

Introduction to Deep Learning



Arijit Mondal

**Dept. of Computer Science & Engineering
Indian Institute of Technology Patna**

`arijit@iitp.ac.in`

Regularization

Introduction

- In machine learning, target is to make an algorithm performs well not only on training data but also on new data
- Many strategies exist to reduce test error at the cost of training error
- Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error
- Objectives
 - To encode prior knowledge
 - Constraints and penalties are designed to express generic preference for simpler model

Regularization in DL

- In DL regularization works by trading increased bias for reduced variance
- Consider the following scenario
 - Excluded the true data generating process
 - Underfitting, inducing bias
 - Matched the true data generating process
 - Desired one
 - Included the generating process but also many other generating process
 - Overfitting, variance dominates
 - Goal of regularizer is to take an model overfit zone to desired zone

Norm penalties

- Most of the regularization approaches are based on limiting the capacity of the model
- Objective function becomes $\tilde{J}(\theta; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \alpha\Omega(\theta)$
 - α — Hyperparameter denotes relative contribution
 - Minimization of \tilde{J} implies minimization of J
 - Ω penalizes only the weight of affine transform
 - Bias remain unregularized
 - Regularizing bias may lead to underfitting

L^2 parameter regularization

- Weights are closer to origin as $\Omega(\theta) = \frac{1}{2}\|\mathbf{w}\|_2^2$
 - Also known as ridge regression or Tikhonov regression
- Objective function $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$

L^2 parameter regularization

- Weights are closer to origin as $\Omega(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{w}\|_2^2$
 - Also known as ridge regression or Tikhonov regression
- Objective function $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- Gradient $\nabla_{\mathbf{w}}\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$

L^2 parameter regularization

- Weights are closer to origin as $\Omega(\theta) = \frac{1}{2}\|\mathbf{w}\|_2^2$
 - Also known as ridge regression or Tikhonov regression
- Objective function $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- Gradient $\nabla_{\mathbf{w}}\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- New weights
$$\mathbf{w} = \mathbf{w} - \epsilon(\alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y}))$$

L^2 parameter regularization

- **Weights are closer to origin as** $\Omega(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{w}\|_2^2$
 - Also known as **ridge regression** or **Tikhonov regression**
- **Objective function** $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- **Gradient** $\nabla_{\mathbf{w}}\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- **New weights**

$$\mathbf{w} = \mathbf{w} - \epsilon(\alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})) = \mathbf{w}(1 - \epsilon\alpha) - \epsilon\nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$$

L^2 parameter regularization

- **Weights are closer to origin as** $\Omega(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{w}\|_2^2$
 - Also known as **ridge regression** or **Tikhonov regression**
- **Objective function** $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- **Gradient** $\nabla_{\mathbf{w}}\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- **New weights**
$$\mathbf{w} = \mathbf{w} - \epsilon(\alpha\mathbf{w} + \nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})) = \mathbf{w}(1 - \epsilon\alpha) - \epsilon\nabla_{\mathbf{w}}J(\mathbf{w}; \mathbf{X}, \mathbf{y})$$
- **Assuming quadratic nature of curve in the neighborhood of**
$$\mathbf{w}^* = \arg \min_{\mathbf{w}} J(\mathbf{w})$$
 - $J(\mathbf{w})$ — **unregularized cost**
 - **Perfect scenario for linear regression with MSE**

Jacobian & Hessian

- Derivative of a function having single input and single output — $\frac{dy}{dx}$
- Derivative of function having vector input and vector output that is, $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$
 - Jacobian $J \in \mathbb{R}^{n \times m}$ of f defined as $J_{i,j} = \frac{\partial}{\partial x_j} f(\mathbf{x})_i$
- Second derivative is also required sometime
 - For example, $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $\frac{\partial^2}{\partial x_i \partial x_j} f$
 - If second derivative is 0, then there is no curvature
- Hessian matrix $H(f)(x)_{ij} = \frac{\partial^2}{\partial x_i \partial x_j} f(\mathbf{x})$

Jacobian & Hessian

- Derivative of a function having single input and single output — $\frac{dy}{dx}$
- Derivative of function having vector input and vector output that is, $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$
 - Jacobian $J \in \mathbb{R}^{n \times m}$ of f defined as $J_{i,j} = \frac{\partial}{\partial x_j} f(\mathbf{x})_i$
- Second derivative is also required sometime
 - For example, $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $\frac{\partial^2}{\partial x_i \partial x_j} f$
 - If second derivative is 0, then there is no curvature
- Hessian matrix $H(f)(x)_{ij} = \frac{\partial^2}{\partial x_i \partial x_j} f(\mathbf{x})$
 - Jacobian of gradient
 - Symmetric

Directional derivative

- The directional derivative of a scalar function $f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$ along a vector $\mathbf{v} = (v_1, \dots, v_n)$ is given by

$$\nabla_{\mathbf{v}}f(\mathbf{x}) = \lim_{h \rightarrow 0} \frac{f(\mathbf{x} + h\mathbf{v}) - f(\mathbf{x})}{h}$$

- If f is differentiable at point \mathbf{x} then

$$\nabla_{\mathbf{v}}f(\mathbf{x}) = \nabla f(\mathbf{x}) \cdot \mathbf{v}$$

Taylor series expansion

- A real valued function differentiable at point x_0 can be expressed as

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \frac{f^{(3)}(x_0)}{3!}(x-x_0)^3 + \dots$$

Taylor series expansion

- A real valued function differentiable at point x_0 can be expressed as

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \frac{f^{(3)}(x_0)}{3!}(x-x_0)^3 + \dots$$

- When input is a vector

$$f(\mathbf{x}) \approx f(\mathbf{x}^{(0)}) + (\mathbf{x} - \mathbf{x}^{(0)})^T \mathbf{g} + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(0)})^T \mathbf{H}(\mathbf{x} - \mathbf{x}^{(0)})$$

- \mathbf{g} — gradient at $\mathbf{x}^{(0)}$, \mathbf{H} — Hessian at $\mathbf{x}^{(0)}$

Taylor series expansion

- A real valued function differentiable at point x_0 can be expressed as

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \frac{f^{(3)}(x_0)}{3!}(x-x_0)^3 + \dots$$

- When input is a vector

$$f(\mathbf{x}) \approx f(\mathbf{x}^{(0)}) + (\mathbf{x} - \mathbf{x}^{(0)})^T \mathbf{g} + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(0)})^T \mathbf{H}(\mathbf{x} - \mathbf{x}^{(0)})$$

- \mathbf{g} — gradient at $\mathbf{x}^{(0)}$, \mathbf{H} — Hessian at $\mathbf{x}^{(0)}$

- If ϵ is the learning rate, then $f(\mathbf{x}^{(0)} - \epsilon \mathbf{g}) \approx f(\mathbf{x}^{(0)}) - \epsilon \mathbf{g}^T \mathbf{g} + \frac{1}{2} \epsilon^2 \mathbf{g}^T \mathbf{H} \mathbf{g}$

Quadratic approximation

- Let $\mathbf{w}^* = \arg \min_{\mathbf{w}} J(\mathbf{w})$ be optimum weights for minimal unregularized cost
- If the objective function is quadratic then

$$\hat{J}(\theta) = J(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^T \mathbf{H}(\mathbf{w} - \mathbf{w}^*)$$

- \mathbf{H} is the Hessian matrix of J with respect to \mathbf{w} at \mathbf{w}^*
- No first order term as \mathbf{w}^* is minimum
- \mathbf{H} is positive semidefinite
- Minimum of \hat{J} occurs when $\nabla_{\mathbf{w}} \hat{J}(\mathbf{w}) = \mathbf{H}(\mathbf{w} - \mathbf{w}^*) = 0$
- With weight decay we have

$$\alpha \tilde{\mathbf{w}} + \mathbf{H}(\tilde{\mathbf{w}} - \mathbf{w}^*) = 0 \Rightarrow (\mathbf{H} + \alpha \mathbf{I})\tilde{\mathbf{w}} = \mathbf{H}\mathbf{w}^* \Rightarrow \tilde{\mathbf{w}} = (\mathbf{H} + \alpha \mathbf{I})^{-1} \mathbf{H}\mathbf{w}^*$$

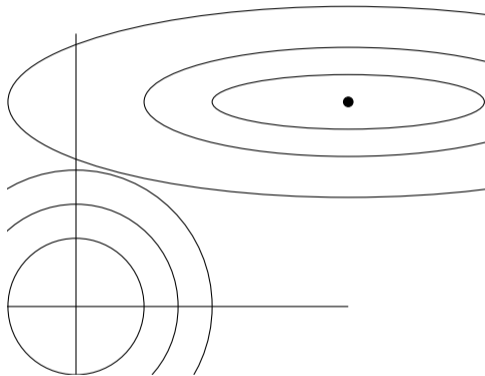
Quadratic approximation (contd)

- As $\alpha \rightarrow 0$, regularized solution $\tilde{\mathbf{w}}$ approaches to \mathbf{w}^*
- As $\alpha \rightarrow \infty$
 - H is symmetric, therefore $H = \mathbf{Q}\Lambda\mathbf{Q}^T$. Now we have

$$\begin{aligned}\tilde{\mathbf{w}} &= (\mathbf{Q}\Lambda\mathbf{Q}^T + \alpha\mathbf{I})^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T\mathbf{w}^* \\ &= [\mathbf{Q}(\Lambda + \alpha\mathbf{I})\mathbf{Q}^T]^{-1}\mathbf{Q}\Lambda\mathbf{Q}^T\mathbf{w}^* \\ &= \mathbf{Q}(\Lambda + \alpha\mathbf{I})^{-1}\Lambda\mathbf{Q}^T\mathbf{w}^*\end{aligned}$$

- Weight decay rescale \mathbf{w}^* along the eigen vector of H
 - Component of \mathbf{w}^* that is aligned to i -th eigen vector, will be rescaled by a factor of $\frac{\lambda_i}{\lambda_i + \alpha}$
 - $\lambda_i \gg \alpha$ — regularization effect is small

L^2 Norm



Linear regression

- For linear regression cost function is $(X\mathbf{w} - \mathbf{y})^T(X\mathbf{w} - \mathbf{y})$
- Using L^2 regularization we have $(X\mathbf{w} - \mathbf{y})^T(X\mathbf{w} - \mathbf{y}) + \frac{1}{2}\alpha\mathbf{w}^T\mathbf{w}$

Linear regression

- For linear regression cost function is $(X\mathbf{w} - \mathbf{y})^T(X\mathbf{w} - \mathbf{y})$
- Using L^2 regularization we have $(X\mathbf{w} - \mathbf{y})^T(X\mathbf{w} - \mathbf{y}) + \frac{1}{2}\alpha\mathbf{w}^T\mathbf{w}$
- Solution for normal equation $\mathbf{w} = (X^T X)^{-1} X^T \mathbf{y}$

Linear regression

- For linear regression cost function is $(Xw - y)^T(Xw - y)$
- Using L^2 regularization we have $(Xw - y)^T(Xw - y) + \frac{1}{2}\alpha w^T w$
- Solution for normal equation $w = (X^T X)^{-1} X^T y$
- Solution for with weight decay $w = (X^T X + \alpha I)^{-1} X^T y$

L^1 regularization

- Formally it is defined as $\Omega(\boldsymbol{\theta}) = \|\mathbf{w}\|_1 = \sum_i |w_i|$
- Regularized objective function will be $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha \|\mathbf{w}\|_1 + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$

L^1 regularization

- Formally it is defined as $\Omega(\theta) = \|\mathbf{w}\|_1 = \sum_i |w_i|$
- Regularized objective function will be $\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha \|\mathbf{w}\|_1 + J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
- The gradient will be $\nabla_{\mathbf{w}} \tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \alpha \mathbf{sign}(\mathbf{w}) + \nabla_{\mathbf{w}} J(\mathbf{w}; \mathbf{X}, \mathbf{y})$
 - Gradient does not scale linearly compared to L^2 regularization
- Taylor series expansion with approximation provides $\nabla_{\mathbf{w}} \hat{J}(\mathbf{w}) = H(\mathbf{w} - \mathbf{w}^*)$
- Simplification can be made by assuming H to be diagonal
 - Apply PCA on the input dataset

L^1 regularization

- Quadratic approximation of L^1 regularization objective function becomes $\hat{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = J(\mathbf{w}^*; \mathbf{X}, \mathbf{y}) + \sum_i \left[\frac{1}{2} H_{i,i} (\mathbf{w}_i - \mathbf{w}_i^*)^2 + \alpha |\mathbf{w}_i| \right]$
- So, analytical solution in each dimension will be $w_i = \text{sign}(w_i^*) \max \left\{ |w_i^*| - \frac{\alpha}{H_{i,i}}, 0 \right\}$
- Consider the situation when $w_i^* > 0$
 - If $w_i^* \leq \frac{\alpha}{H_{i,i}}$, optimal value for w_i will be 0 under regularization
 - If $w_i^* > \frac{\alpha}{H_{i,i}}$, w_i moves towards 0 with a distance equal to $\frac{\alpha}{H_{i,i}}$

Constrained optimization

- Cost function regularized by norm penalty is given by

$$\tilde{J}(\theta; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \alpha\Omega(\theta)$$

- Let us assume $f(\mathbf{x})$ needs to be optimized under a set of equality constraints $g^{(i)}(\mathbf{x}) = 0$ and inequality constraints $h^{(j)}(\mathbf{x}) \leq 0$, then generalized Lagrangian is then defined as

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = f(\mathbf{x}) + \sum_i \lambda_i g^{(i)}(\mathbf{x}) + \sum_j \alpha_j h^{(j)}(\mathbf{x})$$

- If there exists a solution then

$$\min_x \max_{\boldsymbol{\lambda}} \max_{\boldsymbol{\alpha} \geq 0} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = \min_x f(\mathbf{x})$$

- This can be solved by $\nabla_{\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = 0$

Constraint optimization (contd.)

- Suppose $\Omega(\theta) < k$ needs to be satisfied. Then regularization equation becomes

$$L(\theta, \alpha; \mathbf{X}, \mathbf{y}) = J(\theta; \mathbf{X}, \mathbf{y}) + \alpha(\Omega(\theta) - k)$$

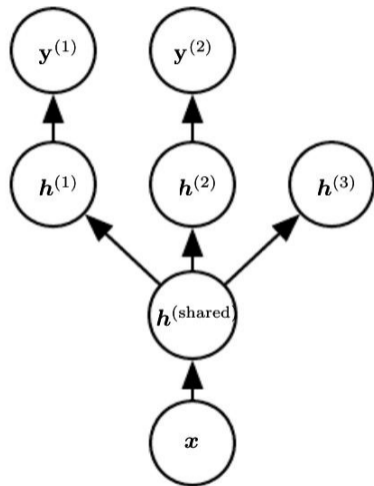
- Solution to the constrained problem

$$\theta^* = \arg \min_{\theta} \max_{\alpha > 0} L(\theta, \alpha)$$

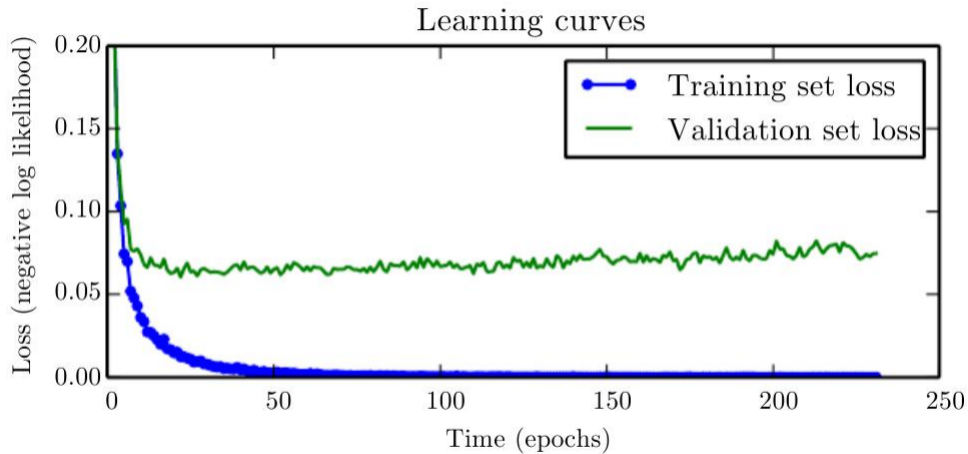
Dataset augmentation

- If data are limited, fake data can be added to training set
 - Computer vision problem
 - Speech recognition
- Easiest for classification problem
- Very effective in object recognition problem
 - Translating
 - Rotating
 - Scaling
 - Need to be careful for 'b' and 'd' or '6' and '9'
- Injecting noise to input data can be viewed as data augmentation

Multitask learning



Early stopping



Early stopping approach

- Initialize the parameters
- Run training algorithm for n steps and update $i = i + n$
- Compute error on the validation set (v')
- If v' is less than previous best, then update the same. Start step 2 again
- If v' is more than the previous best, then increment the count that stores the number of such occurrences. If the count is less than a threshold go to step 2, otherwise exit.

Early stopping (contd)

- Number of training step is a hyperparameter
 - Most hyperparameters that control model capacity have U-shaped curve
- Additional cost for this approach is to store the parameters
- Requires a validation set
 - It will have two passes
 - First pass uses only training data for update of the parameters
 - Second pass uses both training and validation data for update of the parameters
 - Possible strategies
 - Initialize the model again, retrain on all data, train for the same number of steps as obtained by early stopping in pass 1
 - Keep the parameters obtained from the first round, continue training using all data until the loss is less than the training loss at the early stopping point
- It reduces computational cost as it limits the number of iteration
- Provides regularization without any penalty

Early stopping as regularizer

- Let us assume τ training iteration, ϵ learning rate
 - $\epsilon\tau$ — measures effective capacity
- We have, $\hat{J}(\theta) = J(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)H(\mathbf{w} - \mathbf{w}^*)$ and $\nabla_{\mathbf{w}}\hat{J}(\mathbf{w}) = H(\mathbf{w} - \mathbf{w}^*)$
- Assume $\mathbf{w}^{(0)} = 0$

Early stopping as regularizer (contd.)

- **Approximate behavior of gradient descent provides**

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \mathbf{H}(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \mathbf{H}(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{H})(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \mathbf{H}(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{H})(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T)(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \mathbf{H}(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{H})(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T)(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{Q}^T(\mathbf{w}^{(\tau)} - \mathbf{w}^*) = (\mathbf{I} - \epsilon \mathbf{\Lambda}) \mathbf{Q}^T(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

Early stopping as regularizer (contd.)

- Approximate behavior of gradient descent provides

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \nabla_{\mathbf{w}} \hat{J}(\mathbf{w}^{(\tau-1)})$$

$$\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \epsilon \mathbf{H}(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{H})(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{w}^{(\tau)} - \mathbf{w}^* = (\mathbf{I} - \epsilon \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T)(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{Q}^T(\mathbf{w}^{(\tau)} - \mathbf{w}^*) = (\mathbf{I} - \epsilon \mathbf{\Lambda}) \mathbf{Q}^T(\mathbf{w}^{(\tau-1)} - \mathbf{w}^*)$$

$$\mathbf{Q}^T \mathbf{w}^{(\tau)} = [\mathbf{I} - (\mathbf{I} - \epsilon \mathbf{\Lambda})^\tau] \mathbf{Q}^T \mathbf{w}^*$$

Early stopping as regularizer (contd)

- Assuming $\mathbf{w}^{(0)} = \mathbf{0}$ and ϵ is small value such that $|1 - \epsilon\lambda_i| < 1$
- From L^2 regularization, we have

$$\mathbf{Q}^T \tilde{\mathbf{w}} = (\mathbf{\Lambda} + \alpha \mathbf{I})^{-1} \mathbf{\Lambda} \mathbf{Q}^T \mathbf{w}^*$$

$$\mathbf{Q}^T \tilde{\mathbf{w}} = [\mathbf{I} - (\mathbf{\Lambda} + \alpha \mathbf{I})^{-1} \alpha] \mathbf{Q}^T \mathbf{w}^*$$

- Therefore we have, $(\mathbf{I} - \epsilon \mathbf{\Lambda})^\tau = (\mathbf{\Lambda} + \alpha \mathbf{I})^{-1} \alpha$
- Hence, $\tau \approx \frac{1}{\epsilon \alpha}$, $\alpha \approx \frac{1}{\tau \epsilon}$

Bagging

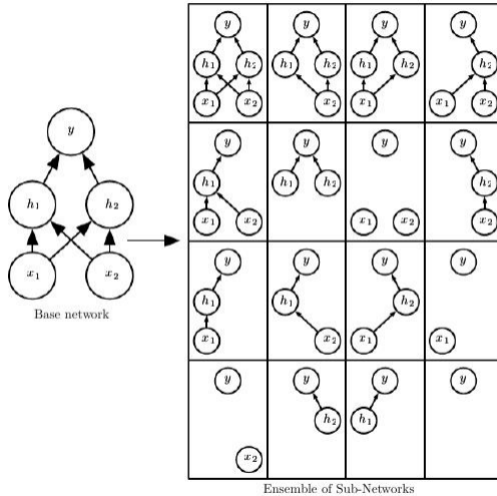
- Also known as Bootstrap aggregating
- Reduces generalization error by combining several models
- Train multiple models then vote on output for the test example
 - Also known as model averaging, ensemble method
- Suppose we have k regression model and each model makes an error ϵ_i such that $\mathbb{E}(\epsilon_i) = 0$, $\mathbb{E}(\epsilon_i^2) = v$, $\mathbb{E}(\epsilon_i \epsilon_j) = c$
- Error made by average prediction $\frac{1}{k} \sum_i \epsilon_i$

Bagging (contd.)

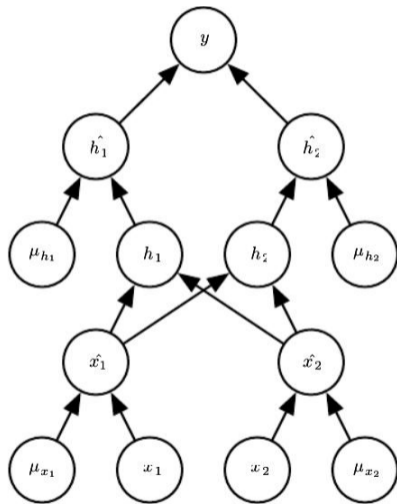
- Expected mean square error

$$\mathbb{E} \left[\left(\frac{1}{k} \sum_i \epsilon_i \right)^2 \right] = \frac{1}{k^2} \mathbb{E} \left[\sum_i \left(\epsilon_i^2 + \sum_{i \neq j} \epsilon_i \epsilon_j \right) \right] = \frac{v}{k} + \frac{k-1}{k} c$$

Dropout



Dropout



Adversarial training

