自动编码器

(Autoencoder)

刘远超

哈尔滨工业大学

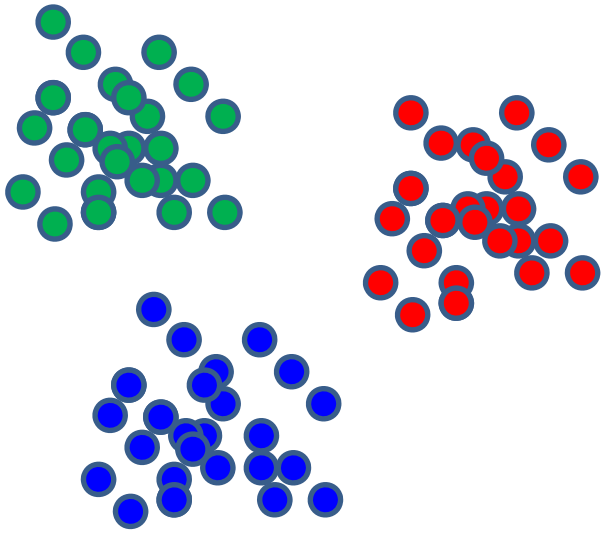
计算机科学与技术学院

有监督学习

- 数据: (x, y)， 其中x是数据特征，y是标签
- 目的: 学习映射 ： x -> y
- 例子：
  - 分类问题
  - 回归问题
  - 图像标题生成等

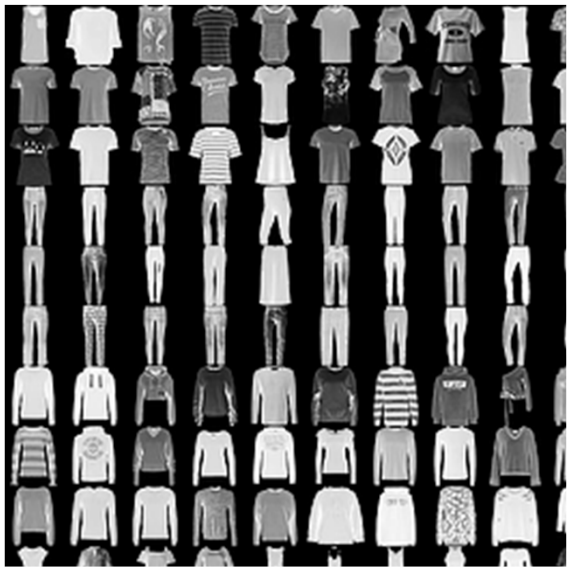
无监督学习

- 其特点是训练样本数据的标记信息未知，因此其目标往往是要通过对无标签训练样本的学习来揭示出数据的内在及规律。
  - 数据: x
  - 只有数据，没有标签
  - 目标:学习数据的结构
  - 例子：聚类、密度估计、异常检测等



生成式模型（Generative Models）

- 从训练数据中学习$p_{model}(x)$，其与训练数据中的分布$p_{data}(x)$尽可能接近；
- 然后从$p_{model}(x)$中生成新的样本



训练数据~ $p_{data}(x)$

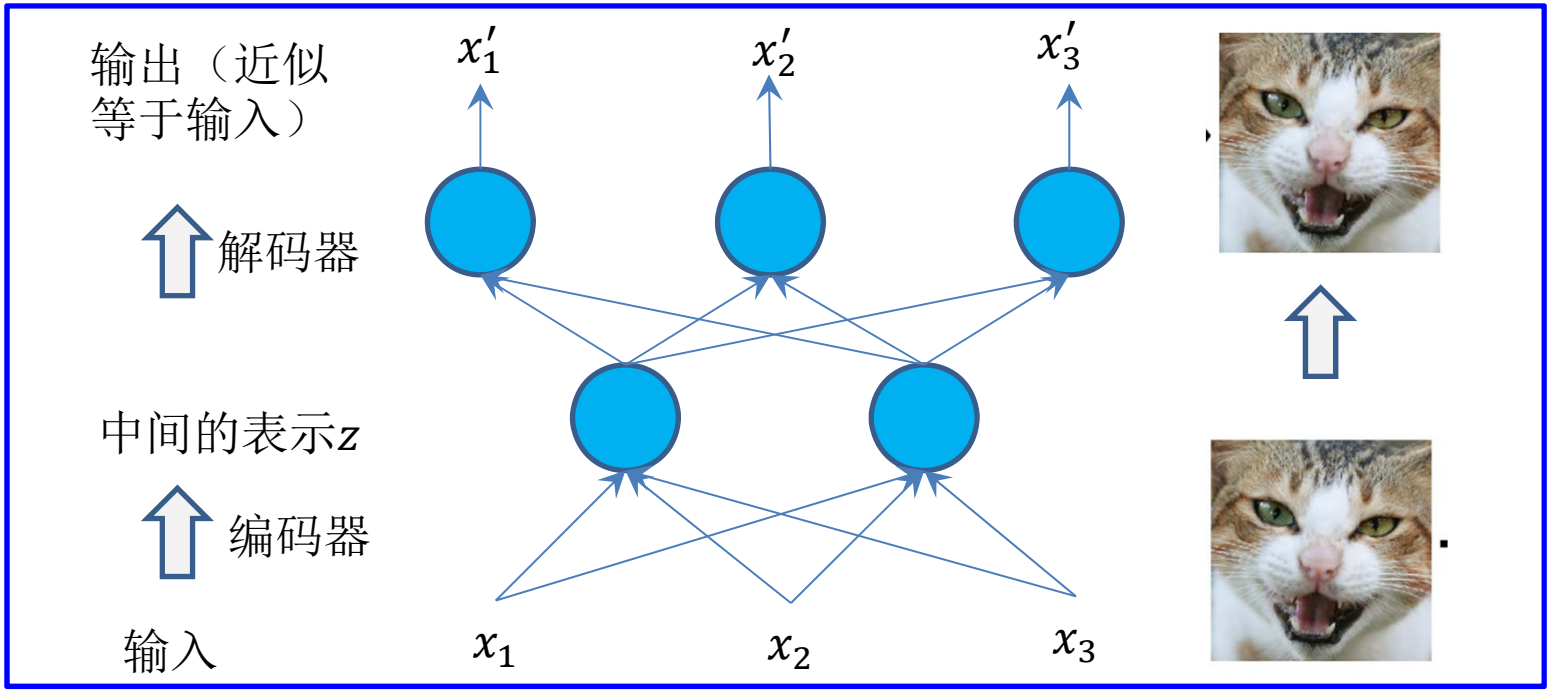
生成的样本数据~ $p_{model}(x)$

自动编码器(Autoencoder)

- 自动编码器是一种无监督的神经网络模型，其目标是通过训练网络忽略信号“噪声”，从而学习得到数据的低维度表示（编码）。
- 通常分为两步：

  1）学习到输入数据的隐含特征，即编码(coder) ϕ: $x \to z$;

  2）用隐含特征重构原始的输入数据，即解码(decoder) ψ: $z \to x$



只有一个隐藏层的自动编码器

- **编码：** 将x 映射为z，例如采用单层感知机：

$$\boldsymbol{z = \sigma(Wx + b)}$$

这里$\sigma\cdot$为激活函数，如sigmoid或 ReLU等；W为权重矩阵，b为偏差向量。

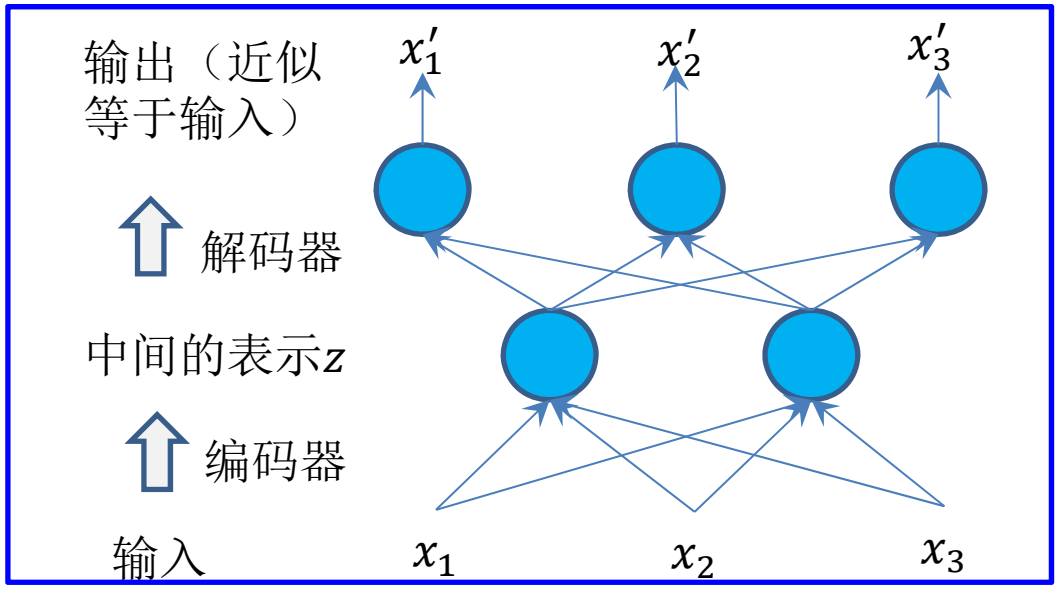
z 通常被称为代码（code）、潜在变量（latent variables）或潜在表示（latent representation）。

- **解码：** 将 z映射到与x形状相同的重构 x'，例如采用单层感知机：

$$\boldsymbol{x' = \sigma'(W'z + b')}$$

解码器中的，σ'、W'、b'也可以和编码器中的σ、W、b不同。

自动编码器的损失函数



- 自动编码器也需要通过训练来减少数据重建的误差（如平方误差）：

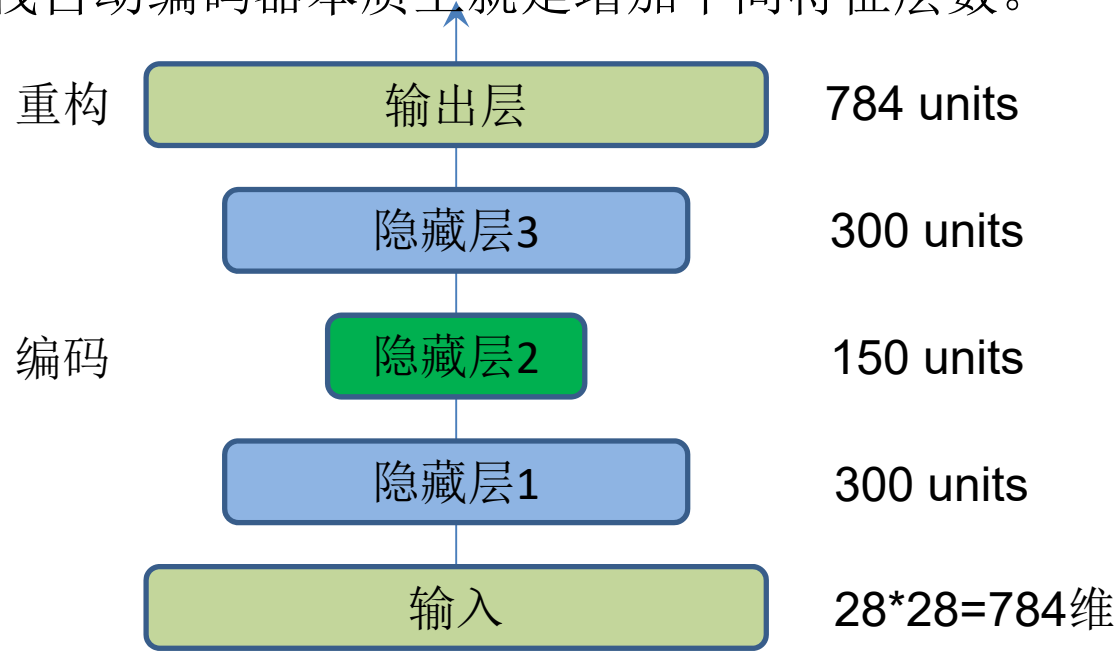
$$\boldsymbol{\phi, \psi} = \underset{\boldsymbol{\phi,\psi}}{\arg\min} \, ||\boldsymbol{X} - (\boldsymbol{\psi} \circ \boldsymbol{\phi})\boldsymbol{X}||^2$$

例如对于单层的感知机，则损失函数通常为

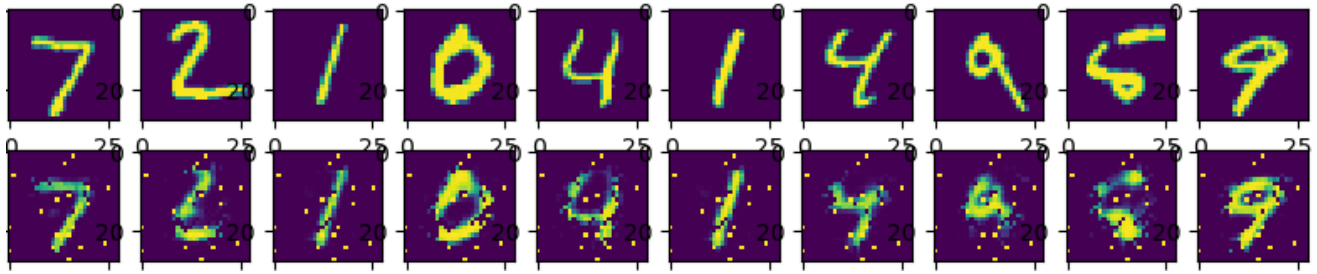
$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{x'}) = ||\boldsymbol{x} - \boldsymbol{x'}||^2 = ||\boldsymbol{x} - \boldsymbol{\sigma'}\left(\boldsymbol{W'}(\boldsymbol{\sigma}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})) + \boldsymbol{b'}\right)||^2$$

堆栈自动编码器

- 堆栈自动编码器本质上就是增加中间特征层数。

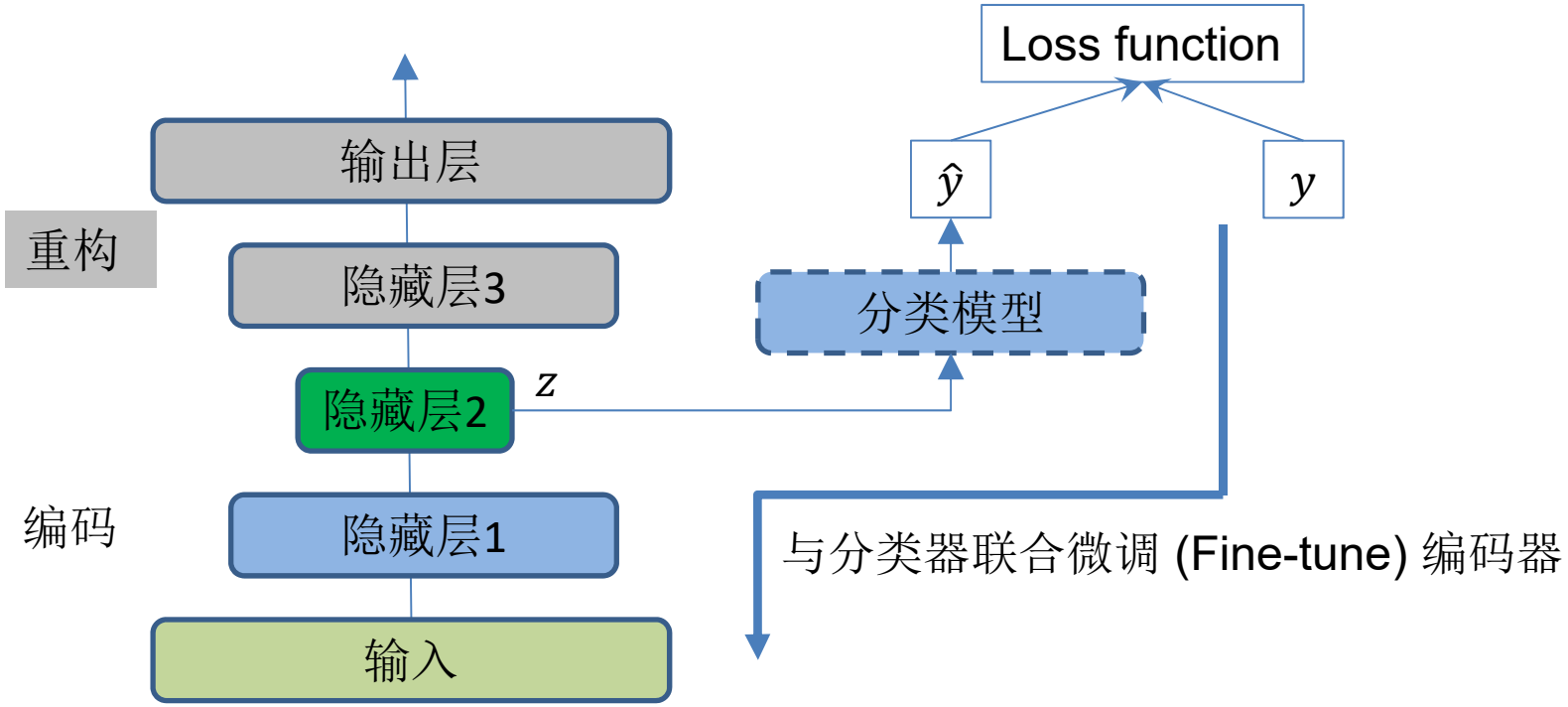


对MNIST构造自动编码器（两个隐含层）



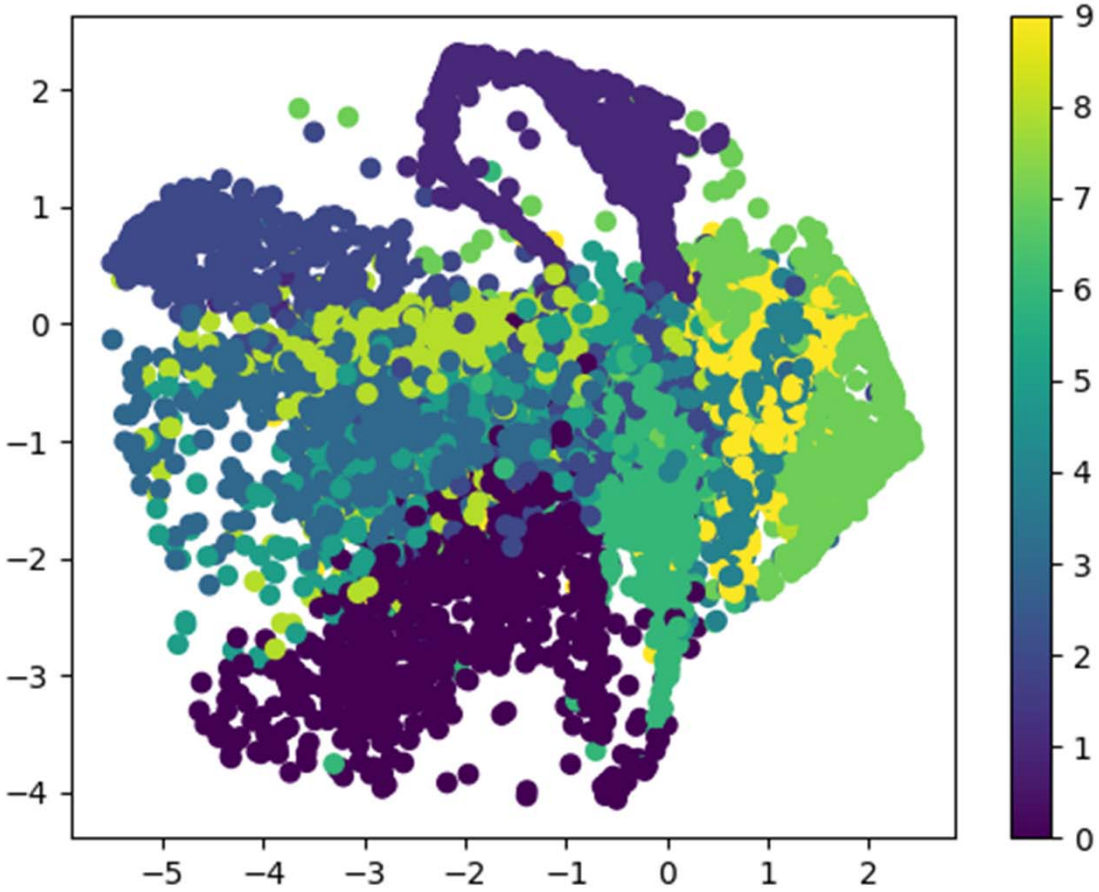
用自动编码得到新特征

- 模型训练完成后，可以将解码器去掉，仅仅使用编码器得到重构的特征。得到的特征可应用在有监督的模型中，如分类模型。
- 这样做的好处是，可以用许多无标签的数据学习得到通用的特征表示。因为，监督学习通常的标注数据较少。



基于自动编码器的图像聚类可视化

- MNIST数据：从784维→128→64→10→2维。



MNIST数据的二维平面聚类（不同颜色表示不同的数字）

Thanks!

变分自动编码器

(Variational Autoencoders)

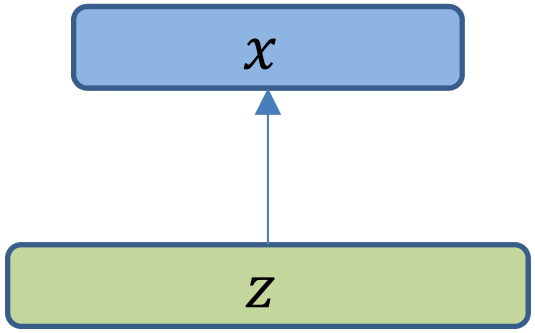
刘远超

哈尔滨工业大学

计算机科学与技术学院

变分自动编码器-图像生成

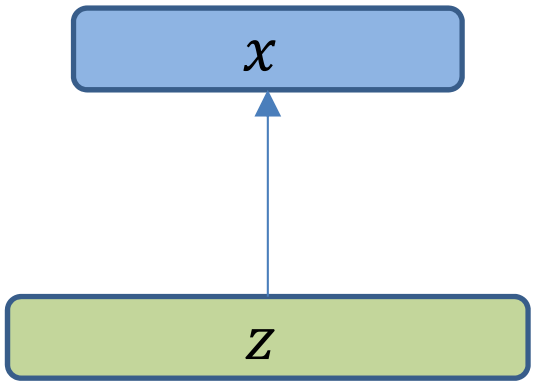
- **问题：**既然自动编码器可以重构原始的输入图像，那么如何生成新的图像呢？



- 变分自动编码器（Variational Autoencoders，VAE）：
  - 假设潜在变量 z 服从某种先验分布（高斯分布）。模型训练完毕后，可以从这种先验分布中采样得到潜在变量，然后在解码器中得到新的样本。
  - 在自动编码器基础上加入了随机因子。

Diederik P Kingma, Max Welling (2013). Auto-Encoding Variational Bayes. ICLR 2014

变分自动编码器的原理



$p_{\theta^*}(x|z^{(i)})$：即在给定z的条件下，从条件分布$p(x|z)$中采样生成样本。

z从$p_{\theta^*}(z)$中采样：这个分布可以采用高斯分布，因为其是对z中的每个元素的一种自然的先验（prior）假设。

- 需要估计得到上述模型中的优化参数θ^*。
  - 一种常见策略是最大化训练数据的似然函数，即

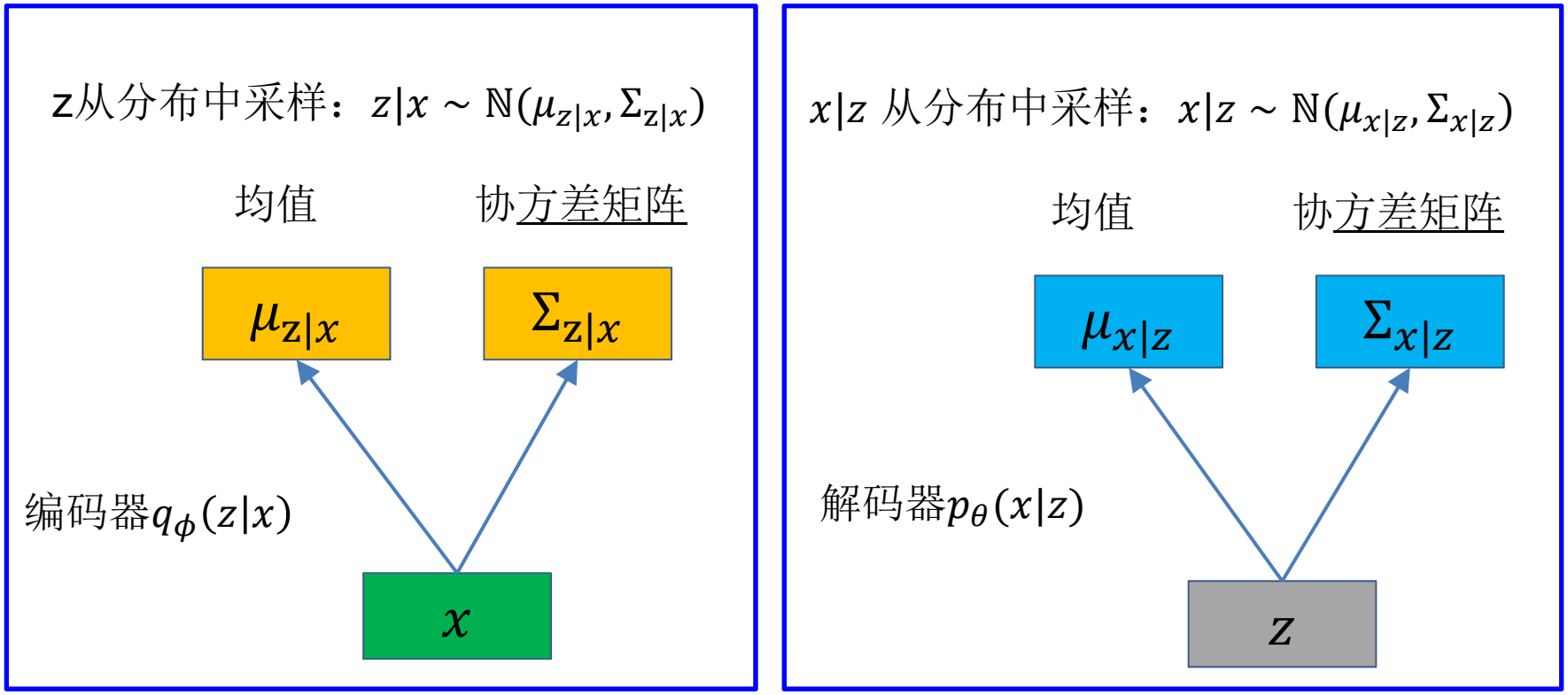
    $\mathrm{p}_\theta(x) = \int p_\theta(z)\, p_\theta(x|z)dz$。

    办法：找出该似然函数的一个下界，对该下界进行优化。

Diederik P Kingma, Max Welling (2013). Auto-Encoding Variational Bayes. ICLR 2014

变分自动编码器中的分布及采样

- 由于是在建模数据的概率生成，所以编码器和解码器都是概率的



Diederik P Kingma, Max Welling (2013). Auto-Encoding Variational Bayes. ICLR 2014

变分自动编码器的对数似然函数

●现在，（最大化）训练数据的似然函数，以找到模型的参数θ

$$\boldsymbol{log p_\theta(x^{(i)})} = \mathrm{E}_{z\sim q_\phi(z|x^{(i)})}[log p_\theta(x^{(i)})]$$

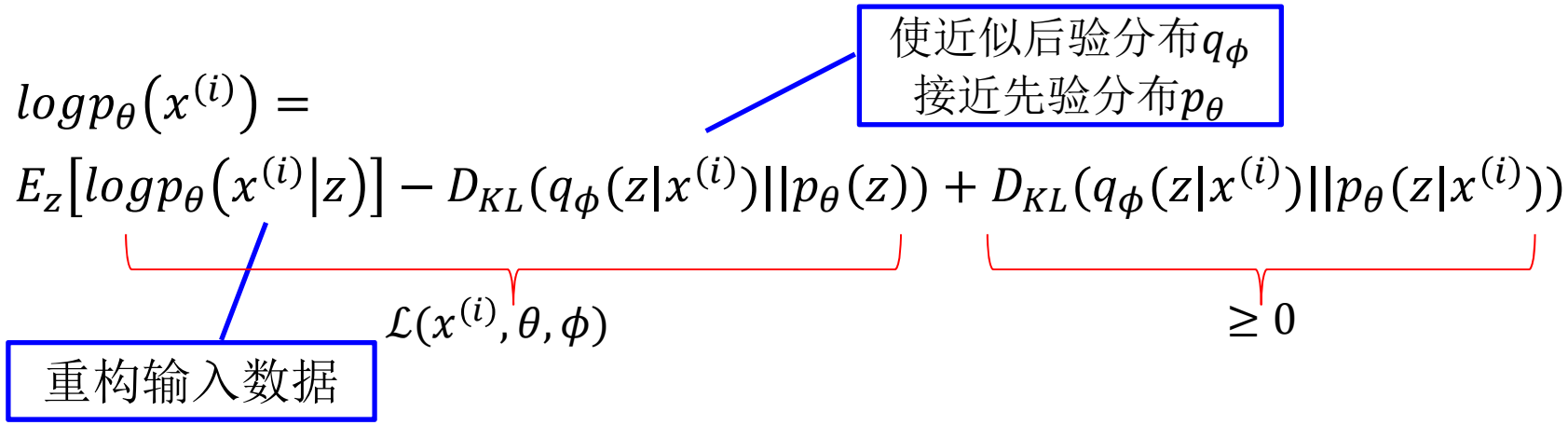
$$= E_z[log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})}]$$

$$= E_z[log \frac{p_\theta(x^{(i)}|z)p_\theta(z)q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})q_\phi(z|x^{(i)})}]$$

$$= E_z[log p_\theta(x^{(i)}|z)] - E_z\left[log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)}\right] + E_z[log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})}]$$

$$= \boldsymbol{E_z[log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))}$$

变分自动编码器的对数似然函数



可见：

- $logp_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)$，即前两项的和$\mathcal{L}(x^{(i)}, \theta, \phi)$是一个可以处理的下界。可以对其求梯度，并进行优化。
- 因此变分编码器实际上是在优化数据的对数似然的下界，并求解参数：

$$\theta^*, \phi^* = \arg\max_{\theta,\ \phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)$$

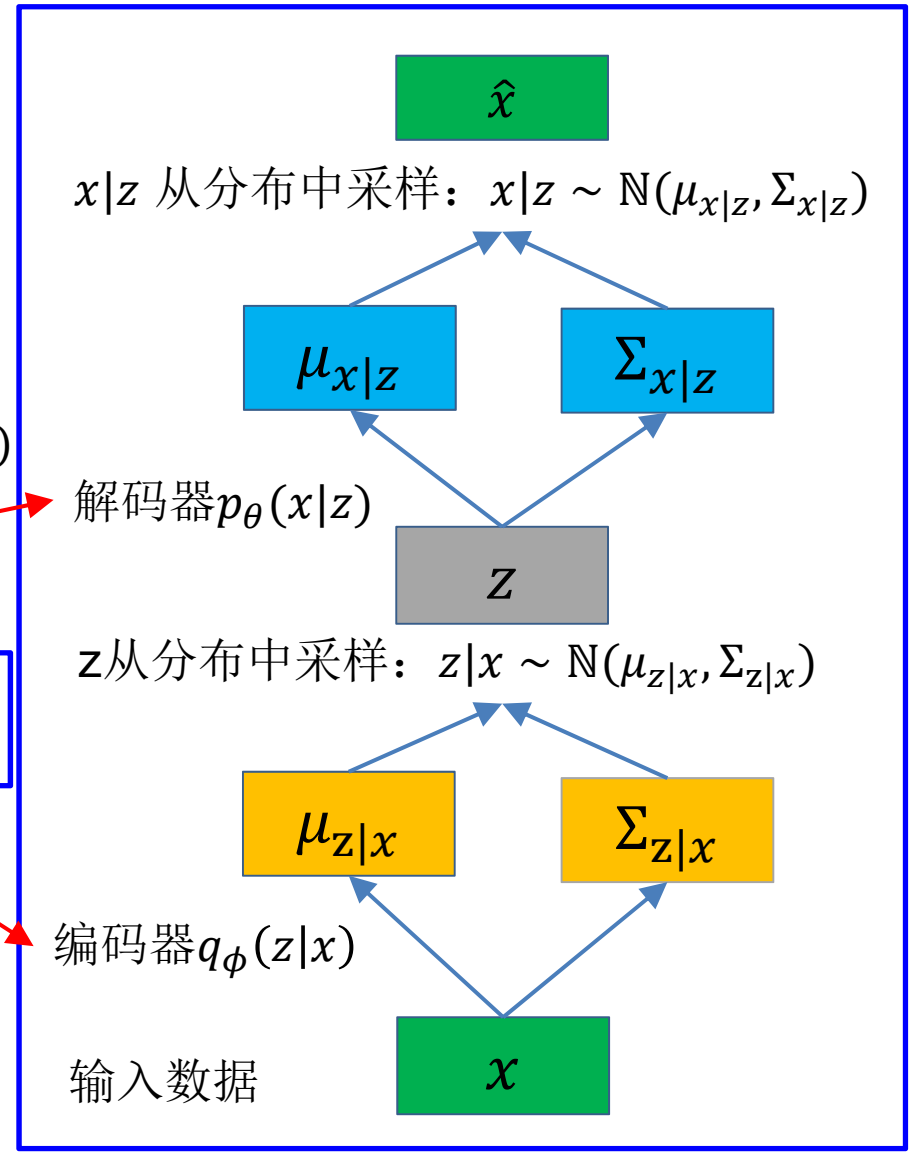
变分自动编码器的训练过程

现在，我们已经知道要优化似然函数的下界：

$$\mathcal{L}(x^{(i)}, \theta, \phi)$$
$$= E_z[\log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$$

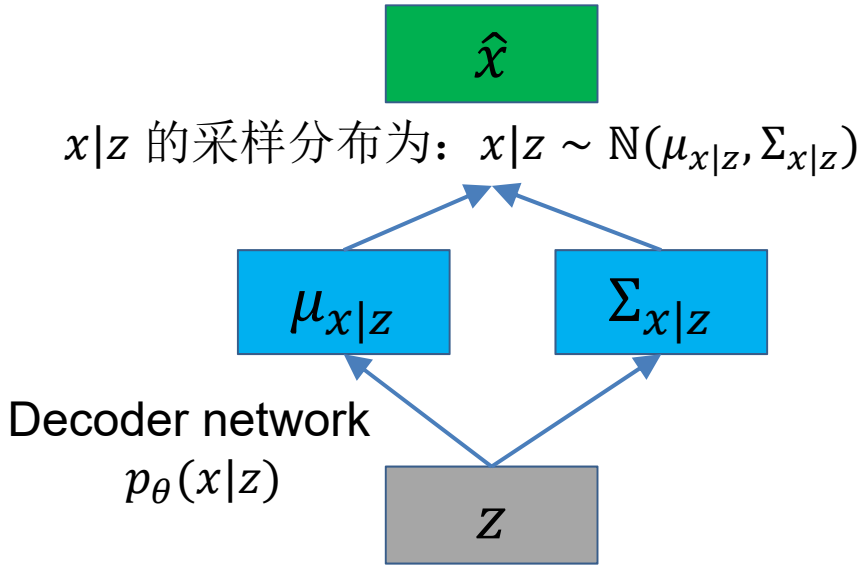
最大化原始输入据被重构的对数似然

使近似后验分布q_ϕ接近先验分布p_θ



生成新的样本

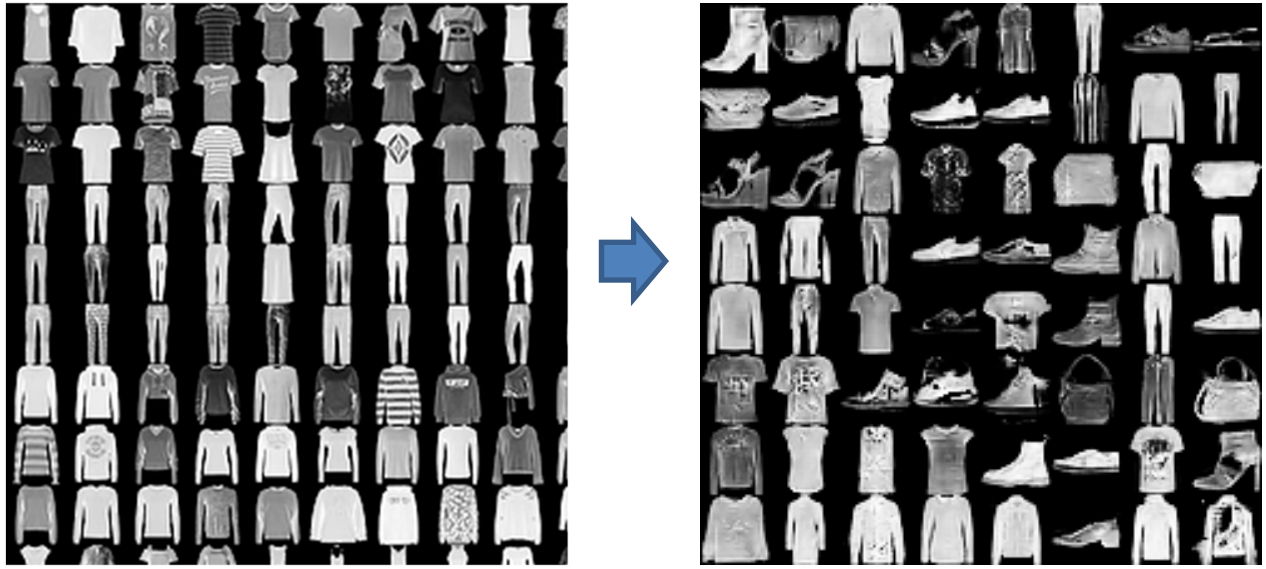
生成数据只需要解码器：

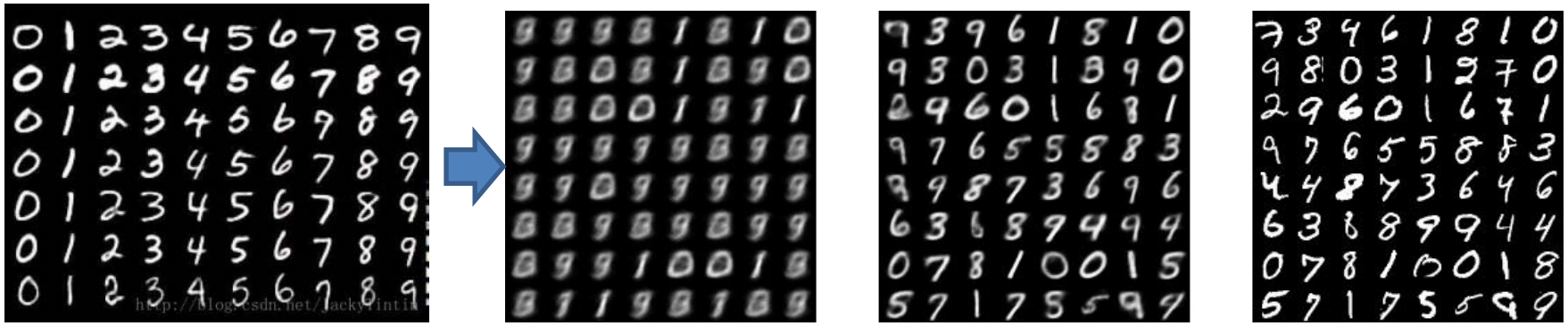


z的采样分布为：$z|x \sim \mathbb{N}(\mathbf{0}, \boldsymbol{I})$

- 生成阶段先对z进行采样。从先验标准正态分布$\mathbb{N}(\mathbf{0}, \boldsymbol{I})$中采样，而不再从训练过程中的后验概率分布中采样。
- 然后，再采样得到生成的数据 $\hat{x}$ 。

VAE生成的图像举例





Thanks!

生成对抗网络

(Generative Adversarial Networks)

刘远超

哈尔滨工业大学

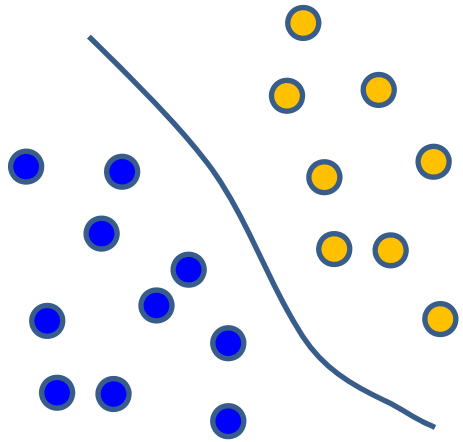
计算机科学与技术学院

判别式模型和生成式模型

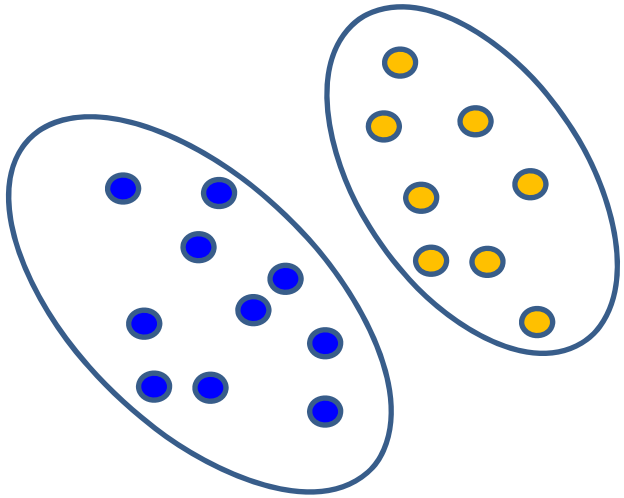
假设研究对象的观测变量为 $\boldsymbol{X}$，类别变量为 $\boldsymbol{Y}$，则

- 判别式模型（Discriminative models, 或conditional models）：
  - 按照一定的判别准则，从数据中直接学习决策函数$\boldsymbol{Y = f(X)}$或者条件概率分布$\boldsymbol{P(Y|X;\theta)}$作为预测的模型。
  - 典型的判别模型包括：k近邻法、决策树、最大熵模型、支持向量机等；
- 生成式模型（Generative models）：
  - 从数据中学习联合概率分布$\boldsymbol{P(X,Y)}$，其之后可以转变为$\boldsymbol{P(Y|X)}$作为预测的模型，例如利用条件概率分布$\boldsymbol{P(Y|X)=P(X,Y)/P(X)}$。
  - 典型的生成模型有：朴素贝叶斯法、高斯混合模型、马尔科夫模型等。

判别式模型和生成式模型的区别



决策边界



判别式模型：只是对给定的样本进行分类。不关心数据如何生成。

生成式模型：根据生成假设，哪个类别最有可能生成这个样本？

简单举例

- 假设输入数据是 $x \in \{1,2\}$, x 的标签$y \in \{0,1\}$, 且有如下 4个数据点 $(x,y) = \{(1,0),(1,0),(2,0),(2,1)\}$。则对于上述数据，根据经验测量估计

联合概率分布$\boldsymbol{p(x,y)}$如下：

	$y=0$	$y=1$
$x=1$	$1/2$	0
$x=2$	$1/4$	$1/4$

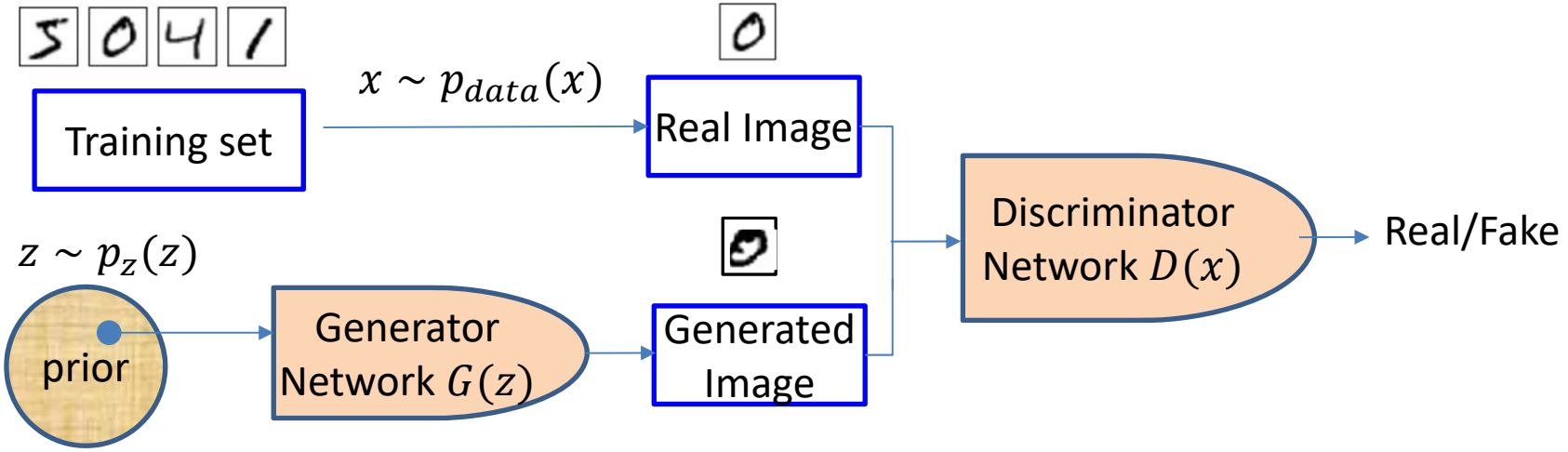
条件概率分布$\boldsymbol{p(y|x)}$ 则为:

	$y=0$	$y=1$
$x=1$	1	0
$x=2$	$1/2$	$1/2$

生成对抗网络—现实世界的启发

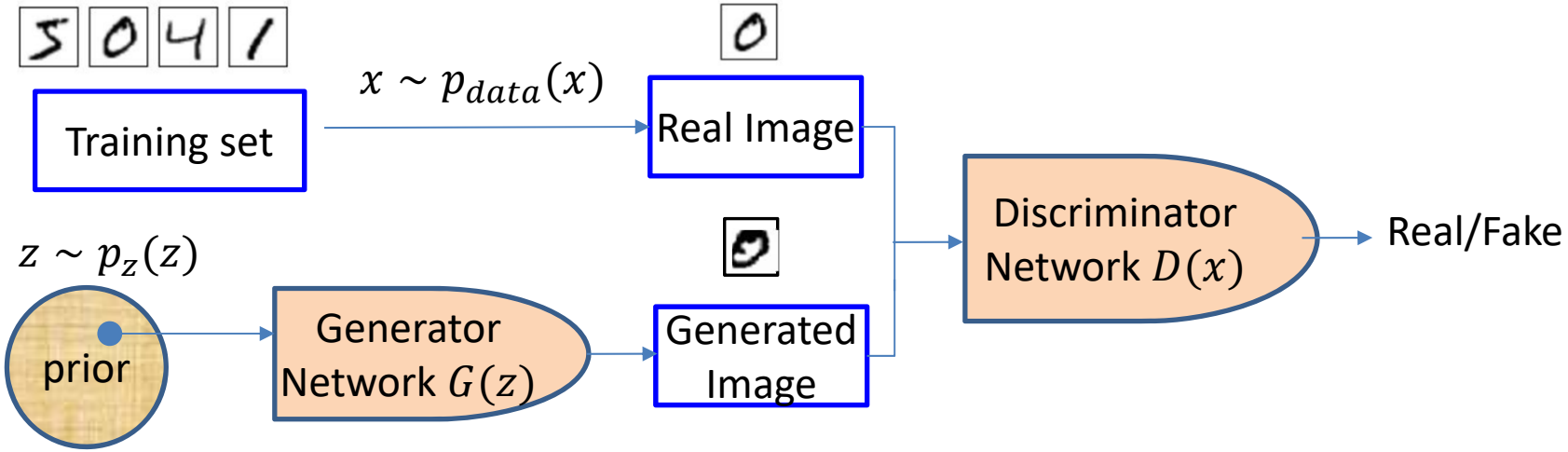
生成对抗网络GAN的基本原理

- GAN **(Generative Adversarial Networks)**包括两部分：生成器和判别器。
  - 生成器: 从随机噪声中生成图像（随机噪声通常从均匀分布或高斯分布中获取）
  - 判别器: 其输入为生成器生成的图像和来自训练集中的真实图像，并对其进行判别。得到输出值为一个0到1之间的数，表示图像为真实图像的概率，real为1，fake为0。



Goodfellow, Ian; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Yoshua (2014). Generative Adversarial Networks (PDF). Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014). pp. 2672–2680.

判别模型的目标函数



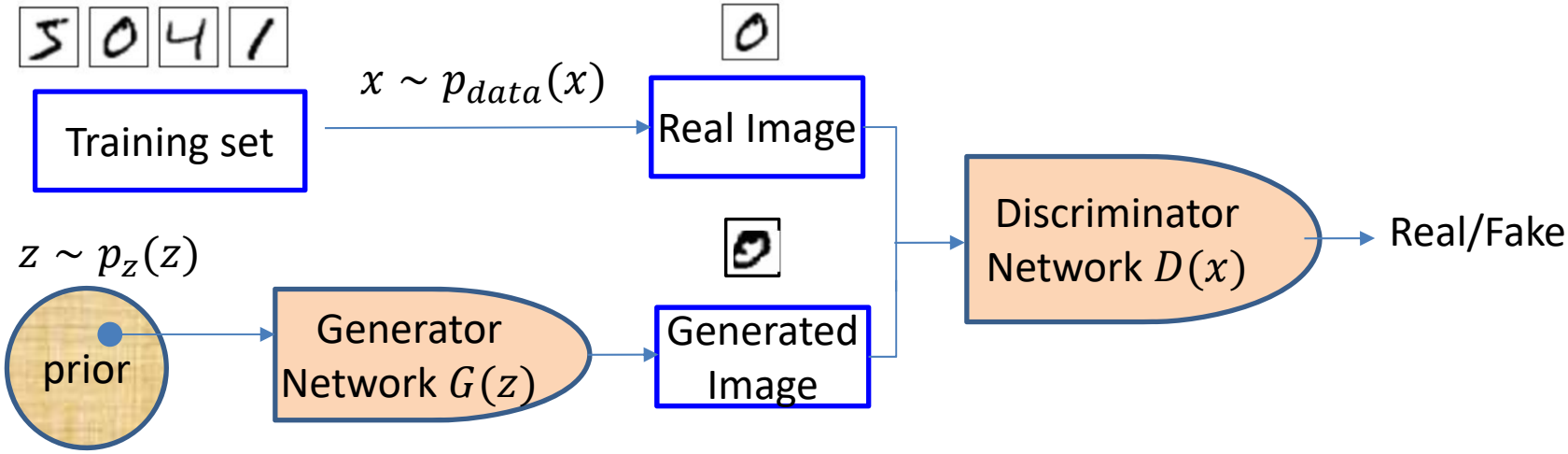
$$\min_G \max_D V(D,G) = \mathrm{E}_{x\sim p_{data}(x)}[logD(x)] + \mathrm{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

当对判别器D进行优化时，要让总数值最大，即

$$\max_D V(D,G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1-D(G(z)))]$$

- 等号右边第一项，$D(x)$表示 x 来自真实数据而不是生成数据的概率，所以损失函数为$logD(x)$，因为其判别结果越接近于1越好；
- 等号右边第二项， z 是随机的输入，$G(z)$表示生成的样本，因此我们希望判别结果$D(G(z))$越接近于0越好。

生成模型的目标函数



$$\min_G \max_D V(D,G) = \mathrm{E}_{x\sim p_{data}(x)}[logD(x)] + \mathrm{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

当对生成器G进行优化时，要让函数取值最小，因此有

$$\min_G V(D,G) = E_{z\sim p_z(z)}[log(1-D(G(z)))]$$

- 生成模型得到的图像肯定是假的，所以其损失函数是上述损失函数的第二部分。好的生成器应该使得判别结果$D(G(z))$尽量接近1，即让判别器以为是真实的图片。
- 所以G的训练目标是将D错误的概率最大化。

GAN的训练算法

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

固定生成器G，更新判别器D；

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

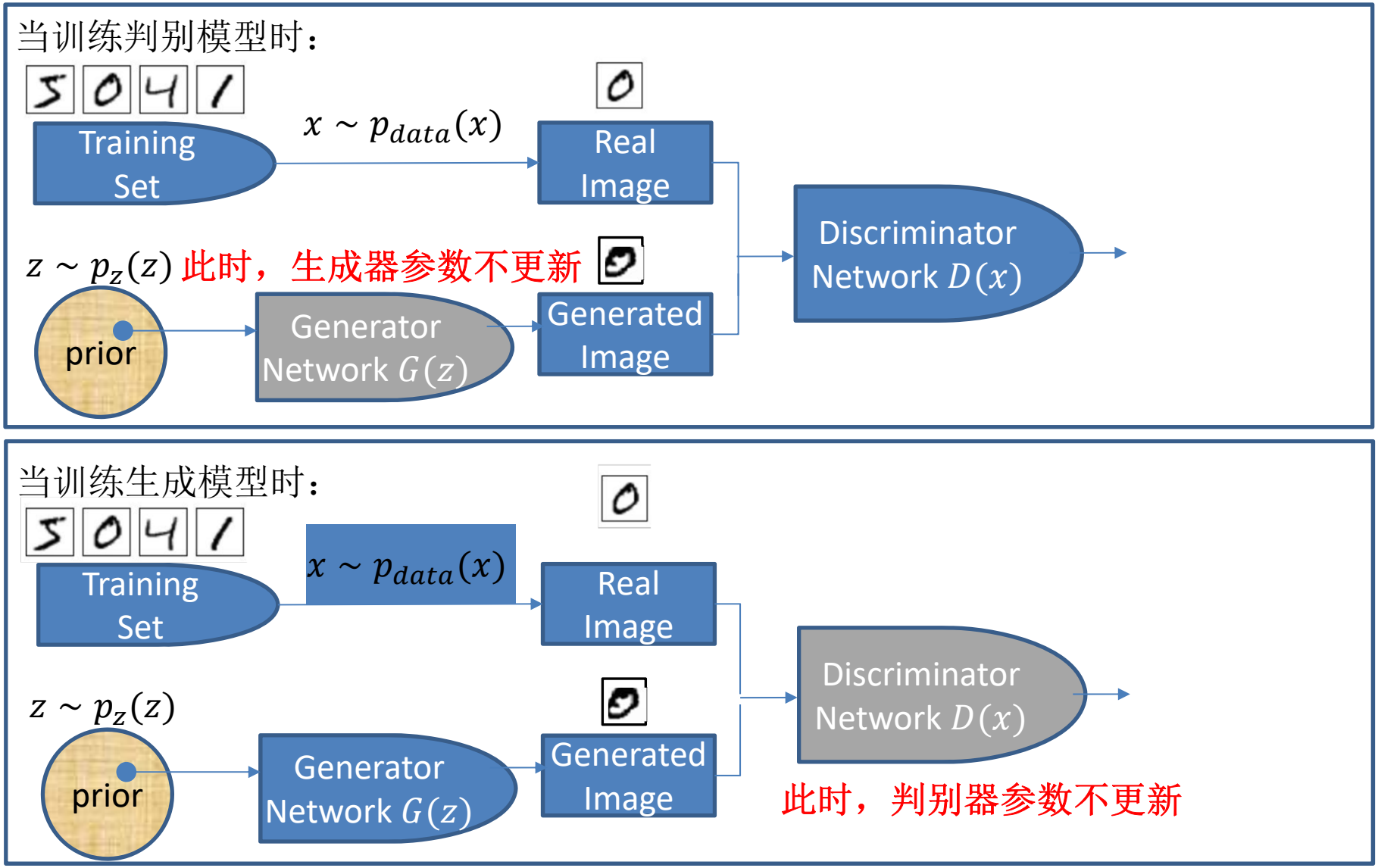
固定判别器D，更新生成器G。

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Goodfellow, Ian; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Yoshua (2014). Generative Adversarial Networks (PDF). Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014). pp. 2672–2680.

GAN训练中的参数更新



GAN在MNIST上的运行效果

- *https://github.com/hwalsuklee/tensorflow-generative-model-collections*
- TensorFlow 1.0
- Python 3



Thanks!