S&DS 365 / 565

Data Mining and Machine Learning

Dictionary Learning

connect to unsupervised Neural Network

Yale

Outline

- Sparse coding aka Dictionary Learning
- Deep networks and sparse coding

Dictionary Learning (Sparse Coding)

- Roots in computational neuroscience (Olshausen and Field, 1996)
- A type of non-orthogonal PCA
- Key ingredients: redundancy, sparsity → explored in regularization
- Together yield data-driven features that are localized, often interpretable 局部的
- Plays a role in deep learning

Recall: Principal Components Analysis

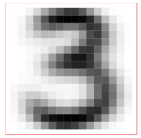
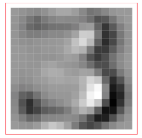
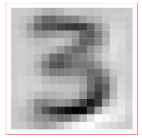
We have a data set $y^{(i)}, i = 1, \ldots, n$ of n points in d dimensions.

Principal components analysis finds the best k-term linear reconstruction of the data.

Principal vectors are orthogonal

Handwritten Digits (3s) – Top 2 components

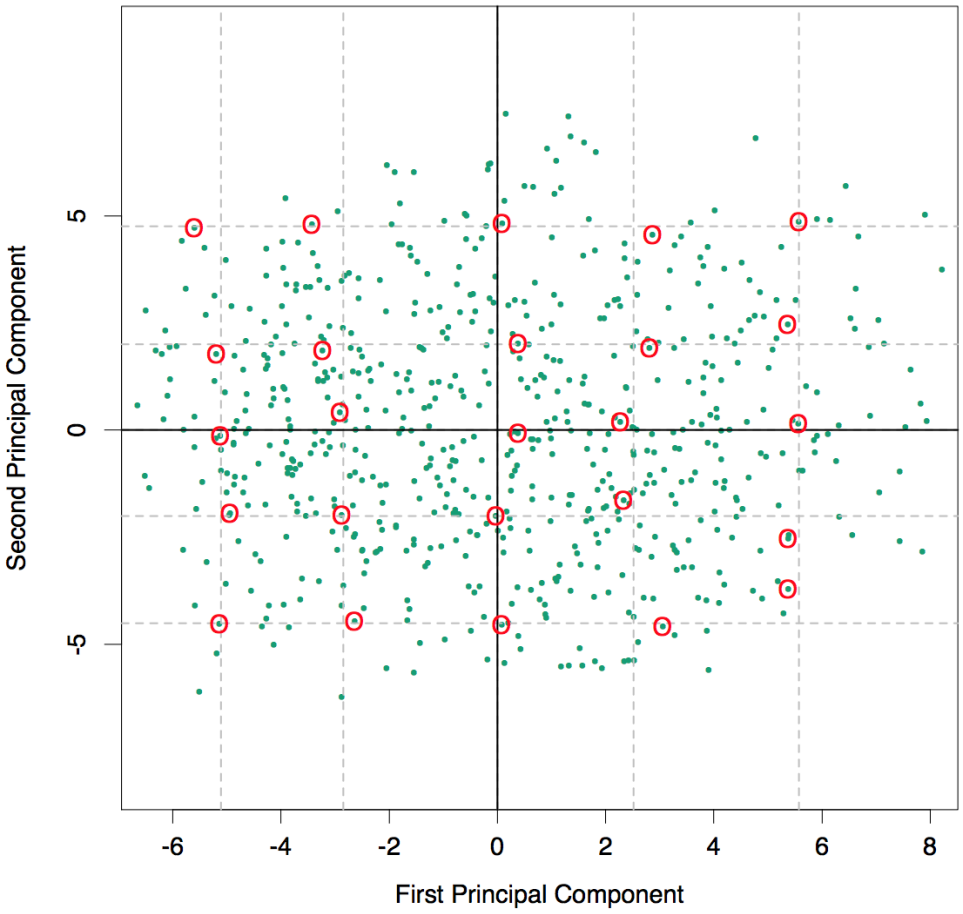
$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

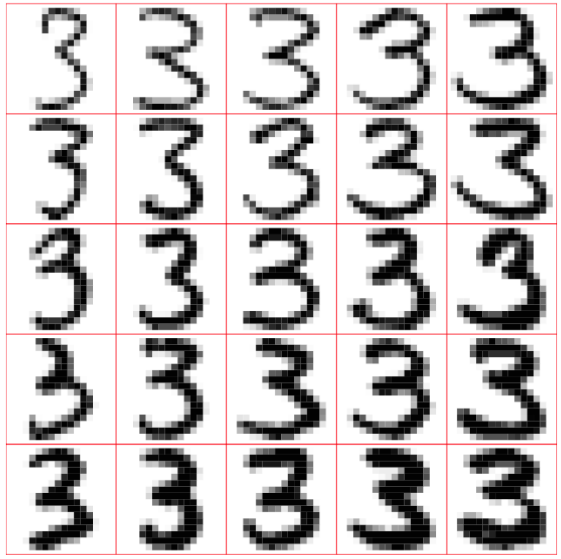
$$= \text{  } + \lambda_1 \cdot \text{  } + \lambda_2 \cdot \text{  }.$$

mean of data X

singular vector 1 (u_1)

singular vector 2 (u_2)





PCA $\mapsto$ Dictionary Learning

We'll change this in two ways

called dictionary elements v_j

- First, the vectors (columns) don't need to be orthogonal
- Second, we will require λ to be *sparse* small # of λ is non-zero

$|D|$: size of dictionary

want $|D|$ to be large
choose best v_j that represent data

$$\tilde{y}^{(i)} = \sum_{j=1}^{|D|} \lambda_j v_j, \quad \text{where few } \lambda_j \text{ are non-zero!}$$

Dictionary Learning

Mathematical formulation of dictionary learning:

optimize over β and X

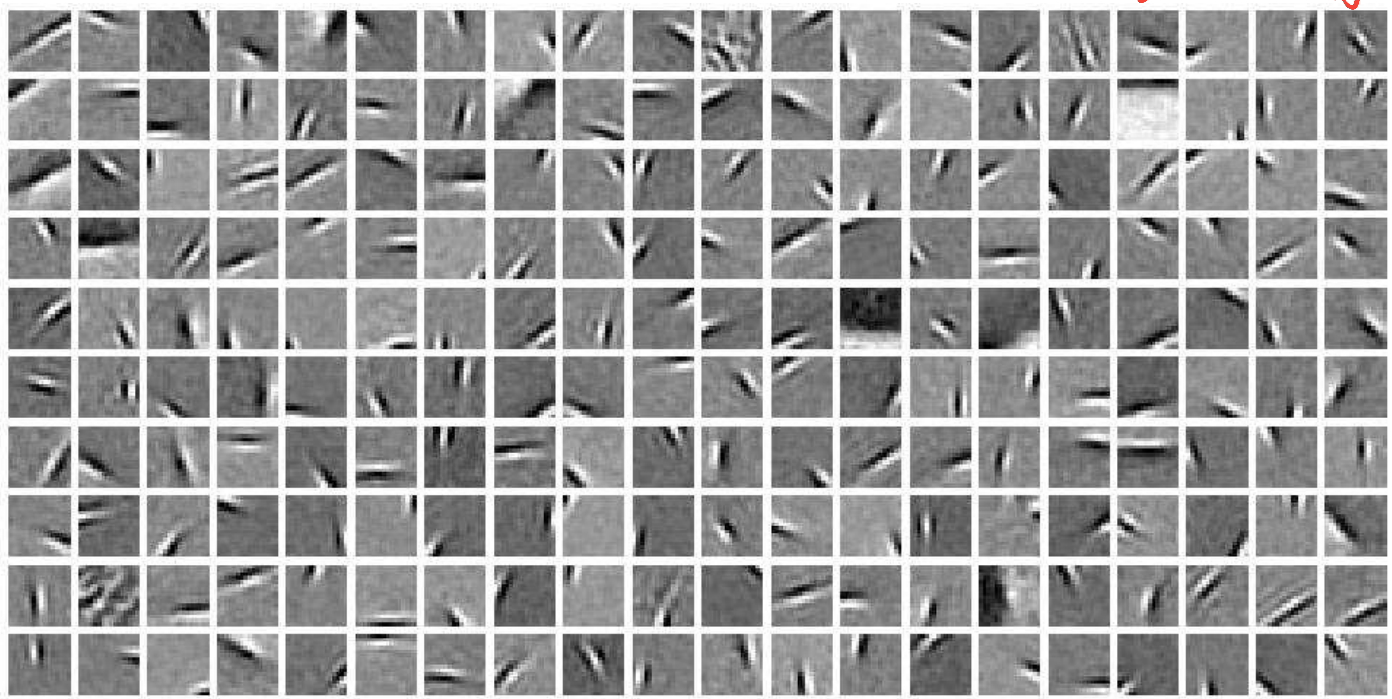
$$\min_{\beta \text{ sparse},\, X} \sum_i \left\| y^{(i)} - X\beta^{(i)} \right\|_2^2$$

l_2-norm squared

in linear regression, we already know X, only optimize β

train on small patches of image

✓ identify edges in different texture images



Dictionary Learning

When human see an image, how neuron in brain activated ? Not all neuron will be activated at the same time

Motivation: understand neural coding (Olshausen and Field, 1996).

original image

sparse representation

Codewords/patch 8.14, RSS 0.1894

Original Paper

letters to nature

Nature **381**, 607 - 609 (13 June 1996); doi:10.1038/381607a0

Mathematical model of neuron reaction pattern when an animal see images

Emergence of simple-cell receptive field properties by learning a sparse code for natural images

BRUNO A. OLSHAUSEN* & DAVID J. FIELD

Department of Psychology, Uris Hall, Cornell University, Ithaca, New York 14853, USA
*Present address: Center for Neuroscience, UC Davis, Davis, California 95616, USA.

THE receptive fields of simple cells in mammalian primary visual cortex can be characterized as being spatially localized, oriented[1–4] and bandpass (selective to structure at different spatial scales), comparable to the basis functions of wavelet transforms[5,6]. One approach to understanding such response properties of visual neurons has been to consider their relationship to the statistical structure of natural images in terms of efficient coding[7–12]. Along these lines, a number of studies have attempted to train unsupervised learning algorithms on natural images in the hope of developing receptive fields with similar properties[13–18], but none has succeeded in producing a full set that spans the image space and contains all three of the above properties. Here we investigate the proposal[8,12] that a coding strategy that maximizes sparseness is sufficient to account for these properties. We show that a learning algorithm that attempts to find sparse linear codes for natural scenes will develop a complete family of localized, oriented, bandpass receptive fields, similar to those found in the primary visual cortex. The resulting sparse image code provides a more efficient representation for later stages of processing because it possesses a higher degree of statistical independence among its outputs.

Observations

- Edge detectors at different scales and orientations
- Very similar to “Gabor wavelets”
- Much like activation patterns found in early layers of visual cortex in mammals

PCA vs. Sparse Coding

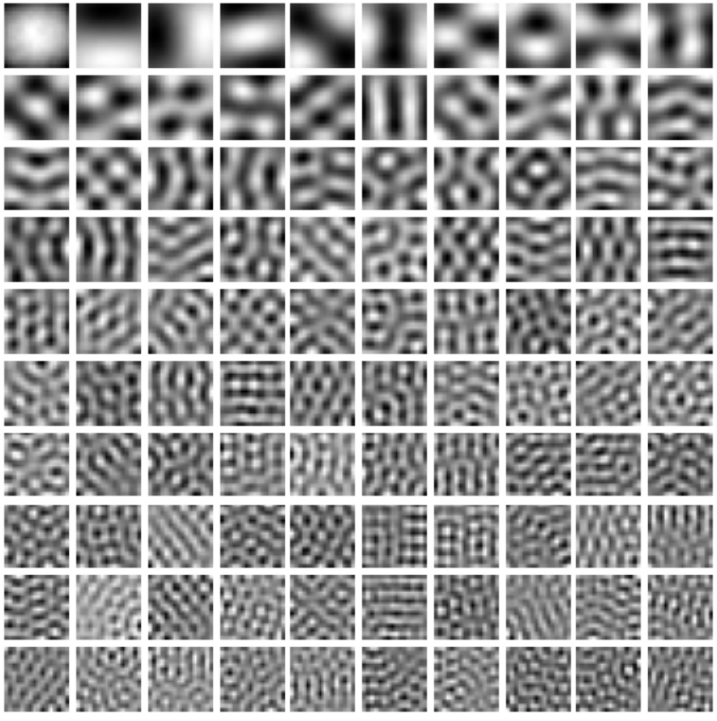
less informative

sin ∿ connect to Fourier

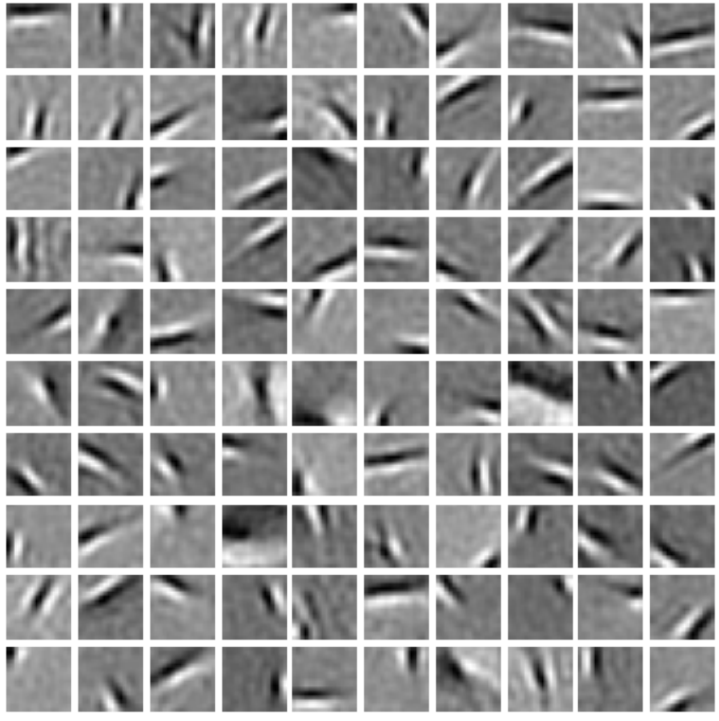
very informative

wavelet

PCA



Receptive fields



Properties

- Provides high dimensional, nonlinear representation
- Sparsity enables codewords to specialize, isolate “features”
- A type of topic modeling!
- Variants successful in image analysis

Method

- We optimize both X and β
- The dictionary elements X_j are shared across all data. (Analogues of principal vectors)
- The coefficients $\beta^{(i)}$ are specific to each data point. (Analogues of the principal components $\lambda^{(i)}$)

Method

Estimated using a form of (guess what?) stochastic gradient descent.

- Select a subset of the data vectors $y^{(i)}$.
- For the current dictionary, fit the coefficients $\beta^{(i)}$ using sparse linear regression.
- Now compute the gradient with respect to X, and do a gradient step in the direction of the gradient.

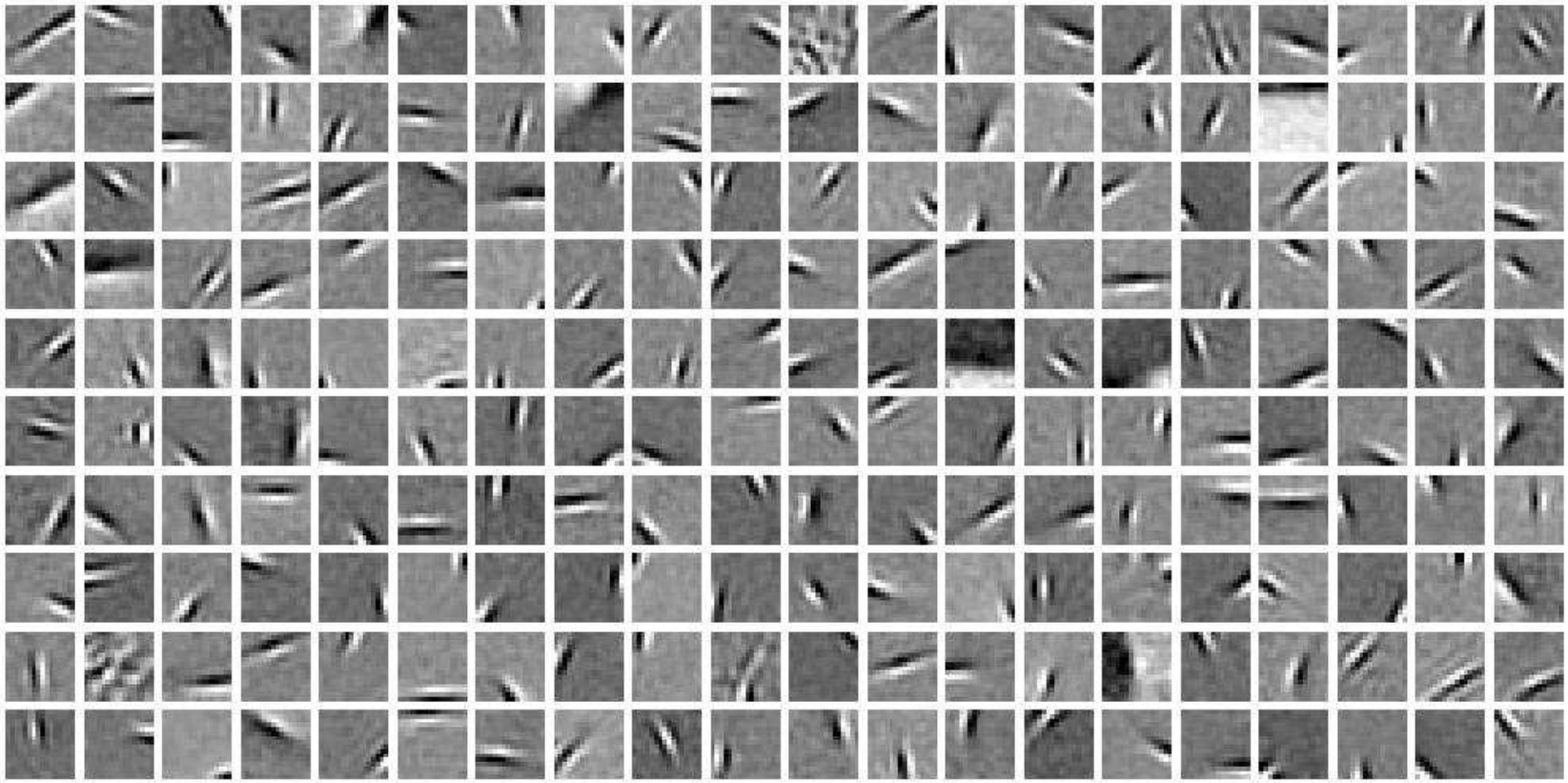
alternative minimization, first solve for $\beta^{(i)}$, then solve for X, then solve for $\beta^{(i)}$

$$\hat{\beta}^{(i)} = \sum_i \| y^{(i)} - X\beta^{(i)} \|_2^2$$

- fix X => LR

Learned dictionary from natural images

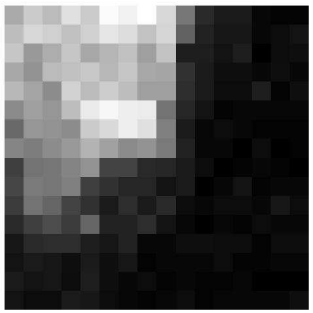
give edge detector at different scales and orientations



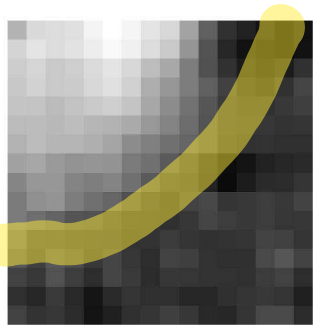
./sparse-coding.gif

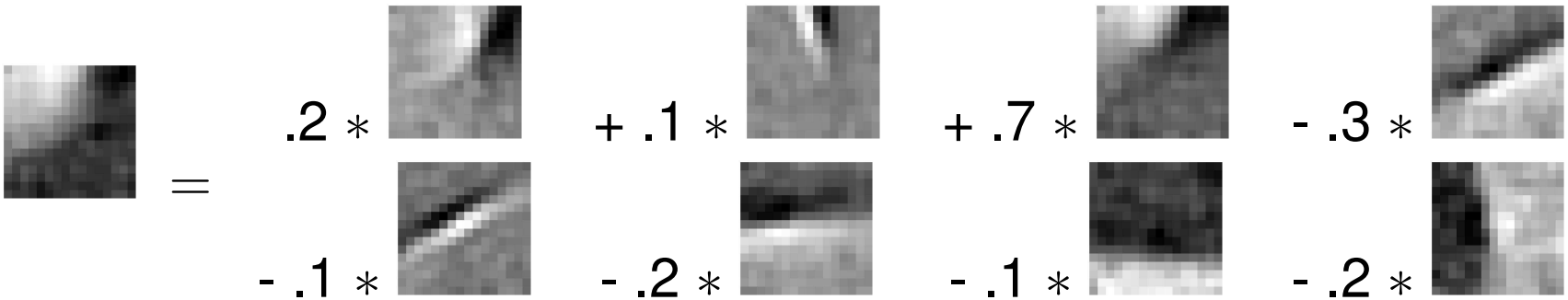
Reconstruction example

Original patch



Reconstruction





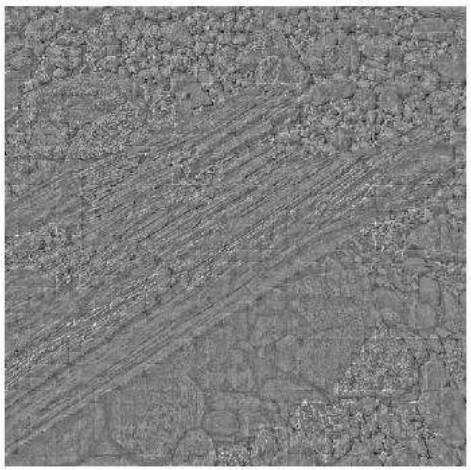
using 8 codewords (dict elements x_j)

Compression performance

original

sparsely reconstructed

error



Original image (left), sparsely reconstructed image (center), difference image (right).

Andrew Ng presentation

https://www.youtube.com/watch?v=n1ViNeWhC24

1:40–6:15 (where do features come from?), 20:10–25:15 (invents edge detection), 31:00–33:40 (applied hierarchically) &t=1m40s

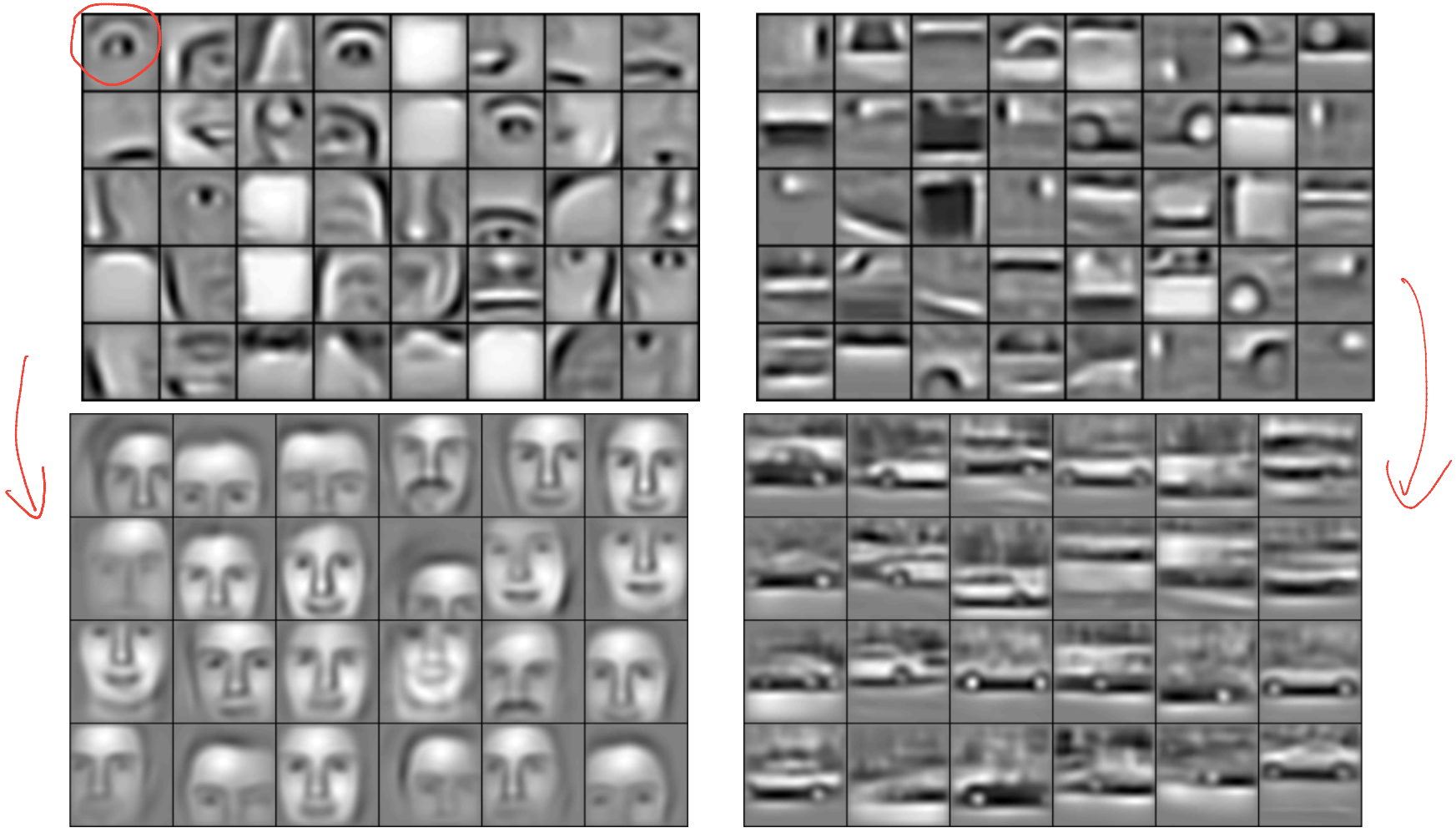
Applying sparse coding hierarchically

a dictionary element

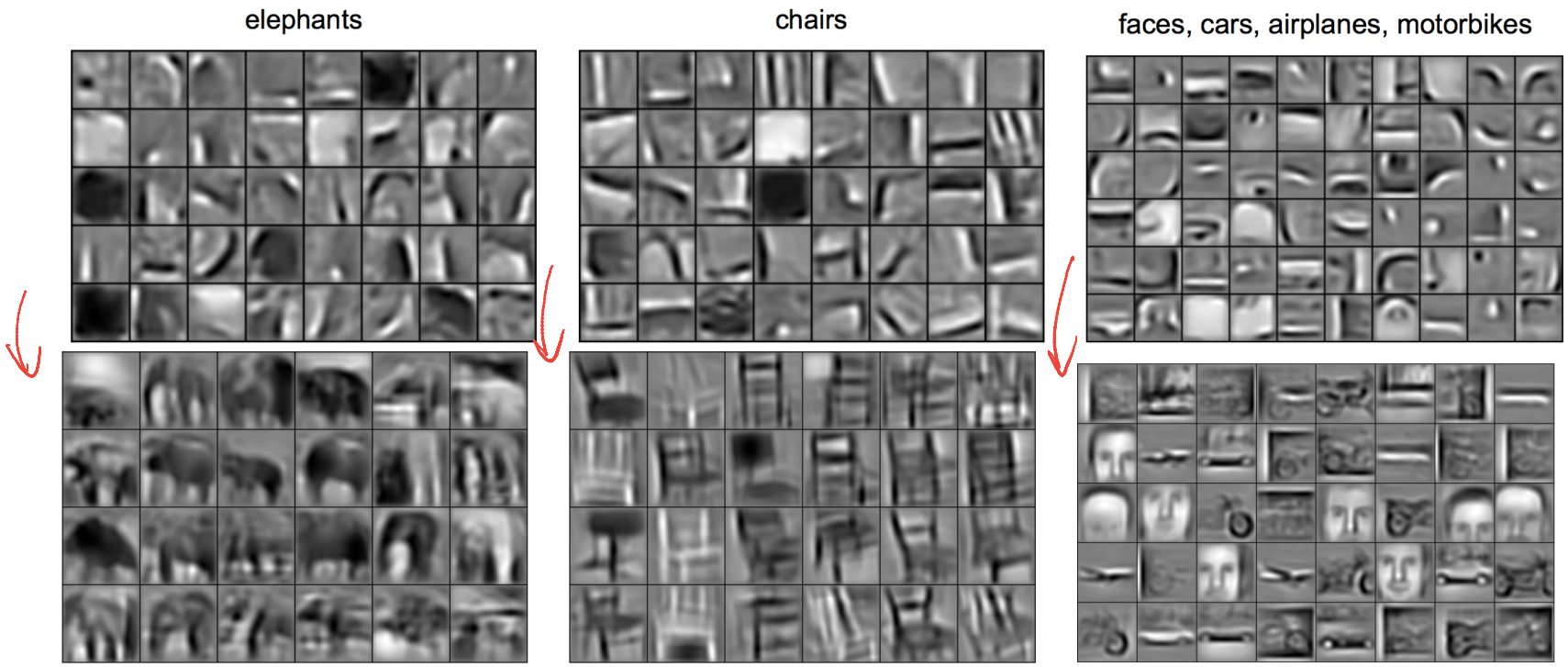
early layers: edges → hierarchy pieces of cars

faces

cars

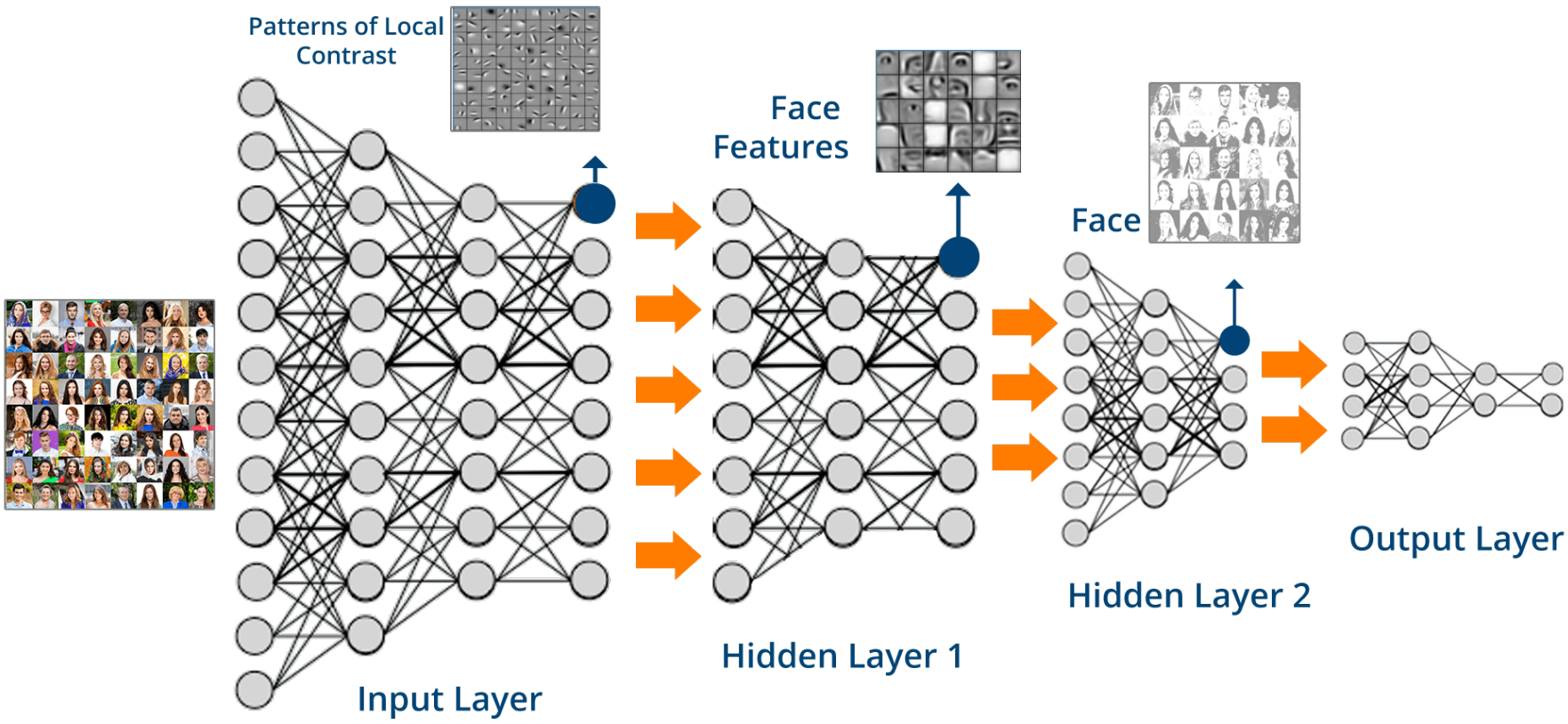


Applying sparse coding hierarchically

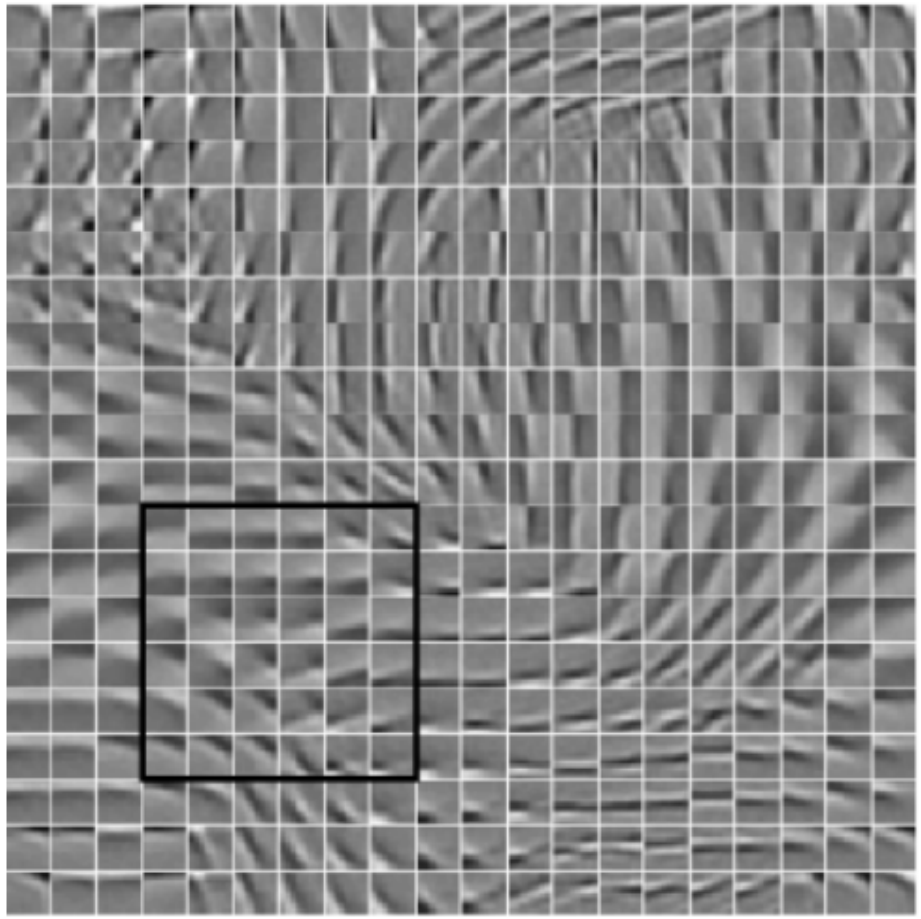


Lee et al., “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations”

Deep learning applies this in layers



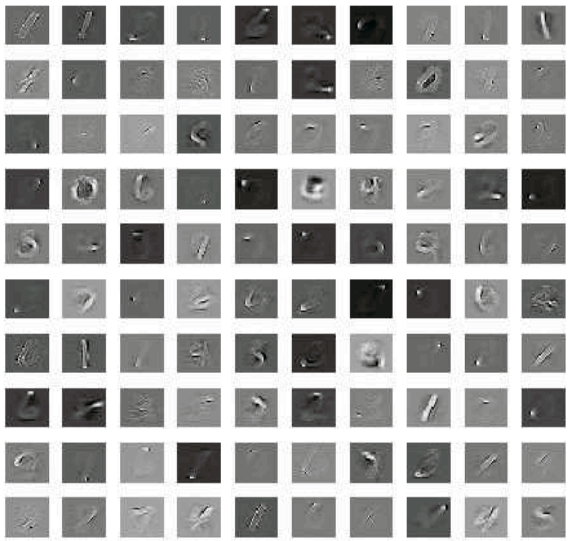
Interesting variants of dictionary learning



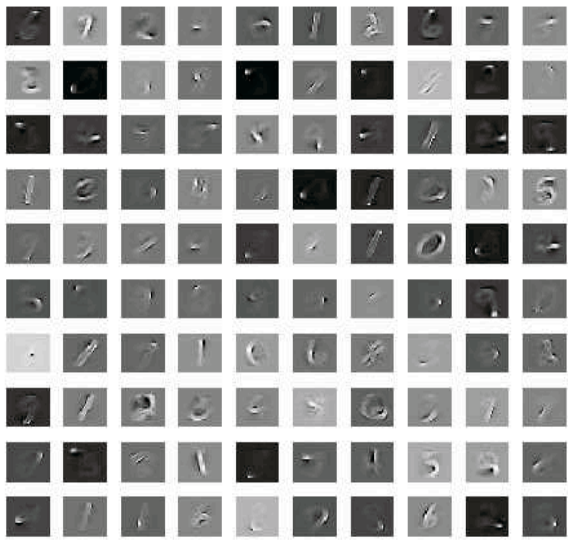
https://www.cs.nyu.edu/~yann/research/sparse/index.html

Sparse coding for MNIST

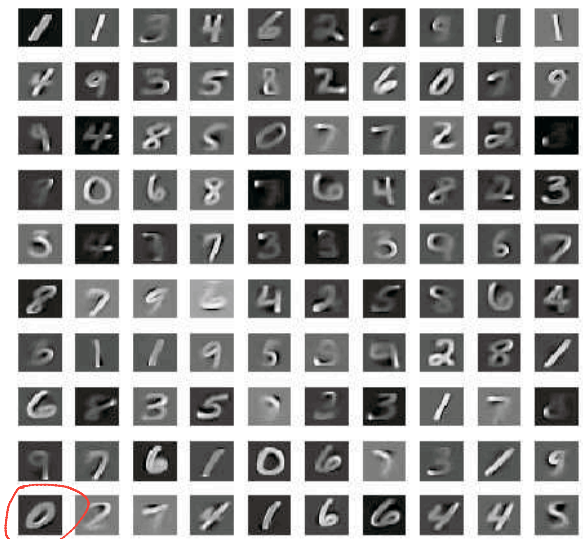
Classification



Error: 4.54%



Error: 3.75%



Error: **2.64%**

- Best accuracy when learned codewords are like digits

Dictionary Learning vs. Topic Modeling

can do dict learning for doc

consider Y as a random variable tells probability a certain word will appear in a doc

constraint for X sum over X's entry

|V|: vocabulary size

Dictionary learning: $Y \sim \sum_j \beta_j X_j, \quad 1^T X_j = 1, \quad X \succeq 0$

dict element X_j is a topic distribution

β_j: probability this distribution (X_j) will influence Y

- Linear decomposition
- Sparsity from penalization of β

in this case, $X_j \in [0,1]^{|V|}$

$(X_j)_{(k)}$ is the probability that word (k) is picked.

The probability of picking word k is $\sum_j \beta_j (X_j)_{(k)}$ ①

θ_j: probability a word comes from a topic j

$p(\cdot|j)$: distribution of word i conditional on topic j

Topic modeling: $Y \sim \sum_j \theta_j p(\cdot \mid j)$

- Mixture model
- Sparsity from Dirichlet

in this case, word i has distribution $p(\cdot|j)$ with probability θ_j.

The i^{th} word, is word k with probability $\sum_j \theta_j p(k|j)$ ②

① ② same linear model

Difference: when we have a specific Y

① Dict learning: has some β_j naturally exist, β_j are picked arbitrary

② Topic modeling: has a nice probalistic hierarchical structure where if I have a specific eg of Y, I first draw θ_j from some distribution, that tell me the distribution of topics for a specific doc

no probability distribution

Topics from Dictionary Learning

NeurIPS data, 1,500 documents, 50 topics

Topic 1	neuron	node	map	motion	distribution
	synaptic	nodes	space	direction	gaussian
	activity	network	feature	velocity	probability
	synapses	tree	representation	moving	density
	potential	graph	mapping	stage	mean
Topic 2	algorithm	signal	word	model	bound
	gradient	noise	recognition	parameter	number
	step	filter	speech	prediction	result
	convergence	processing	character	modeling	threshold
	update	component	speaker	structure	theorem
Topic 3	circuit	spike	training	neural	action
	chip	information	error	network	policy
	analog	rate	generalization	net	optimal
	current	firing	test	recurrent	states
	voltage	noise	performance	computation	step

Dictionary Learning vs. Topic Modeling

	dictionary learning	*topic modeling*
perspective	classical	Bayesian
approach	regularization	latent variables
thinking	sparsity	generative model
methods	optimization	simulation

Dictionary Learning vs. Topic Modeling

	dictionary learning	*topic modeling*
perspective	classical	Bayesian
approach	regularization	latent variables
thinking	sparsity	generative model
methods	optimization	simulation

Math of Dictionary Learning

1) Alternating Minimization (Like in K-means clustering)

2) Partial Gradient Descent

3) Full Gradient Descent

We will talk about mini-batch SGD.

Alternating Minimization

optimize over A and B $\min_{A,B} f(A,B)$

Might be hard to optimize A and B together.

However, for fixed A or fixed B it's easy.

Example: K-means, Dictionary Learning

Algorithm A_0 = random initialization or (smart init) initialization

$B_1 = \arg\min_B f(A_0, B)$ also called coordinate descent

$A_1 = \arg\min_A f(A, B_1)$

repeat

keep new B close to previous B

$B_{k+1} = \arg\min_B f(A_k, B) \quad (+\lambda \|B - B_k\|^2)$

$A_{k+1} = \arg\min_A f(A, B_{k+1}) \quad (+\lambda \|A - A_k\|^2)$

Application: K-means

a special case in dict learning

In K-means Goal:

$$\min_{\mu^{(j)}, \pi^{(i)}} \sum_{i=1}^{n} \| y^{(i)} - \mu^{\pi^{(i)}} \|^2$$

$\mu^{(j)}$: centroid

where $\mu^{(j)} \in \mathbb{R}^d$, $y^{(i)} \in \mathbb{R}^d$, $\pi^{(i)} \in [K]$

where $j \in [K]$, K is the # of clusters.

Computationally intractable

Use alternating minimization instead:

- Use a smart init. of $\mu^{(j)}$. (mean of data) But random can also be used, but not as good.

Smart initialization

Alternating min:

$$\pi^{(i)} = \arg\min_{j \in [K]} \| y^{(i)} - \mu^{(j)} \|^2$$

$$\mu^{(j)} = \frac{1}{|C_j|} \sum_{i \in C_j} y^{(i)}, \quad C_j = \{ i \mid \pi^{(i)} = j \}$$

Special Case of Dictionary Learning!

$$\begin{pmatrix} | \\ y^{(i)} \\ | \end{pmatrix} = \begin{pmatrix} - & x_1^T & - \\ & \vdots & \\ - & x_k^T & - \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ i \\ \vdots \\ 0 \end{pmatrix} \rightarrow j^{th} \text{ entry}$$

$$y^{(i)} = X B^{(i)}, \quad X \in \mathbb{R}^{d \times K} \text{ and}$$

The representation of $y^{(i)}$ is just a column of X

X: cluster centers

$y^{(i)}$: sample $\in \mathbb{R}^d$

$B^{(i)}$: only 1 element is non-zero $= 1$ $\begin{bmatrix} 0 \\ 0 \\ i \\ 0 \end{bmatrix}$

$B^{(i)}$ is 1- sparse, $\in \mathbb{R}^K$

and $\sum_l B_l^{(i)} = 1$,

More generally have Dictionary Learning. size of dict

① initialize random or smart $X \in \mathbb{R}^{d \times D}$, usually $D \gg d$

有时 $d \approx K^2 \log(D)$

② $B^{(i)} = \underset{B \mid B \text{ is sparse}}{\operatorname{argmin}} \| y^{(i)} - X B^{(i)} \|^2$

K is sparsity of $B^{(i)}$.

$K \approx e^d$

see next slide: Forward selection or Lasso regression to solve for $\beta^{(i)}$

③ $X = \underset{X}{\operatorname{argmin}} \sum_{i=1}^{n} \| y^{(i)} - X B^{(i)} \|^2 \quad \left(+ \frac{\mu}{2} \| X \|_F^2 \right)$

can also add regularizer for X

Ridge

$= \frac{\mu}{2} \sum_{a,b} X_{(ab)}^2$.

Fobenius Norm = sum of squared entry of matrix

just Linear regression

solve by just least squares

Solving for $B^{(i)}$

$$B^{(i)} = \underset{B \mid B \text{ is sparse}}{\operatorname{argmin}} \; \| y^{(i)} - X B^{(i)} \|^2$$

1) Use forward selection.

2) Use Lasso (Regularization)

$$B^{(i)} = \underset{B}{\operatorname{argmin}} \; \frac{1}{2} \| y^{(i)} - X B^{(i)} \|^2 + \lambda \| B^{(i)} \|_1$$

Regularize

$$\| v \|_1 = \sum_{j=1}^{D} | v_{(j)} |$$

L1-norm

Partial Gradient Descent : only update X

1. Randomly init X
2. Learn B as in previous slide
3. Gradient step X — slowly update X

$$X^{(K+1)} = X^{(K)} - \eta^{(k)} \nabla_X \mathcal{L}(X^{(K)}, B) \quad (-\alpha X^{(K)})$$

↑ gradient with respect to X

↑ only use if you add the Ridge penalty

4. Repeat

(Neural Net learning this is called weight decay)

Full on Gradient Descent : update both X and β

Loss function $$\mathcal{L}(X, B) = \sum_i \| y^{(i)} - X B^{(i)} \|^2, \quad B \in \mathbb{R}^{D \times n}$$

optimize $$\min_{X,\beta} \mathcal{L}(X, B) + \frac{\mu}{2} \|X\|_F^2 + \lambda \sum_i \| B^{(i)} \|_1$$

$\frac{\mu}{2}\|X\|_F^2$: L-2 norm; $\lambda \sum_i \|B^{(i)}\|_1$: L-1 norm

① initialize X randomly, initialize B randomly

② Update X and β

$$X^{(k+1)} = X^{(k)} - \eta^{(k)} \nabla_X \mathcal{L}(X^{(k)}, B^{(k)}) - \mu X$$

$\eta^{(k)}$ $\gamma^{(k)}$: different step size for X and β update

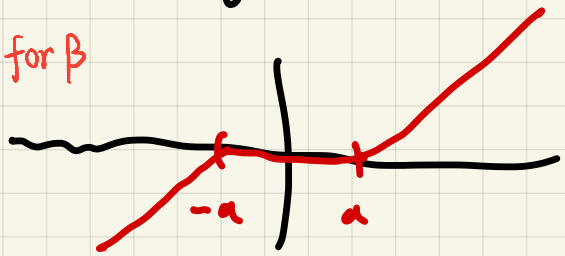
$$\tilde{B}^{(k+1)} = B^{(k)} - \gamma^{(k)} \nabla_B \mathcal{L}(X^{(k+1)}, B^{(k)}) \quad \text{(regular gradient descent)}$$

$$B^{(k+1)} = S_{\gamma^{(k)}\lambda}(\tilde{B}^{(k+1)}) \quad \text{(shrinkage)}$$

$$= \begin{cases} \tilde{B}^{(k+1)} - \lambda & (\tilde{B}^{(k+1)} > \lambda) \\ 0 & (-\lambda < \tilde{B}^{(k+1)} < \lambda) \\ \tilde{B}^{(k+1)} + \lambda & (\tilde{B}^{(k+1)} < -\lambda) \end{cases}$$

where $$S_a(v) = \text{sign}(v) \cdot \max(|v| - a, 0)$$

Using L1-norm will slow down update for β so people use shrinkage instead



Mini-batch SGD

In general optimizing

$$f(x) = \sum_{i=1}^{n} f_i(x)$$

Optimize over x

$$\hat{x} = \min_x f(x)$$

① Grad Descent: $x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla_x f(x^{(k)})$ $S=n$

② SGD / stochastic gradient descent: $x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla_x f_I(x^{(k)})$ $S=1$

$$I \sim \text{Unif}[n]$$

③ Mini-batch SGD: $x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla_x \sum_{j \in S} \frac{f_j(x^{(k)})}{|S|}$

used in dict learning

where $S \subseteq [n]$

subset is randomly selected.