神经网络的训练技巧

主讲：王皓 副研究员| 硕士导师

邮箱：wanghao3@ustc.edu.cn

主页：http://staff.ustc.edu.cn/~wanghao3

本章内容

➢归一化（normalization）
- 数据归一化
- 逐层归一化

➢训练目标和方式
- 多任务、迁移学习、课程学习

➢数据增强

➢正则化
- Early stop
- Dropout

归一化

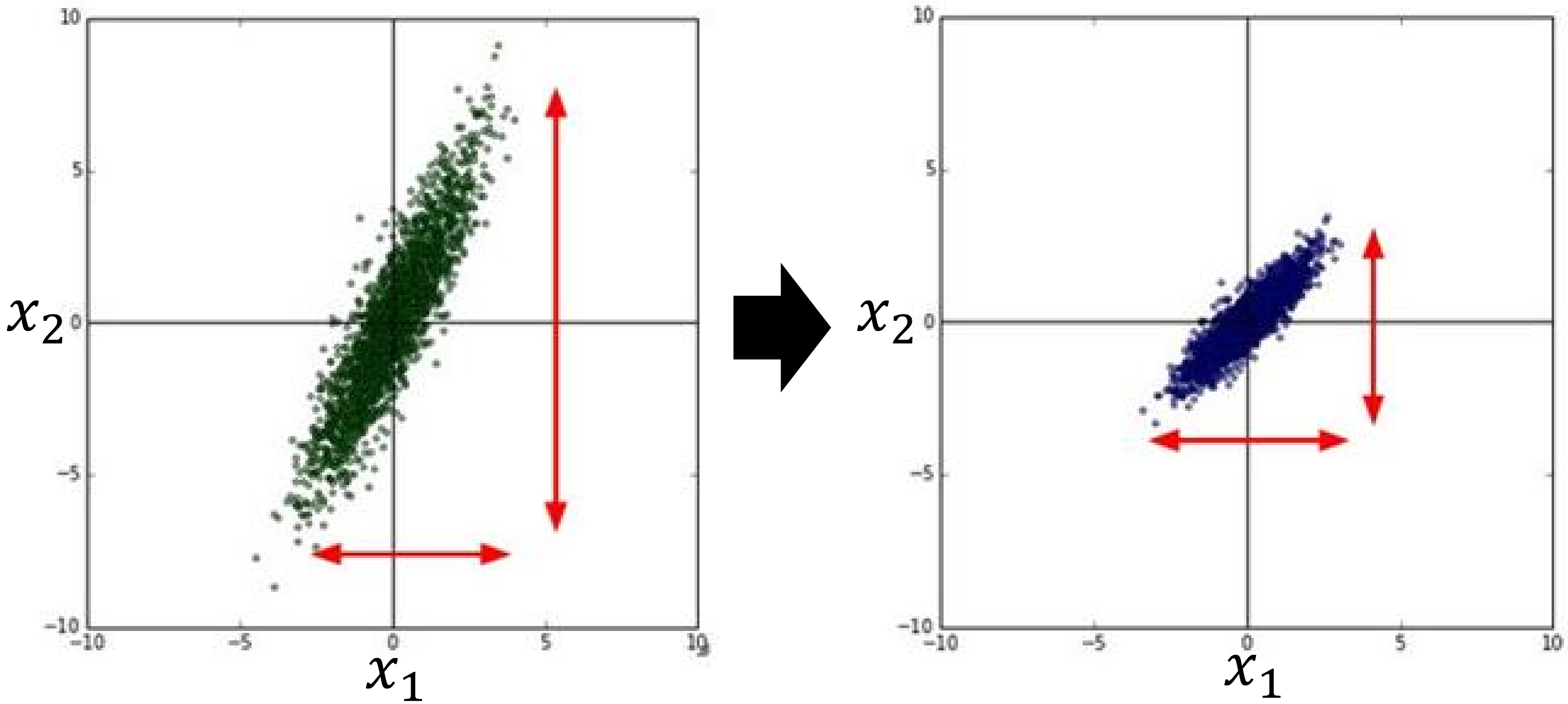
归一化

- 数据归一化
  - 标准化（Standardization）/Z值归一化（Z-Score Normalization）
  - 最大最小值归一化（Min-Max Normalization）

- 逐层归一化
  - Batch Normalization，BN
  - Layer Normalization，LN
  - Instance Normalization，IN
  - Group Normalization，GN

数据归一化

$$\hat{y} = b + w_1 x_1 + w_2 x_2$$

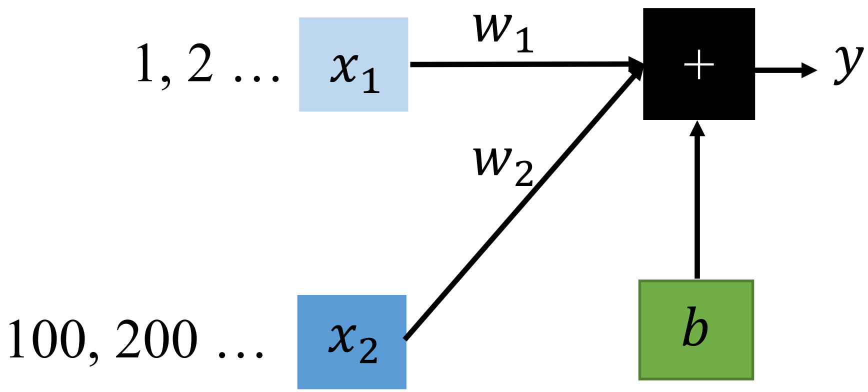


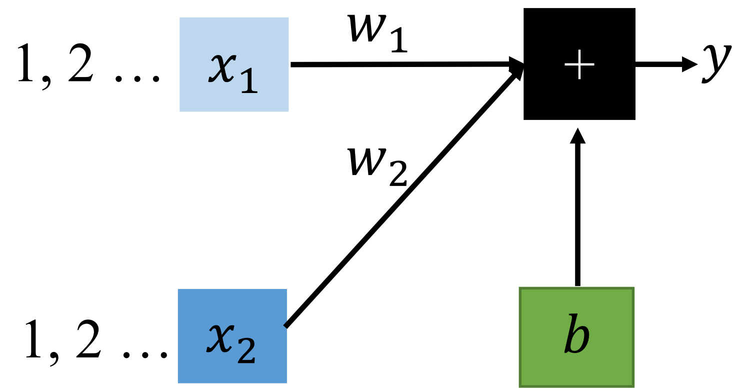
数据归一化：使得不同的特征有相同的尺度

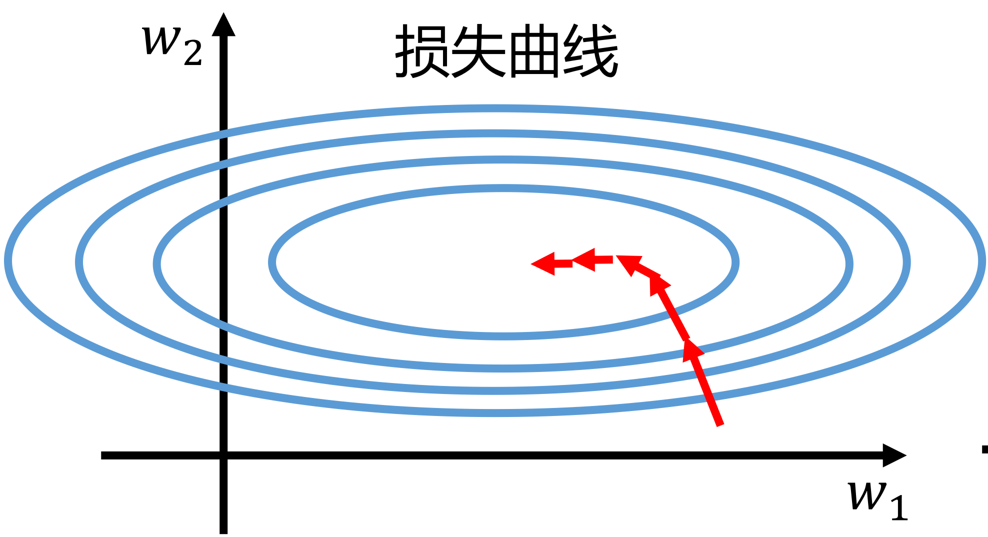
数据归一化

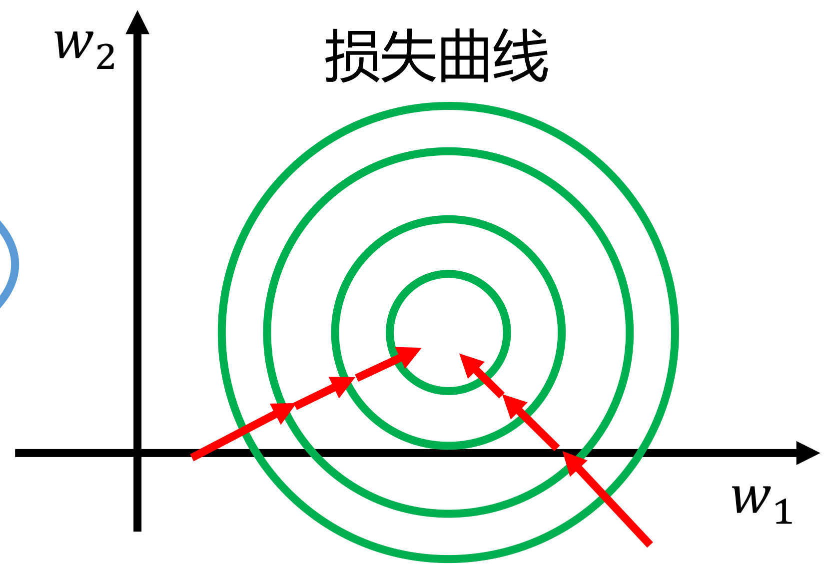
$$\hat{y} = b + w_1x_1 + w_2x_2$$

$$\mathcal{L} = (y - \hat{y})^2$$

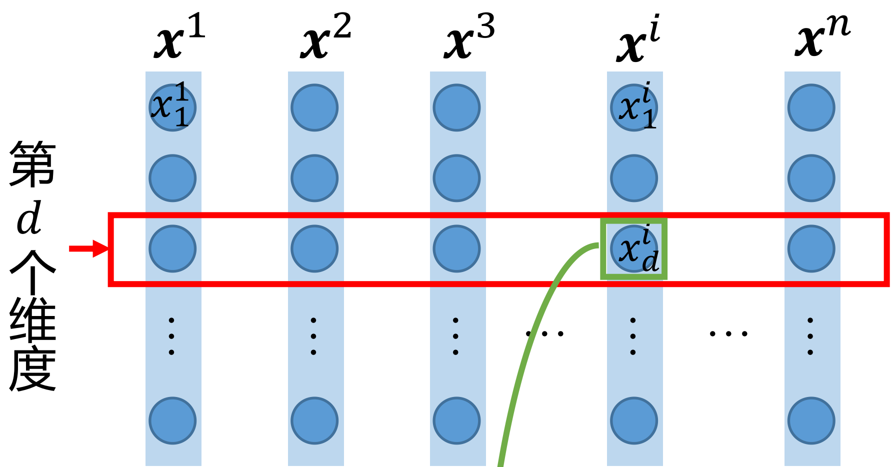








数据归一化——标准化



计算均值$m_d = \frac{1}{n}\sum_{i=1}^{n} x_d^i$

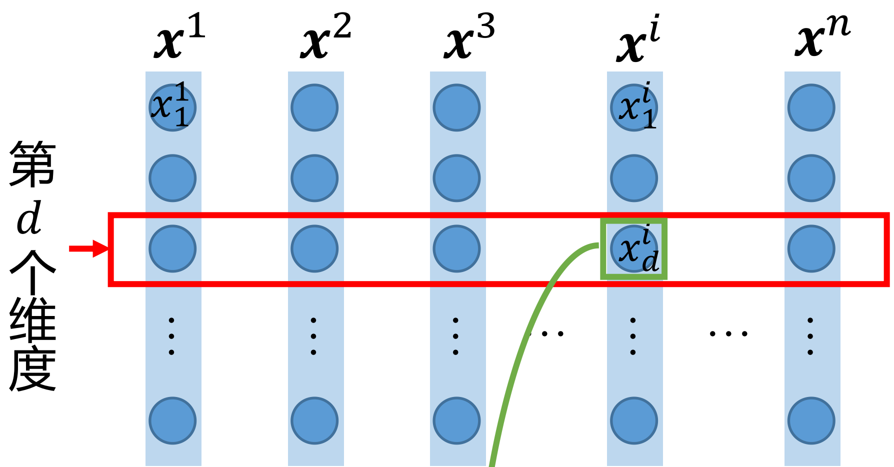
计算方差$\sigma_d^2 = \frac{1}{n}\sum_{i=1}^{n} \left(x_d^i - m_d\right)^2$

$$x_d^i \leftarrow \frac{x_d^i - m_d}{\sigma_d}$$

归一化后数据的所有维度均为

0均值 1方差

数据归一化——最大最小值归一化



计算最大值 $\mathrm{max_d}$

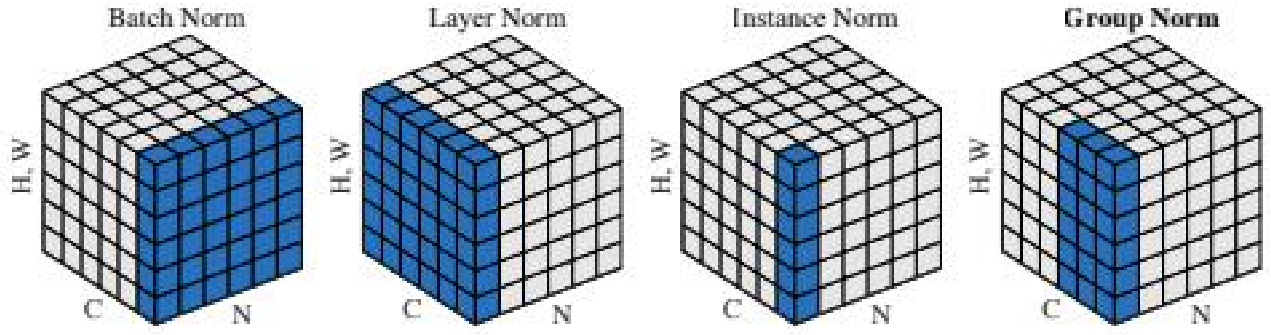
计算最小值 $\mathrm{min_d}$

$$x_d^i \leftarrow \frac{x_d^i - \mathrm{min_d}}{\mathrm{max_d} - \mathrm{min_d}}$$

归一化后数据的所有维度均在[0,1]

逐层归一化

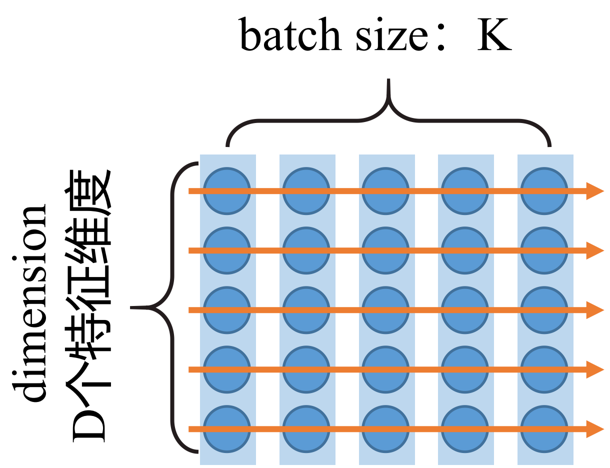
- 更好的尺度不变性：
  - 如果一个神经层的输入分布发生了改变，那么其参数需要重新学习，这种现象叫作内部协变量偏移
  - 为了缓解上述问题，可以对每一个神经层的输入进行归一化操作，使其分布保持稳定
- 更平滑的优化地形：
  - 使得大部分神经层的输入处于不饱和区域，从而让梯度变大，避免梯度消失问题
  - 使梯度变得更加稳定，从而允许使用更大的学习率，并提高收敛速度



Batch Normalization (BN)

神经网络中第l层的净输入为$\boldsymbol{z}^l$，神经元的输出为$\boldsymbol{a}^l$，

$$\boldsymbol{a}^{(l)} = f_l\left(\boldsymbol{z}^{(l)}\right) = f_l\left(\boldsymbol{W}^{(l)}\boldsymbol{a}^{(l-1)} + \boldsymbol{b}^{(l)}\right)$$



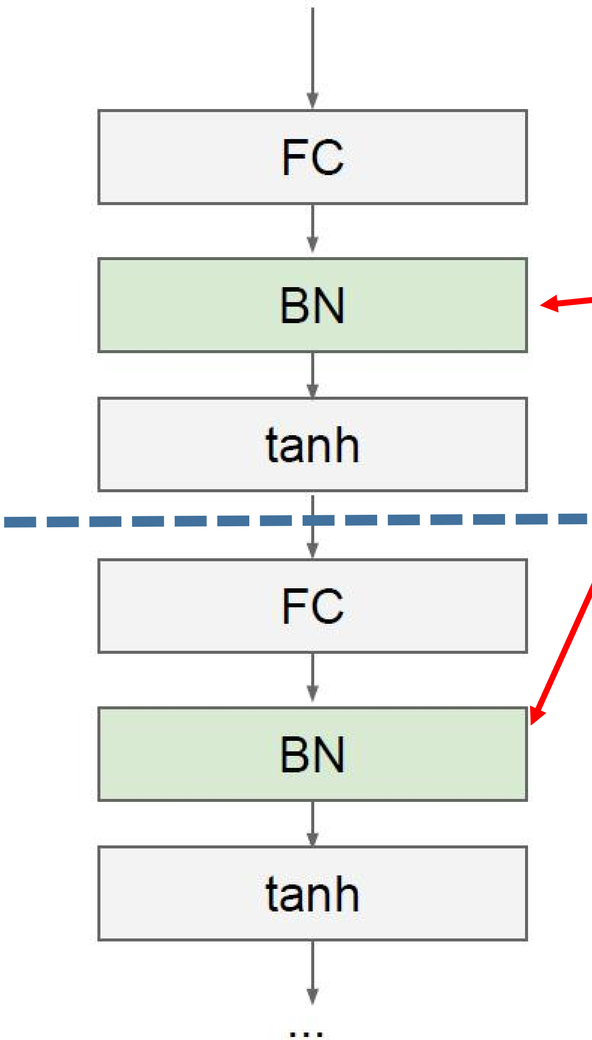
1. **给定一个包含K个样本的小批量样本集合，独立计算该层每维度的均值和方差**

$$\boldsymbol{\mu}_{\mathcal{B}} = \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{z}_k$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{z}_k - \boldsymbol{\mu}_{\mathcal{B}}) \odot (\boldsymbol{z}_k - \boldsymbol{\mu}_{\mathcal{B}})$$

2. 归一化 $\boldsymbol{z}_k = \dfrac{\boldsymbol{z}_k - \boldsymbol{\mu}_{\mathcal{B}}}{\boldsymbol{\sigma}_{\mathcal{B}}}$

Batch Normalization (BN)



BN层一般在仿射变换之后，
非线性激活函数之前

- 可以应用在深度神经网络中的任何一个中间层，并不需要对所有层进行归一化
- 每层的输入通常是前一层的非线性激活函数的输出
- 由于非线性激活函数，每层输入通常是非高斯的，不适合进行标准化

Batch Normalization (BN)

- 标准化单元的均值和标准差会降低该单元的表达能力
- 为了使得归一化不对网络的表示能力造成负面影响，引入$\boldsymbol{\gamma}$和$\boldsymbol{\beta}$两个可学习参数使得各单元的净输出有任意均值和方差

$$\boldsymbol{z}_k = \frac{\boldsymbol{z}_k - \boldsymbol{\mu}_{\mathcal{B}}}{\boldsymbol{\sigma}_{\mathcal{B}}} \Rightarrow \boldsymbol{z}_k = \frac{\boldsymbol{z}_k - \boldsymbol{\mu}_{\mathcal{B}}}{\boldsymbol{\sigma}_{\mathcal{B}}} \odot \boldsymbol{\gamma} + \boldsymbol{\beta}$$

- 网络可能学习出这两个参数，即得到一个恒等映射

$$\boldsymbol{\gamma} = \boldsymbol{\sigma}_{\mathcal{B}} \qquad \boldsymbol{\beta} = \boldsymbol{\mu}_{\mathcal{B}}$$

旧参数	新参数
• $\boldsymbol{\mu}_{\mathcal{B}}$和$\boldsymbol{\sigma}_{\mathcal{B}}$取决于低层神经网络的复杂关联	• $\boldsymbol{\gamma}$和$\boldsymbol{\beta}$解除了与下层计算的密切耦合，直接通过梯度下降来学习

Batch Normalization (BN)

➢测试过程和训练过程有所不同：均值和方差不是在测试集的mini-batch上计算，而是整个数据集上的均值和方差。

➢实践中，可以通过移动平均来计算样本均值和方差

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

BN的作用

- 改善梯度的传播
- 允许比较大的学习率
- 减小初始化的影响
- 一种隐形的正则化方法

- batch size不能太小，否则会导致算法性能下降
- 对于样本维度不统一的数据无法很好处理，例如RNN

Batch Normalization (BN)

全连接网络的BN

$x: N \times D$

Normalize

$\mu, \sigma: 1 \times D$

$\gamma, \beta: 1 \times D$

$y = \gamma(x-\mu)/\sigma+\beta$

卷积网络的BN

$x: N \times C \times H \times W$

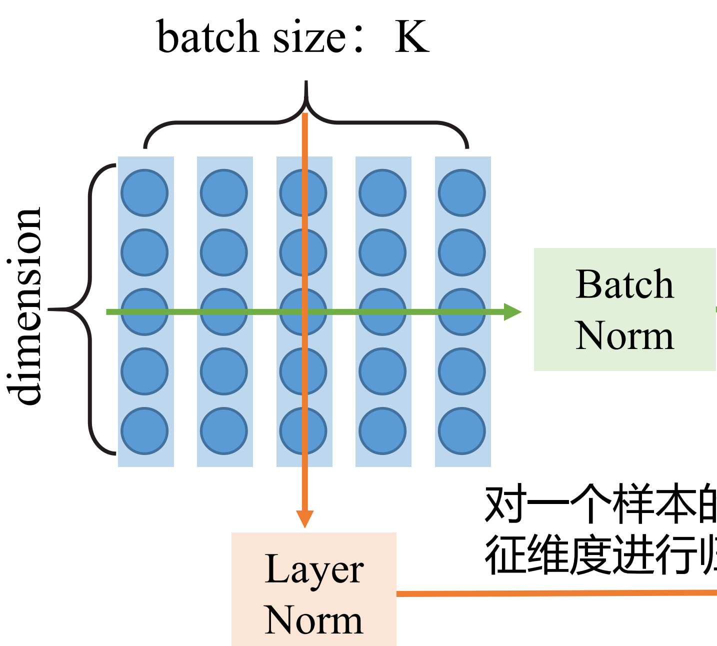
Normalize

$\mu, \sigma: 1 \times C \times 1 \times 1$

$\gamma, \beta: 1 \times C \times 1 \times 1$

$y = \gamma(x-\mu)/\sigma+\beta$

Layer Normalization (LN)



$x: N \times D$

Normalize

$\mu, \sigma: 1 \times D$

$\gamma, \beta: 1 \times D$

$y = \gamma(x-\mu)/\sigma+\beta$

全连接网络的BN

对一个样本的所有特征维度进行归一化

$x: N \times D$

Normalize

$\mu, \sigma: N \times 1$

$\gamma, \beta: 1 \times D$

$y = \gamma(x-\mu)/\sigma+\beta$

全连接网络的LN

训练和测试时操作一致
针对序列数据，常用在递归网络中

Instance Normalization (IN)

➢常用于处理图像数据，作用于每个样本的像素或通道等维度

<u>卷积网络的BN</u>

$$x:\ N\times C\times H\times W$$

Normalize

$$\mu,\sigma:\ 1\times C\times 1\times 1$$

$$\gamma,\beta:\ 1\times C\times 1\times 1$$

$$y=\gamma(x-\mu)/\sigma+\beta$$

<u>卷积网络的IN</u>

$$x:\ N\times C\times H\times W$$

Normalize

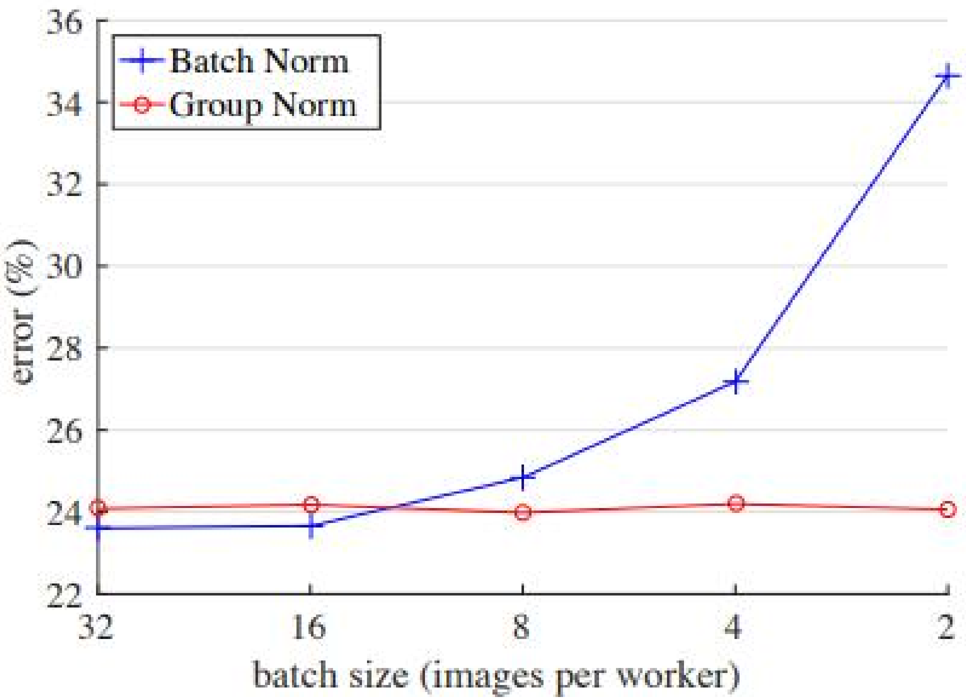
$$\mu,\sigma:\ N\times C\times 1\times 1$$

$$\gamma,\beta:\ 1\times C\times 1\times 1$$

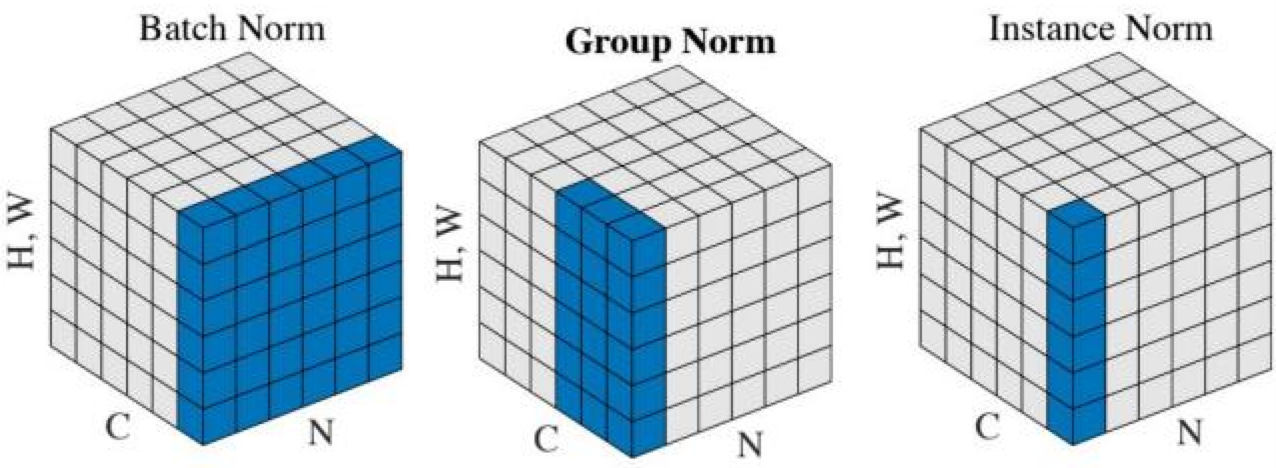
$$y=\gamma(x-\mu)/\sigma+\beta$$

Group Normalization (GN)

- 针对BN在batch size较小时错误率较高而提出的改进算法
- GN将通道分成几组，并在每组内计算归一化的均值和方差，其准确性在很宽的批量大小范围内都很稳定
- GN归一化操作的计算不依赖batch size的大小



归一化层的比较

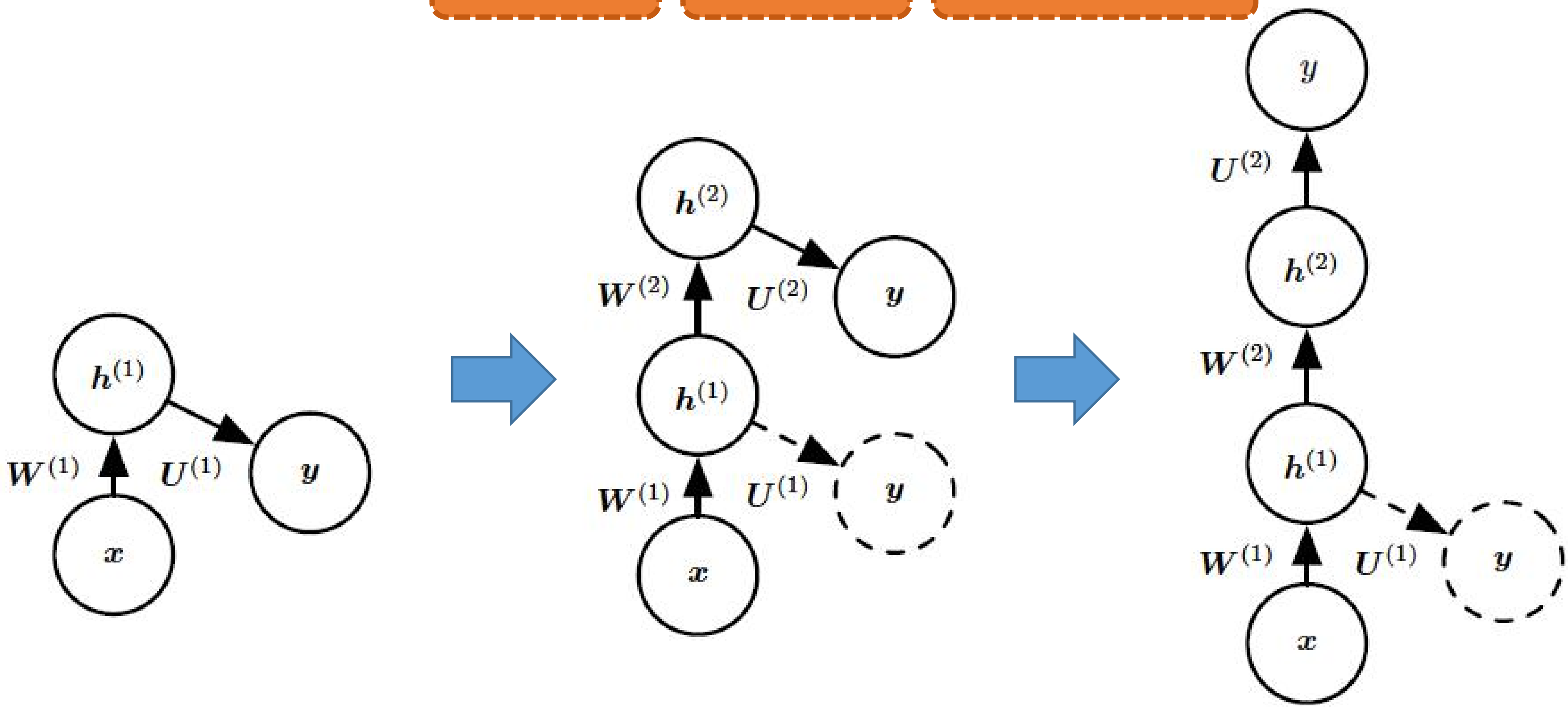


- 当GN将通道分成1组，此时即为LN
- 当GN将通道分成C组，此时即为IN

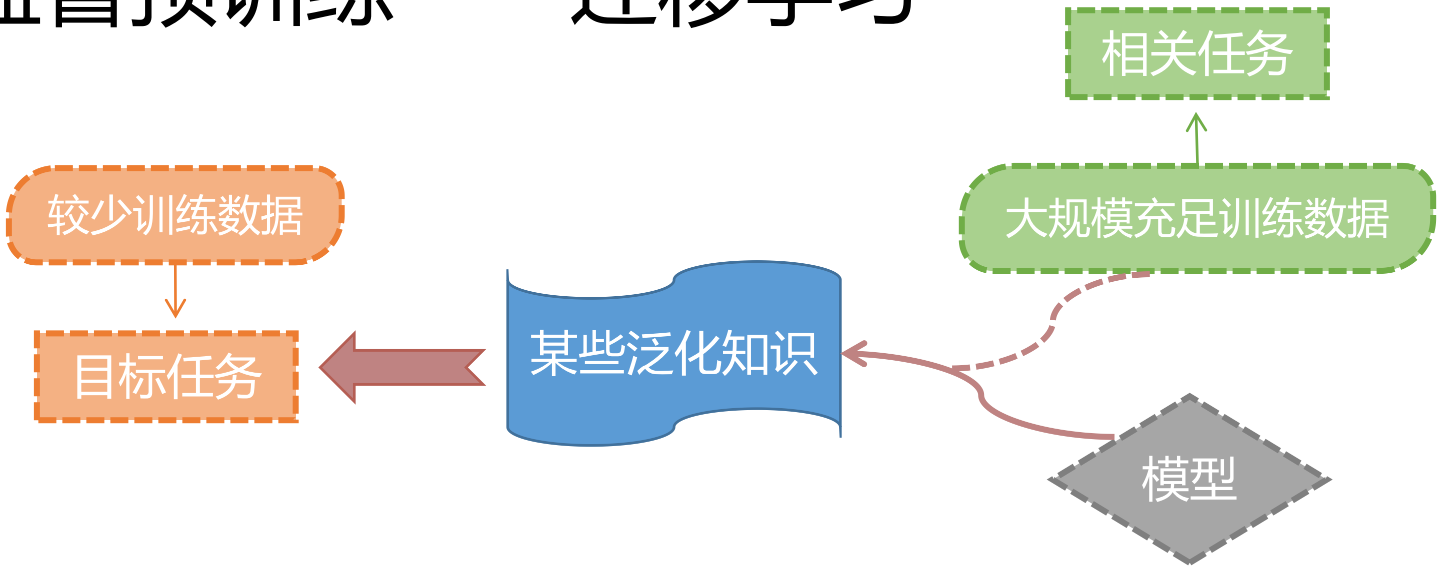
训练目标和方式

监督预训练

- 不同任务的模型往往都是从零开始来训练的，一切知识都从需要与真实数据分布保持一致的训练数据中得到。
- 但实际应用中： 模型复杂 任务困难 训练数据过少



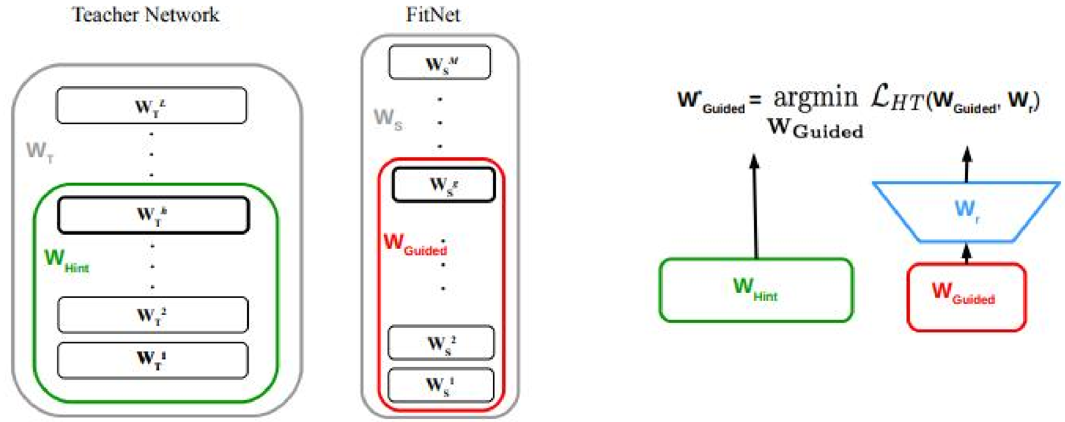
监督预训练——迁移学习



- 在一组任务上预训练了8 层权重的深度卷积网络
  - 1000 个ImageNet 对象类的子集
- 用该网络的前k 层初始化同样规模的网络，其余层随机初始化
  - 联合训练以执行不同的任务（1000 个ImageNet 对象类的另一个子集）
  - 训练样本少于第一个任务

监督预训练——FitNets

- ➢教师网络：深度浅（5层）且足够宽的网络，较容易训练
- ➢学生网络：深度深（11-19层）且窄的网络，很难用SGD训练
- ➢学生网络预测原任务的输出而且预测教师网络中间层的值
  - 学生网络中间层去回归教师网络的中间层

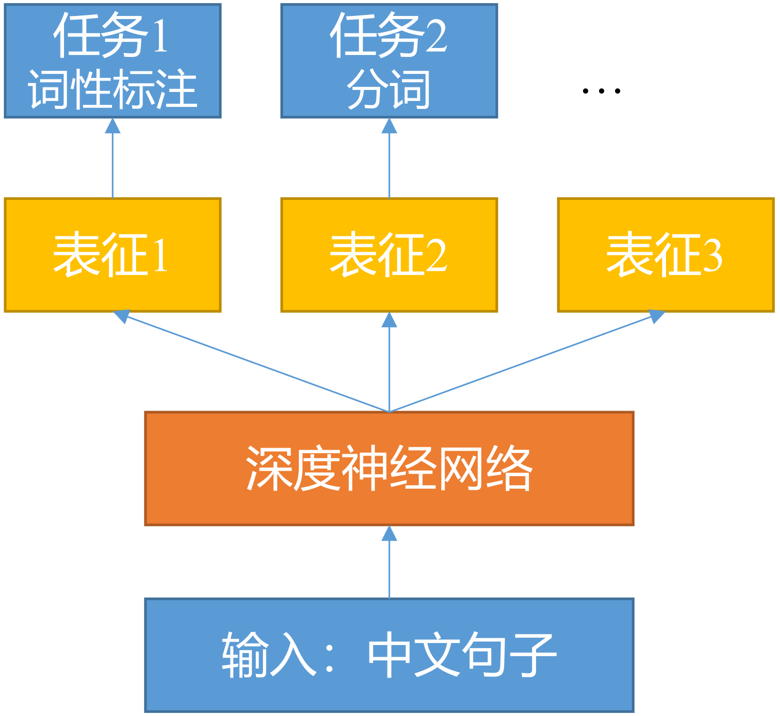


(a) Teacher and Student Networks　　(b) Hints Training

$$\mathcal{L}_{HT}(\mathbf{W}_{\mathbf{Guided}}, \mathbf{W}_{\mathbf{r}}) = \frac{1}{2}||u_h(\mathbf{x}; \mathbf{W}_{\mathbf{Hint}}) - r(v_g(\mathbf{x}; \mathbf{W}_{\mathbf{Guided}}); \mathbf{W}_{\mathbf{r}})||^2$$

多任务学习

➢通过合并几个任务中的样例来提高泛化的一种方式



课程学习

- 基于规划学习过程的想法，首先学习简单的概念，然后逐步学习依赖于这些简化概念的复杂概念
  - 在大规模的神经语言模型任务上使用课程学习，可以获得更好的结果

- 课程学习被证实为与人类教学方式一致
  - 教师刚开始会展示更容易、更典型的示例，然后帮助学习者在不太显然的情况下提炼决策面
  - 在人类教学上，基于课程学习的策略比基于样本均匀采样的策略更有效

课程学习

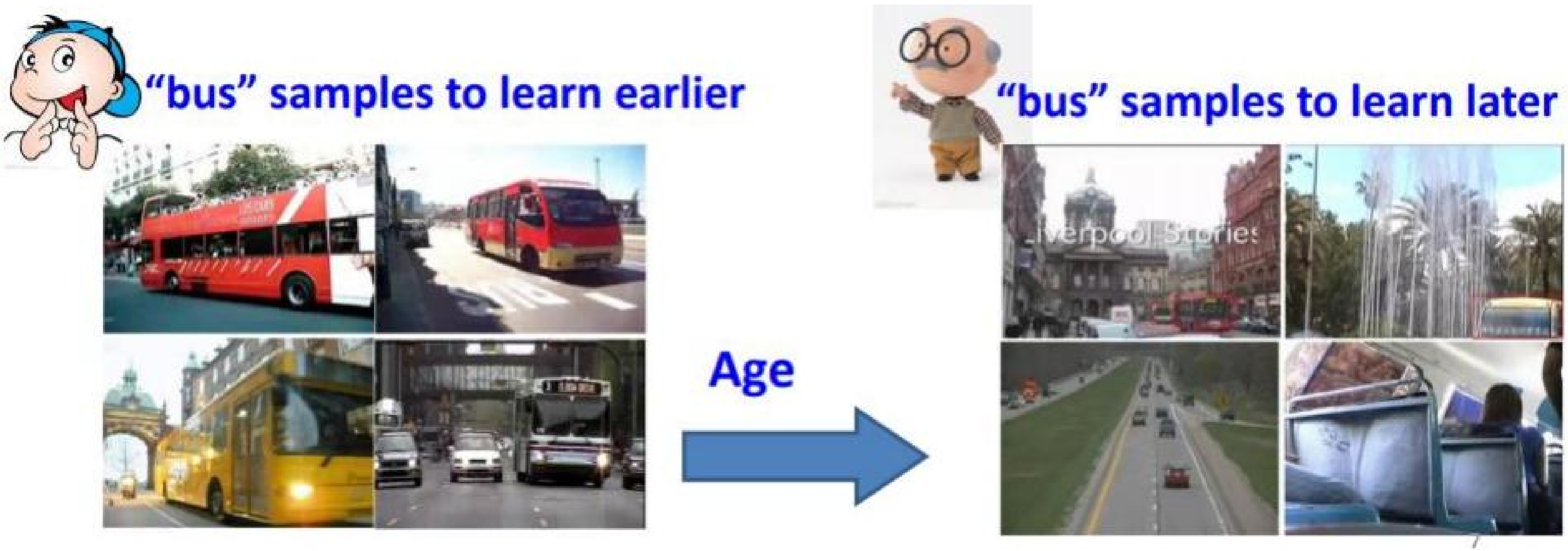
- 首先学习简单的、普适性的知识结构，然后逐渐增加难度，过渡到学习更复杂、更专业化的知识。

Samples of “Dog” to learn earlier.

Samples of “Dog” to learn later.

课程学习

- 首先学习简单的、普适性的知识结构，然后逐渐增加难度，过渡到学习更复杂、更专业化的知识。



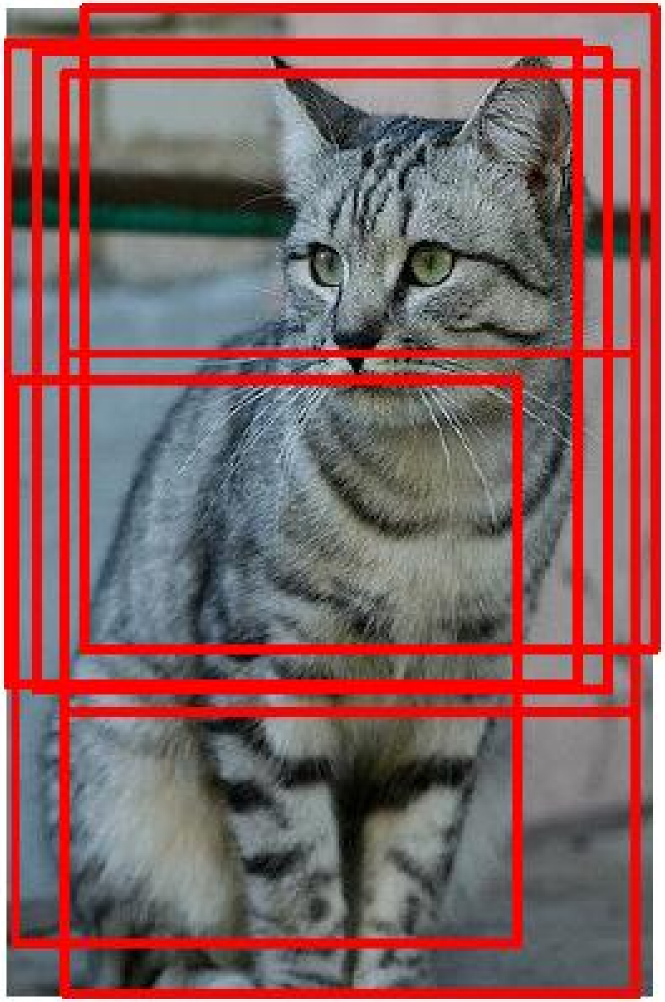
数据增强

数据集增强——水平翻转

- 需要保证水平翻转不会改变类别
  - 反例：OCR中的字符b和d，6和9

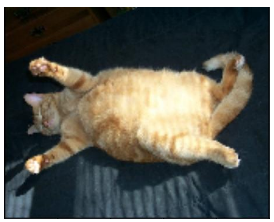
数据集增强——随机裁剪和缩放

- ➢训练
  - 从[256,480]的范围内随机取 L
  - 按比例缩放，其中短边长度为 L
  - 从缩放的图上采样224 × 224的图
- ➢测试
  - 五个尺度上的缩放{224,256,384,480,640}
  - 每个大小的图片，用10个224 × 224的裁剪
    - 2（水平翻转）×（4 角 + 1 中心）



数据集增强——变色

- ➢调整色调，饱和度和亮度 （例如 [0.5, 1.5]）

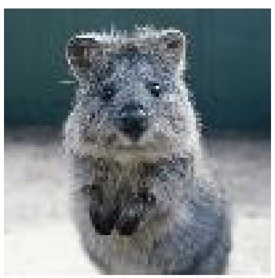


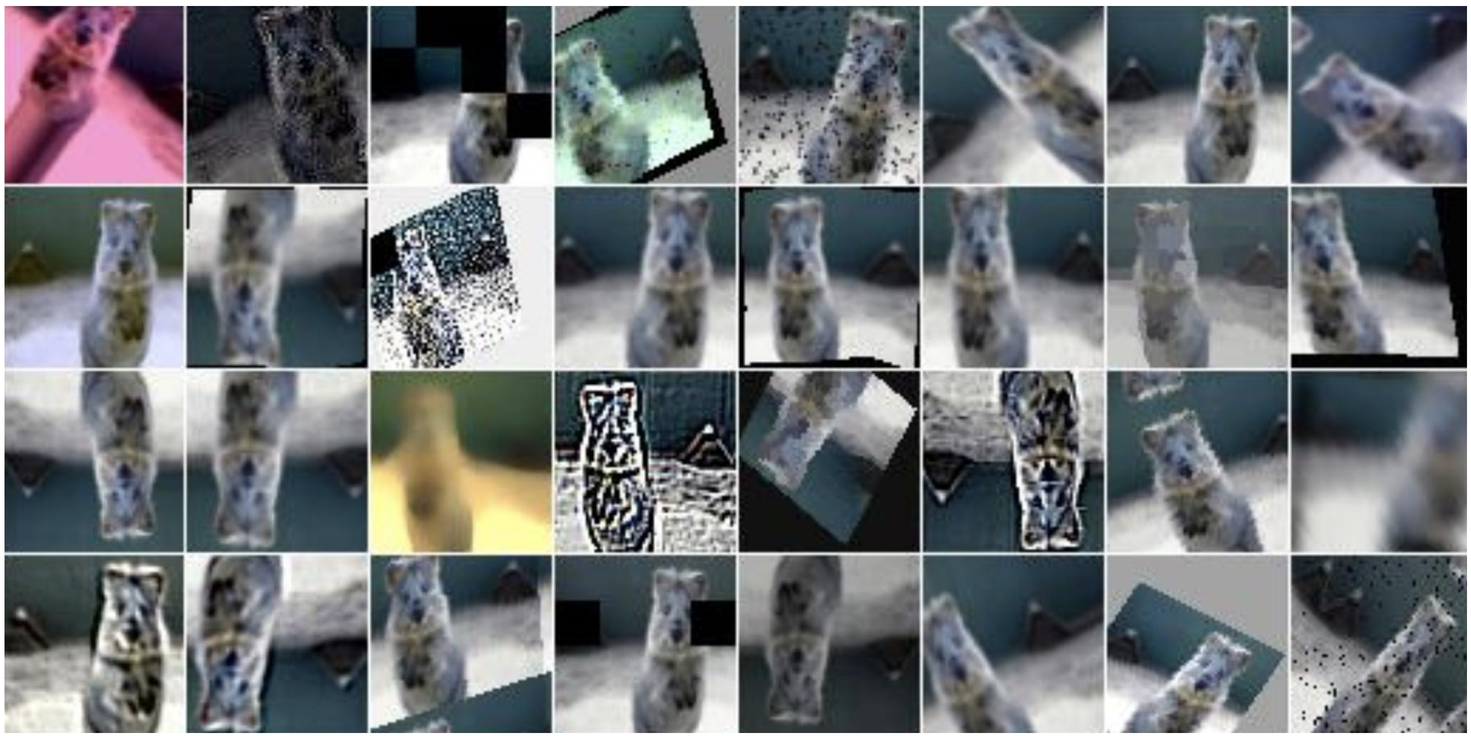
亮度

色调

数据集增强——更多方法

➢平移、旋转、伸缩、裁剪、扭曲的任意组合





数据集增强——注入噪声

- 向神经网络的输入层注入噪声也是一种数据集增强的方法

原图

加入$\mathcal{N}(0, 0.1^2)$的噪声

也可以向隐藏单元施加噪声

数据集增强——标签平滑

- 在输出标签中添加噪声来避免模型过拟合
- 大多数数据集的 y 标签都有一定错误，错误的 y 不利于最大化 $\log p(y|x)$
- 一个样本$\boldsymbol{x}$的标签一般用onehot向量表示

$$y = [0, \ldots, 0, 1, 0, \ldots, 0]$$

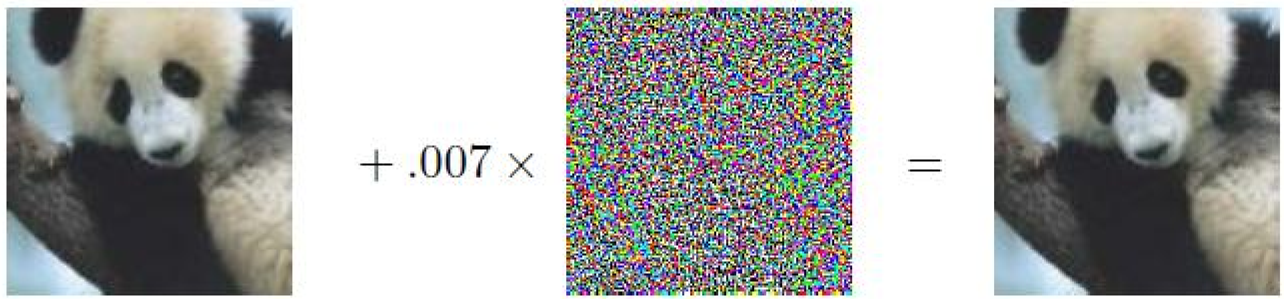
- 引入一个噪声对标签进行平滑，即假设样本以ϵ的概率为其它类。平滑后的标签为

$$y = \left[\frac{\epsilon}{k-1}, \ldots, \frac{\epsilon}{k-1}, 1-\epsilon, \frac{\epsilon}{k-1}, \ldots, \frac{\epsilon}{k-1}\right]$$

软目标（Soft Target）

对抗学习

- 对抗样本：x 与 x' 非常相似，人类难以察觉差异，但神经网络会作出不同的预测



x	$\text{sign}(\nabla_x J(\theta, x, y))$	$x + \epsilon\text{sign}(\nabla_x J(\theta, x, y))$
$y =$ 熊猫	线虫	长臂猿
w/ 57.7% confidence	w/ 8.2% confidence	w/ 99.3 % confidence

对抗学习	在对抗扰动的训练集样本上训练网络，以减少原有独立同分布的测试集的错误率

正则化

正则化

- 正则化（Regularization）是一类通过限制模型复杂度，从而避免过拟合，提高泛化能力的方法，比如引入约束、增加先验、提前停止等
- L1&L2正则化：通过约束参数的L1 和L2范数来减小模型在训练数据集上的过拟合现象

$$\mathcal{L}'(\boldsymbol{\theta}) = \underbrace{\mathcal{L}(\boldsymbol{\theta})} + \frac{1}{2}\lambda \underline{\|\boldsymbol{\theta}\|_2^2}$$

正则化项

损失函数，比如平方损失，交叉熵损失等等

模型参数：$\boldsymbol{\theta} = \{w_1, w_2, \ldots\}$

L1 正则:

$$\|\boldsymbol{\theta}\|_1 = |w_1| + |w_2| + \cdots$$

L2 正则:

$$\|\boldsymbol{\theta}\|_2^2 = (w_1)^2 + (w_2)^2 + \cdots$$

一般不考虑bias

L1正则化

- L1正则化：

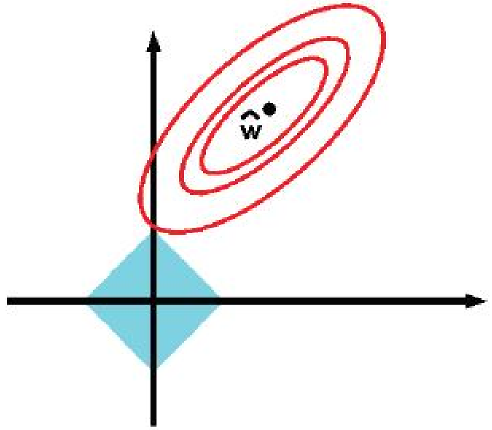
$$\mathcal{L}'(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \frac{1}{2}\lambda\|\boldsymbol{\theta}\|_1 \qquad \|\boldsymbol{\theta}\|_1 = |w_1| + |w_2| + \cdots$$

- 梯度：$\nabla\mathcal{L}' = \nabla\mathcal{L} + \lambda\,\mathrm{sign}(\boldsymbol{\theta})$

- 梯度下降更新：

$$\boldsymbol{\theta}^t \leftarrow \boldsymbol{\theta}^{t-1} - \eta\nabla\mathcal{L}' = \boldsymbol{\theta}^{t-1} - \eta\nabla\mathcal{L} - \eta\lambda\,\mathrm{sign}(\boldsymbol{\theta}^{t-1})$$

$$= \boldsymbol{\theta}^{t-1} - \eta\lambda\,\mathrm{sign}(\boldsymbol{\theta}^{t-1}) - \eta\nabla\mathcal{L}$$



- 如果$w^{t-1} > 0$，w^{t-1} 变小
- 如果$w^{t-1} < 0$，w^{t-1} 变大

$w^{t-1} \rightarrow 0$

具有稀疏的特性

L2正则化

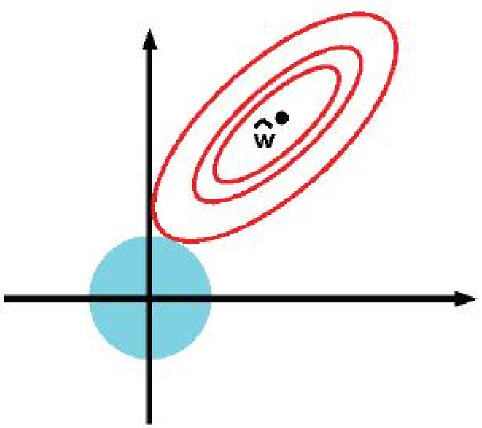
- L2正则化：

$$\mathcal{L}'(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \frac{1}{2}\lambda\|\boldsymbol{\theta}\|_2^2 \qquad \|\boldsymbol{\theta}\|_2^2 = (w_1)^2 + (w_2)^2 + \cdots$$

- 梯度：$\nabla\mathcal{L}' = \nabla\mathcal{L} + \lambda\,\boldsymbol{\theta}$

- 梯度下降更新：

$$\boldsymbol{\theta}^t \leftarrow \boldsymbol{\theta}^{t-1} - \eta\nabla\mathcal{L}' \;= \boldsymbol{\theta}^{t-1} - \eta\nabla\mathcal{L} - \eta\lambda\,\boldsymbol{\theta}^{t-1}$$

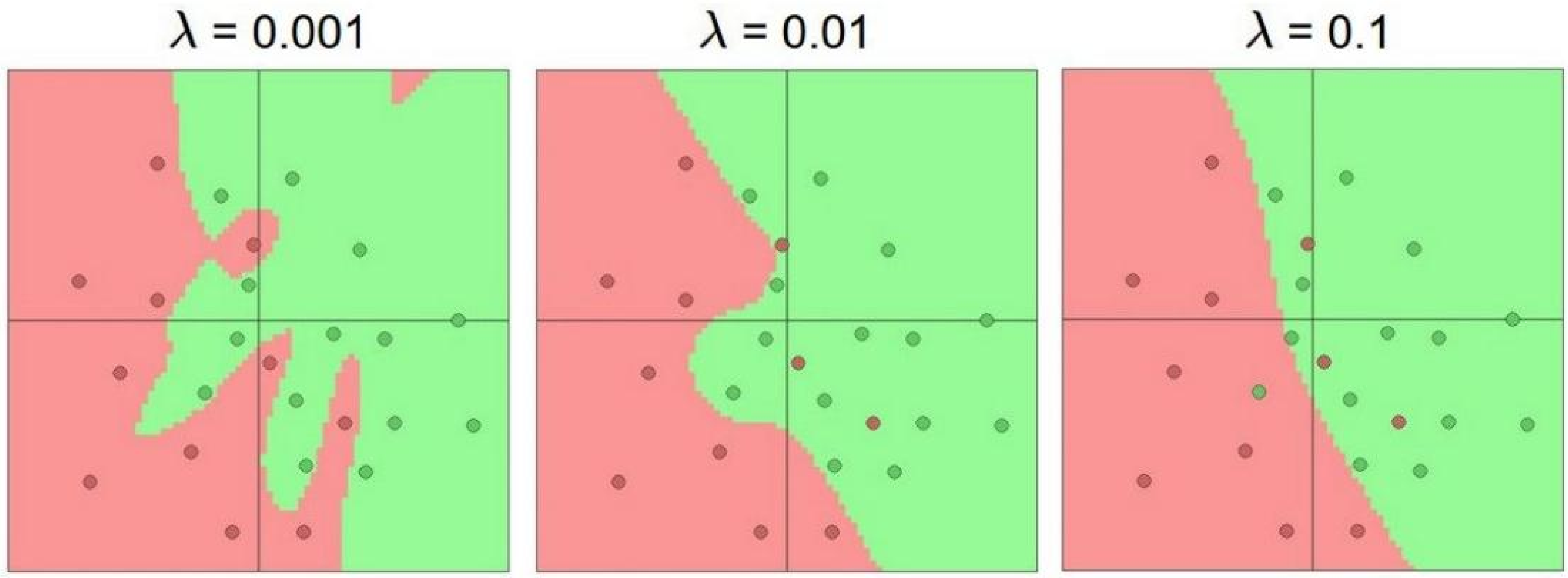


$$= \underbrace{(1-\eta\lambda)}\boldsymbol{\theta}^{t-1} - \eta\nabla\mathcal{L} \qquad \eta < 1, \lambda < 1$$

越来越小，但由于$-\eta\nabla L$项，使得参数不会变为0

权重衰减（weight decay）可用L2正则化来实现

神经网络示例



http://playground.tensorflow.org/

添加噪声

➢假设网络权重添加随机扰动$\boldsymbol{\epsilon}_W \sim \mathcal{N}(0, \eta \boldsymbol{I})$, 扰动模型为：$\hat{y}_{\epsilon_W}(\boldsymbol{x})$

➢将平方损失作为目标函数

$$\tilde{J}_W = \mathbb{E}_{p(x,y,\epsilon_W)}\left[\left(\hat{y}_{\epsilon_W}(\boldsymbol{x}) - y\right)^2\right]$$

$$= \mathbb{E}_{p(x,y,\epsilon_W)}\left[\hat{y}^2_{\epsilon_W}(\boldsymbol{x}) - 2y\hat{y}_{\epsilon_W}(\boldsymbol{x}) + y^2\right]$$

一阶泰勒展开近似

$$\hat{y}_{\epsilon_W}(\boldsymbol{x}) \approx y_W(\boldsymbol{x}) + \nabla_W y_W(\boldsymbol{x})^T \epsilon_W$$

$$\approx \mathbb{E}_{p(x,y,\epsilon_W)}[y^2_W(\boldsymbol{x}) + 2y_W(\boldsymbol{x})\nabla y_W(\boldsymbol{x})^\top \epsilon_W + \epsilon_W^\top \nabla y_W(\boldsymbol{x}) \nabla y_W(\boldsymbol{x})^\top \epsilon_W]$$

$$+ \mathbb{E}_{p(x,y,\epsilon_W)}[-2yy_W(\boldsymbol{x}) - 2y\nabla y_W(\boldsymbol{x})^\top \epsilon_W + y^2]$$

$$= \mathbb{E}_{p(x,y)}[(y_W(\boldsymbol{x}) - y)^2] + \mathbb{E}_{p(x,y)}\left[\mathrm{tr}\left(\nabla y_W(\boldsymbol{x}) \nabla y_W(\boldsymbol{x})^\top \mathbb{E}_{p(\epsilon_W)}[\epsilon_W \epsilon_W^\top]\right)\right]$$

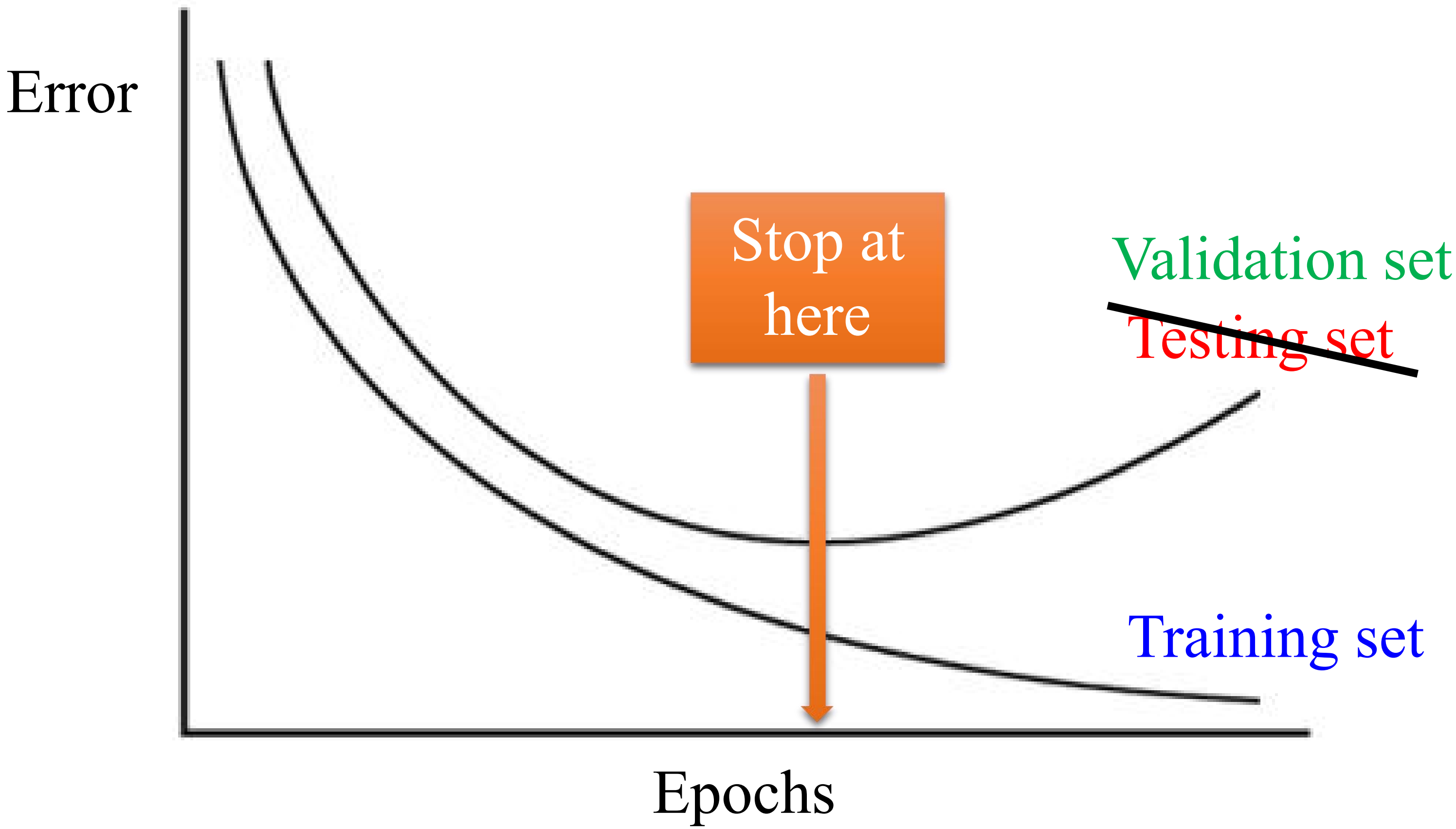
$$= J_W + \eta \mathbb{E}_{p(x,y)}[\|\nabla y_W(\boldsymbol{x})\|^2]$$

$$\mathbb{E}_{p(\epsilon_W)}[\epsilon_W \epsilon_W^\top] = \eta \boldsymbol{I}$$

鼓励参数进入 权重小扰动对输出相对影响较小的参数空间区域

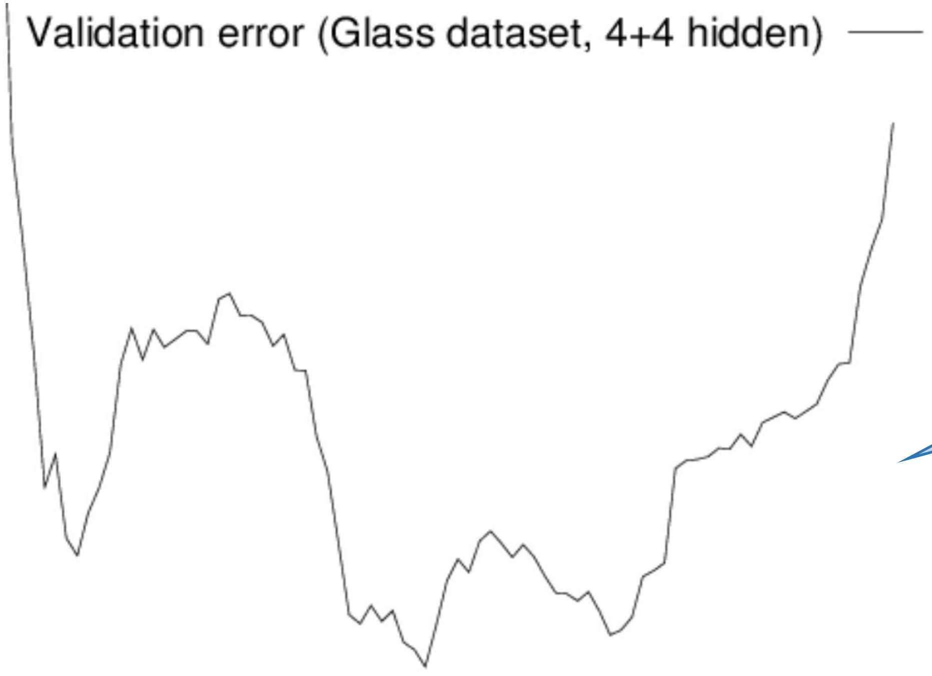
提前终止

- ➢验证集上准确率下降（损失上升）的时候停止训练
- ➢训练很长时间，保存在验证集上最优的模型



提前终止

- 早停标准：什么时候停？寻找更好的训练时间和泛化错误之间的权衡。

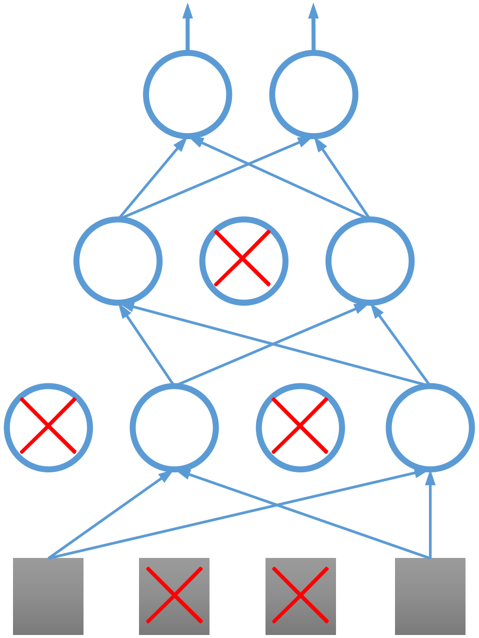


模型在验证集上的表现可能短暂的变差之后继续变好

真实的验证集误差变化曲线

Dropout

- 在训练时，取得一个batch后，以概率p设置一些神经元为0
  - 网络结构发生了变化
  - 用新网络在batch上计算梯度
  - 在新网络上进行参数更新
  - 每次取新batch时，都需要重新随机对神经元置为0



Dropout

- 测试时，没有dropout，即所有的神经元都处于激活状态
- 缩放激活函数的输出，使得每个神经元测试时输出等于训练时的期望输出

权重比例推断规则
如果训练时dropout 的概率为 p，那么测试时所有的权重要乘以 $1-p$

Dropout—训练和测试时实现

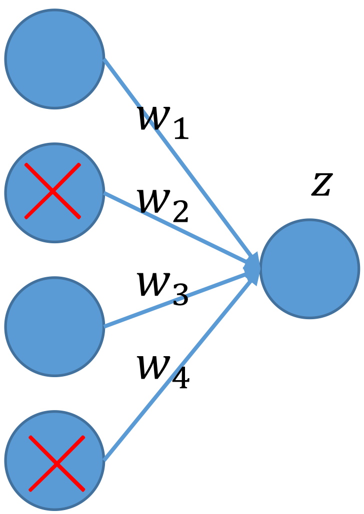
```
p = 0.5 # probability of keeping a unit active. higher = less dropout

def train_step(X):
  """ X contains the data """

  # forward pass for example 3-layer neural network
  H1 = np.maximum(0, np.dot(W1, X) + b1)
  U1 = np.random.rand(*H1.shape) < p # first dropout mask
  H1 *= U1 # drop!
  H2 = np.maximum(0, np.dot(W2, H1) + b2)
  U2 = np.random.rand(*H2.shape) < p # second dropout mask
  H2 *= U2 # drop!
  out = np.dot(W3, H2) + b3

def predict(X):
  # ensembled forward pass
  H1 = np.maximum(0, np.dot(W1, X) + b1) * p # NOTE: scale the activations
  H2 = np.maximum(0, np.dot(W2, H1) + b2) * p # NOTE: scale the activations
  out = np.dot(W3, H2) + b3
```

该概率为保留概率

Dropout—直观解释

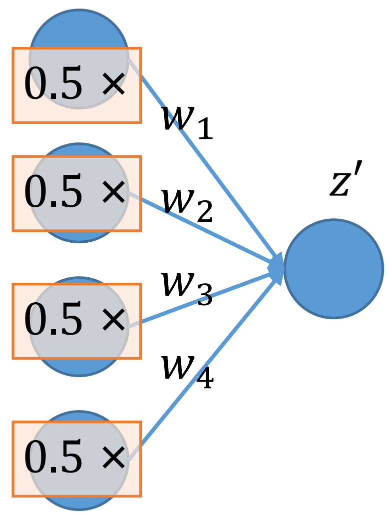
训练时dropout

dropout 概率为0.5



测试时dropout

No dropout



× 直接用训练后的权重

$\Rightarrow$ $z' \approx 2z$

✓ 权重乘以$1-p$

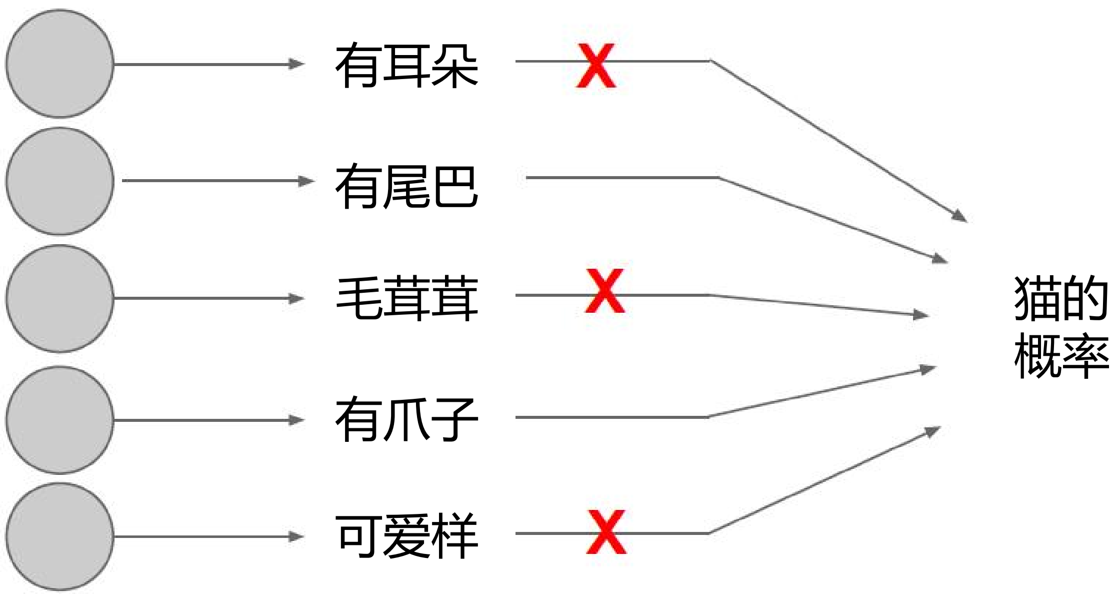
$\Rightarrow$ $z' \approx z$

Dropout动机

1 强制网络有冗余特征表示

2 防止特征的co-adaptation

co-adaptation: feature detectors只有在一些其它特定的feature detectors存在时才能发挥作用的情况



Dropout动机

➢人脸识别情形下的动机：

1

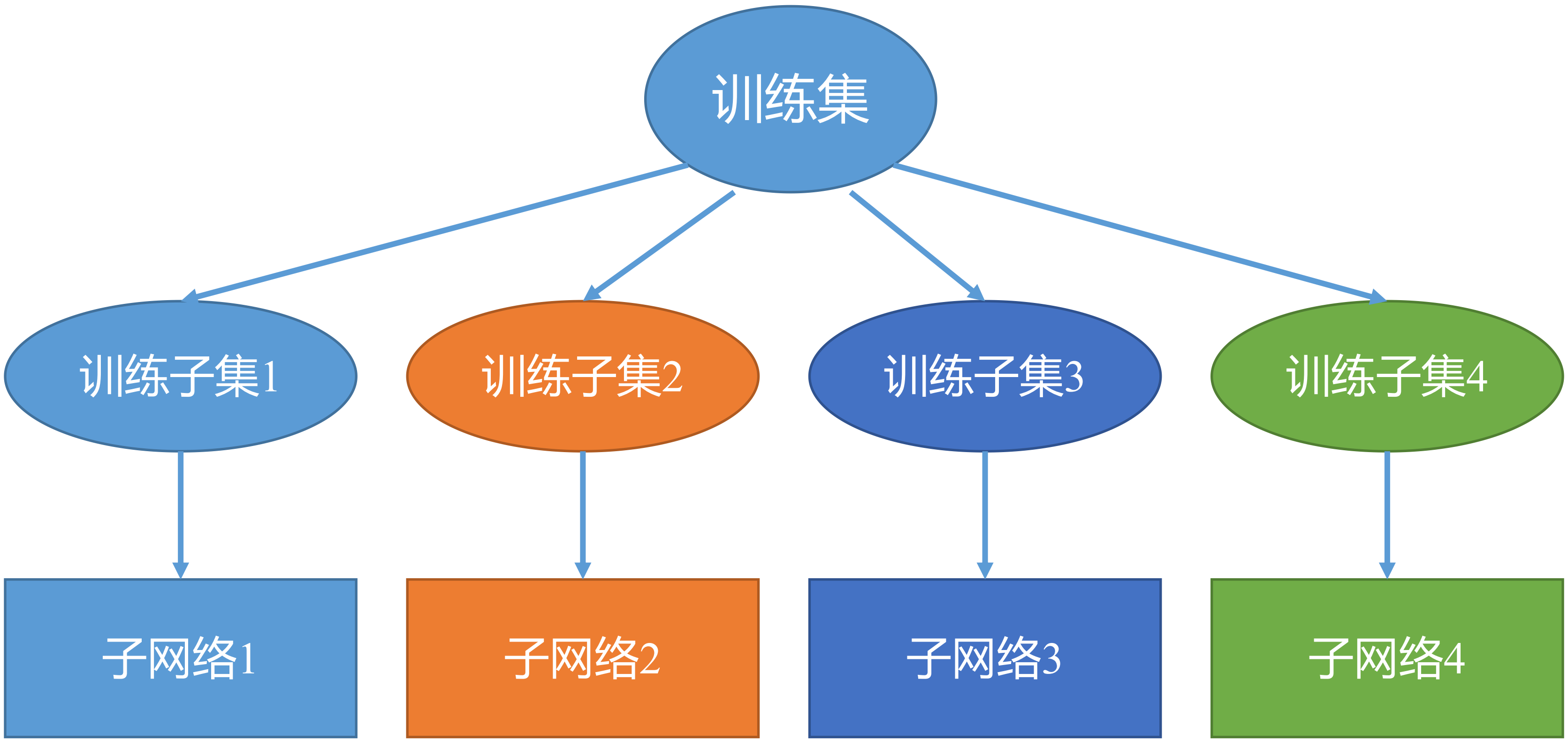
模型学得通过鼻检测脸的隐藏单元h_i，丢失h_i对应于擦除图像中有鼻子的信息

2

模型必须学习另一种h_i，要么是鼻子存在的冗余编码，要么是像嘴这样的脸部的另一特征

Dropout 是一种模型集成（Bagging）

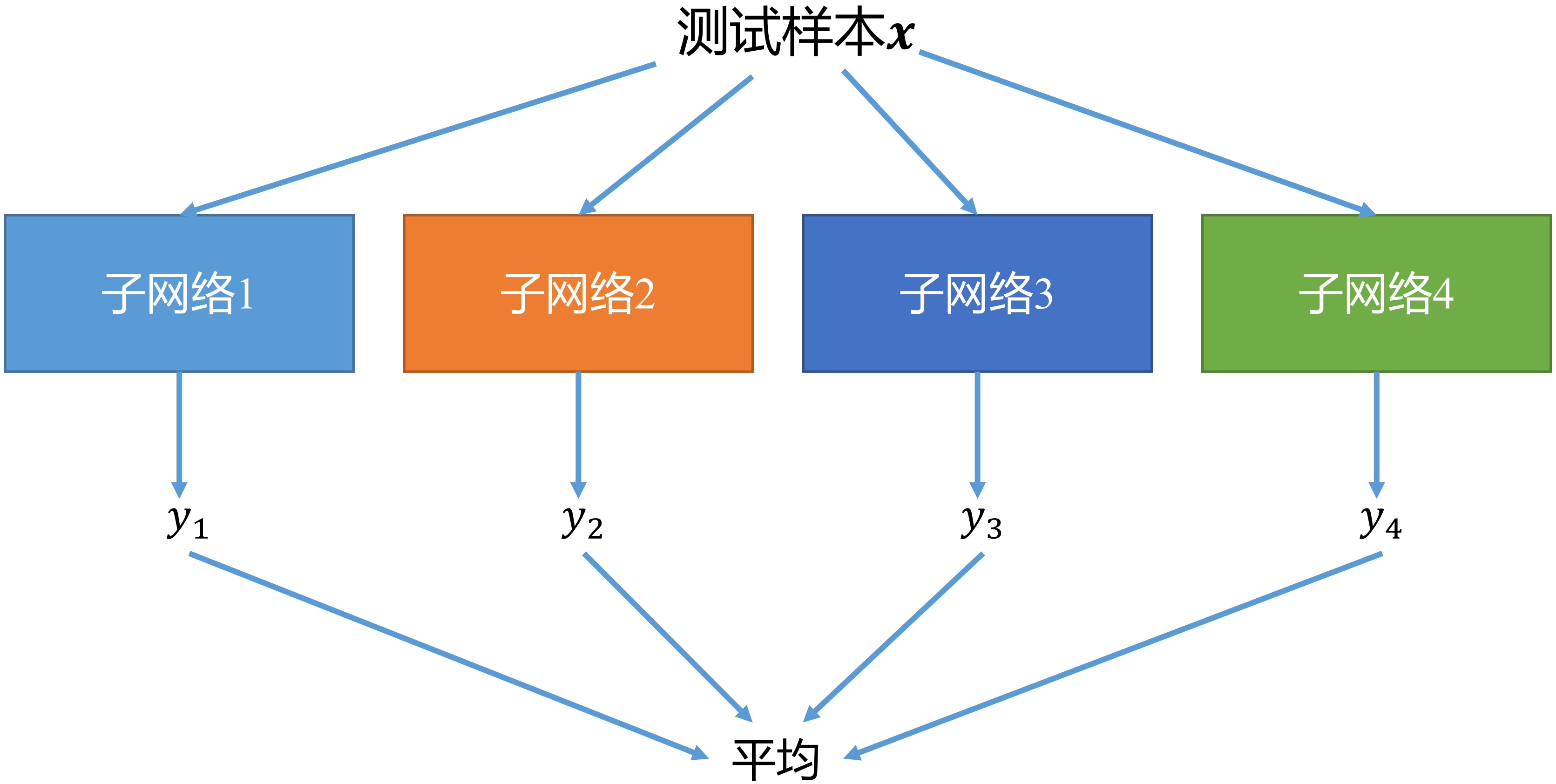
- Bagging:



训练一系列不同结构的神经网络

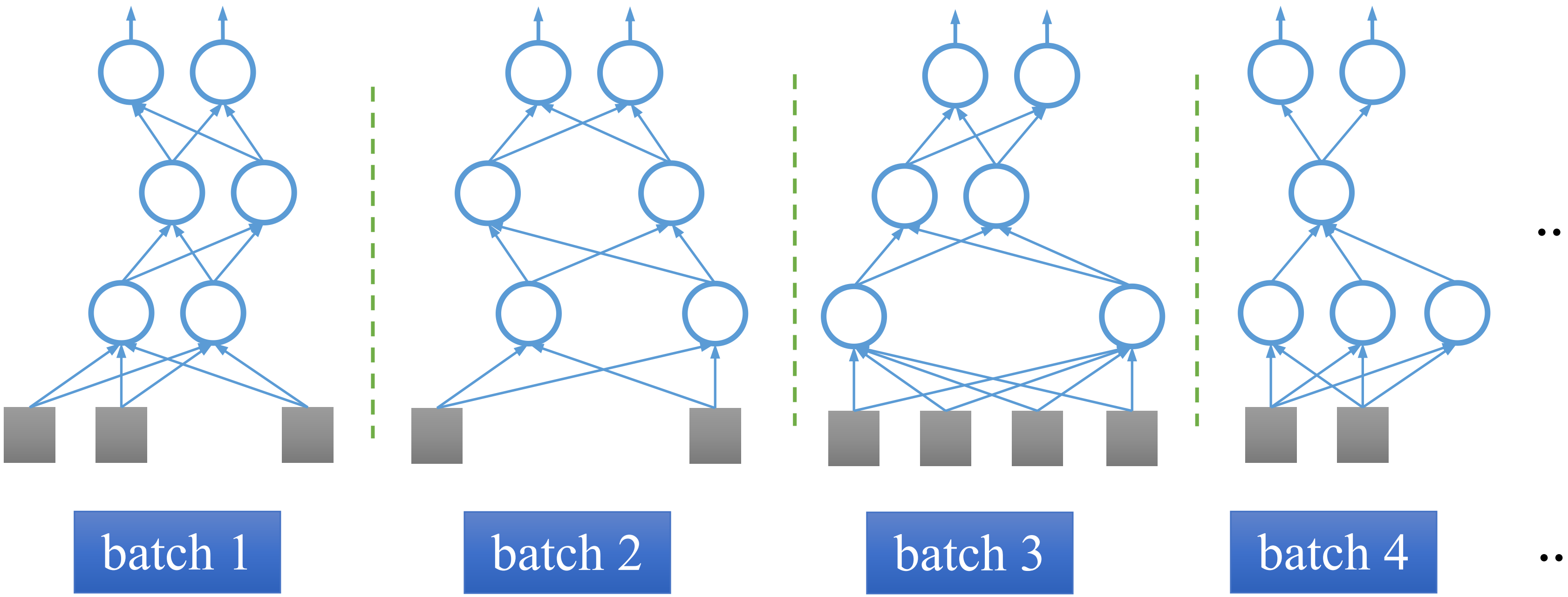
Dropout 是一种模型集成（Bagging）

➢ Bagging:



Dropout 是一种模型集成（Bagging）

- **训练时**：每做一次Dropout，相当于从原始的网络中采样得到一个子网络

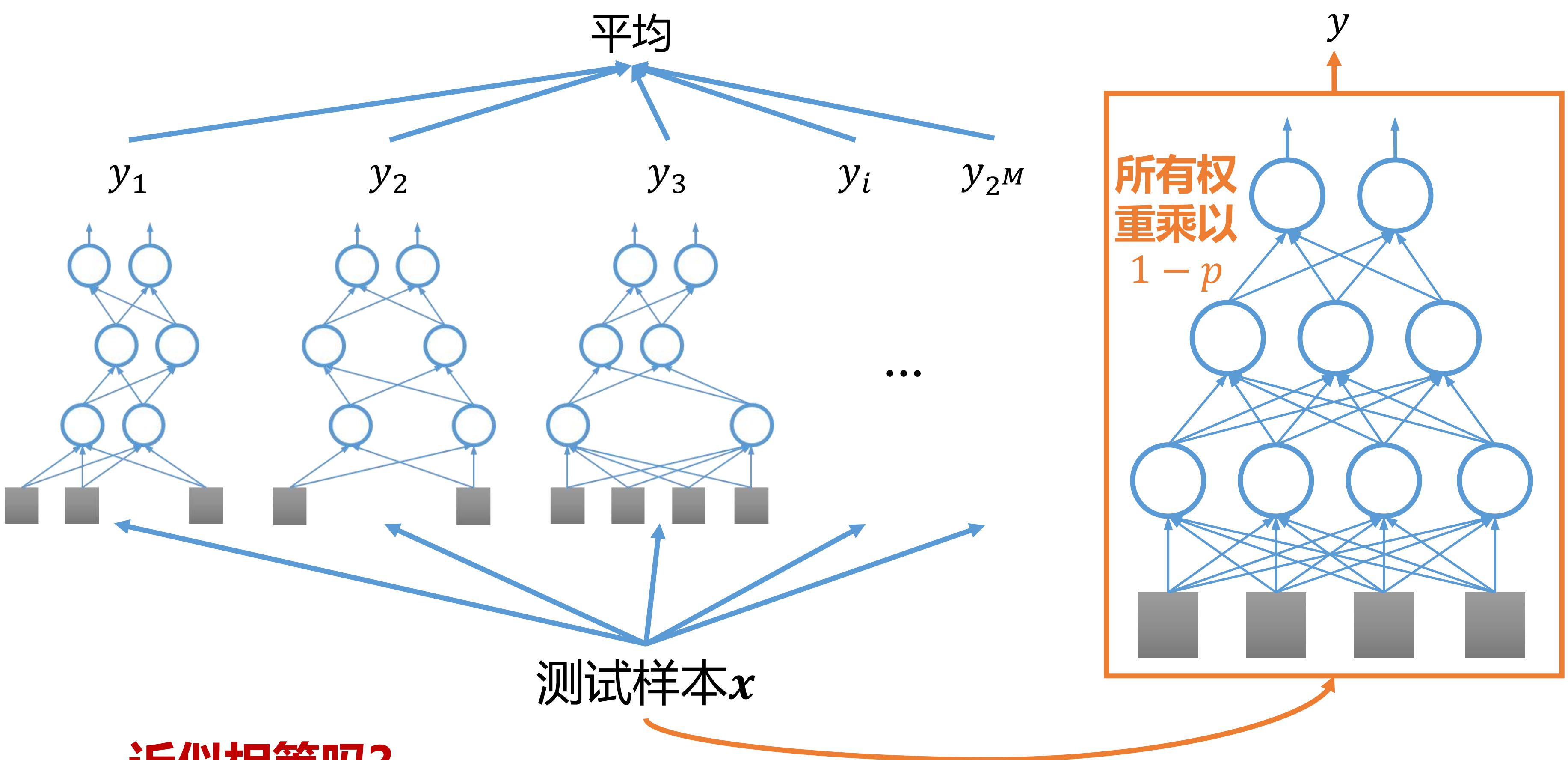


- 若神经网络有M个神经元，则共有2^M个不同结构的网络
- 一个batch数据训练一个结构的网络，只更新一次参数
- 不同结构的网络共享部分参数

Dropout 是一种模型集成（Bagging）

Dropout	Bagging
所有模型共享参数	所有模型相互独立
每一步训练一小部分的网络	每个模型在相应数据集上训练到收敛

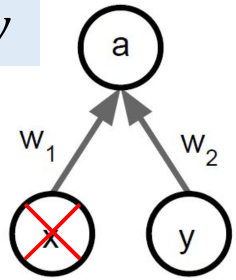
Dropout 是一种模型集成（Bagging）

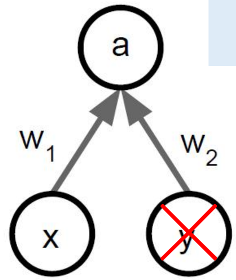


近似相等吗?

Dropout 是一种模型集成（Bagging）

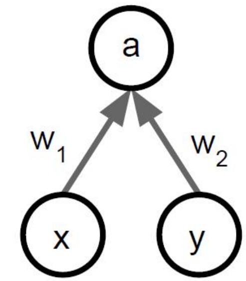
$a = 0x + w_2 y$



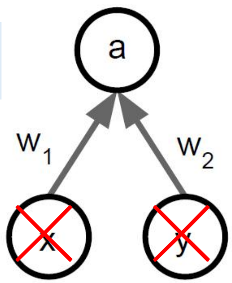


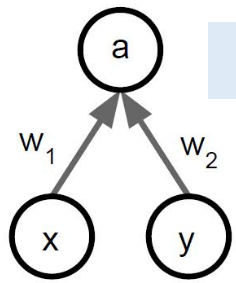
$a = w_1 x + 0y$

$a = w_1 x + w_2 y$



$a = 0x + 0y$





$a = w_1 x + w_2 y$

训练时： $$\mathbb{E}[a] = \frac{1}{4}(w_1 x + w_2 y) + \frac{1}{4}(w_1 x + 0y) + \frac{1}{4}(0x + 0y) + \frac{1}{4}(0x + w_2 y) = \frac{1}{2}(w_1 x + w_2 y)$$

测试时： $\mathbb{E}[a] = w_1 x + w_2 y$ **所有权重都乘以 1/2**

Dropout 是一种模型集成（Bagging）

- 权重比例推断规则对许多无非线性隐藏单元的模型族是精确的
- 考虑 $P(\mathrm{y}=y|\boldsymbol{v},\boldsymbol{\mu})=\mathrm{softmax}(\boldsymbol{W}^T(\boldsymbol{v}\odot\boldsymbol{\mu})+\boldsymbol{b})_y$

$$\tilde{P}_{\text{ensemble}}(\mathrm{y}=y|\boldsymbol{v})=\sqrt[2^n]{\prod_{\boldsymbol{\mu}\in\{0,1\}^n}P(\mathrm{y}=y|\boldsymbol{v};\boldsymbol{\mu})}$$

使用集成成员预测分布的几何平均而不是算术平均

$$=\sqrt[2^n]{\prod_{\boldsymbol{\mu}\in\{0,1\}^n}\frac{\exp\left(\boldsymbol{W}_{y,:}^T(\boldsymbol{v}\odot\boldsymbol{\mu})+b_y\right)}{\sum_{y'}\exp\left(\boldsymbol{W}_{y',:}^T(\boldsymbol{v}\odot\boldsymbol{\mu})+b_{y'}\right)}}$$

$$\propto\sqrt[2^n]{\prod_{\boldsymbol{\mu}\in\{0,1\}^n}\exp\left(\boldsymbol{W}_{y,:}^T(\boldsymbol{v}\odot\boldsymbol{\mu})+b_y\right)}$$

$$=\exp\left(\frac{1}{2^n}\sum_{\boldsymbol{\mu}\in\{0,1\}^n}\left(\boldsymbol{W}_{y,:}^T(\boldsymbol{v}\odot\boldsymbol{\mu})+b_y\right)\right)$$

$$=\exp\left(\frac{1}{2}\boldsymbol{W}_{y,:}^T\boldsymbol{v}+b_y\right)$$

Dropout 是一种模型集成（Bagging）

- 权重比例推断规则对许多无非线性隐藏单元的模型族是精确的
  - softmax函数回归分类
  - 条件正态输出的回归网络
  - 隐层不含非线性的深度网络

Dropout是一种正则化

- 考虑基于指数分布族的广义线性模型，囊括线性回归、对率回归

$$P_{\boldsymbol{\beta}}(y|\boldsymbol{x}) = h(y)\exp\{y\,\boldsymbol{x}^\top\boldsymbol{\beta} - A(\boldsymbol{x}^\top\boldsymbol{\beta})\}$$

$$A(\boldsymbol{x}^\top\boldsymbol{\beta}) = \log\int h(y)\exp\{y\,\boldsymbol{x}^\top\boldsymbol{\beta}\}\,\mathrm{d}y$$ 为对数配分函数，用于归一化

- 指数分布族的重要性质

$$A'(\boldsymbol{x}^\top\boldsymbol{\beta}) = \mathbb{E}_{P_{\boldsymbol{\beta}}(y|\boldsymbol{x})}[y]$$

$$A''(\boldsymbol{x}^\top\boldsymbol{\beta}) = \mathbb{V}_{P_{\boldsymbol{\beta}}(y|\boldsymbol{x})}[y]$$

Dropout是一种正则化

➢对于样本$(\boldsymbol{x}, y)$，其损失定义为$\ell_{\boldsymbol{x},y}(\boldsymbol{\beta}) = -\log P_{\boldsymbol{\beta}}(y|\boldsymbol{x})$

➢考虑Dropout 噪声 $\tilde{\boldsymbol{x}} = \boldsymbol{x} \odot \boldsymbol{\xi}$

ξ_j	**0**	$\mathbf{1/(1-\delta)}$
概率p	δ	$1-\delta$

$$\mathbb{E}[\tilde{\boldsymbol{x}}] = \boldsymbol{x}$$

$$\mathbb{V}[\tilde{\boldsymbol{x}}] = \frac{\delta}{1-\delta}\mathrm{diag}(\boldsymbol{x}\boldsymbol{x}^\top)$$

$$\mathbb{E}[\tilde{x}_j] = x_j \quad \mathbb{V}[\tilde{x}_j] = \frac{\delta}{1-\delta}x_j^2$$

➢含噪声的损失函数定义为

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \mathbb{E}_{\xi}\left[\ell_{\tilde{\boldsymbol{x}}_i, y_i}(\boldsymbol{\beta})\right] \\
&= -\sum_{i=1}^{n}\left(y_i\, \boldsymbol{x}_i^\top \boldsymbol{\beta} - \mathbb{E}_{\xi}\left[A\left(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{\beta}\right)\right]\right) \\
&= \sum_{i=1}^{n} \boxed{-\left(y_i\, \boldsymbol{x}_i^\top \boldsymbol{\beta} - A\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right)} + \boxed{\left(\mathbb{E}_{\xi}\left[A\left(\tilde{\boldsymbol{x}}_i^\top \boldsymbol{\beta}\right)\right] - A\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right)} \\
&= \sum_{i=1}^{n} \boxed{\ell_{\boldsymbol{x}_i, y}(\boldsymbol{\beta})} + \boxed{R(\boldsymbol{\beta})}
\end{aligned}$$

正则项

Dropout是一种正则化

- 正则项 $$R(\boldsymbol{\beta})=\sum_{i=1}^{n}\mathbb{E}_{\xi}\left[A\left(\widetilde{\boldsymbol{x}}_i^{\top}\boldsymbol{\beta}\right)\right]-A\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)$$

- 考虑$A(\widetilde{\boldsymbol{x}}_i^{\top}\boldsymbol{\beta})$在$\boldsymbol{x}_i^{\top}\boldsymbol{\beta}$附近的二阶泰勒展开

$$\mathbb{E}_{\xi}\left[A\left(\widetilde{\boldsymbol{x}}_i^{\top}\boldsymbol{\beta}\right)\right]\approx A\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)+A'\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)\mathbb{E}_{\xi}\left[\left(\widetilde{\boldsymbol{x}}_i^{\top}\boldsymbol{\beta}-\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)\right]+1/2A''\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)\mathbb{E}_{\xi}\left[\left(\widetilde{\boldsymbol{x}}_i^{\top}\boldsymbol{\beta}-\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)^{2}\right]$$

$$=A\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)+\frac{1}{2}A''\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)\boldsymbol{\beta}^{\top}\mathbb{V}_{\xi}(\widetilde{\boldsymbol{x}}_i)\boldsymbol{\beta}$$

$$\Rightarrow\quad R(\boldsymbol{\beta})=\frac{1}{2}\frac{\delta}{1-\delta}\boldsymbol{\beta}^{\top}\sum_{i=1}^{n}A''\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)\mathrm{diag}\left(\boldsymbol{x}_i\boldsymbol{x}_i^{\top}\right)\boldsymbol{\beta}$$

$$=\frac{1}{2}\frac{\delta}{1-\delta}\boldsymbol{\beta}^{\top}\mathrm{diag}(\boldsymbol{X}^{\top}V(\boldsymbol{\beta})\boldsymbol{X})\boldsymbol{\beta}$$

记$V(\boldsymbol{\beta})$为$n\times n$对角阵，
对角线上元素为$A''\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)$

Dropout是一种正则化

- 对率回归，记$P_{\boldsymbol{\beta}}(y=1|\boldsymbol{x}_i)=p_i$，

$$则A''(\boldsymbol{x}_i^\top\boldsymbol{\beta})=\mathbb{V}_{P_{\boldsymbol{\beta}}(y|\boldsymbol{x}_i)}[y]=p_i(1-p_i)$$

$$R(\boldsymbol{\beta})=\frac{1}{2}\frac{\delta}{1-\delta}\boldsymbol{\beta}^\top\mathrm{diag}(\boldsymbol{X}^\top V(\boldsymbol{\beta})\boldsymbol{X})\boldsymbol{\beta}$$
$$=\frac{1}{2}\frac{\delta}{1-\delta}\sum_{i,j}x_{ij}^2\,\beta_j^2\,p_i(1-p_i)$$

Dropout偏好置信度高的预测和小的β

DropConnect

- DropConnect：将节点中的每个与其相连的输入权值以1-p的概率变成0
- Dropout：随机的将隐层节点的输出变成0