Chapter 7

Jiankou Li

2017 年 2 月 27 日

- 正则化：旨在减少学习算法泛化误差而不是训练误差的任何修改
- 相关概念：泛化、欠拟合、过拟合、偏差、方差
- 形式：向模型添加额外约束、向目标函数增加额外项、集成方法

Chap7-Introduction

- 欠拟合和偏差：不包括真实数据生成过程-
- 匹配真实数据生成过程
- 过拟合和方差：除了包含真实数据生成过程，还包含许多其它的生成过程

Chap7-Introduction

- fit a square peg into a round hole
- 持方枘而欲内圆凿
- 控制模型复杂性不是找到合适的模型，而是一个适当正则化的大型模型

Chap7-Introduction

- l_1 and l_2
- Early Stopping
- Dropout
- Data augmentation and Adversarial Training
- Manifold (Tangent Prop and Double Prop)
- Multi-Task Learning、Semi-Supervised Learning and Parameter Sharing

- Cost function

$$J'(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta) \tag{1}$$

- w and θ
- Number of hypeparameter

- l_2 regularized objection function

$$J'(w) = J(w) + \frac{\alpha}{2}||w||_2^2 \tag{2}$$

- Gradient

$$\nabla_w J'(w) = \alpha w + \nabla J_w(w) \tag{3}$$

- A single gradient step to update

$$w \leftarrow w - \epsilon(\alpha w + \nabla J_w(w)) \tag{4}$$

- Written in another way

$$w \leftarrow (1 - \epsilon\alpha)w - \epsilon\nabla_w J(w) \tag{5}$$

- Quadratic approximation

$$J'(w) = J(w^*) + \frac{1}{2}(w - w^*)^T H(w - w^*) \tag{6}$$

where $w^* = argmin_w J(w)$

- Gradient

$$\nabla J'(w) = H(w - w^*) \tag{7}$$

- Add weight decay and solve the minimum of the regularized version of J'

$$\alpha w + H(w - w^*) = 0 \tag{8}$$

- We have

$$w' = (H + \alpha I)^{-1} H w^* \tag{9}$$

- Decompose H, $H = Q\Lambda Q^T$, we have

$$\begin{aligned} w' &= (H+\alpha I)^{-1} H w^* \\ &= (Q\Lambda Q^T + \alpha Q Q^T)^{-1} Q\Lambda Q^T w^* \\ &= [Q(\Lambda + \alpha I) Q^T]^{-1} Q\Lambda Q^T w^* \\ &= Q(\Lambda + \alpha I)^{-1} \Lambda Q^T w^* \end{aligned} \tag{10}$$

- We have

$$Q^T w' = (\Lambda + \alpha I)^{-1} \Lambda Q^T w^* \tag{11}$$

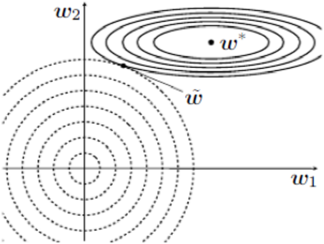

Figure: l_2 regularization (Figure 7.1)

$$\gamma = \sum_i \frac{\lambda_i}{\lambda_i + \alpha} \tag{12}$$

特征值代表了曲率，每个方向的衰减系数为$\frac{\lambda_i}{1+\lambda_i}$，特征值越大表示相应方向坡度越陡，衰减程度比较小；相反，特征越小表示相应方向的坡度越平滑，相应的梯度对数据越不敏感，衰减程度越大。L_2正则衰减了对数据不敏感的方向，保留了受数据影响大的方向。

- Cost function for linear regression

$$L(w) = \frac{1}{2}\sum_{n=1}^{N}(w^T\phi(x) - y)^2 = (Xw - y)^T(Xw - y) \tag{13}$$

- l_2 regularized cost functdion

$$L'(w) = (Xw - y)^T(Xw - y) + \frac{1}{2}\alpha w^T w \tag{14}$$

- The solution changes from

$$w = (X^T X)^{-1} X^T y \tag{15}$$

- To

$$w = (X^T X + \alpha I)^{-1} X^T y \tag{16}$$

- A scala g is a subgradient of $f(\theta)$ if it follows:

$$f(\theta) - f(\theta_0) \geq g(\theta - \theta_0) \; \forall \theta \in I \tag{17}$$

where I is some interval containing θ_0

- The subgradients of the function f at θ_0 is defined a set [a,b], denoted $|\partial f(\theta)|_{\theta_0}$

$$\text{where } a = \lim_{\theta \to \theta_0^-} \frac{f(\theta) - f(\theta_0)}{\theta - \theta_0}, \; b = \lim_{\theta \to \theta_0^+} \frac{f(\theta) - f(\theta_0)}{\theta - \theta_0}, \tag{18}$$

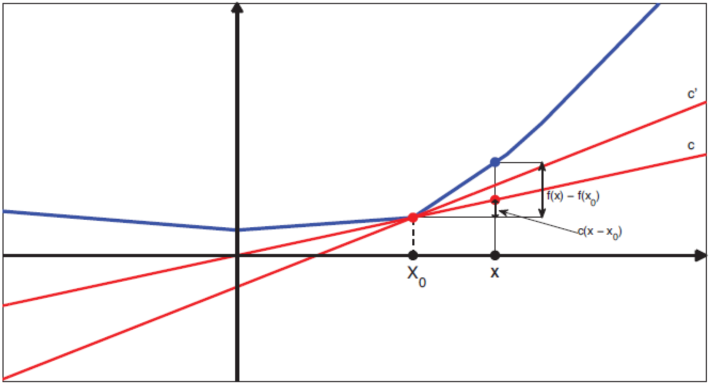

Figure: Subgradient[1]

[1] Mlapp Figure 13.4 http://en.wikipedia.org/wiki/Subderivative

- Let $f(\theta) = |\theta|$, the subgradients is given by

$$\partial f(\theta) = \begin{cases} \{-1\} & \text{if } \theta < 0 \\ [-1, 1] & \text{if } \theta = 0 \\ \{1\} & \text{if } \theta > 0 \end{cases} \tag{19}$$

- Cost function for regression:

$$J(w) = (\sum_{i=1}^{N} w_i x_i - y)^2 = (w^T x - y)^2 \tag{20}$$

- Partial derivative:

$$\frac{\partial J(w)}{\partial w_j} = a_j w_j - c_j \tag{21}$$

where $a_j = 2x_j^2, c_j = 2x_j(y - w_{-j}^T x_{-j})$

- Adding in the penalty term:

$$\begin{aligned}
\partial_{w_j} J(w) &= (a_j w_j - c_j) + \alpha \partial_{w_j} ||w||_1 && (22)\\
&= \begin{cases} \{a_j w_j - c_j - \alpha\} & \text{if } w_i < 0 \\ [-c_j - \alpha, -c_j + \alpha] & \text{if } w_i = 0 \\ \{a_j w_j - c_j + \alpha\} & \text{if } w_i > 0 \end{cases} && (23)
\end{aligned}$$

- Solutions of w:

$$w_j = \begin{cases} \dfrac{c_j + \alpha}{a_j} & \text{if } c_j < -\alpha \\ 0 & \text{if } c_j \in [-\alpha, \alpha] \\ \dfrac{c_j - \alpha}{a_j} & \text{if } c_j > \alpha \end{cases} \qquad (24)$$

- We have

$$w_j = \text{soft}(\frac{c_j}{a_j}, \frac{\lambda}{a_j}), \tag{25}$$

where

$$\text{soft}(a, b) = \text{sign}(a) max\{(|a| - b, 0\} \tag{26}$$

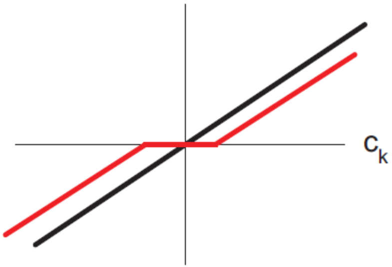

Figure: Soft thresholding[2], the flat region is interval $[-\alpha, \alpha]$

[2]Machine learning a probabilistic perspective Figure 13.5

- Cost function for l_1 regularization:

$$J(w) = J(w^*) + \sum_i [\frac{1}{2} H_{i,i}(w_i - w_i^*)^2 + \alpha |w_i|] \tag{27}$$

- Subgradients:

$$
\begin{aligned}
\frac{\partial J(w)}{\partial w_i} &= H_{i,i}(w_i - w_i^*) + \lambda \nabla_{w_i} |w_i| \qquad (28) \\
&= \begin{cases} \{H_{i,i}(w_i - w_i^*) - \lambda\} & \text{if } w_i < 0 \\ [-H_{i,i}w_i^* - \lambda, -H_{i,i}w_i^* + \lambda] & \text{if } w_i = 0 \\ \{H_{i,i}(w_i - w_i^*) + \lambda\} & \text{if } w_i < 0 \end{cases}
\end{aligned}
$$

- Solution for minimizing function:

$$
\begin{aligned}
w_j &= \begin{cases} w_i^* + \frac{\alpha}{H_{i,i}} & \text{if } w_i^* < -\frac{\alpha}{H_{i,i}} \\ 0 & \text{if } w_i \in [-\frac{\alpha}{H_{i,i}}, \frac{\alpha}{H_{i,i}}] \\ w_i^* - \frac{\alpha}{H_{i,i}} & \text{if } w_i^* > \frac{\alpha}{H_{i,i}} \end{cases} \qquad (29) \\
&= \text{sign}(w_i^*)\text{max}\{|w_i^*| - \frac{\alpha}{H_{i,i}}, 0\}
\end{aligned}
$$

Chap 7-l_1 Parameter Regularization

- l_2 weight decay
- l_1 weight decay and feature selection[3]

[3] Stability selection N.Meinshausen, P.Buhlmann

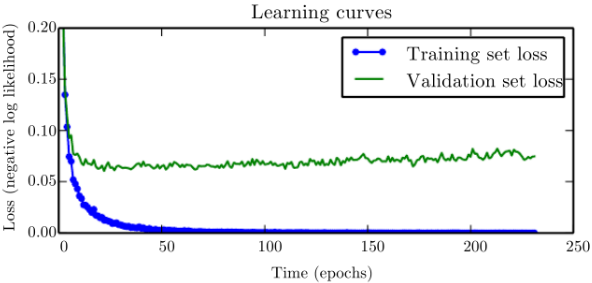

Figure: Learning curves[4]

[4] Deep Learning Figure 7.3

Chap7-Early Stopping

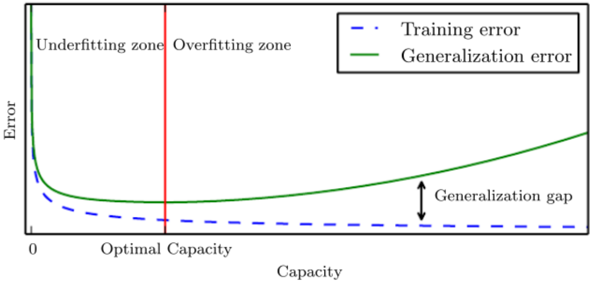

Figure: Typical relationship between capacity and error[5]

[5]Deep Learning Figure 5.3

Chap7-Early Stopping

- Single run : many values
- Separate Cpu
- Small validation set
- Evaluate less frequently

- Using all Data:
    - Retrain on all of the data
    - Continue training using all of the data
- Reducing the computational cost:
    - Limiting the number of training iterations
    - Without computation of additional terms

Chap7-Early Stopping

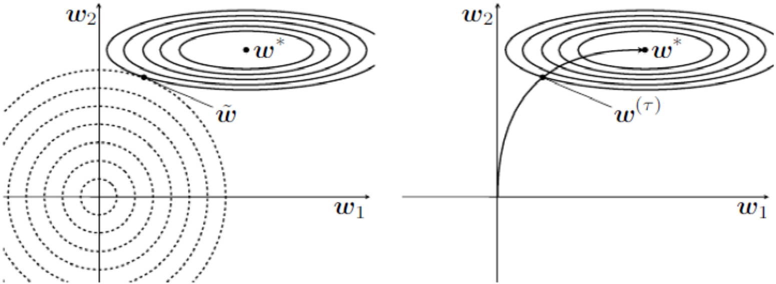

Figure: Early Stopping acts as a regularizer[6]

[6]Deep Learning Figure 7.4

Chap7-Early Stopping

- Quadratic approximation:

$$J'(w) = J(w^*) + \frac{1}{2}(w - w^*)^T H(w - w^*) \tag{30}$$

- Gradient descent

$$w^\tau = w^{\tau-1} - \epsilon \nabla J'(w) = w^{\tau-1} - \epsilon H(w^{\tau-1} - w^*) \tag{31}$$

$$w^\tau - w^* = (I - \epsilon H)(w^{\tau-1} - w^*) \tag{32}$$

- Eigen decomposition $H = Q\Lambda Q^T$

$$w^\tau - w^* = (I - \epsilon Q\Lambda Q^T)(w^{\tau-1} - w^*) \tag{33}$$

$$Q^T(w^\tau - w^*) = (I - \epsilon\Lambda)Q^T(w^{\tau-1} - w^*) \tag{34}$$

- After τ iterations

$$Q^T w^\tau = [I - (I - \epsilon\Lambda)^\tau]Q^T w^* \tag{35}$$

- l_2 regularization

$$Q^T w = (\Lambda + \alpha I)^{-1} \Lambda Q^T w^* \quad (36)$$

$$Q^T w = [I - (\Lambda + \alpha I)^{-1} \alpha] Q^T w^* \quad (37)$$

- After τ iterations

$$Q^T w^\tau = [I - (I - \epsilon\Lambda)^\tau] Q^T w^* \quad (38)$$

- if follows satisfies

$$(I - \epsilon\Lambda)^\tau = (\Lambda + \alpha I)^{-1} \alpha \quad (39)$$

then they are equivalent;

- we need:

$$(1 - \epsilon\lambda_i)^\tau = (\lambda_i + \alpha)^{-1} \alpha \quad (40)$$

$$\tau \ln(1 - \epsilon\lambda_i) = -\ln(1 + \frac{\lambda_i}{\alpha}) \quad (41)$$

$$-\tau\epsilon\lambda_i \approx -\frac{\lambda_i}{\alpha} \quad (42)$$

- Finally we need:

$$\epsilon\tau \approx \frac{1}{\alpha} \quad (43)$$

- Early Stopping automatically determines the correct amount of regularization
- l_2 regularization requires many training experiments with different values of its hyperparameter.

- Bagging: a technique for reducing generalization error by combing models
- Suppose each model makes an error ϵ_i on each example, $\epsilon \sim N(0, \Sigma)$,

$$E((\frac{1}{k}\sum_i \epsilon_i)^2) = \frac{1}{\epsilon_i^2}E(\sum_i(\epsilon_i^2 + \sum_{i \neq j}\epsilon_i\epsilon_j)) = \frac{1}{k}v + \frac{k-1}{k}c \tag{44}$$

where $E[\epsilon_i^2] = v, E[\epsilon_i\epsilon_j] = c$

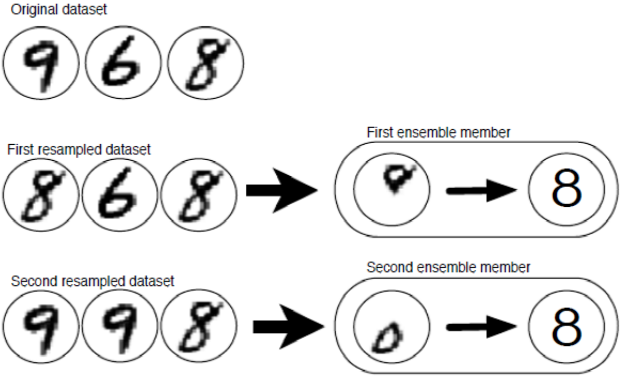

Figure: Bagging[7]

[7]Deep Learning Figure 7.5

- 神经网络的特点：参数多，表达能力强，容易过拟合；
- Dropout，在每次迭代中按一定概率将网络中的节点及其相应的边临时去掉，可以看作神经网络中的装袋算法，特点是不增加额外的大量运算，可以应用到很多模型；
- Motivation：与其它任意基因组合都能发挥作用的基因更具有鲁棒性，相应与任意隐单元组合都能学到好特征的隐单元泛化能力更强；

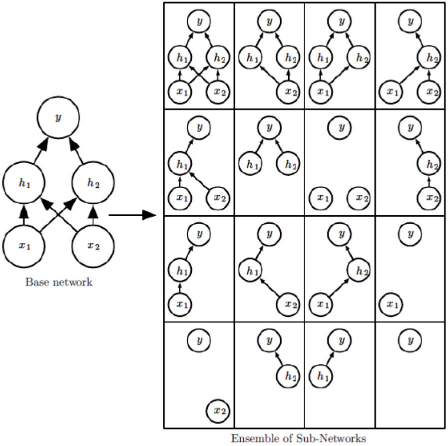

Figure: Dropout[8]

[8] Deep Learning Figure 7.6

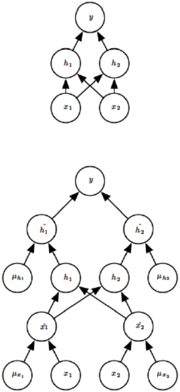

Figure: Dropout[9]

[9] Deep Learning Figure 7.7

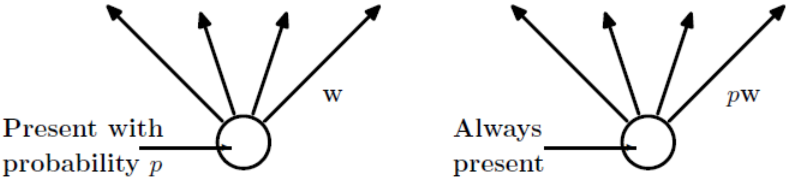

Figure: Left:train， Right:test,[10]

- 训练时，相当于从2^n个网络中采样一个进行训练；
- 测试时，所有的隐单元都保留，每个边的权重变为pw；

[10] Dropout: A simple Way to Prevent Neural Networks from Overfitting Fig2

- Standard network

$$
\begin{aligned}
z_{l+1} &= w_{l+1}^T y_l + b_{l+1} \\
y_{l+1} &= f(z_{l+1})
\end{aligned}
$$

- Dropout network

$$
\begin{aligned}
r_l &\sim Bernoulli(p) \\
y'_l &= r_l * y_l \\
z_{l+1} &= w_{l+1} y' + b_{l+1} \\
y_{l+1} &= f(z_{l+1})
\end{aligned}
$$

Chap7-Dropout-Softmax Regression

- Softmax

$$p(t = y|v) = \text{softmax}(w^T v)_y$$

$$p(t = y|v; d) = \text{softmax}(w^T (d * v))_y$$

- Prediction

$$p(t = y|v) = \frac{P'(t = y|v)}{\sum_{y'} p(t = y'|v)}$$

where $p'(t = y|v) = \sqrt[2^2]{\prod_{d \in \{0,1\}^2} p(t = y|v, d)}$

- Softmax

$$p'(t=y|v) = \sqrt[2^n]{\prod_{d\in\{0,1\}^2} p(t=y|v,d)} \quad (45)$$

$$= \sqrt[2^n]{\prod_{d\in\{0,1\}^n} \text{softmax}(w^T(d*v))_y} \quad (46)$$

$$= \sqrt[2^n]{\prod_{d\in\{0,1\}^n} \frac{\exp(w_{y,:}^T(d*v))}{\sum_{y'} \exp(w_{y',:}^T(d*v))_y}} \quad (47)$$

$$= \frac{\sqrt[2^n]{\prod_{d\in\{0,1\}^n} \exp(w_{y,:}^T(d*v))}}{\sqrt[2^n]{\prod_{d\in\{0,1\}^n} \sum_{y'} \exp(w_{y',:}^T(d*v))_y}} \quad (48)$$

- Softmax

$$
\begin{aligned}
p'(t=y|v) &\propto \sqrt[2^n]{\prod_{d\in\{0,1\}^n} \exp(w_{y,:}^T(d*v))} &(49)\\
&= \exp\Big(\frac{1}{2^n}\sum_{d\in\{0,1\}^n} w_{y,:}^T(d*v)\Big) &(50)\\
&= \exp\Big(\frac{1}{2}w_{y,:}^T v\Big) &(51)
\end{aligned}
$$

- Error Function

$$||y - Xw||^2$$

- Error Function for dropout

$$E_R[||y - (R * X)w||^2] \text{ where } R_{ij} \sim Bernoulli(p)$$

- Reduces to

$$||y - pXw||^2 + p(1-p)||\Gamma w||^2 \text{ where } \Gamma = (diag(X^T X))^{1/2}$$

- Absorb p into w

$$||y - Xw'||^2 + \frac{1-p}{p}||\Gamma w'||^2 \text{ where } w' = pw$$

- Drop Connect(Wan et al. 2013), Stochastic pooling(Sec,9.3)
- Any kind of random
- Ensemble of models that share hidden units
- Destroying extracted features rather than original values

- 使用更多的数据是让模型泛化得更好的最好方法；
- 数据集增加：创建假数据并添加到训练集；
- 适用场景：图象识别、语音识别；
- 图像识别：平移、旋转、缩放；
- 注入噪声：在神经网络输入层、隐藏层注入噪声是数据增强的一种形式，可以极大减少泛化误差；

Chap7-Adversarial Training

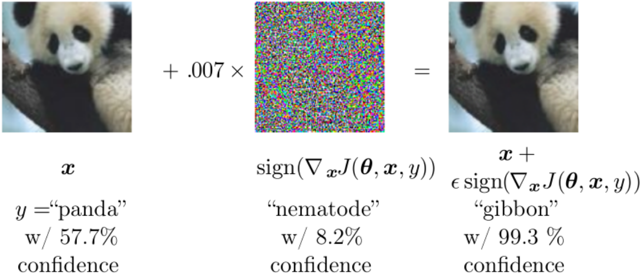

Figure: Adversarial example[11]

[11] Deep Learning Figure 7.8

- 现象：在许多情况下，x'与x非常近似，人类感觉不到差异，网络会给出非常不同的预测
- 原因：过度线性, 每个输入改变ϵ , 那么线性函数可以改变$\epsilon||w||_1$
- 对抗训练有助于说明积极正则化与大型函数族结合的力量

- 流形：连接在一起的区域；数学上，它是指一组点，且每个点都有其邻域；
- 流形学习：概率质量高度集中；

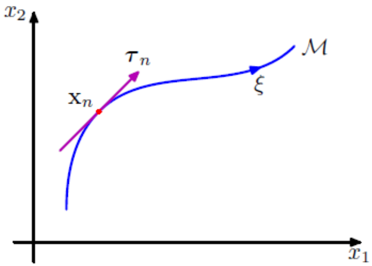

Figure: Tangent propagation[12]

[12] Pattern Recognition and Machine Learning Figure 5.15

- One dimensional manifold M is parameterized by z, $s(x,z)$ is a vector acting on vector x， we have $s(x,0)=x$，
- Tangent vector at point x：

$$\tau = \frac{\partial s(x,z)}{\partial z}|_{z=0} \tag{52}$$

- Derivative of output with respect z：

$$\frac{\partial y}{\partial z}|_{z=0} = \sum_{i=1}^{D} \frac{\partial y}{\partial x_i}\frac{\partial x_i}{\partial z}|_{z=0} = \tau^{\mathrm{T}} u \tag{53}$$

where $u = \nabla y(x)$

- Cost function

$$E' = E + \lambda\Omega \tag{54}$$

where

$$\Omega = \frac{1}{2}(\frac{\partial y}{\partial z}|_{z=0})^2 = \frac{1}{2}(\tau^{\mathrm{T}} u)^2 \tag{55}$$

λ is a balance parameter

- Double backprop (Drucker and LeCnn 1992) regularizes the Jacobian to be small.

	explicit	implicit
specified direction	dataset augmentation	tangent propagation
all directions	adversarial training	double backprop

Table: Four regularization strategy

Chap7-Summary

- Parameter regularization l_1 and l_2
- Early Stopping
- Bagging, Dropout
- Data augmentation and Adversarial Training
- Manifold (Tangent Prop and Double Prop)

Thank You!