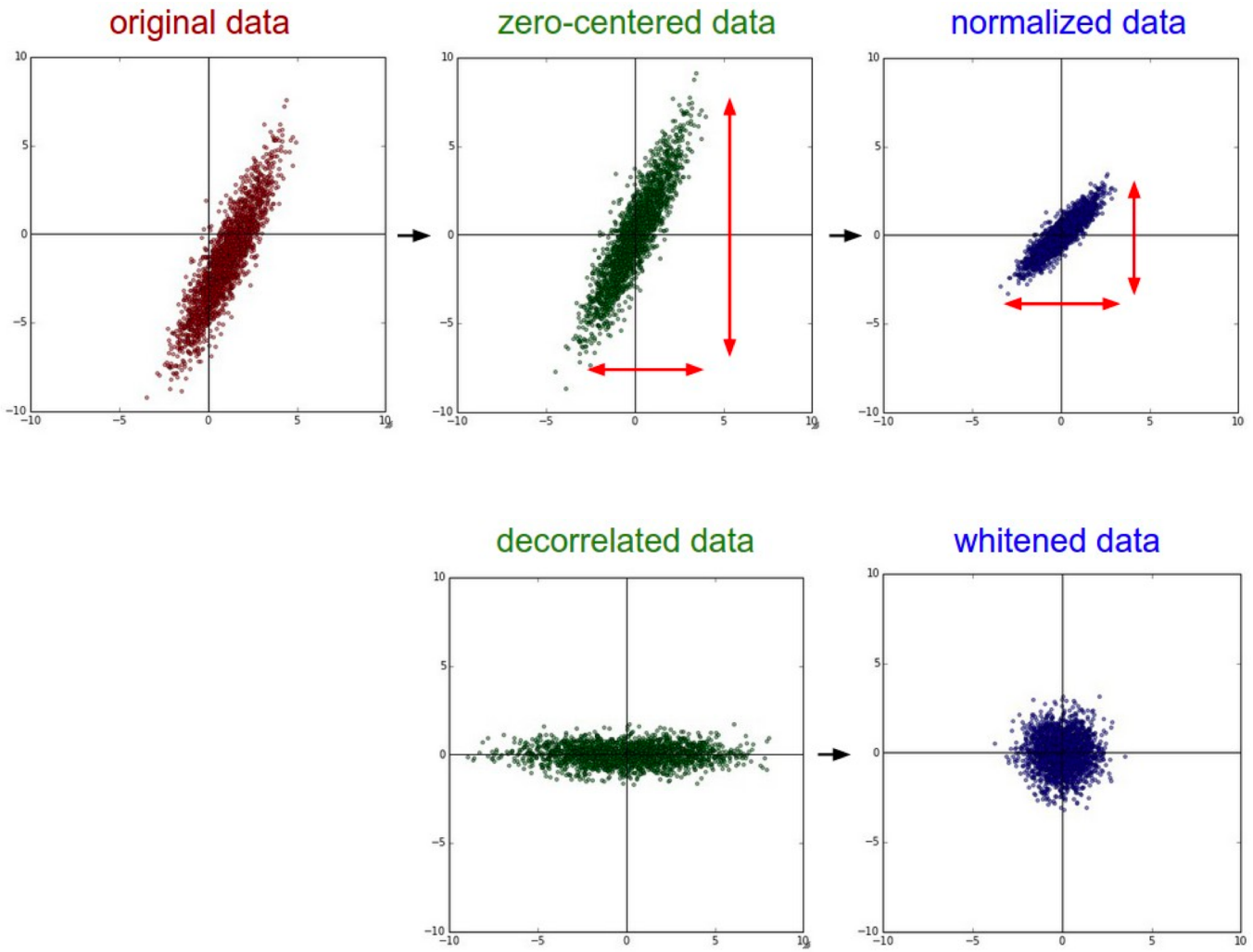
Covariate shift



Testing

Example: A. Ng

Input pre-processing



Batch normalization

- Normalize each input dimension of a layer

$$\bar{x}_i = \frac{x_i - \mathbb{E}x_i}{\sqrt{\mathrm{Var}x_i}}$$

  where mean and variance are computed on a mini-batch

- Shift and scale normalized value by **learnable** parameters

$$y_i = \gamma_i \bar{x}_i + \beta_i$$

Ioffe, Szegedy 2015