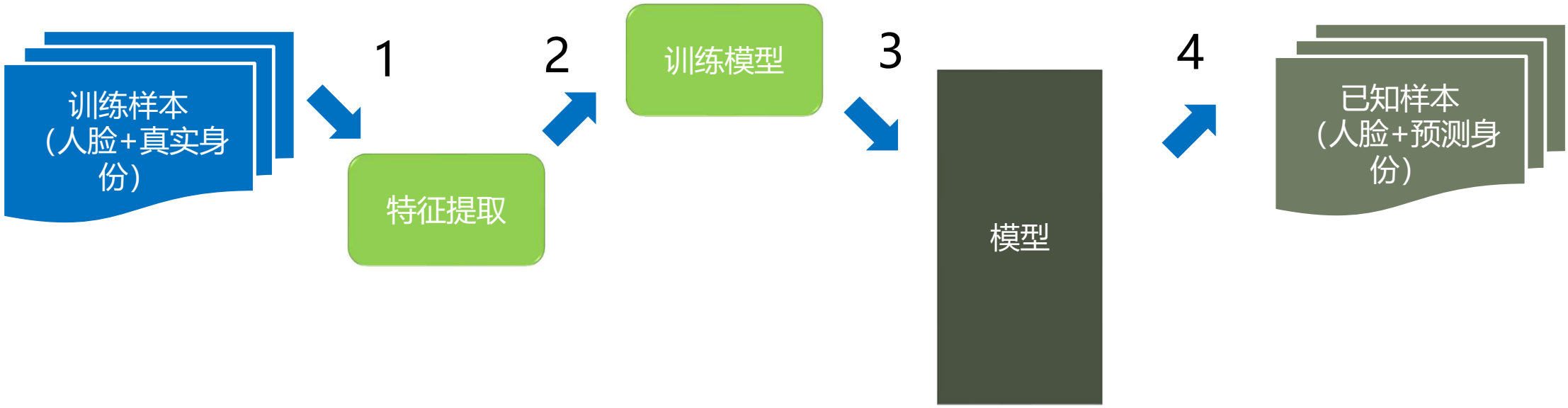
机器学习—SVM，深度自编码器

有监督学习

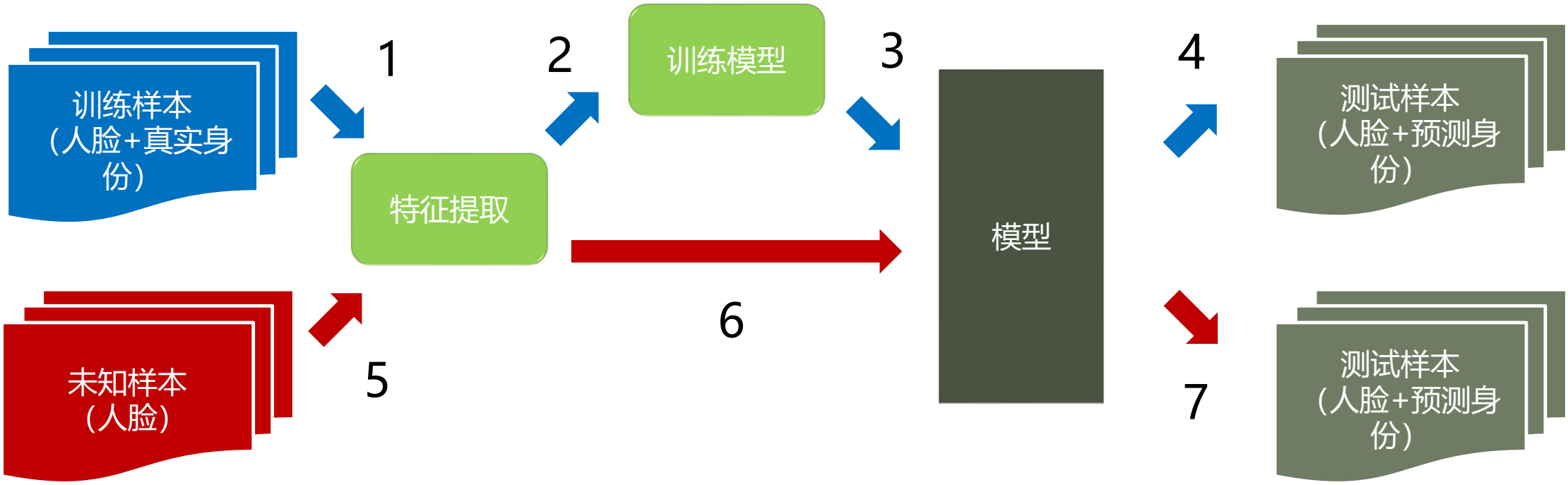
最简单也最普遍的一类机器学习算法就是分类（classification）。对于分类，输入的训练数据有特征（feature），有标签（label，比如1、2、3，比如你我他）。

所谓的学习，其本质就是找到特征和标签间的关系（mapping）。这样当有特征而无标签的未知数据输入时，我们就可以通过已有的关系得到未知数据标签。

学习过程



测试过程


训练样本
（人脸+真实身份）
1
特征提取
2
训练模型
3
模型
4
测试样本
（人脸+预测身份）
未知样本
（人脸）
5
6
7
测试样本
（人脸+预测身份）

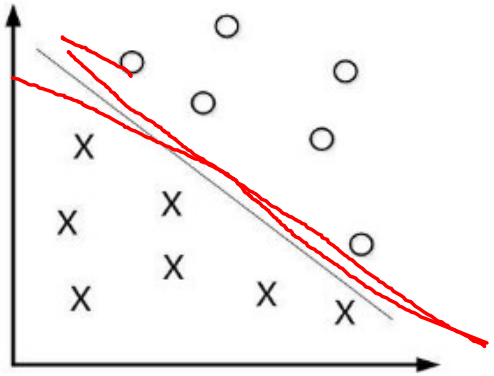
SVM基础

分类问题的数学表示

1、画一条线，把是“他”的人脸和不是“他”的人脸分开

- “他”，标签y为1；不是“他”，标签y为-1

2、数学：f(x) = w*x + b，x是人脸特征的向量，w和b为所求参数，f(x)>0，则x是“他”；f(x)<0则不是；f(x)=0是关键的分界面！



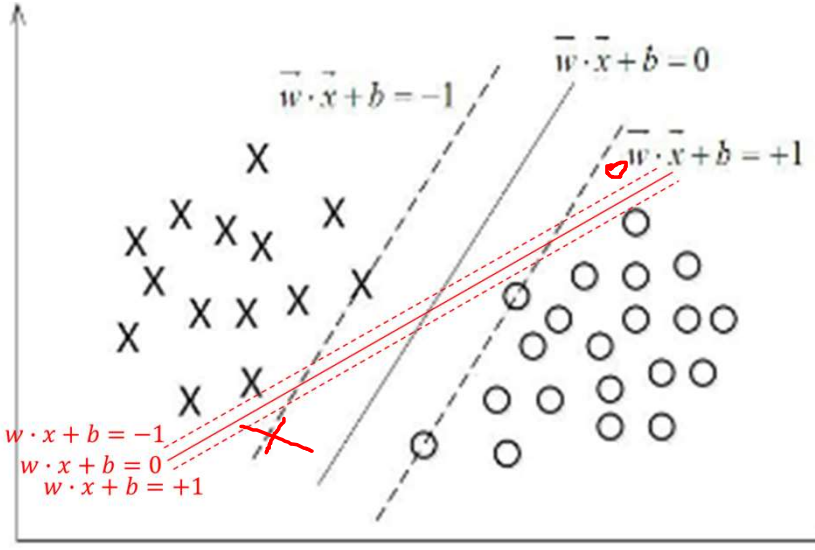
支持向量机
SVM，Support Vector Machine

3、直线f(x)的画法很多。可以画在“X”和“O”之间的任意地方，显然画在中心最合适！

4、支持向量SV，定义：两类样本点中离f(x)=0最近的样本点（样本边界），f(SV) = w*SV + b = t 或者-t；同时除以t得到f(SV) = 1或者-1

所以一般直接定义f(SV) = 1或者-1

SV确定了，f(x)就确定了！！



SVM：几何解释

5、样本边界距离越大，样本之间的区分度越大。

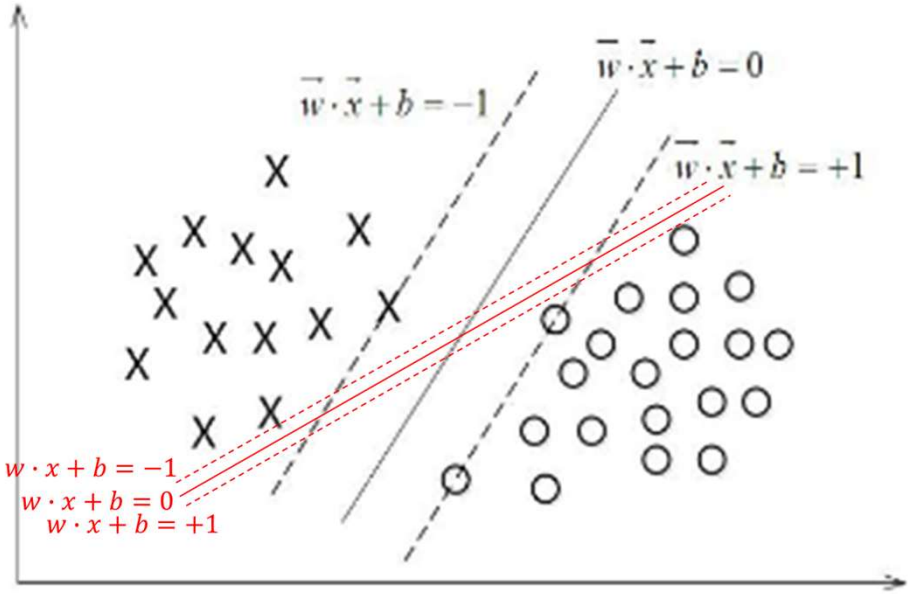
边界的距离是多少？

正样本的边界： $\boldsymbol{w^T x_+ + b = 1}$

负样本的边界： $\boldsymbol{w^T x_- + b = -1}$

两个边界的距离，等于正样本边界上的点 $\boldsymbol{x_+^*}$ **（**$\boldsymbol{w^T x_+^* + b = 1}$**）到负样本边界（**$\boldsymbol{w^T x_- + b = -1}$**）的距离：**

$$\frac{|w^T x_+^* + b + 1|}{\|w\|} = \frac{2}{\|w\|}。$$

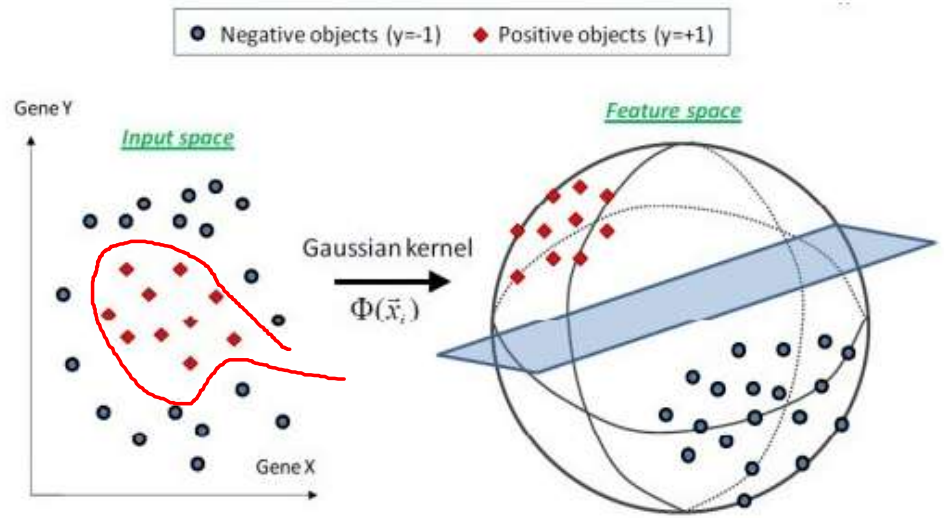


注：$w^T x + b = w_1 * x_1 + w_2 * x_2 + \cdots + w_n * x_n + b = \mathrm{w}^{\mathrm{T}} x + b$

SVM

6、最大化这个距离$\frac{1}{\|w\|}$，等价于最小化这个距离的倒数

$$\min \frac{1}{2}\|w\|^2 \quad s.t., y_i(w^T x_i + b) \geq 1, i = 1, \ldots, n$$



7、线性不可分?

通过核函数映射到高维空间;

given x_i, x_j

$k(x_i, x_j) = similarity(x_i, x_j)$

$$= \exp(-\frac{|x_i - x_j|^2}{2\sigma^2})$$

$$gamma = \frac{1}{2 \cdot \sigma^2}$$

SVM：outlier

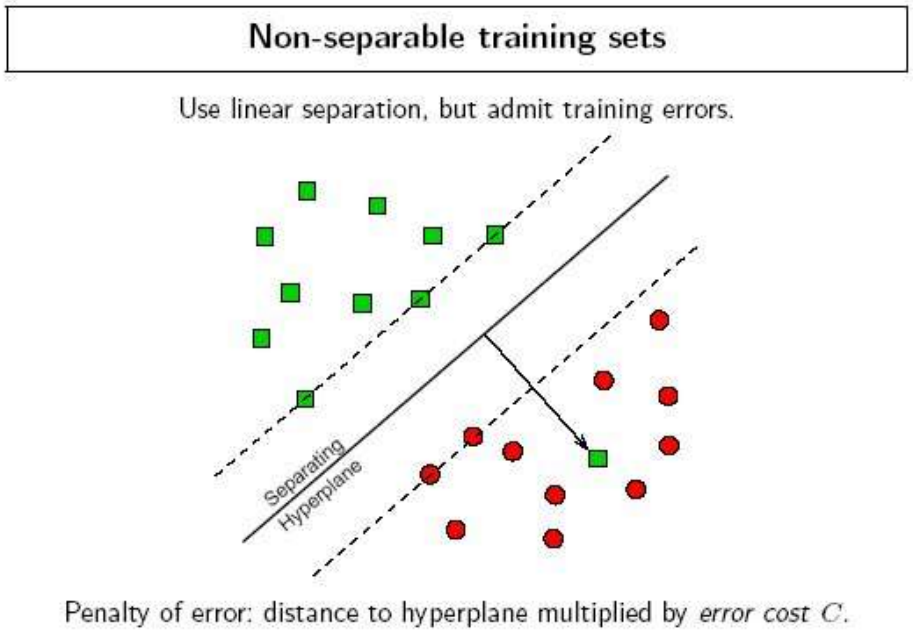
outlier：有些人脸长得确实不像“他”，实在不可分

处理办法：

模型允许错分几个；

用一个参数C来控制错分的样本到分类面的距离之和：

$sum(error_i) < f(C)$



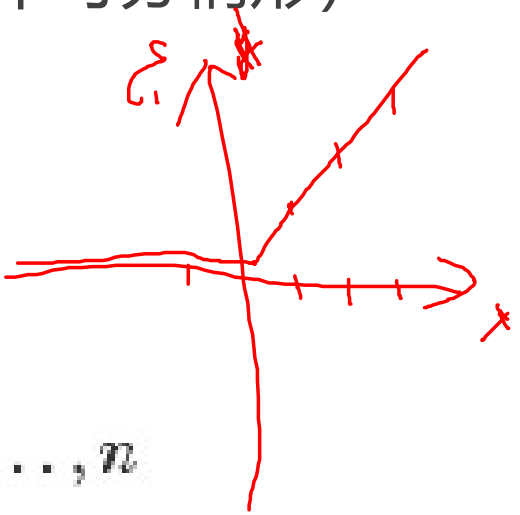
SVM目标函数-原始域问题和对偶问题

原始问题：C-SVC，软间隔分类器（针对线性不可分情形）

二次凸优化问题（在ξ_i的0点处不平滑）

分类器复杂度惩罚

训练误差(hinge loss)



$$\begin{cases} \min_{\vec{w},b}\left(\frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}\xi_i\right) \\ s.t. \quad y_i(\vec{w}^T\overrightarrow{x_i} + b) \geq 1 - \xi_i \quad ,i = 1,2,\ldots,n \\ \quad \xi_i \geq 0 \end{cases}$$

拉格朗日乘子:第一步：将不等式约束转化成标准形式：

$$y_i(\overrightarrow{w}^T\overrightarrow{x_i} + b) - 1 + \xi_i \geq 0$$

第二步：在不等式约束标准形式左右两边各乘一个拉格朗日乘子　第三步：把所有约束加到目标函数中

$$\alpha_i(y_i(\overrightarrow{w}^T\overrightarrow{x_i} + b) - 1 + \xi_i) \geq 0 \qquad \mu_i\xi_i \geq 0$$

SVM对偶问题推导

$$L=\frac{1}{2}\|w\|^2+C\sum\xi_i-\sum_i\alpha_i(y_i(\vec{w}^T\vec{x}_i+b)-1+\xi_i)-\sum_i\mu_i\xi_i$$ 其中$\alpha_i,\mu_i\geq 0$

$$\min_{w,b}\max_{\alpha_i\geq 0}L(w,b,\alpha_i)=\max_{\alpha_i\geq 0}\min_{w,b}L(w,b,\alpha_i)$$

导数=0，得到w和b：

$$\frac{\partial L}{\partial w}=0$$

$$\frac{\partial L}{\partial b}=0$$

$$\frac{\partial L}{\partial \xi_i}=0$$

$$\begin{cases}\vec{w}=\sum_{i=1}^{n}\alpha_i y_i\vec{x_i}\\ \sum_{i=1}^{n}\alpha_i y_i=0\\ C=\alpha_i+\mu_i\end{cases}$$

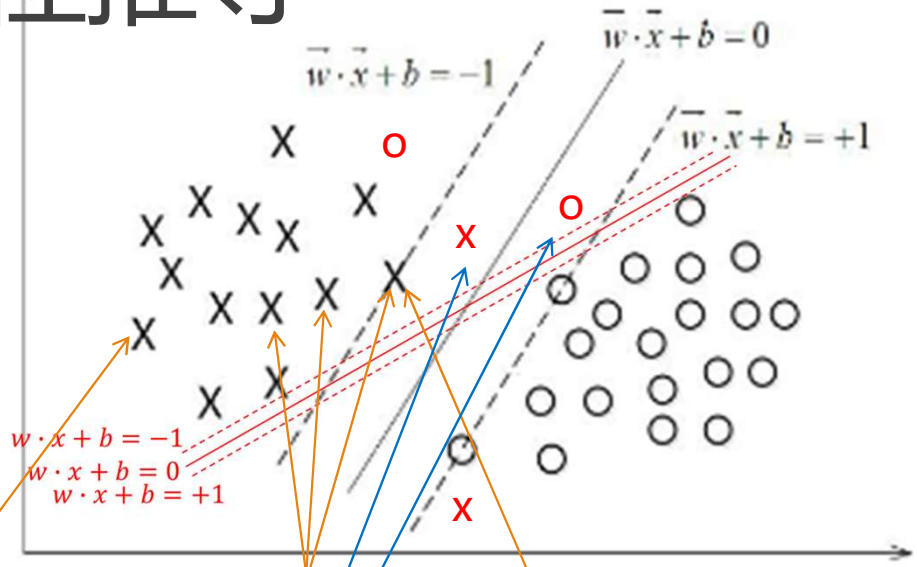
对偶形式的函数：

$$\max_{\alpha_i\geq 0}\min_{w,b}L(w,b,\alpha_i)=\max_{\alpha_i\geq 0}\left\{\sum_{i=1}^{l}\alpha_i-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)\right\}$$

原始-对偶问题等价性推导

KKT条件：

$$\begin{cases} \alpha_i \geq 0, \mu_i \geq 0 \\ y_i(\vec{w}\vec{x}+b)-1+\xi_i \geq 0 \\ \alpha_i(y_i(\vec{w}\vec{x}+b)-1+\xi_i)=0 \\ \xi_i \geq 0, \mu_i\xi_i=0 \end{cases}$$



如果ξ_i=0，即该样本被正确划分即y(wx+b)>=1，则μ_i可为0或者不为0，则0<=α_i<=C

特别地，当y(wx+b)>1时，α_i=0; 当x处在边界之上, 即y(wx+b)=1时，0<=α_i<=C。

如果ξ_i不为0，即该样本被错分，y(wx+b)<1
则μ_i 必须为0，由于$\alpha_i+\mu_i$=C,即α_i=C

SVM对偶问题

平滑，有界二次凸优化问题

$$\begin{cases} \max_{\vec{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \overrightarrow{x_i}^T \overrightarrow{x_j} \\ s.t. \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \quad , i = 1, 2, \ldots, n \\ 0 \leq \alpha_i \leq C \end{cases}$$

$$\min \boldsymbol{\alpha}^T \mathbf{1} - \boldsymbol{\alpha}^T \mathbf{Q} \boldsymbol{\alpha}$$

$$\text{st.} \quad \boldsymbol{\alpha}^T \mathbf{y} = 0, \ 0 \leq \alpha_i \leq C$$

SVM核函数及核化判别函数

常用核函数

名称	表达式	参数
线性核	$\kappa(\vec{x_i},\vec{x_j})=\vec{x_i}^T\vec{x_j}$	
多项式核	$\kappa(\vec{x_i},\vec{x_j})=(\vec{x_i}^T\vec{x_j})^n$	$n\geq 1$为多项式的次数
高斯核(RBF)	$\kappa(\vec{x_i},\vec{x_j})=exp(-\frac{\|\vec{x_i}-\vec{x_j}\|^2}{2\sigma^2})$	$\sigma>0$为高斯核的带宽
拉普拉斯核	$\kappa(\vec{x_i},\vec{x_j})=exp(-\frac{\|x_i-x_j\|}{\sigma})$	$\sigma>0$
Sigmoid核	$\kappa(\vec{x_i},\vec{x_j})=tanh(\beta\vec{x_i}^T\vec{x_j}+\theta)$	thah为双曲正切函数

$$f(x)=w\cdot x+b$$
$$=\left(\sum_{i=1:n}\alpha_i y_i x_i\right)\cdot x+b$$
$$=\sum_{i=1:n}\alpha_i y_i\left(x_i^T\cdot x\right)+b$$
$$=\sum_{i=1:n}\alpha_i y_i k\left(x_i,x\right)+b$$

SVM求解

$$\begin{cases} \max_{\vec{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \vec{x_i}^T \vec{x_j} \\ s.t. \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \quad , i = 1, 2, \ldots, n \\ \quad 0 \le \alpha_i \le C \end{cases}$$

SMO：对偶域

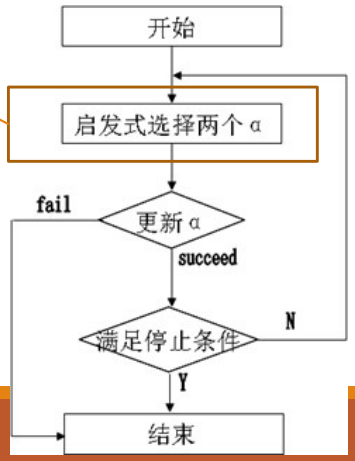
求解过程中，目标函数严格满足KKT条件

$$\alpha_i(y_i(<w,x_i>+b)-1+\xi_i)=0$$

$$\mu_i\xi_i=(\alpha_i-C)\xi_i=0$$

决定效率的关键之一

提升效率关键之二：kernel 缓存



1. Find α^1 as the initial feasible solution. Set $k = 1$.

2. If α^k is an optimal solution of (1), stop. Otherwise, find a *two-element* working set $B = \{i, j\} \subset \{1, \ldots, l\}$. Define $N \equiv \{1, \ldots, l\} \backslash B$ and α_B^k and α_N^k to be sub-vectors of α^k corresponding to B and N, respectively.

3. Solve the following sub-problem with the variable α_B:

$$\begin{aligned} \min_{\alpha_B} \quad & \frac{1}{2} \begin{bmatrix} \alpha_B^T & (\alpha_N^k)^T \end{bmatrix} \begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix} \begin{bmatrix} \alpha_B \\ \alpha_N^k \end{bmatrix} - \begin{bmatrix} \mathbf{e}_B^T & \mathbf{e}_N^T \end{bmatrix} \begin{bmatrix} \alpha_B \\ \alpha_N^k \end{bmatrix} \\ & = \frac{1}{2} \alpha_B^T Q_{BB} \alpha_B + (-\mathbf{e}_B + Q_{BN} \alpha_N^k)^T \alpha_B + \text{ constant} \\ & = \frac{1}{2} \begin{bmatrix} \alpha_i & \alpha_j \end{bmatrix} \begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ij} & Q_{jj} \end{bmatrix} \begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + (-\mathbf{e}_B + Q_{BN} \alpha_N^k)^T \begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + \text{ constant} \\ \text{subject to} \quad & 0 \le \alpha_i, \alpha_j \le C, \qquad (2) \\ & y_i \alpha_i + y_j \alpha_j = -\mathbf{y}_N^T \alpha_N^k, \end{aligned}$$

where $\begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix}$ is a permutation of the matrix Q.

4. Set α_B^{k+1} to be the optimal solution of (2) and $\alpha_N^{k+1} \equiv \alpha_N^k$. Set $k \leftarrow k + 1$ and goto Step 2.

SVM求解-次梯度方法(subgradient)

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{(\mathbf{x},y)\in S} \ell(\mathbf{w};(\mathbf{x},y)) \ ,$$

$$\ell(\mathbf{w};(\mathbf{x},y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\} \ ,$$

$$\nabla_t = \lambda \mathbf{w}_t - \mathbb{1}[y_{i_t}\langle \mathbf{w}_t, \mathbf{x}_{i_t}\rangle < 1]\, y_{i_t}\mathbf{x}_{i_t}$$

$$\mathbf{w}_{t+1} \leftarrow (1 - \frac{1}{t})\mathbf{w}_t + \eta_t \mathbb{1}[y_{i_t}\langle \mathbf{w}_t, \mathbf{x}_{i_t}\rangle < 1]\, y_{i_t}\mathbf{x}_{i_t}$$

INPUT: S, λ, T

INITIALIZE: Set $\mathbf{w}_1 = 0$

FOR $t = 1, 2, \ldots, T$

Choose $i_t \in \{1, \ldots, |S|\}$ uniformly at random.

Set $\eta_t = \frac{1}{\lambda t}$

If $y_{i_t}\langle \mathbf{w}_t, \mathbf{x}_{i_t}\rangle < 1$, then:

Set $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t\lambda)\mathbf{w}_t + \eta_t y_{i_t}\mathbf{x}_{i_t}$

Else (if $y_{i_t}\langle \mathbf{w}_t, \mathbf{x}_{i_t}\rangle \geq 1$):

Set $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t\lambda)\mathbf{w}_t$

[Optional: $\mathbf{w}_{t+1} \leftarrow \min\left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{t+1}\|}\right\} \mathbf{w}_{t+1}$]

OUTPUT: $\mathbf{w}_{T+1}$

SVM求解-次梯度方法(subgradient)

mini-batch 扩展：用多个训练数据对来训练

INPUT: S, λ, T, k

INITIALIZE: Set $\mathbf{w}_1 = 0$

FOR $t = 1, 2, \ldots, T$

Choose $A_t \subseteq [m]$, where $|A_t| = k$, uniformly at random

Set $A_t^+ = \{i \in A_t : y_i \langle \mathbf{w}_t, \mathbf{x}_i \rangle < 1\}$

Set $\eta_t = \frac{1}{\lambda t}$

Set $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t \lambda)\mathbf{w}_t + \frac{\eta_t}{k} \sum_{i \in A_t^+} y_i \mathbf{x}_i$

[Optional: $\mathbf{w}_{t+1} \leftarrow \min\left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{t+1}\|}\right\} \mathbf{w}_{t+1}$]

OUTPUT: $\mathbf{w}_{T+1}$

复杂函数的次梯度计算方法

Loss function	Subgradient
$\ell(z, y_i) = \max\{0, 1 - y_i z\}$	$\mathbf{v}_t = \begin{cases} -y_i \mathbf{x}_i & \text{if } y_i z < 1 \\ \mathbf{0} & \text{otherwise} \end{cases}$
$\ell(z, y_i) = \log(1 + e^{-y_i z})$	$\mathbf{v}_t = -\frac{y_i}{1+e^{y_i z}} \mathbf{x}_i$
$\ell(z, y_i) = \max\{0, \lvert y_i - z \rvert - \epsilon\}$	$\mathbf{v}_t = \begin{cases} \mathbf{x}_i & \text{if } z - y_i > \epsilon \\ -\mathbf{x}_i & \text{if } y_i - z > \epsilon \\ \mathbf{0} & \text{otherwise} \end{cases}$
$\ell(z, y_i) = \max_{y \in \mathcal{Y}} \delta(y, y_i) - z_{y_i} + z_y$	$\mathbf{v}_t = \phi(\mathbf{x}_i, \hat{y}) - \phi(\mathbf{x}_i, y_i)$ where $\hat{y} = \arg\max_y \delta(y, y_i) - z_{y_i} + z_y$
$\ell(z, y_i) = \log\left(1 + \sum_{r \neq y_i} e^{z_r - z_{y_i}}\right)$	$\mathbf{v}_t = \sum_r p_r \phi(\mathbf{x}_i, r) - \phi(\mathbf{x}_i, y_i)$ where $p_r = e^{z_r} / \sum_j e^{z_j}$

如何测试已训练模型的好坏?

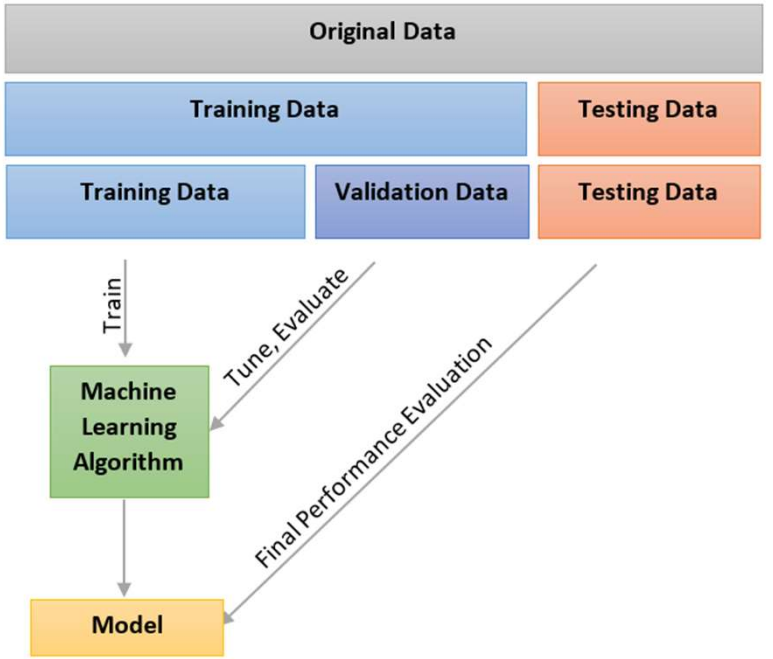
假设你已经训练了一个人脸识别模型，你怎么知道模型的正确率?

如何测试已训练模型的好坏?

假设你已经训练了一个人脸识别模型，你怎么知道模型的正确率?

数据集分割:

训练集和测试集



Matlab中的svm初步

model = svmtrain(X,Y,'Kernel_Function',@(X,Y)kfun_rbf(X,Y,gamma),'box constraint',c);

%只能做两类分类（A的脸和B的脸）。X为输入PCA降维后的矩阵，Y为X每一行对应的人脸标签（是A：1，是B：0），函数的结果是个model模型

class = svmclassify(model, X)

%根据已训练的model，对X进行分类，返回X的每一行对应的预测是否A的人脸（是：1，不是：0）

第一个函数：训练多分类器

function [multiSVMstruct] =multiSVMtrain(traindata,labels,gamma,C)

%输入：一组人脸的矩阵traindata，一共n行，m列，即n张图片

%labels表示traindata对应的标签

%gamma和C是前文提到svm的参数

%输出：multiSVMstruct：

%multiSVMstruct{i}{j}记录着第i个人和第j个人的分类模型（是第i个：1，是第j个：0）

multiSVMstruct{i}{j} = svmtrain([traindata中所有第i个人的图片,traindata中第j个人的图片], [1,1,1,1,1,0,0,0,0,0]，...)；

% [1,1,1,1,1,0,0,0,0,0]告诉svmtrain前五张图是第i个人，后五张是第j个

第二个函数：根据多分类器识别人脸

```
function [class] = multiSVMpredict(multiSVMstruct, testface, nclass)
%给定一组人脸的矩阵testface，一共n行，m列，即n张图片
%输出人脸的类别，class(i)表示第i张图片是第几个人
%nclass表示人数
%调用svmclassify
```

gamma和C的调参：超参数

一般调参的时候，尝试

gamm和C都在我给你们的文件夹中的face.m文件的开头

gamma = 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}

C = 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}

确定最优参数范围，再缩小范围微调一遍（比如范围中心×3^{-n} → 3^{n}），一般调节两遍就足够了

深度自编码器与特征学习

Age of Big **Unlabeled** Data

Opportunity: High-throughput data and computation

Particle Accelerators

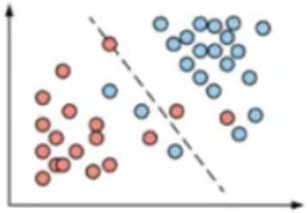
Materials Project

GenBank

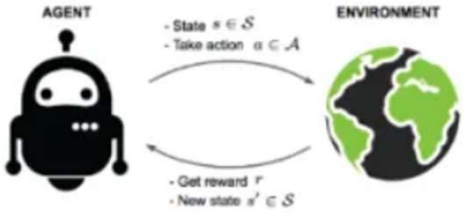
Challenge: Supervision signals such as labels are *expensive*

e.g., time, money, expertise, safety costs

Learning With Limited Supervision

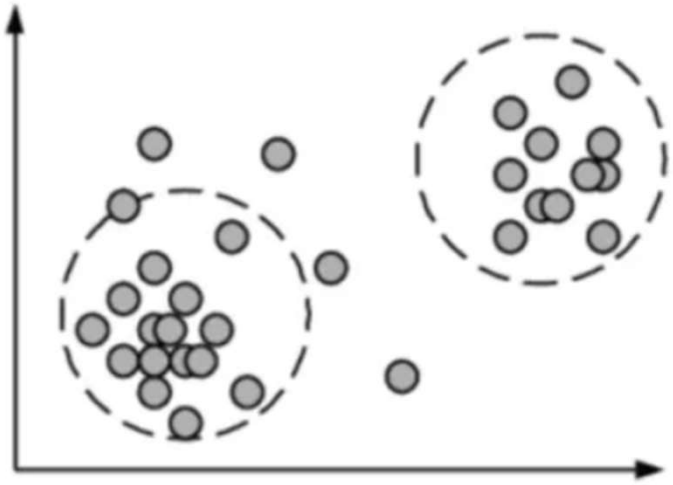


Classification
w/ active querying



Reinforcement Learning
w/ sparse rewards

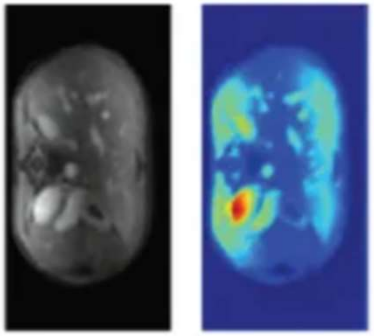
Can we learn with *no* supervision?



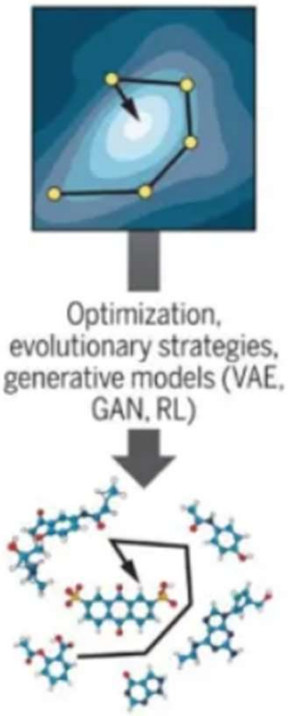
Recent Advances in Unsupervised Learning

Karras et al.

Huang et al.

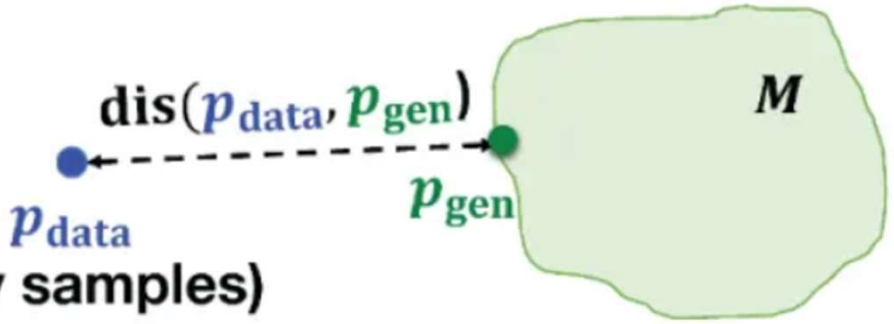


Mardani et al.



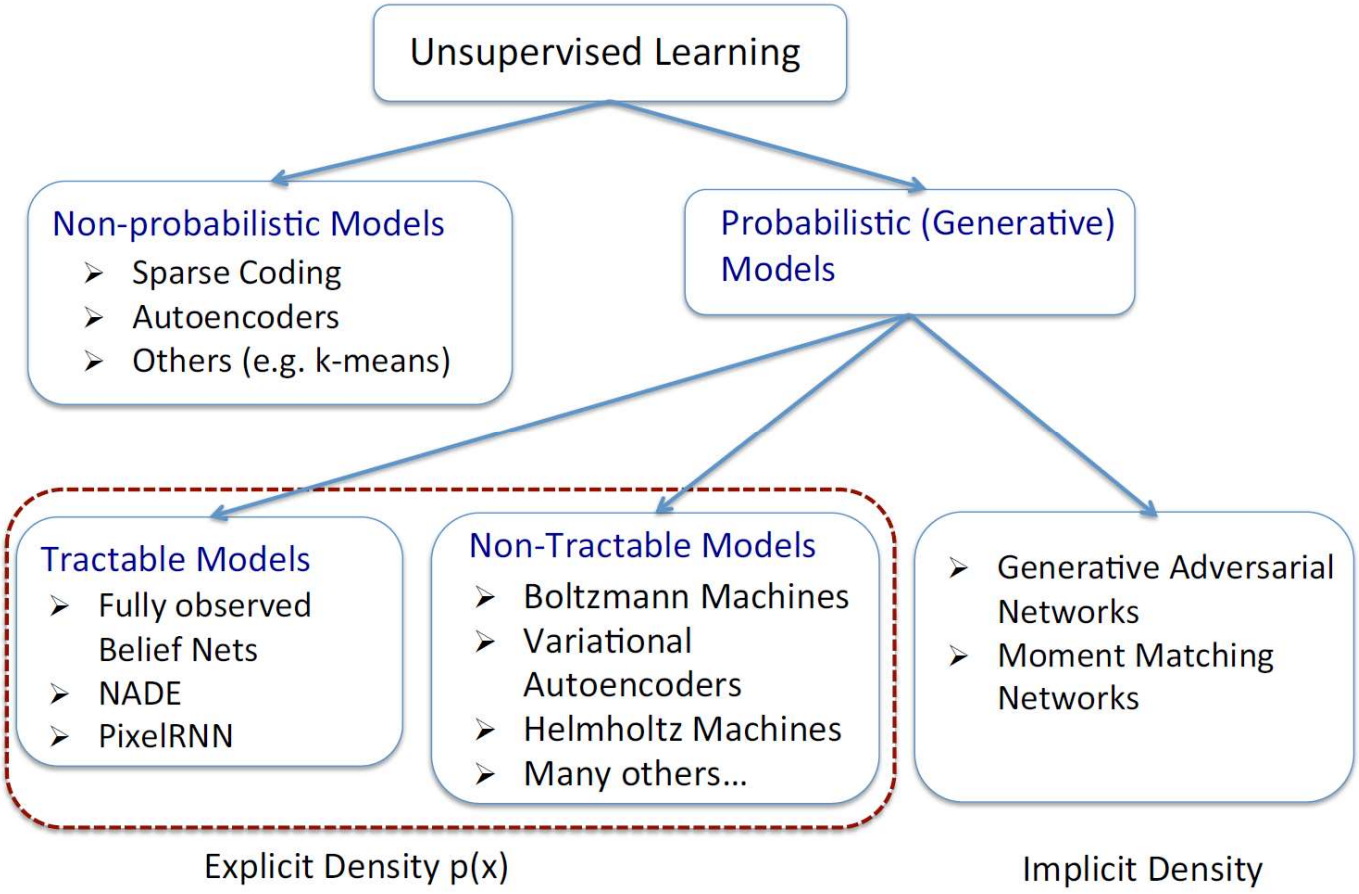
Sanchez-Lengelin et al.

Generative Models: Fit Data Distributions



$$\min_{p_{\mathbf{gen}} \in M} \mathbf{dis}(p_{\mathbf{data}}, p_{\mathbf{gen}})$$

稀疏编码


Unsupervised Learning
Non-probabilistic Models
Sparse Coding
Autoencoders
Others (e.g. k-means)
Probabilistic (Generative) Models
Tractable Models
Fully observed Belief Nets
NADE
PixelRNN
Non-Tractable Models
Boltzmann Machines
Variational Autoencoders
Helmholtz Machines
Many others…
Generative Adversarial Networks
Moment Matching Networks
Explicit Density p(x)
Implicit Density

Sparse Coding

• Sparse coding (Olshausen & Field, 1996). Originally developed to explain early visual processing in the brain (edge detection).

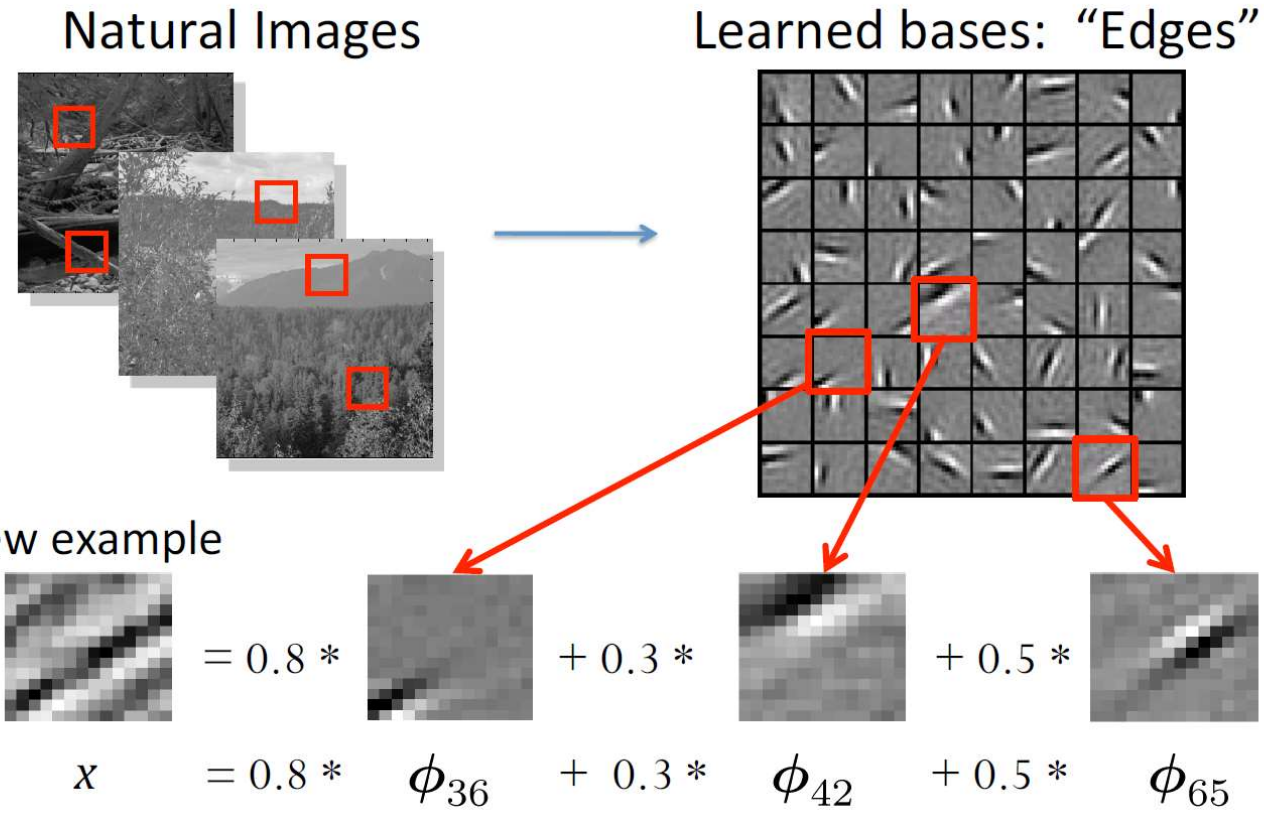
• Objective: Given a set of input data vectors $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, learn a dictionary of bases $\{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_K\}$, such that:

$$\mathbf{x}_n = \sum_{k=1}^{K} a_{nk} \boldsymbol{\phi}_k,$$

Sparse: mostly zeros

• Each data vector is represented as a sparse linear combination of bases.

Sparse Coding



[0, 0, … **0.8**, …, **0.3**, …, **0.5**, …] = coefficients (feature representation)

Slide Credit: Honglak Lee

Sparse Coding: Training

- Input image patches: $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N \in \mathbb{R}^D$
- Learn dictionary of bases: $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_K \in \mathbb{R}^D$

$$\min_{\mathbf{a},\boldsymbol{\phi}} \sum_{n=1}^{N} \left\| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk}\boldsymbol{\phi}_k \right\|_2^2 + \lambda \sum_{n=1}^{N}\sum_{k=1}^{K} |a_{nk}|$$

Reconstruction error Sparsity penalty

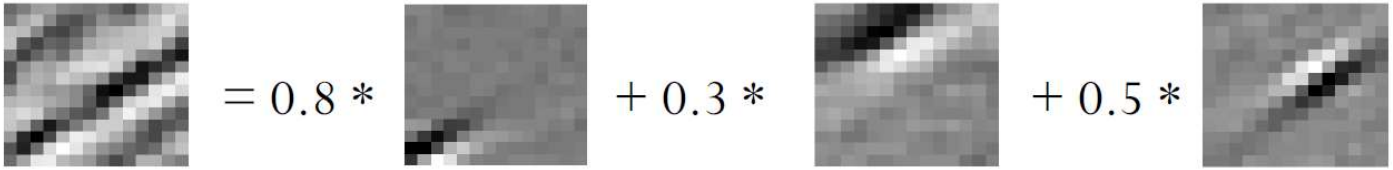
- Alternating Optimization:

1. Fix dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_K$ and solve for activations **a** (a standard Lasso problem).
2. Fix activations **a**, optimize the dictionary of bases (convex QP problem).

Sparse Coding: Testing Time

- Input: a new image patch x* , and K learned bases $\phi_1, \phi_2, ..., \phi_K$
- Output: sparse representation **a** of an image patch x*.

$$\min_{\mathbf{a}} \left\| \mathbf{x}^* - \sum_{k=1}^{K} a_k \phi_k \right\|_2^2 + \lambda \sum_{k=1}^{K} |a_k|$$

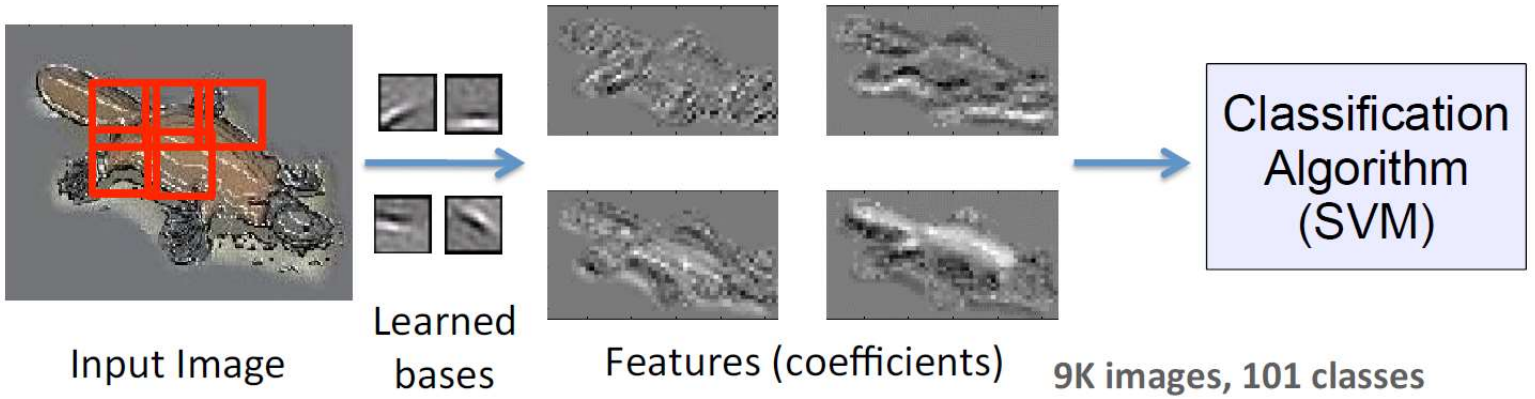


x^* $= 0.8 * \phi_{36} + 0.3 * \phi_{42} + 0.5 * \phi_{65}$

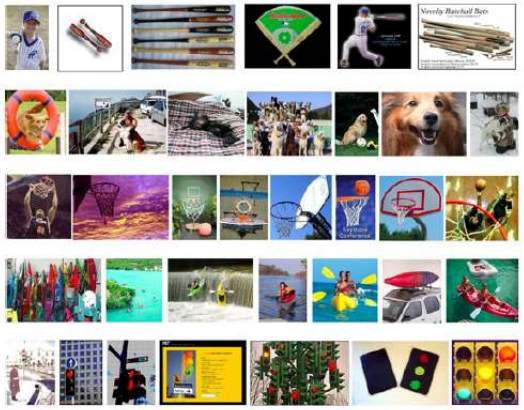
[0, 0, … **0.8**, …, **0.3**, …, **0.5**, …] = coefficients (feature representation)

Image Classification

Evaluated on Caltech101 object category dataset.



9K images, 101 classes



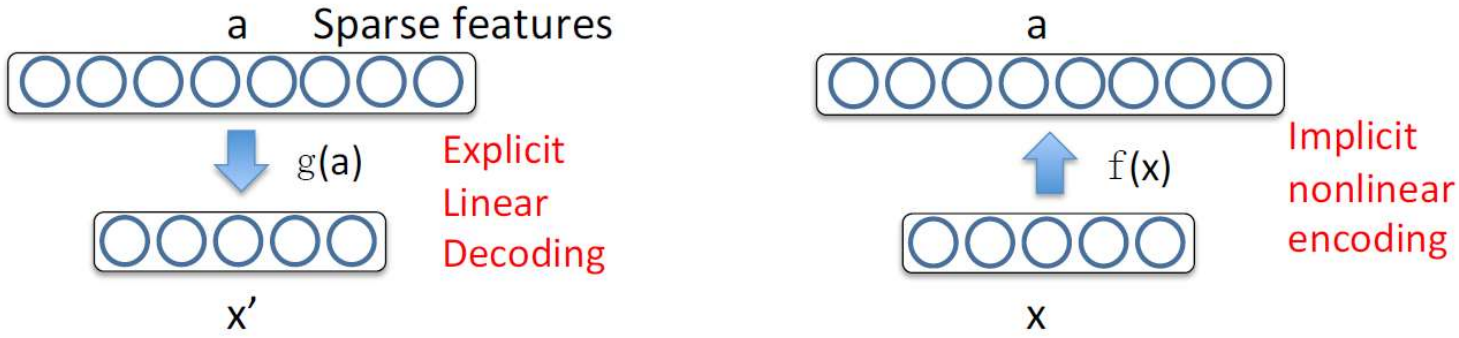
Algorithm	Accuracy
Baseline (Fei-Fei et al., 2004)	16%
PCA	37%
Sparse Coding	**47%**

Slide Credit: Honglak Lee

Lee, Battle, Raina, Ng, 2006

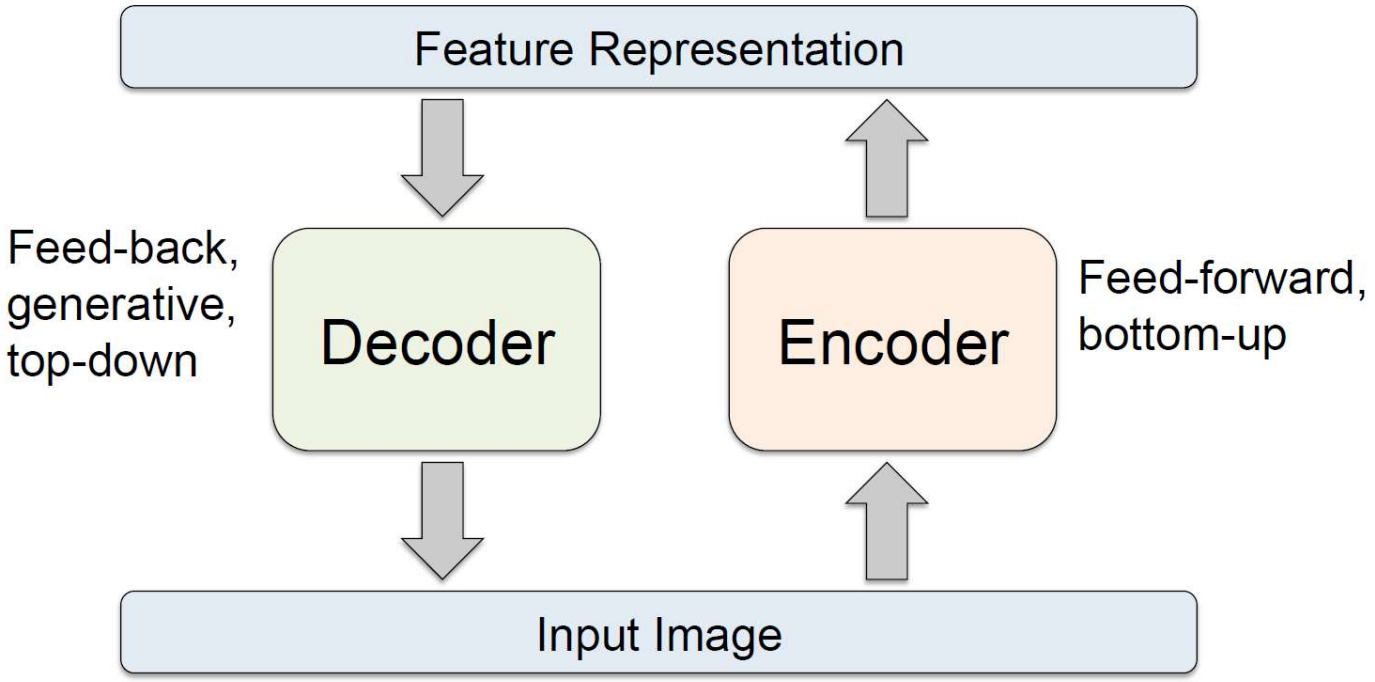
Interpreting Sparse Coding

$$\min_{\mathbf{a},\boldsymbol{\phi}} \sum_{n=1}^{N} \left|\left| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk}\boldsymbol{\phi}_k \right|\right|_2^2 + \lambda \sum_{n=1}^{N}\sum_{k=1}^{K} |a_{nk}|$$

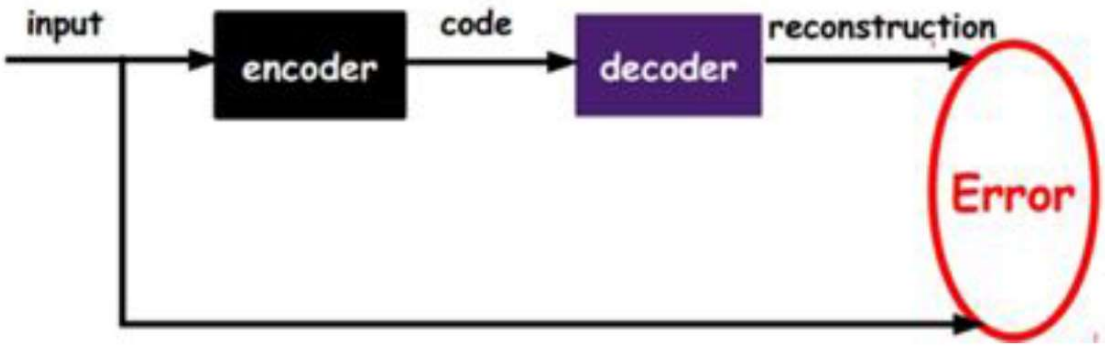


- Sparse, over-complete representation **a.**
- Encoding **a** = f(**x**) is implicit and nonlinear function of **x**.
- Reconstruction (or decoding) **x'** = g(**a**) is linear and explicit.

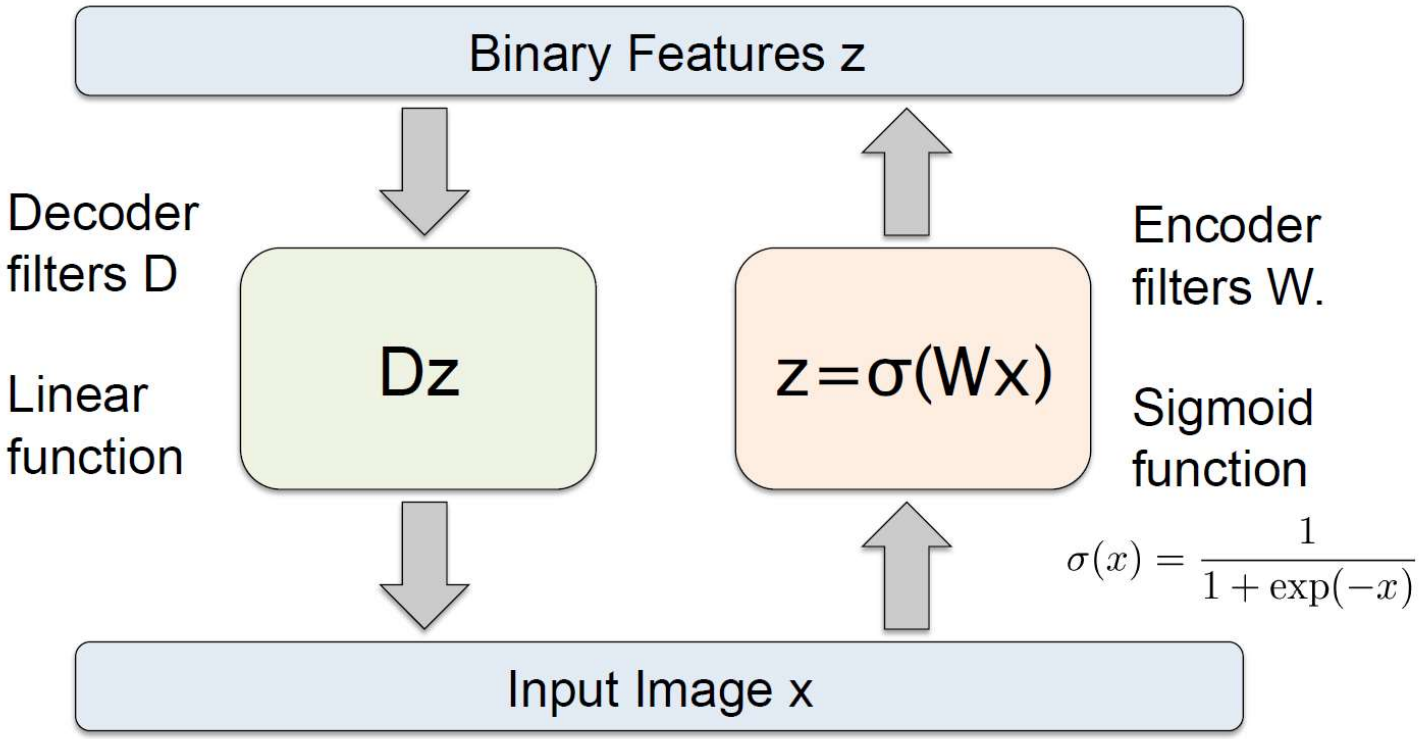
自编码器



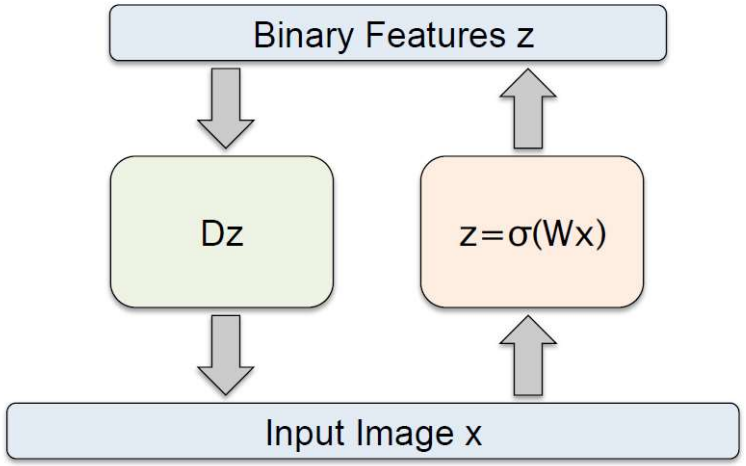
- Details of what goes insider the encoder and decoder matter!
- Need constraints to avoid learning an identity.



自编码器


Binary Features z
Decoder filters D
Linear function
Dz
z=σ(Wx)
Encoder filters W.
Sigmoid function
$\sigma(x) = \frac{1}{1+\exp(-x)}$
Input Image x

自编码器



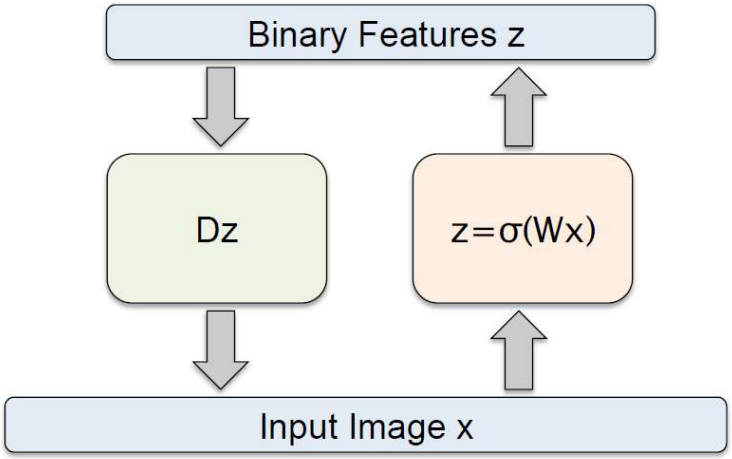
- An autoencoder with D inputs, D outputs, and K hidden units, with K<D.

- Given an input x, its reconstruction is given by:

$$y_j(\mathbf{x}, W, D) = \underbrace{\sum_{k=1}^{K} D_{jk}}_{\text{Decoder}} \underbrace{\sigma\left(\sum_{i=1}^{D} W_{ki} x_i\right)}_{\text{Encoder}}, \quad j = 1, .., D.$$

$$y_j = \sum_{k=1}^{K} D_{jk} z_k \qquad z_k = \sigma\left(\sum_{i=1}^{D} W_{ki} x_i\right)$$

自编码器

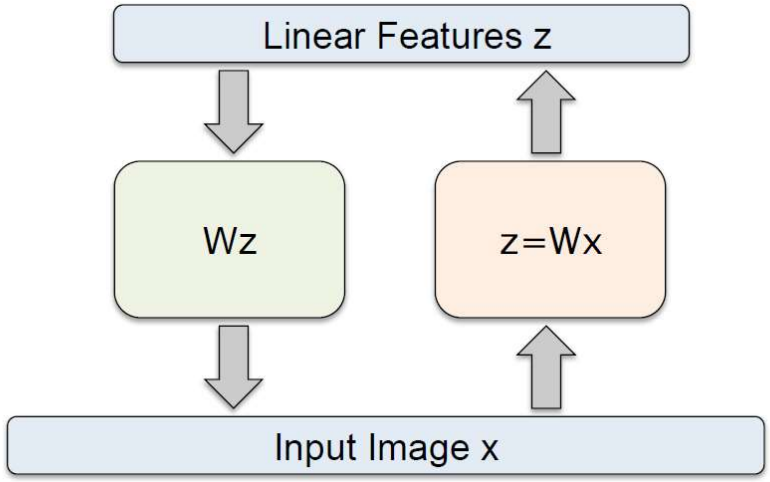


• An autoencoder with D inputs, D outputs, and K hidden units, with K<D.

• We can determine the network parameters W and D by minimizing the reconstruction error:

$$E(W, D) = \frac{1}{2} \sum_{n=1}^{N} ||y(\mathbf{x}_n, W, D) - \mathbf{x}_n||^2.$$

线性自编码器

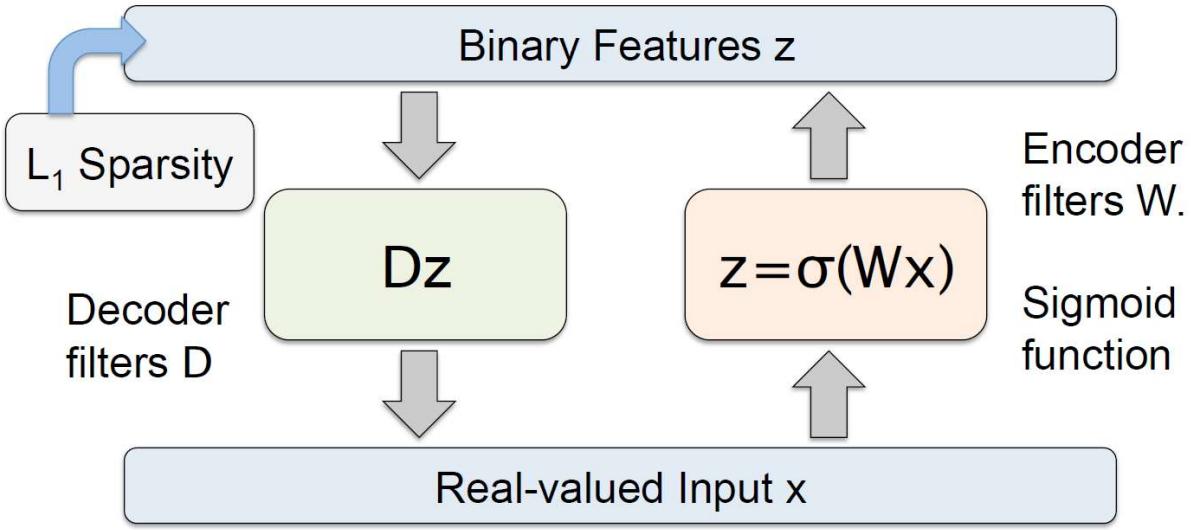


- If the hidden and output layers are linear, it will learn hidden units that are a linear function of the data and minimize the squared error.

- The K hidden units will span the same space as the first k principal components. The weight vectors may not be orthogonal.

- With nonlinear hidden units, we have a nonlinear generalization of PCA.

稀疏自编码器



At training time

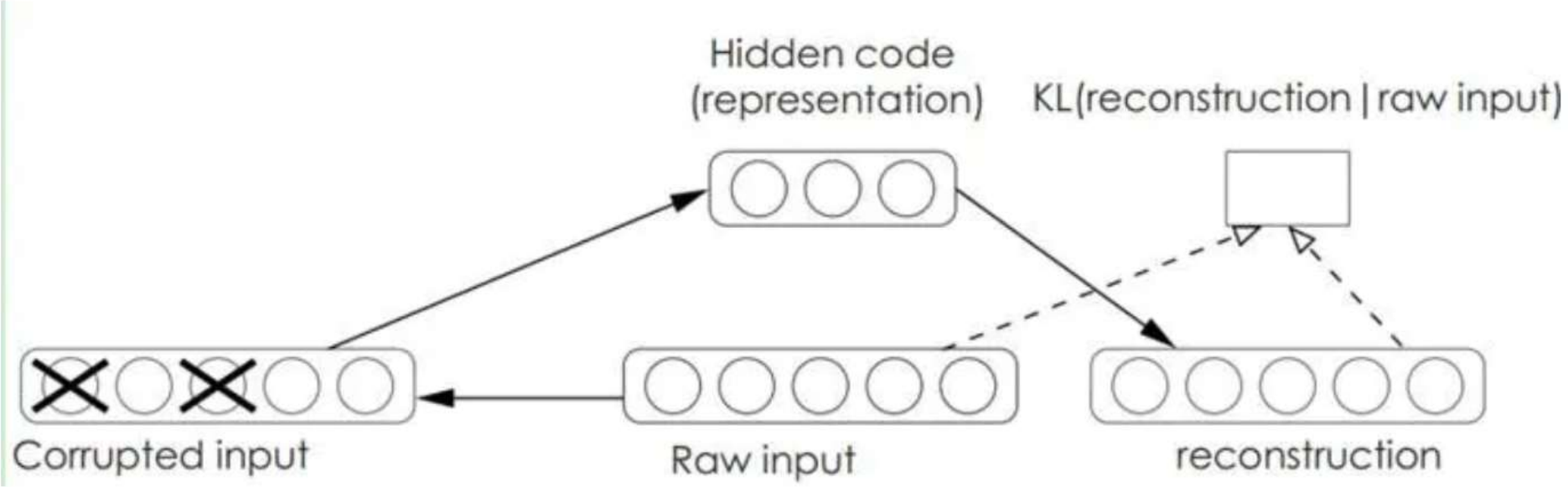
$$\min_{D,W,\mathbf{z}} \underbrace{||D\mathbf{z}-\mathbf{x}||_2^2+\lambda|\mathbf{z}|_1}_{\text{Decoder}}+\underbrace{||\sigma(W\mathbf{x})-\mathbf{z}||_2^2}_{\text{Encoder}}$$

Kavukcuoglu, Ranzato, Fergus, LeCun, 2009

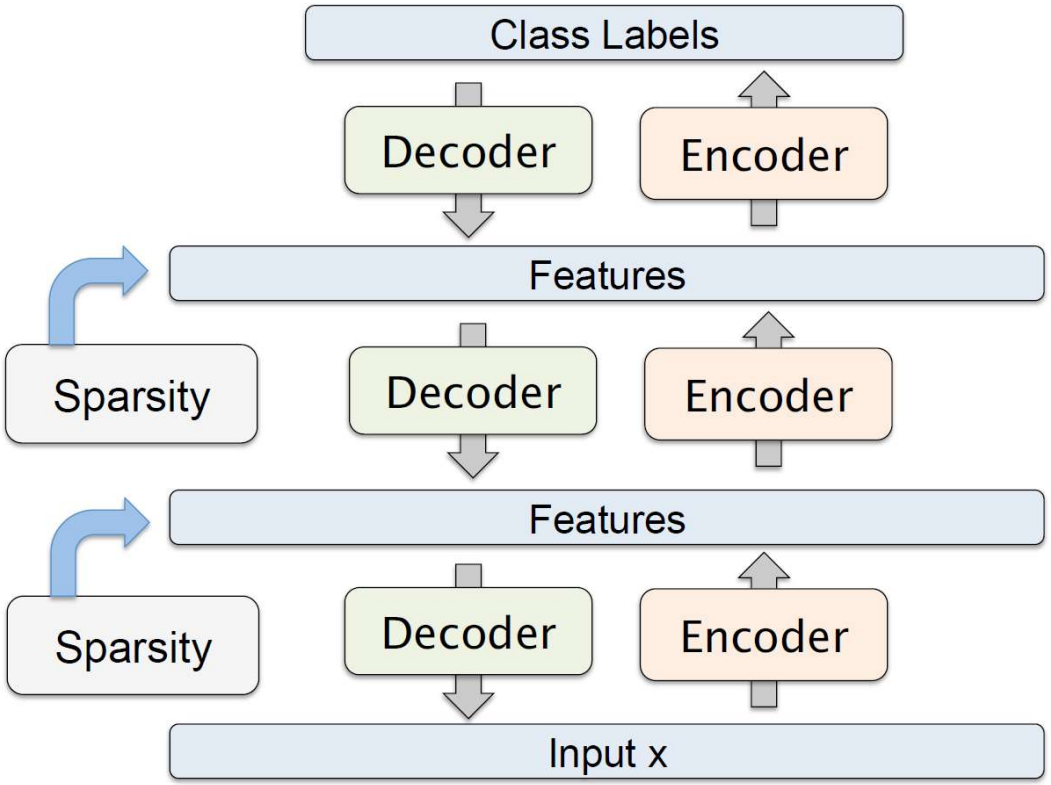
降噪自编码器

在自动编码器的基础上，训练数据加入噪声，所以自动编码器必须学习去去除这种噪声而获得真正的没有被噪声污染过的输入。

- 迫使编码器去学习输入信号的更加鲁棒的表达，这也是它的泛化能力比一般编码器强的原因。
- 可以通过梯度下降算法去训练。



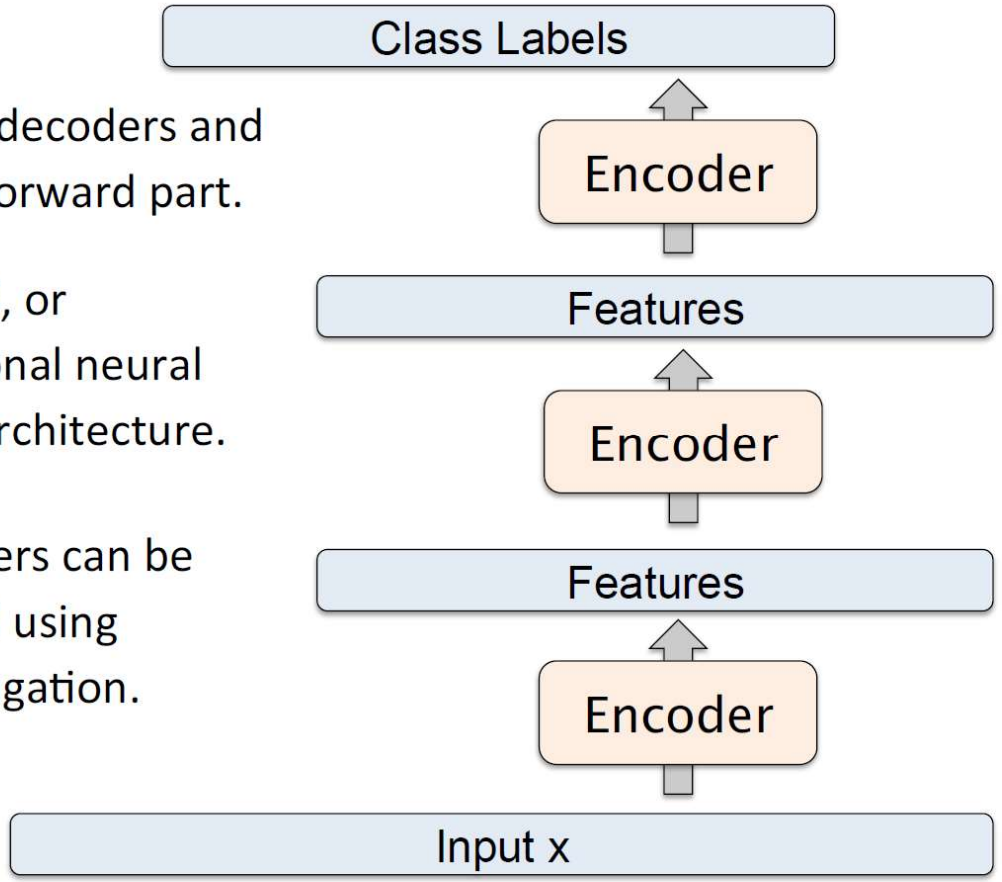
堆叠自编码器



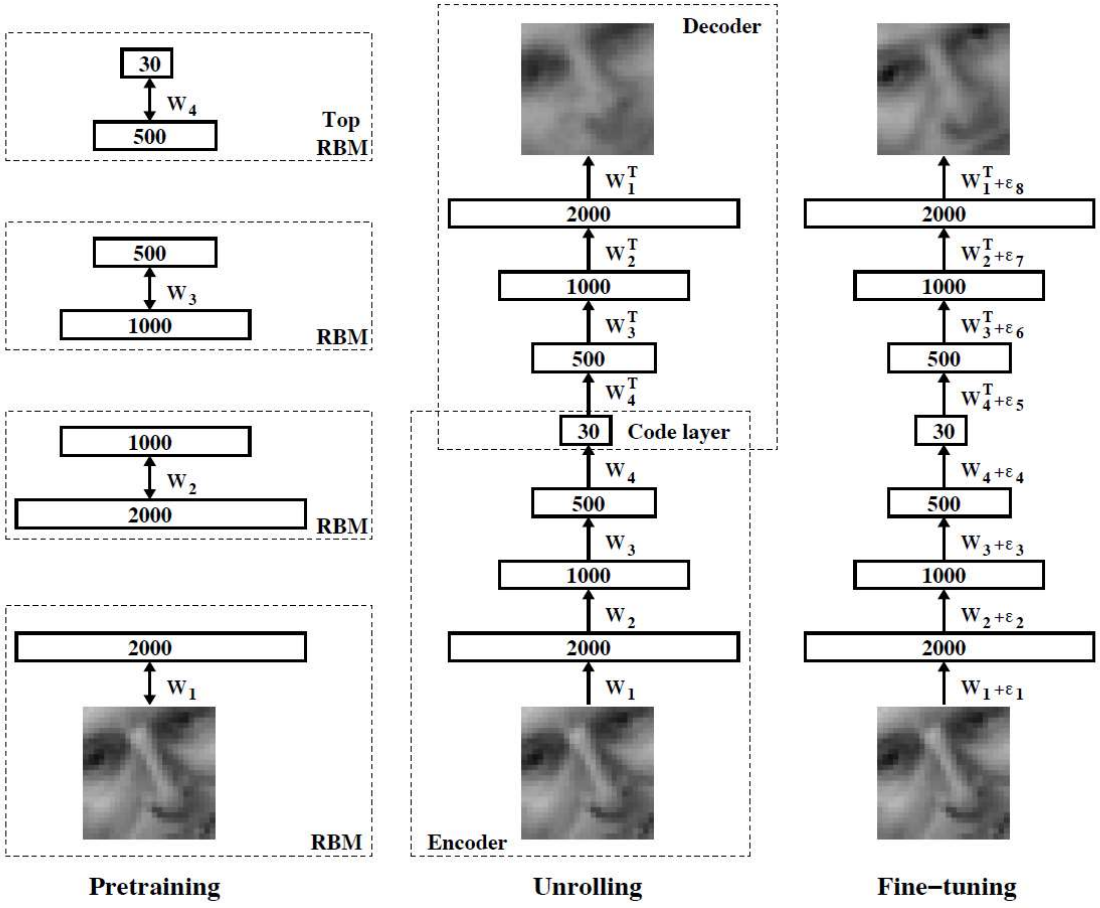
贪婪的逐层学习
Layer-wise training

堆叠自编码器-用类别标签微调

- Remove decoders and use feed-forward part.
- Standard, or convolutional neural network architecture.
- Parameters can be fine-tuned using backpropagation.



深度自编码器


30
W4
500
Top
RBM
500
W3
1000
RBM
1000
W2
2000
RBM
2000
W1
RBM
Pretraining
Decoder
W1T
2000
W2T
1000
W3T
500
W4T
30
Code layer
W4
500
W3
1000
W2
2000
W1
Encoder
Unrolling
W1T+ε8
2000
W2T+ε7
1000
W3T+ε6
500
W4T+ε5
30
W4+ε4
500
W3+ε3
1000
W2+ε2
2000
W1+ε1
Fine−tuning

深度自编码器

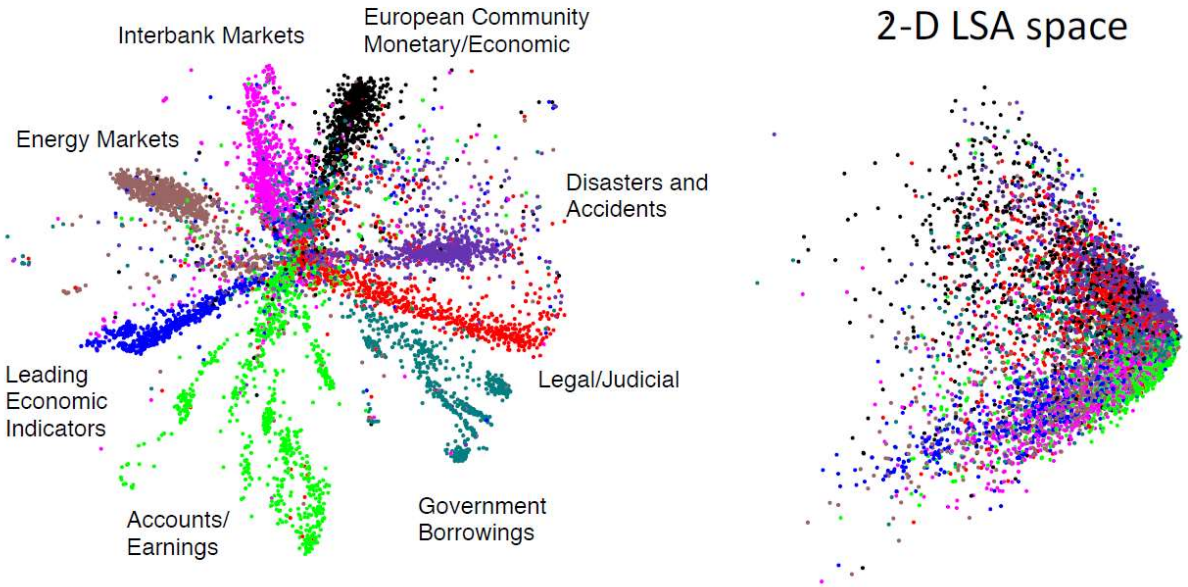
• 25x25 – 2000 – 1000 – 500 – 30 autoencoder to extract 30-D real-valued codes for Olivetti face patches.

• **Top**: Random samples from the test dataset.

• **Middle**: Reconstructions by the 30-dimensional deep autoencoder.

• **Bottom**: Reconstructions by the 30-dimentinoal PCA.

深度自编码器—信息检索



• The Reuters Corpus Volume II contains 804,414 newswire stories (randomly split into **402,207 training** and **402,207 test).**

• "Bag-of-words" representation: each article is represented as a vector containing the counts of the most frequently used 2000 words.

(Hinton and Salakhutdinov, Science 2006)

全可观测模型

- Explicitly model conditional probabilities:

$$p_{\text{model}}(\boldsymbol{x}) = p_{\text{model}}(x_1) \prod_{i=2}^{n} p_{\text{model}}(x_i \mid x_1, \ldots, x_{i-1})$$

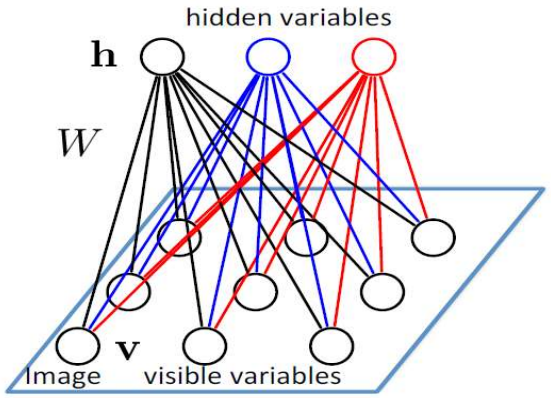
Each conditional can be a complicated neural network

- A number of successful models, including
  - NADE, RNADE (Larochelle, et.al. 20011)
  - Pixel CNN (van den Ord et. al. 2016)
  - Pixel RNN (van den Ord et. al. 2016)

Pixel CNN

受限玻尔兹曼机

Graphical Models: Powerful framework for representing dependency structure between random variables.



Feature Detectors

Pair-wise **Unary**

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} v_i h_j + \sum_{i=1}^{D} v_i b_i + \sum_{j=1}^{F} h_j a_j\right)$$

$$\theta = \{W, a, b\}$$

$$P_\theta(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_\theta(v_i|\mathbf{h}) = \prod_{i=1}^{D} \frac{1}{1+\exp(-\sum_{j=1}^{F} W_{ij} v_i h_j - b_i)}$$

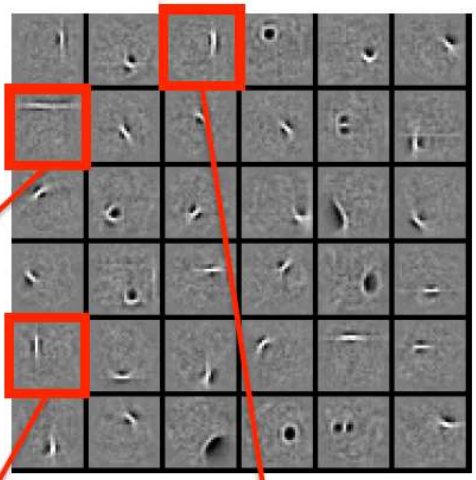
RBM is a Markov Random Field with:

- Stochastic binary visible variables $\mathbf{v} \in \{0,1\}^D$.
- Stochastic binary hidden variables $\mathbf{h} \in \{0,1\}^F$.
- Bipartite connections.

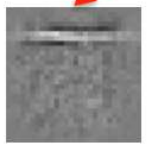
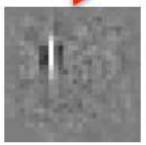
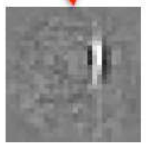
Markov random fields, Boltzmann machines, log-linear models.

Observed Data
Subset of 25,000 characters

Learned W: "edges"
Subset of 1000 features



New Image:

$= \sigma\Big(0.99 \times$  $+ \; 0.97 \times$  $+ \; 0.82 \times$  $\cdots\Big)$

$p(h_7 = 1|v)$ $p(h_{29} = 1|v)$

Sparse representations

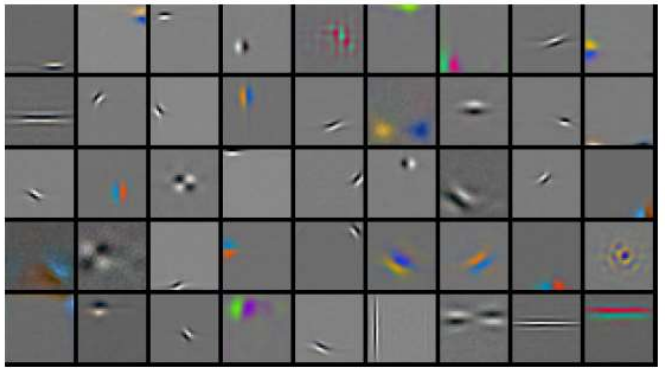
$\sigma(x) = \frac{1}{1+\exp(-x)}$

Logistic Function: Suitable for modeling binary images

RBMs for Real-valued & Count Data

4 million **unlabelled** images

Learned features (out of 10,000)



REUTERS

AP Associated Press

Reuters dataset:
804,414 **unlabeled**
newswire stories
Bag-of-Words

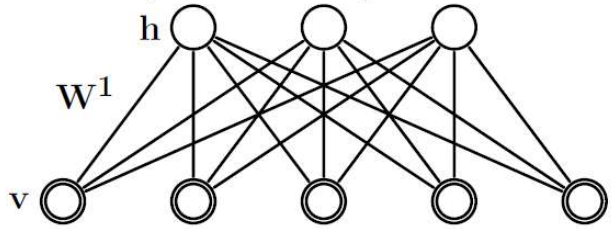
Learned features: ``topics''

russian	clinton	computer	trade	stock
russia	house	system	country	wall
moscow	president	product	import	street
yeltsin	bill	software	world	point
soviet	congress	develop	economy	dow

Collaborative Filtering

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{ijk} W_{ij}^k v_i^k h_j + \sum_{ik} b_i^k v_i^k + \sum_j a_j h_j \right)$$

Binary hidden: user preferences



Multinomial visible: user ratings

Netflix dataset:
480,189 users
17,770 movies
Over 100 million ratings

Learned features: ``genre''

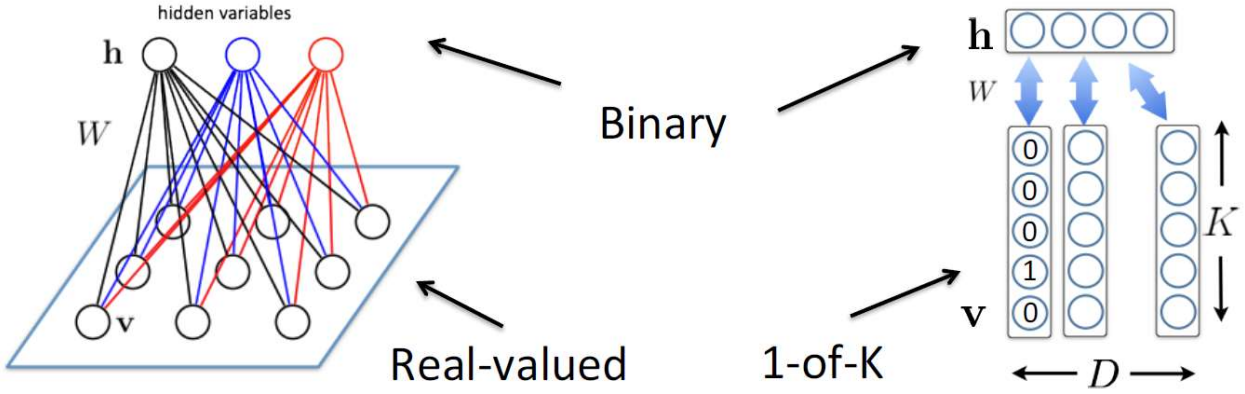
Fahrenheit 9/11	Independence Day
Bowling for Columbine	The Day After Tomorrow
The People vs. Larry Flynt	Con Air
Canadian Bacon	Men in Black II
La Dolce Vita	Men in Black
Friday the 13th	Scary Movie
The Texas Chainsaw Massacre	Naked Gun
Children of the Corn	Hot Shots!
Child's Play	American Pie
The Return of Michael Myers	Police Academy

State-of-the-art performance on the Netflix dataset.

(Salakhutdinov, Mnih, Hinton, ICML 2007)

针对不同的数据模态

- Binary/Gaussian/Softmax RBMs: All have binary hidden variables but use them to model different kinds of data.



- It is easy to infer the states of the hidden variables:

$$P_\theta(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{F} P_\theta(h_j|\mathbf{v}) = \prod_{j=1}^{F} \frac{1}{1+\exp(-a_j - \sum_{i=1}^{D} W_{ij} v_i)}$$

Product of Experts

The joint distribution is given by:

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{ij} W_{ij} v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j\right)$$

Marginalizing over hidden variables:

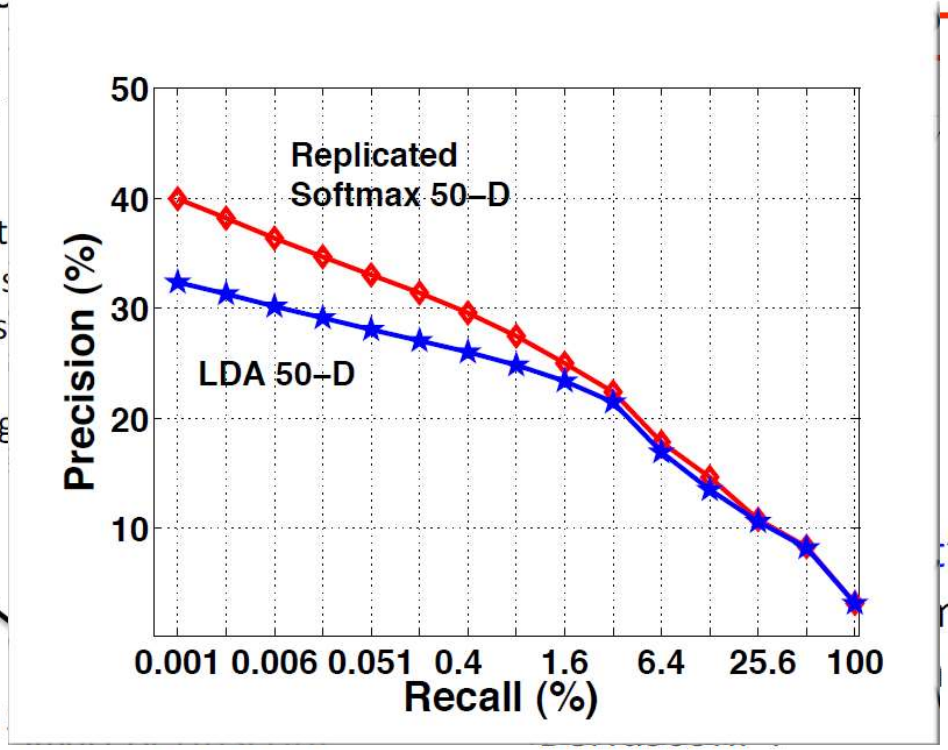
$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}}$$

Product of Experts

$$W_{ij} v_i)\Big)$$

government
authority
power
empire
federation

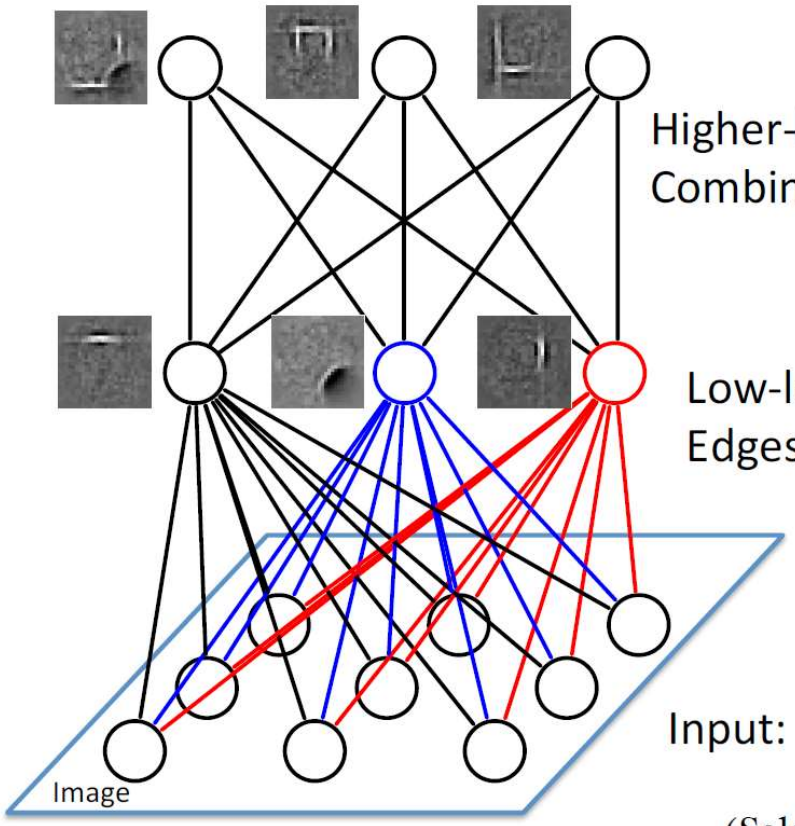
clint
hous
pres
bill
cong



", "corruption"
nbine to give very
word "Silvio

深度玻尔兹曼机

Learn simpler representations,
then compose more complex ones



Higher-level features:
Combination of edges

Low-level features:
Edges

Built from **unlabeled** inputs.

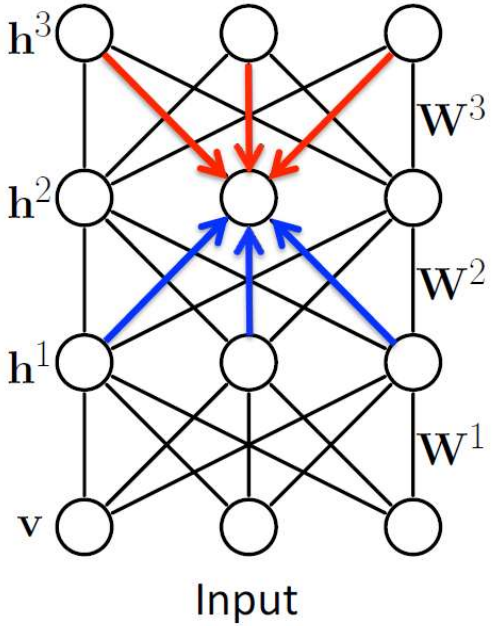
Input: Pixels

(Salakhutdinov 2008, Salakhutdinov & Hinton 2009)

深度玻尔兹曼机

$$P_\theta(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left[\mathbf{v}^\top W^{(1)} \mathbf{h}^{(1)} + \mathbf{h}^{(1)\top} W^{(2)} \mathbf{h}^{(2)} + \mathbf{h}^{(2)\top} W^{(3)} \mathbf{h}^{(3)}\right]$$

Same as RBMs



$\theta = \{W^1, W^2, W^3\}$ model parameters

- Dependencies between hidden variables.
- All connections are undirected.
- Bottom-up and Top-down:

$$P(h_j^2 = 1 | \mathbf{h}^1, \mathbf{h}^3) = \sigma\left(\sum_k W_{kj}^3 h_k^3 + \sum_m W_{mj}^2 h_m^1\right)$$

Top-down　　Bottom-up

- Hidden variables are dependent even when **conditioned on the input.**

作业

下载或实现一个(深度)自编码器，在ORL数据上实现自编码器特征学习。

谢　谢!