自编码器

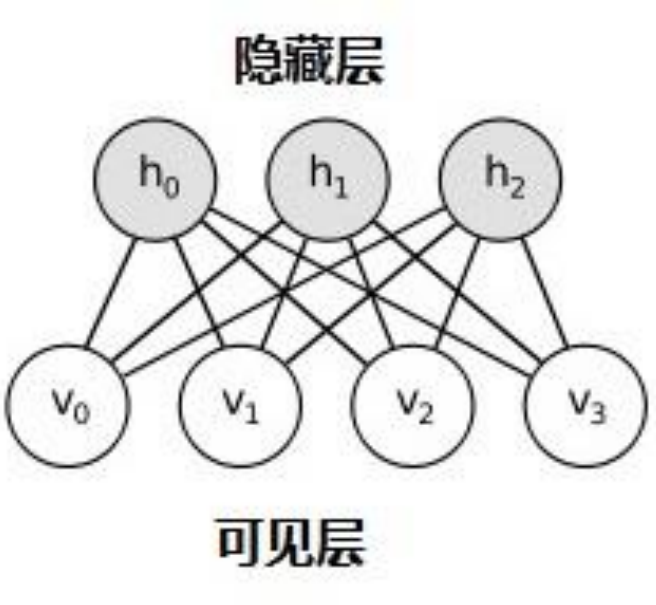
李娜

朱仁煜

大纲

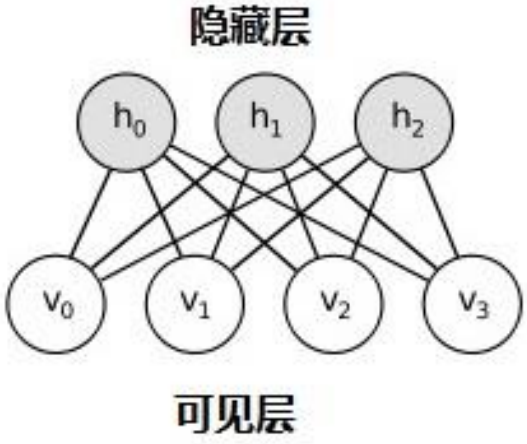
1、自编码器
2、栈式自编码器
3、欠完备自编码器
4、稀疏自编码器
5、去噪自编码器
6、卷积/循环自编码器

受限玻尔兹曼机



1、受限玻尔兹曼机是由可见层和隐藏层构成的两层结构。
2、可见层由可见变量组成，隐藏层由隐藏变量组成。
3、可见层和隐藏层之间相互连接，层内之间无连接。

受限玻尔兹曼机



能量函数：$$E(v,h,\theta)=-\sum_{i=1}^{n}b_iv_i-\sum_{j=1}^{m}c_jh_j-\sum_{i=1}^{n}\sum_{j=1}^{m}w_{ij}v_ih_j$$

联合概率分布：$$p(v,h|\theta)=\frac{1}{Z}exp\{-E(v,h,\theta)\}$$

$$Z=\sum_{v,h}exp\{-E(v,h,\theta)\}$$

对数似然函数函数：

$$\begin{aligned} logL(\theta|v) &= log\tfrac{1}{Z}\textstyle\sum_h exp\{-E(v,h,\theta)\} \\ &= log\textstyle\sum_h exp\{-E(v,h,\theta)\}-log\sum_{v,h}exp\{-E(v,h,\theta)\} \end{aligned}$$

受限玻尔兹曼机

对θ求导计算梯度：

$$
\begin{aligned}
\frac{\partial logL(\theta|v)}{\partial\theta} &= \frac{\partial(log\sum_h exp\{-E(v,h,\theta)\})}{\partial\theta} - \frac{\partial(log\sum_{v,h} exp\{-E(v,h,\theta)\})}{\partial\theta} \\
&= -\frac{1}{\sum_h exp\{-E(v,h,\theta)\}}\sum_h(exp\{-E(v,h,\theta)\}\frac{\partial E(v,h,\theta)}{\partial\theta}) \\
&+ \frac{1}{\sum_{v,h} exp\{-E(v,h,\theta)\}}\sum_{v,h}(exp\{-E(v,h,\theta)\}\frac{\partial E(v,h,\theta)}{\partial\theta}) \\
&= -\sum_h(p(h|v)\frac{\partial E(v,h,\theta)}{\partial\theta}) + \sum_{v,h}(p(v,h)\frac{\partial E(v,h,\theta)}{\partial\theta})
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial logL(\theta|v)}{\partial w_{ij}} &= -\sum_h(p(h|v)\frac{\partial E(v,h,\theta)}{\partial w_{ij}}) + \sum_{v,h} p(v,h)(\frac{\partial E(v,h,\theta)}{\partial w_{ij}}) \\
&= \sum_{h_j}(p(h_j|v)v_i h_j) - \sum_{v_i,h_j} p(v_i,h_j)v_i h_j \\
&= \sum_{h_j}(p(h_j|v)v_i h_j) - \sum_{v_i}(p(v_i)\sum_{h_j} p(h_j|v_i)v_i h_j) \\
&= p(h_j=0|v)v_i\cdot 0 + p(h_j=1|v)v_i\cdot 1 \\
&- (\sum_{v_i} p(v_i)p(h_j=0|v_i)v_i\cdot 0 + \sum_{v_i} p(v_i)p(h_j=1|v_i)v_i\cdot 1) \\
&= p(h_j=1|v)v_i - \sum_{v_i}(p(v_i)p(h_j=1|v_i)v_i)
\end{aligned}
$$

受限玻尔兹曼机

对比散度算法（Algorithm：CD-1）主要步骤

输入：可视层向量 $\mathbf{v}$，隐藏层单元个数 m，学习效率 ε

输出：连接权重矩阵 W、可视层的偏置向量 $\mathbf{b}$、隐藏层的偏置向量 $\mathbf{c}$

初始化：连接权重矩阵 W、可视层的偏置向量 $\mathbf{b}$、隐藏层的偏置向量 $\mathbf{c}$ 为随机的较小数值；

For $i = 1,\ 2, \ldots,\ m$ (对所有隐藏层神经元)

计算 $p(h_i = 1|\mathbf{v})$，即 $p(h_i = 1|\mathbf{v}) = \log istic(c_i + \sum_j W_{j,i} v_j)$

从条件分布 $p(h_i = 1|\mathbf{v})$ 中抽取 $h_i \in \{0,1\}$

End For

For $i = 1,\ 2, \ldots,\ d$ (对所有可视层神经元)

计算 $p(v_i^* = 1|\mathbf{h})$，即 $p(v_i^* = 1|\mathbf{h}) = \log istic(b_i + \sum_j W_{i,j} h_j)$

从条件分布 $p(v_i^* = 1|\mathbf{h})$ 中抽取 $v_i^* \in \{0,1\}$

End For

For $i = 1,\ 2, \ldots,\ m$ (对所有隐藏层神经元)

计算 $p(h_i^* = 1|\mathbf{v}^*)$，即 $p(h_i^* = 1|\mathbf{v}^*) = \log istic(c_i + \sum_j W_{j,i} v_j^*)$

End For

更新各个参数值：

$$\mathrm{W} \leftarrow \mathrm{W} + \varepsilon \times [p(\mathbf{h} = 1|\mathbf{v})\mathbf{v}^T - p(\mathbf{h}^* = 1|\mathbf{v}^*)\mathbf{v}^{*T}]$$

$$\mathbf{b} \leftarrow \mathbf{b} + \varepsilon \times (\mathbf{v} - \mathbf{v}^*)$$

$$\mathbf{c} \leftarrow \mathbf{c} + \varepsilon \times [p(\mathbf{h} = 1|\mathbf{v}) - p(\mathbf{h}^* = 1|\mathbf{v}^*)]$$

受限玻尔兹曼机

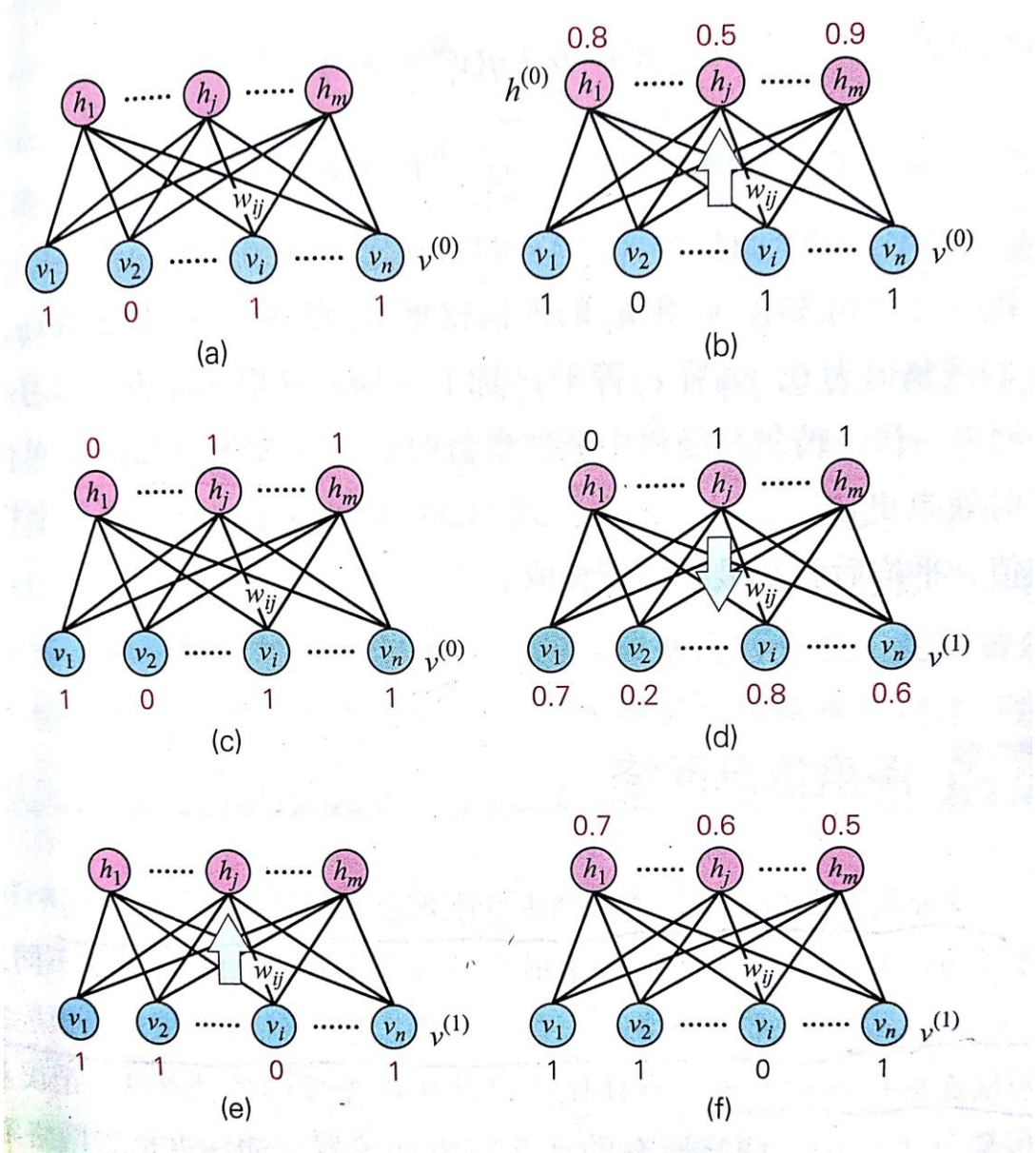
$$
\begin{aligned}
P(h|v) &= \frac{P(h,v)}{P(v)} \\
&= \frac{1}{P(v)}\frac{1}{Z}exp\{\sum_{i=1}^{n} b_i v_i + \sum_{j=1}^{m} c_j h_j + \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} v_i h_j\} \\
&= \frac{1}{Z'}exp\{\sum_{j=1}^{m} c_j h_j + \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} v_i h_j\} \\
&= \frac{1}{Z'}\prod_{j=1}^{n_h} exp\{c_j h_j + w_{ij} v_i h_j\}
\end{aligned}
$$

$$
\frac{1}{Z'} = \frac{1}{P(v)}\frac{1}{Z}exp\{\sum_{i=1}^{n} b_i v_i\}
$$

$$
\begin{aligned}
P(h_j = 1|v) &= \frac{P(h_j=1|v)}{P(h_j=0|v)+P(h_j=1|v)} \\
&= \frac{exp\{c_j + w_{ij} v_i\}}{1+exp\{c_j + w_{ij} v_i\}} \\
&= \frac{1}{1+exp\{-(c_j + w_{ij} v_i)\}} \\
&= \sigma(c_j + w_{ij} v_i)
\end{aligned}
$$

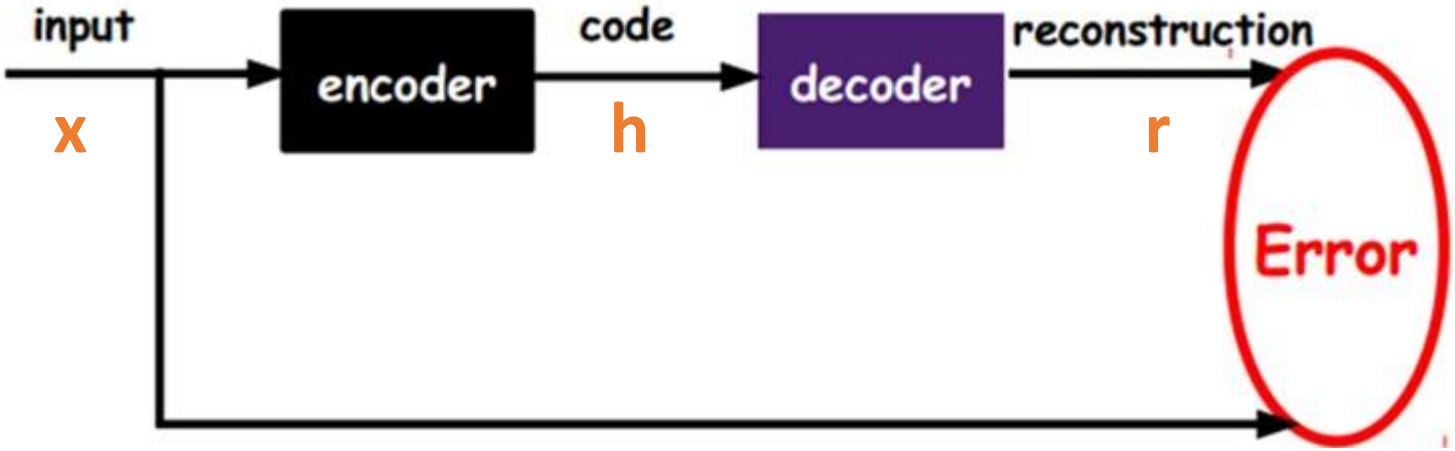
$$
P(v_i = 1|h) = \sigma(b_i + w_{ij} h_i)
$$

受限玻尔兹曼机



Autoencoder

自编码器是一种无监督学习的数据维度压缩和特征表达方法。



编码：$h = f(x)$将输入压缩为潜在空间表示

解码：$r = g(h)$ 重构来自隐空间表示的输入

损失函数：$L(x, g(f(x)))$

Autoencoder和Feedforward NN对比

1、Autoencoder是Feedforward NN的一种，最开始主要用于数据的降维或者特征的抽取，现在也被用于生成模型中。

2、Feedforward主要关注的是输出层和错误率，而Autoencoder主要关注隐层

Autoencoder和RBM对比

1、都起了降维的作用

2、都可以用来对神经网络进行预训练

3、训练都是无监督的

Autoencoder和RBM对比

区别：

1、自编码器希望通过非线性变换找到输入数据的特征表示，它是某种确定论性的模型；而RBM则是围绕概率分布进行的，它通过输入数据的概率分布来提取中间层表示，它是某种概率论性的模型。

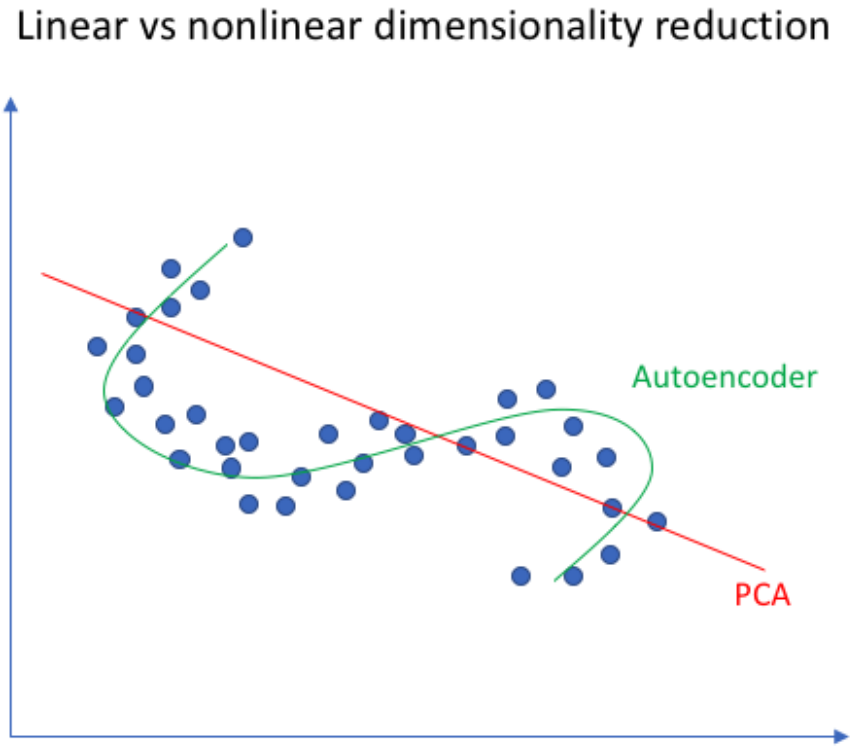
2、AE使用的是BP算法进行优化，而RBM是基于概率模型，使用CD算法进行优化。

	Autoencoder:	RBM:
	$h_i = s(W_i x + b_i)$	$P(h_i=1 \mid v) = s(W_i v + c_i)$
	$r_j = s(W_j^T h + b_{dj})$	$P(v_j=1 \mid h) = s(W_j^T h + b_j)$
Differences:	deterministic mapping h is a function of x.	stochastic mapping h is a random variable

自编码器和PCA对比

1、若Encoder和Decoder是线性的，将目标函数L换成均方误差，Autoencoder和PCA拥有相同的生成子空间。

2、Autoencoeder其实是增强的PCA，Autoencder具有非线性变换单元，因此学习到的Code对Input的表达能力更强。



Autoencoder存在的问题：

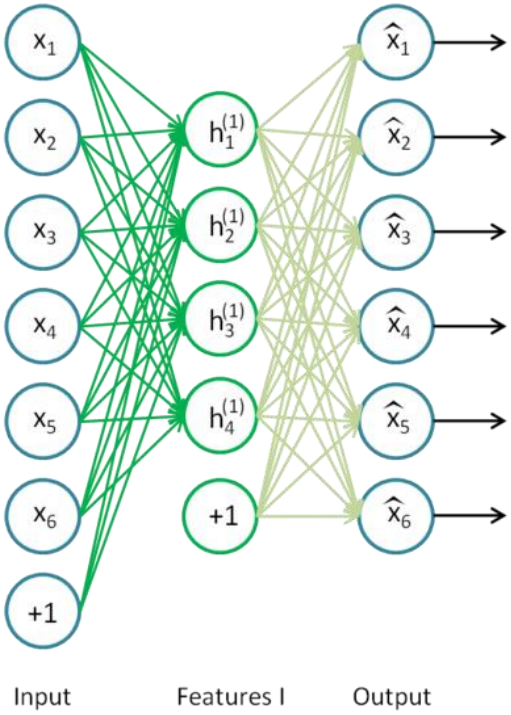
1、因为Input和Output是完全相同的，所以其压缩能力有限，仅适用于与训练样本相似的样本。

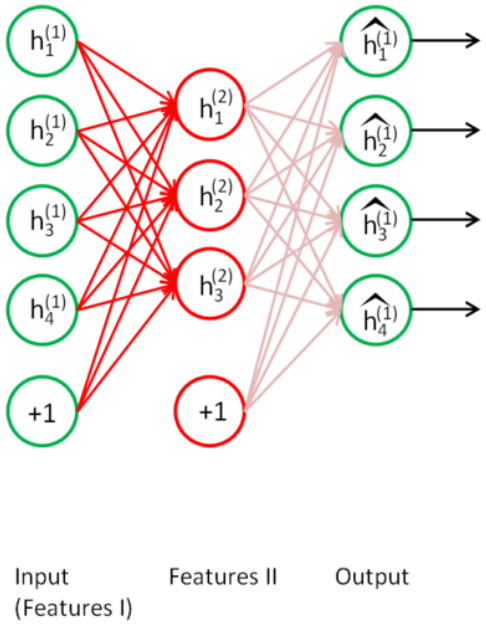
2、Autoencoder的Encoder和Decoder不能被赋予过大的容量，其能力不能太强，否则，他们可能将训练样本完全记忆，产生过拟合。

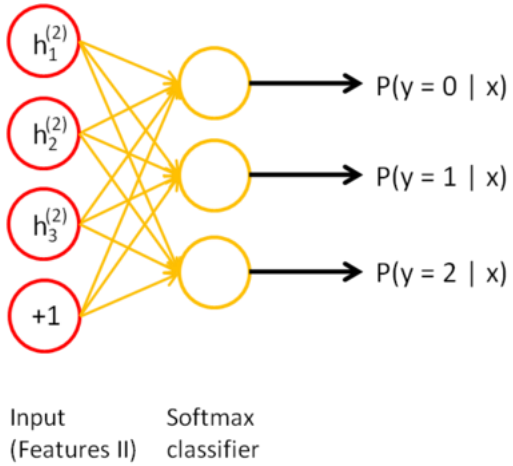
因此对隐层单元数增加限制：

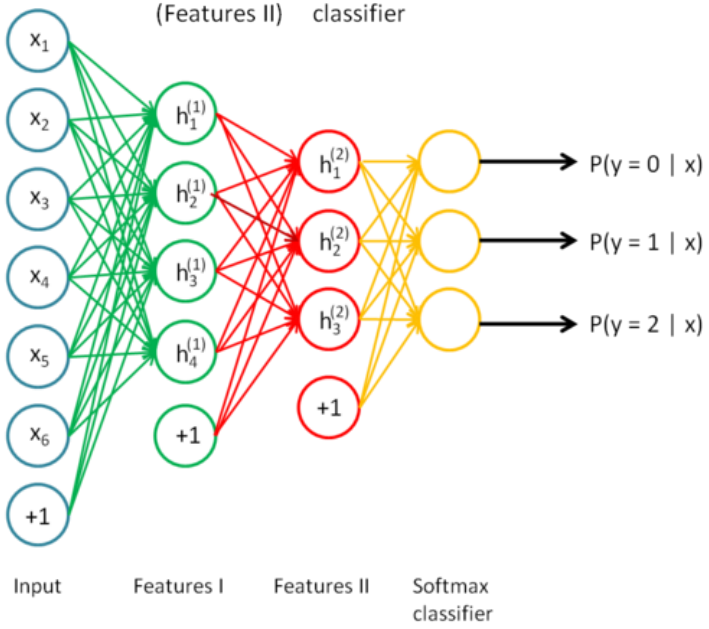
→ 欠完备自编码器

栈式自编码器

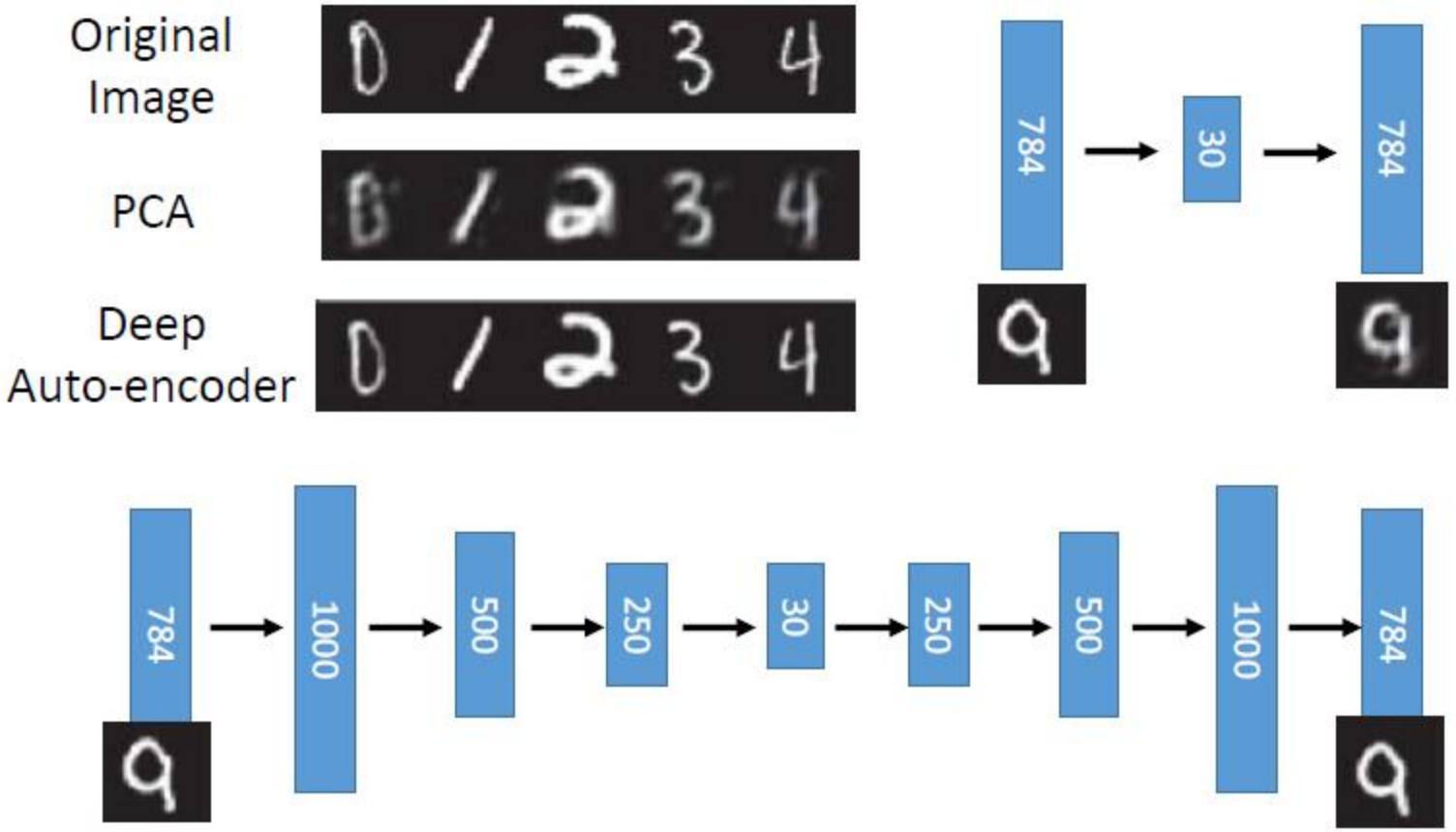

Input
Features I
Output


Input
(Features I)
Features II
Output

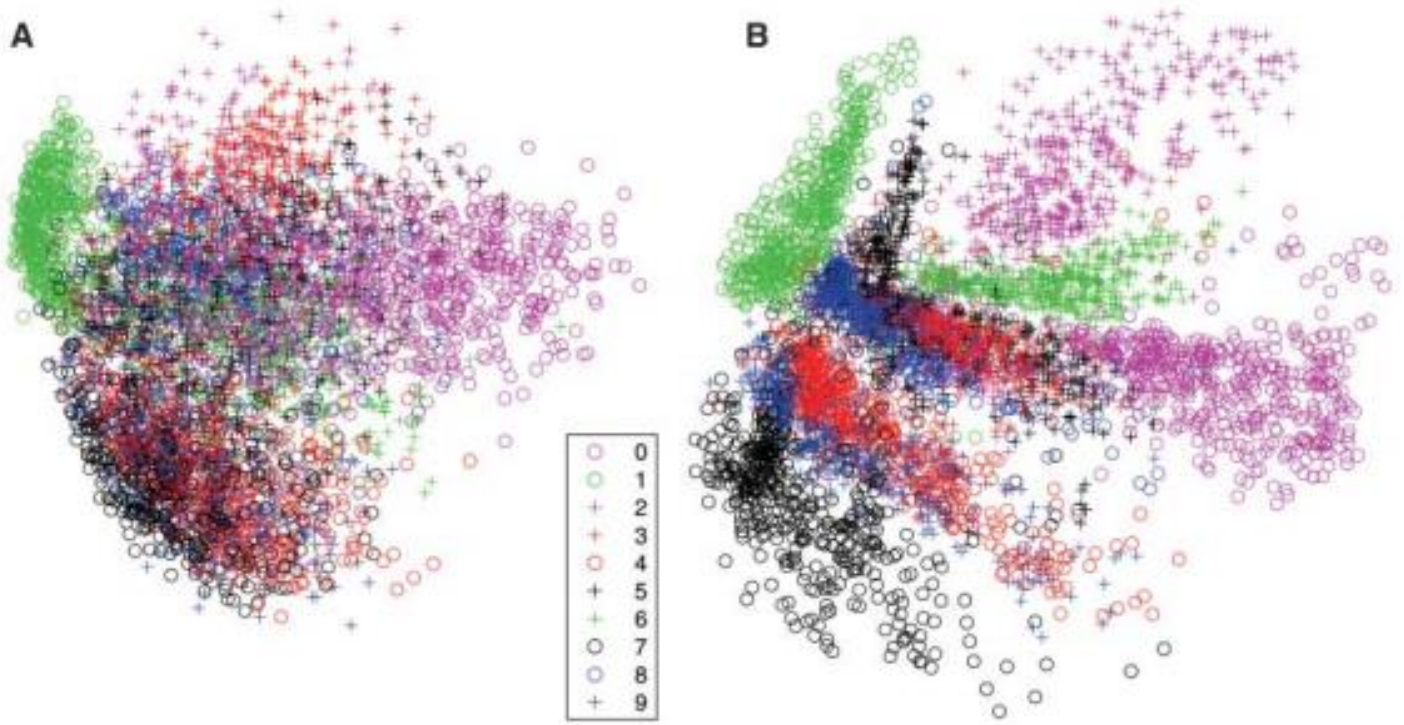

P(y = 0 | x)
P(y = 1 | x)
P(y = 2 | x)
Input
(Features II)
Softmax
classifier


P(y = 0 | x)
P(y = 1 | x)
P(y = 2 | x)
Input
Features I
Features II
Softmax
classifier

栈式自编码器


Original Image
PCA
Deep Auto-encoder
784
30
784
784
1000
500
250
30
250
500
1000
784

栈式自编码器

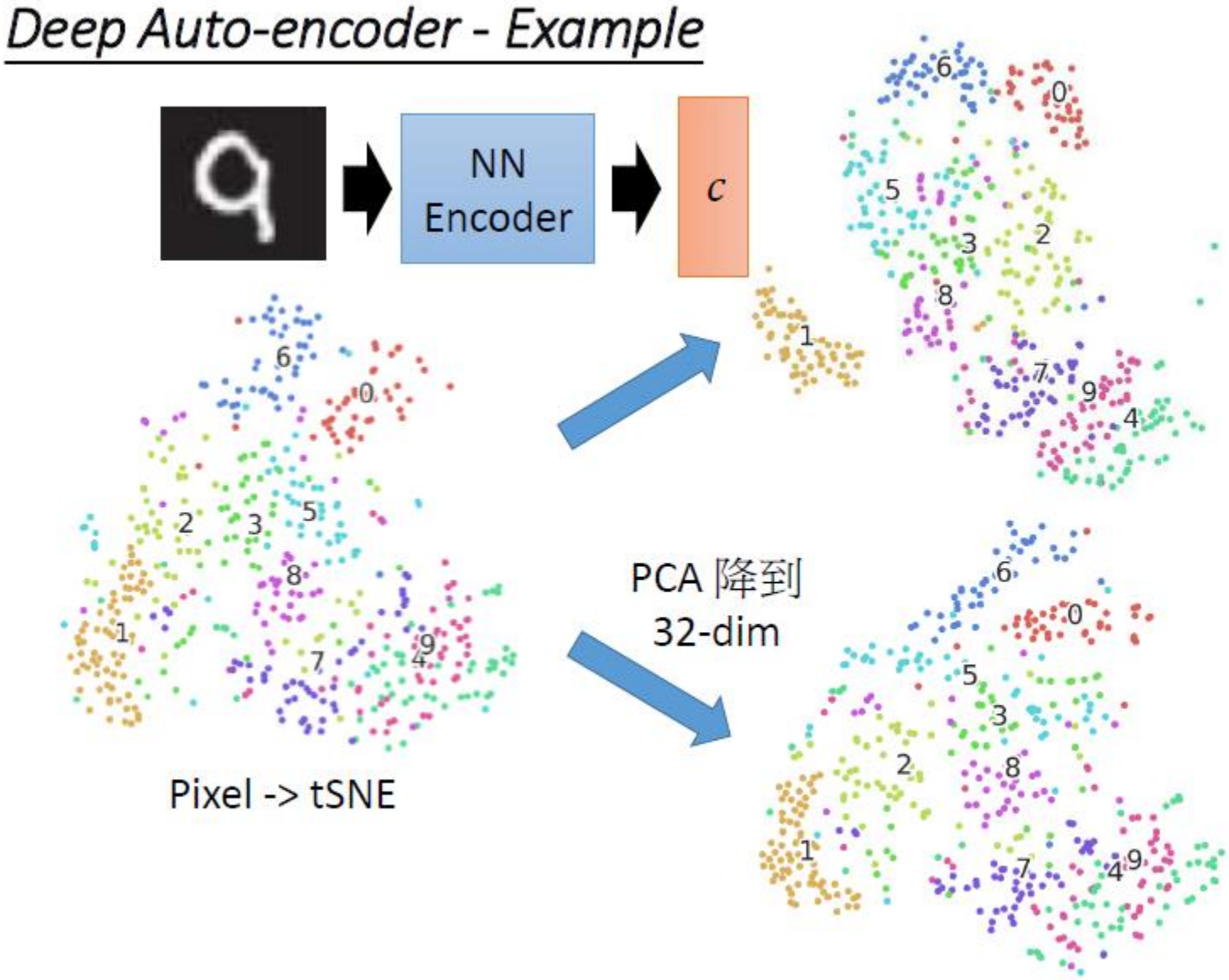


PCA: 784→2

SAE: 784→1000→500→250→2

Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. science, 2006, 313(5786): 504-507.

栈式自编码器


Deep Auto-encoder - Example
NN Encoder
c
Pixel -> tSNE
PCA 降到 32-dim

栈式自编码的特点

1、增加隐层可以学到更复杂的编码，每一层可以学习到不同的信息维度。

2、若层数太深，encoder过于强大，可以将学习将输入映射为任意数（然后decoder学习其逆映射）。这一编码器可以很好的重建数据，但它并没有在这一过程中学到有用的数据表示。

Stack AE和DBN异同点

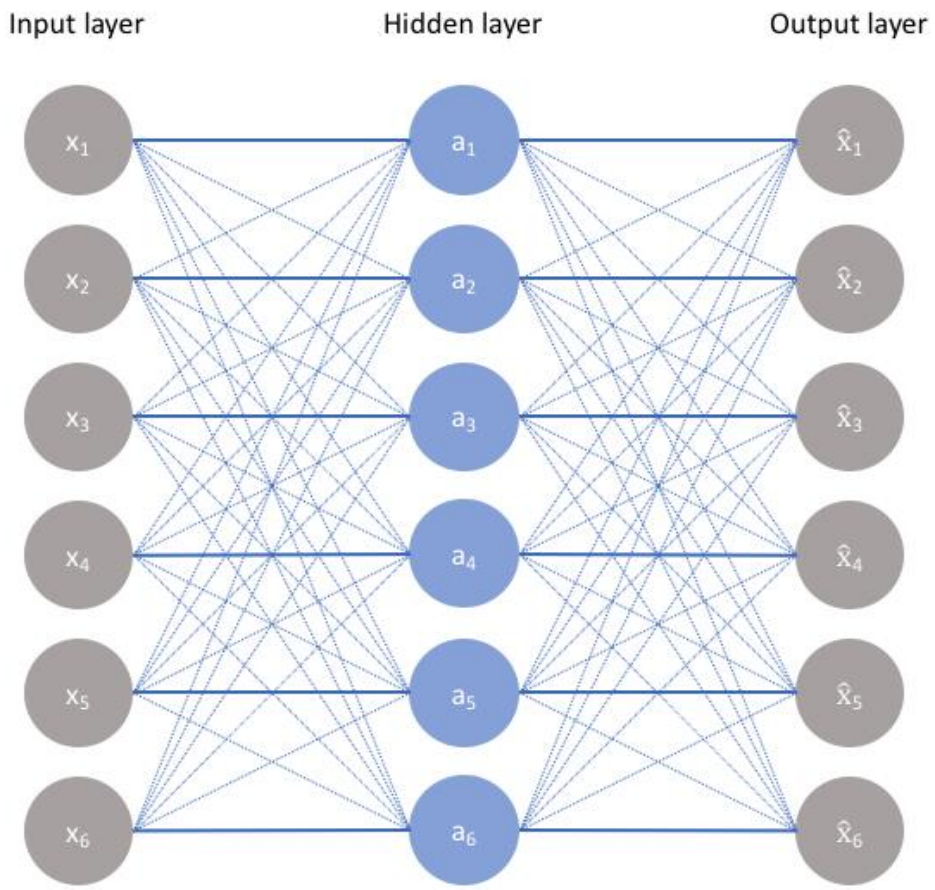
自编码器→栈式自编码器

受限玻尔兹曼机→深度信念网络

相同点：逐层训练

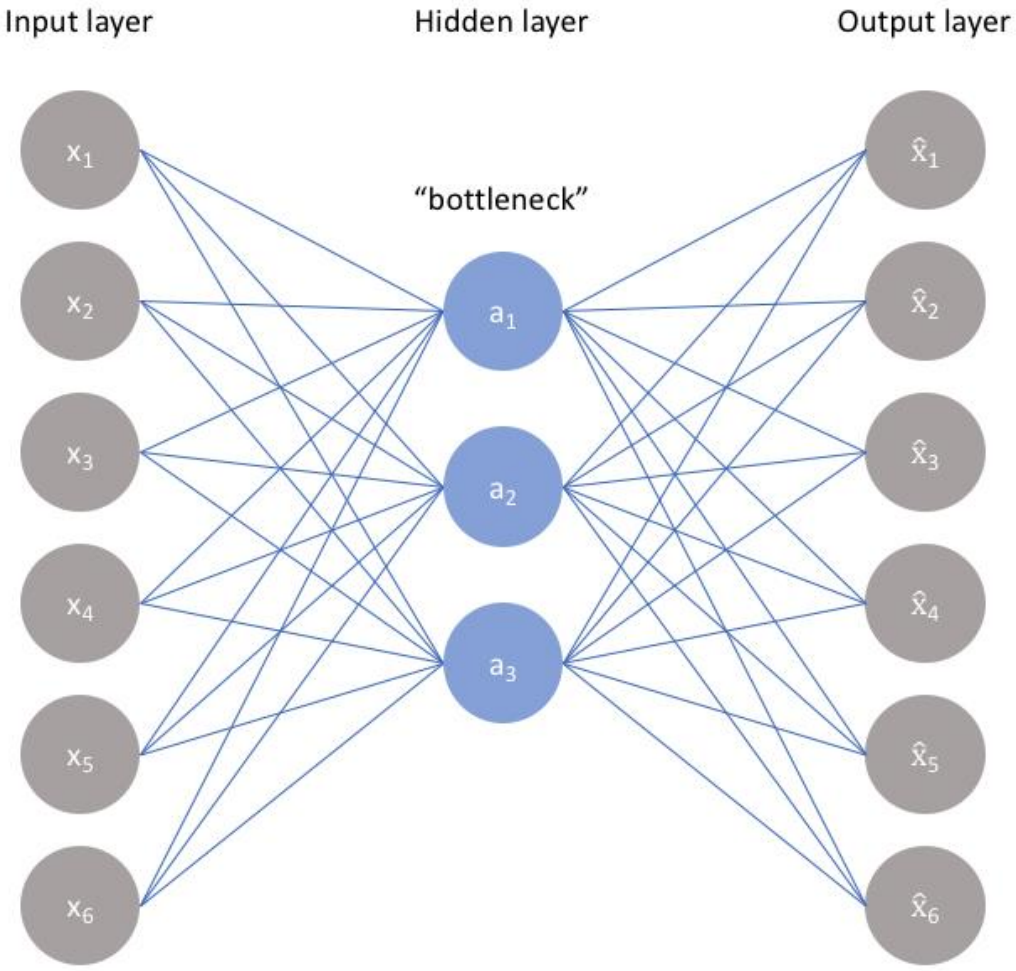
不同点：训练方法不同。

欠完备自编码器



当隐层单元数大于等于输入维度时，网络可能发生完全记忆。

欠完备自编码器



限制隐层的维度一定要比输入维度小

欠完备自编码器特点

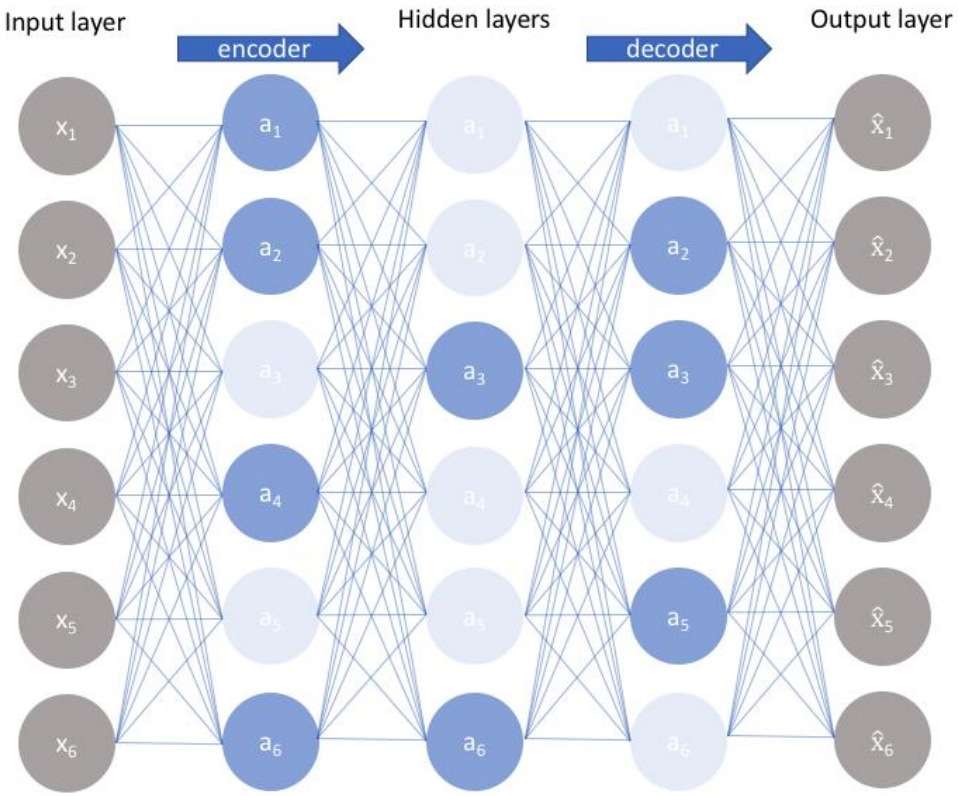
1、防止过拟合。因编码维数小于输入维数，因此可以学习数据分布最显著的特征。

2、若中间隐层单元数特别少，则信息量太少重构过程会比较困难。

因此提出将稀疏正则化引入其中

→ 稀疏自编码器

稀疏自编码器



$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) + \Omega(\boldsymbol{h})$$

1、未限制网络接收数据的能力，即不限制隐层的单元数。

2、稀疏性限制：当神经元的输出接近于1的时候认为被激活，输出接近于0的时候认为被抑制，那么使得大部分神经元被抑制的限制。

稀疏自编码器

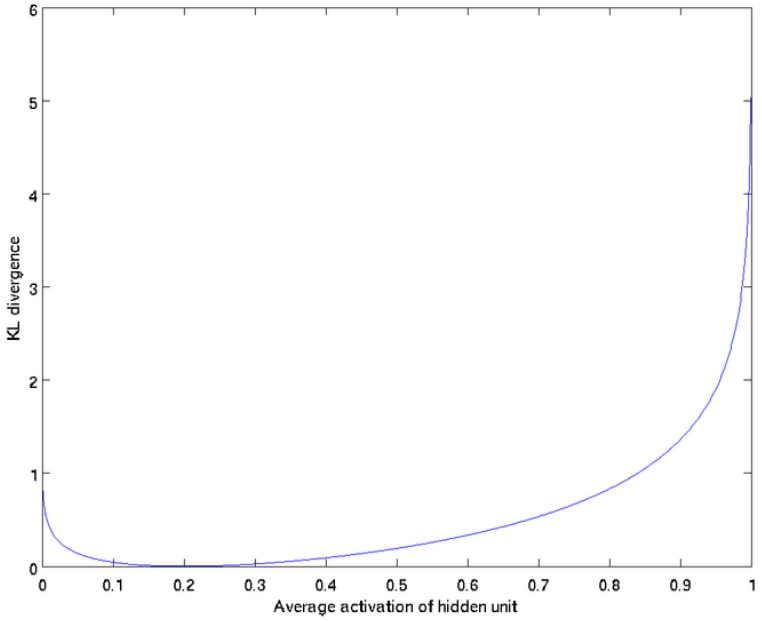
1 假设用$a_j^{(2)}(x)$ 来表示在给定输入x的情况下，自编码网络隐层神经元j的激活度。

2 神经元j在所有训练集上的平均活跃度：$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}(x^{(i)})\right]$

3 KL散度 $\sum_{j=1}^{s_2}\rho\log\frac{\rho}{\hat{\rho}_j}+(1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}.$ ρ 表示稀疏性参数

(一个以ρ为均值和一个以$\hat{\rho}_j$为均值的两个伯努利随机变量的相对熵)

4 损失函数$J_{\text{sparse}}(W,b) = J(W,b)+\beta\sum_{j=1}^{s_2}\text{KL}(\rho\|\hat{\rho}_j),$



稀疏自编码器

$$J_{\text{sparse}}(W,b) = J(W,b) + \beta \sum_{j=1}^{s_2} \text{KL}(\rho \| \hat{\rho}_j),$$

$$\begin{aligned}
\frac{\partial \sum_{j=1}^{s_2} KL(\rho||\hat{\rho}_j)}{\partial z_i^{(2)}} &= \frac{\partial KL(\rho||\hat{\rho}_i)}{\partial z_i^{(2)}} \\
&= \frac{\partial KL(\rho||\hat{\rho}_i)}{\partial \hat{\rho}_i} \cdot \frac{\partial \hat{\rho}_i}{\partial z_i^{(2)}} \\
&= \frac{\partial (\rho \log \frac{\rho}{\hat{\rho}_i} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_i})}{\partial \hat{\rho}_i} \cdot \frac{\partial \hat{\rho}_i}{\partial z_i^{(2)}} \\
&= (-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i}) \cdot f'(z_i^{(2)})
\end{aligned}$$

$$\delta_i^{(2)} = \left(\left(\sum_{j=1}^{s_2} W_{ji}^{(2)} \delta_j^{(3)} \right) + \beta \left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1-\hat{\rho}_i} \right) \right) f'(z_i^{(2)}).$$

稀疏自编码器

Mini-Batch的情况：

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}(x^{(i)})\right]$$

平均激活度是根据所有样本计算出来的，使用小批量时计算效率很低，因此，使用下列计算方法：

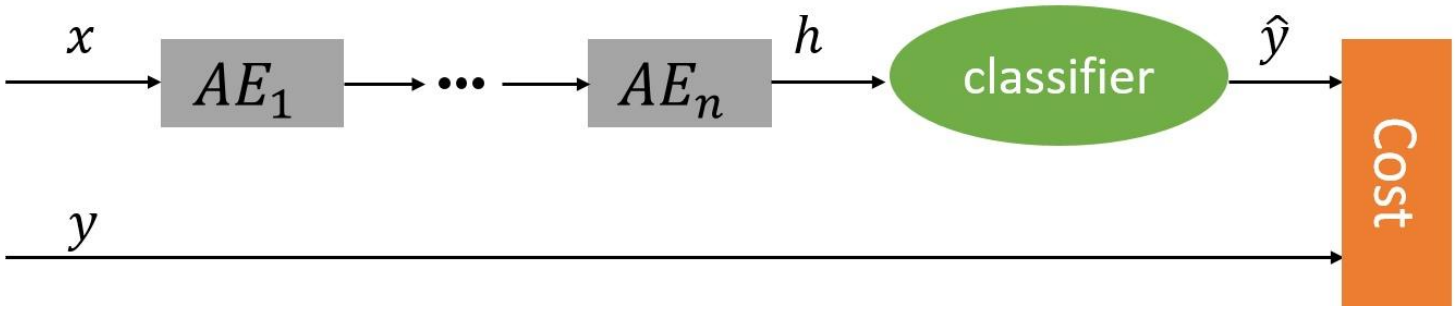
$$\tilde{\rho}_j^t = \lambda\tilde{\rho}_j^{t-1} + (1-\lambda)\tilde{\rho}_j^t$$

$\tilde{\rho}_j^t$ 是t时刻Min-Batch的平均激活度

稀疏自编码器

1、稀疏自编码器一般用来学习特征，用于分类等。

2、允许学习容量很高，同时防止在编码器和解码器学习一个无用的恒等函数。

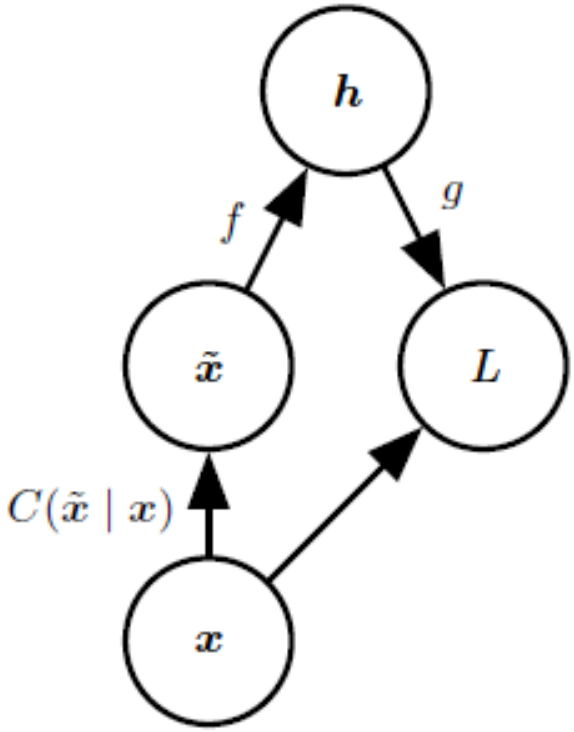


稀疏限制和L1/L2正则化的关系：

1、稀疏限制是对激活函数的结果增加限制，使得尽量多的激活函数的结果为0（如果激活函数是tanh，则为-1）

3、L2/L1是对参数增加限制，使得尽可能多的参数为0。

去噪自编码器



$C(\tilde{x}|x)$

1、高斯噪声

2、随机舍去

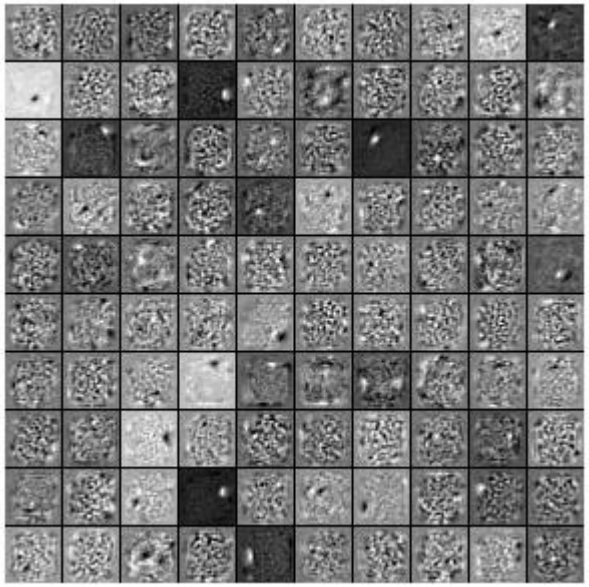
去噪自编码器

Dataset	SVM_{rbf}	SVM_{poly}	DBN-1	SAA-3	DBN-3	SdA-3 (ν)
basic	**3.03±0.15**	3.69±0.17	3.94±0.17	3.46±0.16	3.11±0.15	**2.80±0.14** (10%)
rot	11.11±0.28	15.42±0.32	14.69±0.31	**10.30±0.27**	**10.30±0.27**	**10.29±0.27** (10%)
bg-rand	14.58±0.31	16.62±0.33	9.80±0.26	11.28±0.28	**6.73±0.22**	10.38±0.27 (40%)
bg-img	22.61±0.37	24.01±0.37	**16.15±0.32**	23.00±0.37	**16.31±0.32**	**16.68±0.33** (25%)
rot-bg-img	55.18±0.44	56.41±0.43	52.21±0.44	51.93±0.44	47.39±0.44	**44.49±0.44** (25%)
rect	**2.15±0.13**	**2.15±0.13**	4.71±0.19	2.41±0.13	2.60±0.14	**1.99±0.12** (10%)
rect-img	24.04±0.37	24.05±0.37	23.69±0.37	24.05±0.37	22.50±0.37	**21.59±0.36** (25%)
convex	19.13±0.34	19.82±0.35	19.92±0.35	**18.41±0.34**	**18.63±0.34**	**19.06±0.34** (10%)

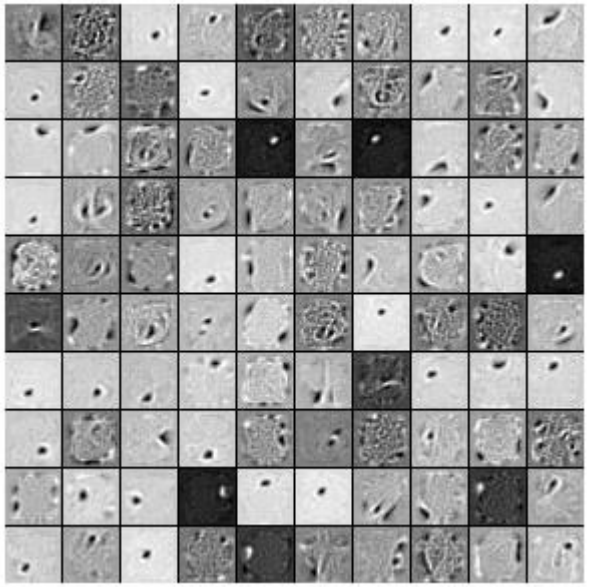
SVM_rbf是使用高斯核的SVM
SVM_poly是使用多项式的SVM
DBN-1是使用1层隐藏单元的深度信念网络
DBN-3是使用3层隐藏单元的深度信念网络
SAA-3是使用栈式自编码器初始化之后的3层深度网络
Sda-3是3层的栈式去噪自编码器

Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th international conference on Machine learning. ACM, 2008: 1096-1103.

去噪自编码器



Autoencoder



Denoising-Autoencoder

DAE学习到的特征更具代表性

去噪自编码器

1、允许学习容量很高，同时防止在编码器和解码器学习一个无用的恒等函数。

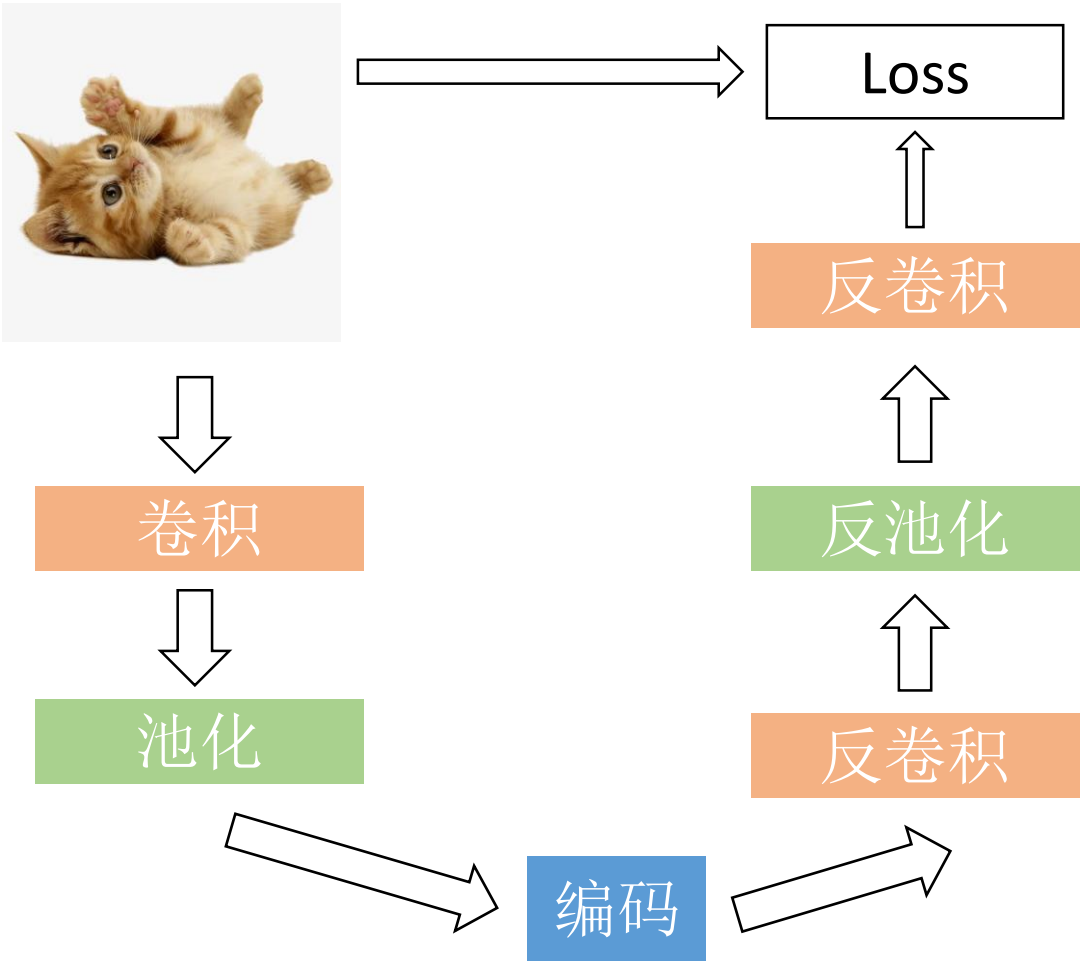
2、经过了加入噪声并进行降噪的训练过程，能够强迫网络学习到更加鲁棒的不变性特征，获得输入的更有效的表达。

去噪自编码器和Droupout对比

1、若添加噪声种类为随机舍去，则去噪自编码器相当于对输入去掉一部分内容，而Droupout是去掉隐层的一部分单元。

2、Dropout在分层预训练权值的过程中是不参与的，只是后面的微调部分会加入；而去噪自编码器是在每层预训练的过程中作为输入层被引入，在进行微调时不参与

卷积自编码器



循环自编码器

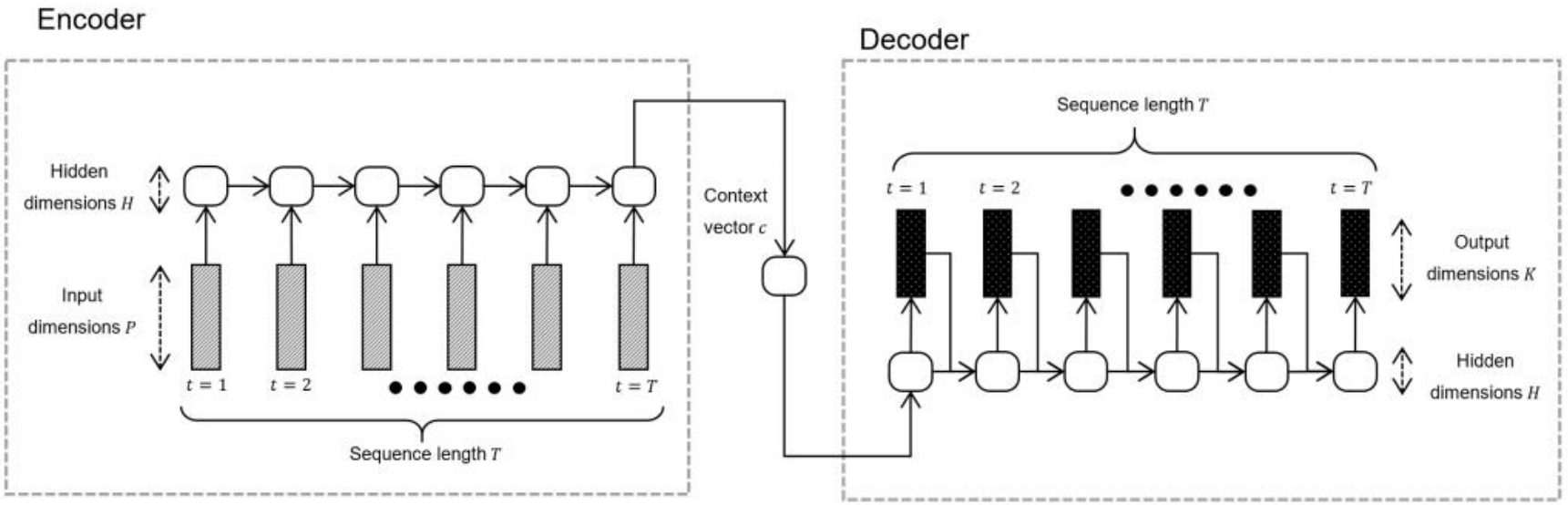


Figure 1: Recurrent auto-encoder model. Both the encoder and decoder are made up of multilayered RNN. Arrows indicate the direction of information flow.