

Visual Recognition: Learning Features

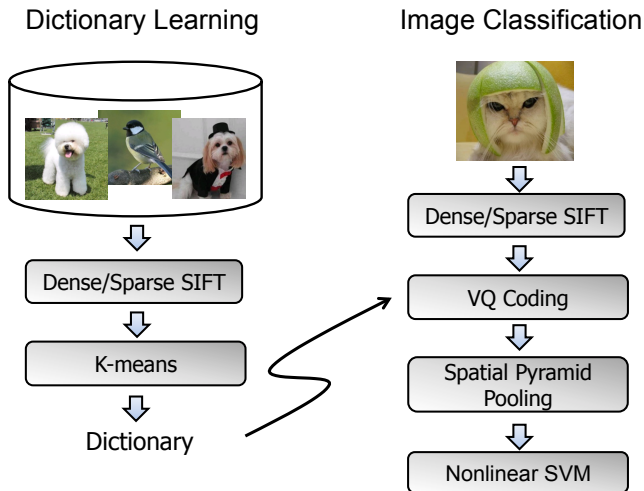
Raquel Urtasun

TTI Chicago

Feb 21, 2012

- Sparse coding
- Deep architectures
- Topic models

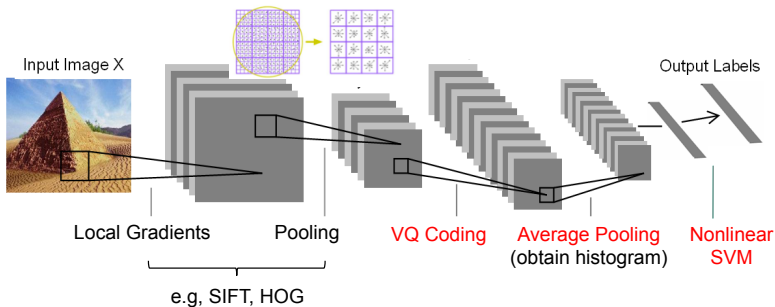
Image Classification using BoW



[Source: K. Yu]

BoW+SPM: the architecture

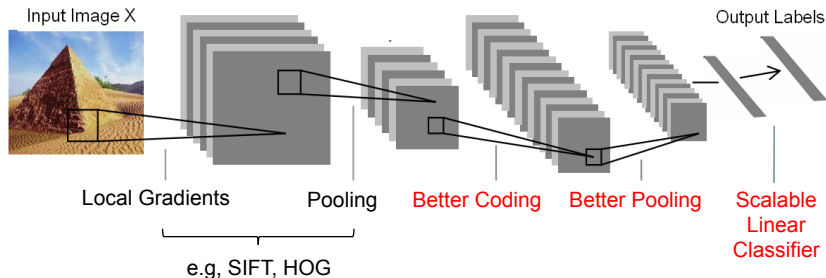
- Nonlinear SVM is not scalable
- VQ coding may be too coarse
- Average pooling is not optimal
- Why not learn the whole thing?



[Source: K. Yu]

Sparse Architecture

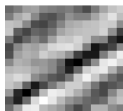
- Nonlinear SVM is not scalable → Scalable linear classifier
- VQ coding may be too coarse → Better coding methods
- Average pooling is not optimal → Better pooling methods
- Why not learn the whole → Deep learning



[Source: A. Ng]

Feature learning problem

- Given a 14×14 image patch x , can represent it using 196 real numbers.
- Problem: Can we find a learn a better representation for this?



- Given a set of images, learn a better way to represent image than pixels.



[Source: A. Ng]

Learning an image representation

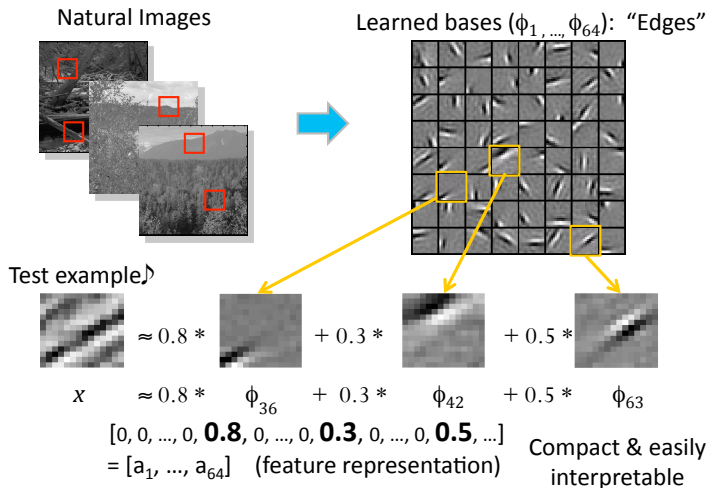
- Sparse coding [Olshausen & Field,1996]
- Input: Images $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (each in $\mathbb{R}^{n \times n}$)
- Learn: Dictionary of bases ϕ_1, \dots, ϕ_k (also $\mathbb{R}^{n \times n}$), so that each input x can be approximately decomposed as:

$$x \approx \sum_{j=1}^k a_j \phi_j$$

such that the a_j 's are mostly zero, i.e., sparse

[Source: A. Ng]

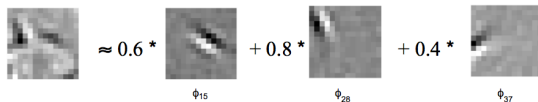
Sparse Coding Illustration



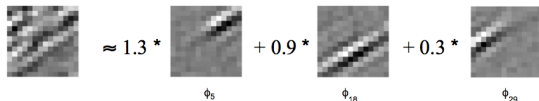
[Source: A. Ng]

More examples

- Method hypothesizes that edge-like patches are the most basic elements of a scene, and represents an image in terms of the edges that appear in it.
- Use to obtain a more compact, higher-level representation of the scene than pixels.



Represent as: [0, 0, ..., 0, 0.6, 0, ..., 0, 0.8, 0, ..., 0, 0.4, ...]



Represent as: [0, 0, ..., 0, 1.3, 0, ..., 0, 0.9, 0, ..., 0, 0.3, ...]

[Source: A. Ng]

Sparse Coding details

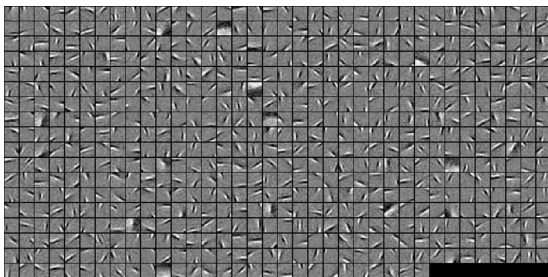
- Input: Images $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (each in $\mathbb{R}^{n \times n}$)
- Obtain dictionary elements and weights by

$$\min_{a, \phi} \sum_{i=1}^m \left(\left\| x^{(i)} - \sum_{j=1}^k a_j^{(i)} \phi_j \right\|_2^2 + \lambda \underbrace{\sum_{j=1}^k |a_j^{(i)}|_1}_{\text{sparsity}} \right)$$

- Alternating minimization with respect to ϕ_j 's and a 's.
- The second is harder, the first one is closed form.

Fast algorithms

- Solving for a is expensive.
- Simple algorithm that works well
 - Repeatedly guess sign (+, - or 0) of each of the a_i 's.
 - Solve for a_i 's in closed form. Refine guess for signs.
- Other algorithms such as projective gradient descent, stochastic subgradient descent, etc



[Source: A. Ng]

Recap of sparse coding for feature learning

Training:

- Input: Images $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (each in $\mathbb{R}^{n \times n}$)
- Learn: Dictionary of bases ϕ_1, \dots, ϕ_k (also $\mathbb{R}^{n \times n}$),

$$\min_{a, \phi} \sum_{i=1}^m \left(\|x^{(i)} - \sum_{j=1}^k a_j^{(i)} \phi_j\|^2 + \lambda \sum_{j=1}^k |a_j^{(i)}| \right)$$

Test time:

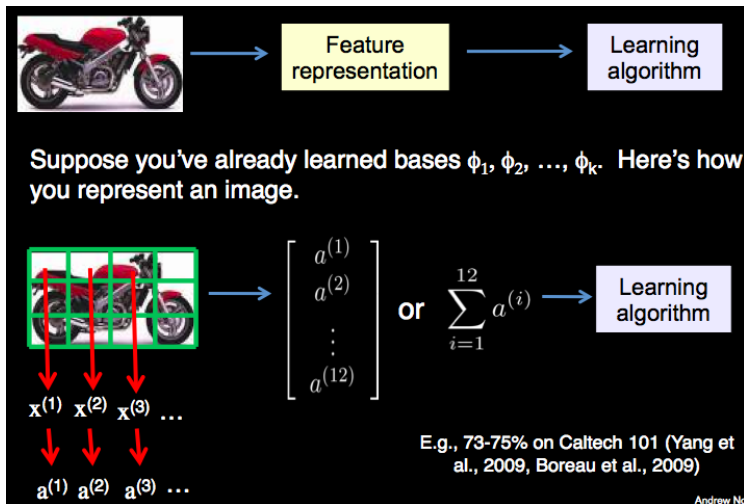
- Input: novel $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ and learned ϕ_1, \dots, ϕ_k .
- Solve for the representation a_1, \dots, a_k for each example

$$\min_a \sum_{i=1}^m \left(\|x - \sum_{j=1}^k a_j \phi_j\|^2 + \lambda \sum_{j=1}^k |a_j| \right)$$

- Much better than pixel representation, but still not competitive with SIFT, etc.
- Three ways to make it competitive:
 - Combine this with SIFT.
 - Advanced versions of sparse coding, e.g., LCC.
 - Deep learning.

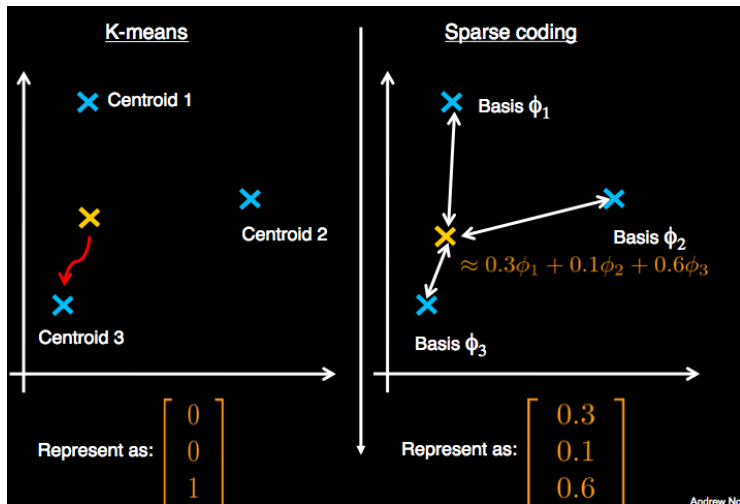
[Source: A. Ng]

Sparse Classification



[Source: A. Ng]

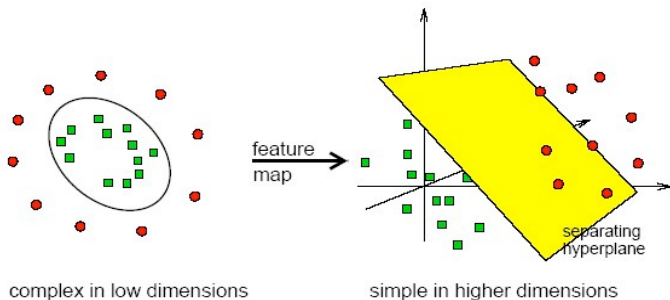
K-means vs sparse coding



[Source: A. Ng]

Why sparse coding helps classification?

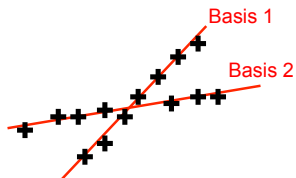
- The coding is a nonlinear feature mapping
- Represent data in a higher dimensional space
- Sparsity makes prominent patterns more distinctive



[Source: K. Yu]

A topic model view to sparse coding

- Each basis is a direction or a topic.
- Sparsity: each datum is a linear combination of only a few bases.
- Applicable to image denoising, inpainting, and super-resolution.

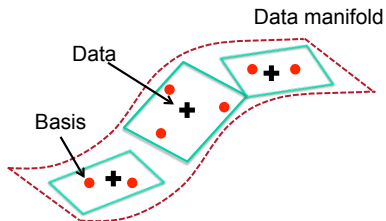


Both figures adapted from CVPR10 tutorial by F. Bach, J. Mairal, J. Ponce and G. Sapiro

[Source: K. Yu]

A geometric view to sparse coding

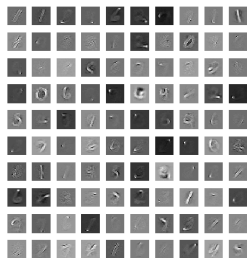
- Each basis is somewhat like a pseudo data point anchor point
- Sparsity: each datum is a sparse combination of neighbor anchors.
- The coding scheme explores the manifold structure of data



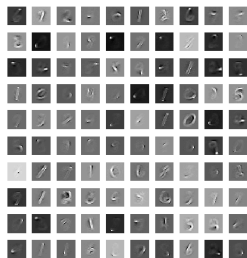
[Source: K. Yu]

Influence of Sparsity

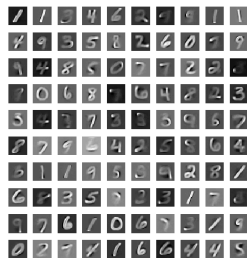
- When SC achieves the best classification accuracy, the learned bases are like digits – each basis has a clear local class association.



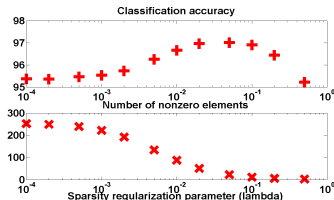
Error: 4.54%



Error: 3.75%



Error: 2.64%

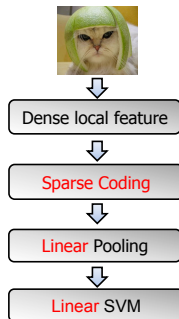


The image classification setting for analysis

- Learning an image classifier is a matter of learning nonlinear functions on patches

$$\underbrace{f(x)}_{f.\text{images}} = w^T x = \sum_{i=1}^m a_i (w^T \phi^{(i)}) = \sum_{i=1}^m a_i \underbrace{f(\phi^{(i)})}_{f.\text{patches}}$$

where $x = \sum_{i=1}^m a_i \phi^{(i)}$

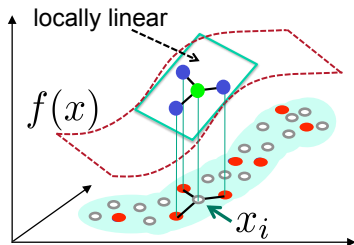


[Source: K. Yu]

Nonlinear learning via local coding

- We assume $x_i \approx \sum_{j=1}^k a_{i,j} \phi_j$ and thus

$$f(x_i) = \sum_{j=1}^k a_{i,j} f(\phi_j)$$

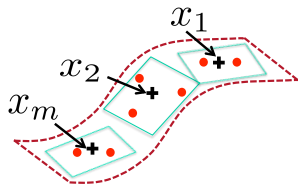


- data points
- bases

[Source: K. Yu]

How to learn the non-linear function

- 1 Learning the dictionary ϕ_1, \dots, ϕ_k from unlabeled data
- 2 Use the dictionary to encode data $x_i \rightarrow a_{i,1}, \dots, a_{i,k}$



- 3 Estimate the parameters

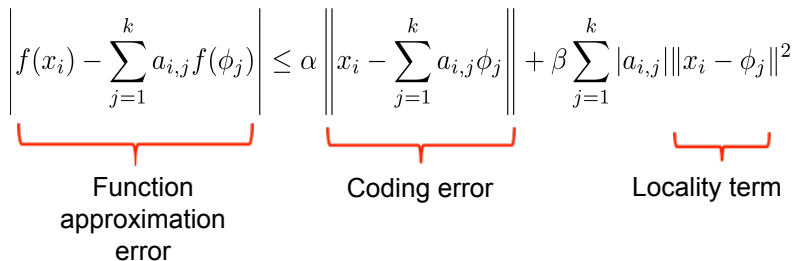
$$\begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_m) \end{bmatrix} \approx \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,k} \\ a_{2,1} & a_{2,2} & \dots & a_{2,k} \\ a_{m,1} & a_{m,2} & \dots & a_{m,k} \end{bmatrix} \times \begin{bmatrix} f(\phi_1) \\ f(\phi_2) \\ f(\phi_k) \end{bmatrix}$$

Nonlinear local learning via learning a global linear function

Local Coordinate Coding (LCC):

- If $f(x)$ is (α, β) -Lipschitz smooth

$$\left| f(x_i) - \sum_{j=1}^k a_{i,j} f(\phi_j) \right| \leq \alpha \left\| x_i - \sum_{j=1}^k a_{i,j} \phi_j \right\| + \beta \sum_{j=1}^k |a_{i,j}| \|x_i - \phi_j\|^2$$

The diagram shows the equation above with three red curly braces underneath. The first brace is under the left-hand side of the inequality and is labeled 'Function approximation error'. The second brace is under the first term on the right-hand side, $\alpha \left\| x_i - \sum_{j=1}^k a_{i,j} \phi_j \right\|$, and is labeled 'Coding error'. The third brace is under the second term on the right-hand side, $\beta \sum_{j=1}^k |a_{i,j}| \|x_i - \phi_j\|^2$, and is labeled 'Locality term'.

Function approximation error Coding error Locality term

A good coding scheme should [Yu et al. 09]

- have a small coding error,
- and also be sufficiently local

[Source: K. Yu]

- The larger dictionary, the higher accuracy, but also the higher comp. cost

Table 2: Error rates (%) of MNIST classification with different $|C|$.

$ C $	512	1024	2048	4096
Linear SVM with sparse coding	2.96	2.64	2.16	2.02
Linear SVM with local coordinate coding	2.64	2.44	2.08	1.90

(Yu et al. 09)

Table 5. The effects of codebook size on ScSPM and LSPM respectively on Caltech 101 dataset.

Codebook size		256	512	1024
30 train	ScSPM	68.26	71.20	73.20
	LSPM	57.42	58.81	58.56
15 train	ScSPM	61.97	63.23	69.70
	LSPM	51.84	53.23	51.74

(Yang et al. 09)

- The same observation for Caltech256, PASCAL, ImageNet

[Source: K. Yu]

- A fast implementation of LCC [Wang et al. 10]
- Dictionary learning using k-means and code for x based on

Step 1 ensure locality: find the K nearest bases $[\phi_j]_{j \in J(x)}$

Step 2 ensure low coding error:

$$\min_a \left\| x - \sum_{j \in J(x)} a_{i,j} \phi_j \right\|^2, \quad s.t. \quad \sum_{j \in J(x)} a_{i,j} = 1$$

[Source: K. Yu]

Table 1. Image classification results on Caltech-101 dataset

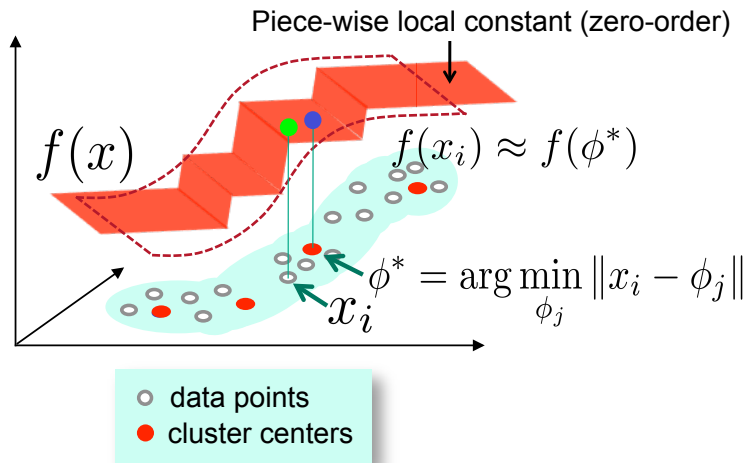
training images	5	10	15	20	25	30
Zhang [25]	46.6	55.8	59.1	62.0	-	66.20
Lazebnik [15]	-	-	56.40	-	-	64.60
Griffin [11]	44.2	54.5	59.0	63.3	65.8	67.60
Boiman [2]	-	-	65.00	-	-	70.40
Jain [12]	-	-	61.00	-	-	69.10
Gemert [8]	-	-	-	-	-	64.16
Yang [22]	-	-	67.00	-	-	73.20
Ours	51.15	59.77	65.43	67.74	70.16	73.44

Table 2. Image classification results using Caltech-256 dataset

training images	15	30	45	60
Griffin [11]	28.30	34.10	-	-
Gemert [8]	-	27.17	-	-
Yang [22]	27.73	34.02	37.46	40.14
Ours	34.36	41.19	45.31	47.68

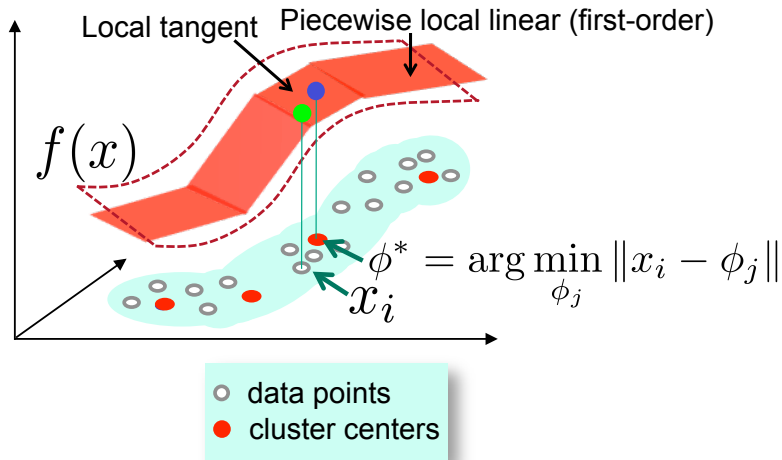
[Source: K. Yu]

Interpretation of BoW + linear classifier



[Source: K. Yu]

Support vector coding [Zhou et al, 10]



[Source: K. Yu]

- Let $[a_i, 1, \dots, a_{i,k}]$ be the VQ coding of x_i

$$f(x_i) \approx \underbrace{\left[a_{i,1}(1, x_i - \phi_1), \dots, a_{i,k}(1, x_i - \phi_k) \right]}_{\text{Super-vector codes of data}} \times \underbrace{\begin{bmatrix} f(\phi_1) \\ \nabla f(\phi_1) \\ \vdots \\ f(\phi_k) \\ \nabla f(\phi_k) \end{bmatrix}}_{\text{Global linear weights to be learned}}$$

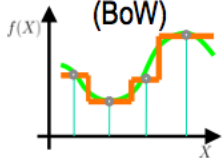
- No.1 position in PASCAL VOC 2009

[Source: K. Yu]

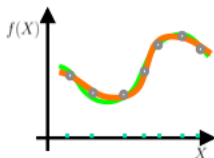
Summary of coding algorithms

- All lead to higher-dimensional, sparse, and localized coding
- All explore geometric structure of data
- New coding methods are suitable for linear classifiers.
- Their implementations are quite straightforward.

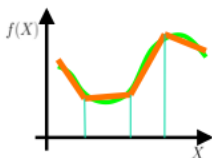
Vector Quantization
(BoW)



(Fast) Local Coordinate Coding



Super-vector Coding



[Source: K. Yu]

- No.1 for 18 of 20 categories
- PASCAL: 20 categories, 10,000 images
- They use only HOG feature on gray images

Classes	Ours	Best of Other Teams	Difference
Aeroplane	88.1	86.6	1.5
Bicycle	68.6	63.9	4.7
Bird	68.1	66.7	1.4
Boat	72.9	67.3	5.6
Bottle	44.2	43.7	0.5
Bus	79.5	74.1	5.4
Car	72.5	64.7	7.8
Cat	70.8	64.2	6.6
Chair	59.5	57.4	2.1
Cow	53.6	46.2	7.4
Diningtable	57.5	54.7	2.8
Dog	59.3	53.5	5.8
Horse	73.1	68.1	5.0
Motorbike	72.3	70.6	1.7
Person	85.3	85.2	0.1
Pottedplant	36.6	39.1	-2.5
Sheep	56.9	48.2	8.7
Sofa	57.9	50.0	7.9
Train	86.0	83.4	2.6
Tvmonitor	68.0	68.6	-0.6

ImageNet Large-scale Visual Recognition Challenge 2010



ImageNet: 1000 categories, 1.2 million images for training

[Source: K. Yu]

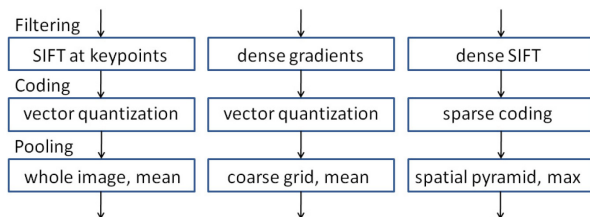
ImageNet Large-scale Visual Recognition Challenge 2010

- 150 teams registered worldwide, resulting 37 submissions from 11 teams
- SC achieved 52% for 1000 class classification.

Teams	Top 5 Hit Rate
Our team: NEC-UIUC	72.8%
Xerox European Lab, France	66.4%
Univ. of Tokyo	55.4%
Univ. of California, Irvine	53.4%
MIT	45.6%
NTU, Singapore	41.7%
LIG, France	39.3%
IBM T. J. Waston Research Center	30.0%
National Institute of Informatics, Tokyo	25.8%
SRI (Stanford Research Institute)	24.9%

[Source: K. Yu]

Learning Codebooks for Image Classification



Replacing Vector Quantization by Learned Dictionaries

- unsupervised: [Yang et al., 2009]
- supervised: [Boureau et al., 2010, Yang et al., 2010]

[Source: Mairal]

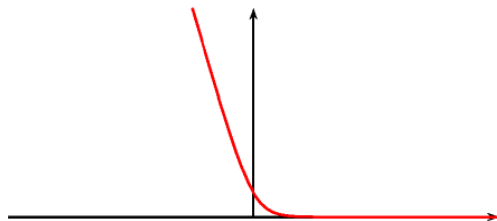
Discriminative Training

“Discriminative” training

[Mairal, Bach, Ponce, Sapiro, and Zisserman, 2008a]

$$\min_{\mathbf{D}_-, \mathbf{D}_+} \sum_i \mathcal{C} \left(\lambda z_i (\mathbf{R}^*(\mathbf{x}_i, \mathbf{D}_-) - \mathbf{R}^*(\mathbf{x}_i, \mathbf{D}_+)) \right),$$

where $z_i \in \{-1, +1\}$ is the label of \mathbf{x}_i .



Logistic regression function

[Source: Mairal]

Discriminative Dictionaries

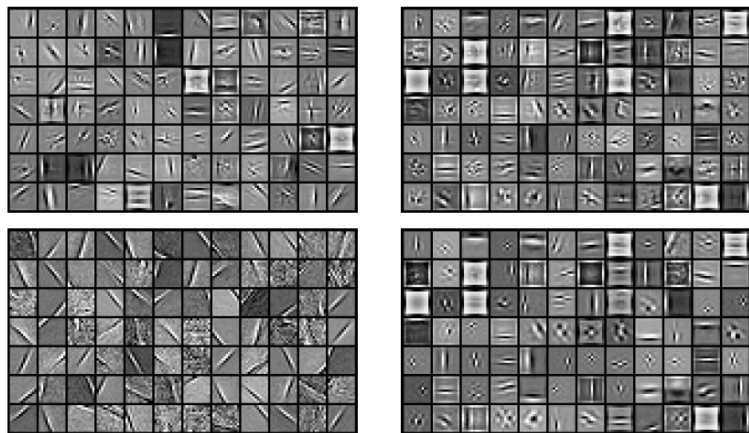


Figure: Top: reconstructive, Bottom: discriminative, Left: Bicycle, Right: Background

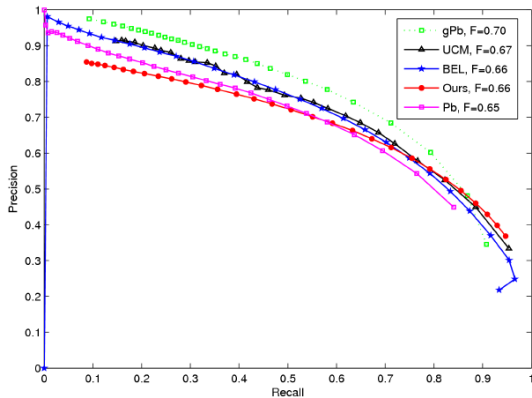
[Source: Mairal]

Application: Edge detection



[Source: Mairal]

Application: Edge detection



[Source: Mairal]

Application: Authentic from fake

Authentic



Fake



Fake

[Source: Mairal]

Predictive Sparse Decomposition (PSD)

- Feed-forward predictor function for feature extraction

$$\min_{a, \phi, K} \sum_{i=1}^m \left(\|x^{(i)} - \sum_{j=1}^k a_j^{(i)} \phi_j\|^2 + \lambda \sum_{j=1}^k |a_j^{(i)}| + \beta \|a - C(X, K)\|_2^2 \right)$$

with e.g., $C(X, K) = g \cdot \tanh(x * k)$

Learning is done by

- 1) Fix K and a , minimize to get optimal ϕ
- 2) Update a and K using optimal ϕ
- 3) Scale elements of ϕ to be unit norm.

[Source: Y. LeCun]

Predictive Sparse Decomposition (PSD)

- 12 x 12 natural image patches with 256 dictionary elements

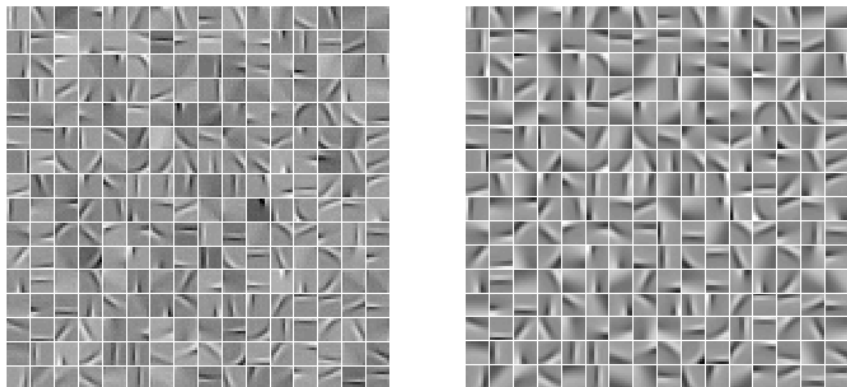


Figure: (Left) Encoder, (Right) Decoder

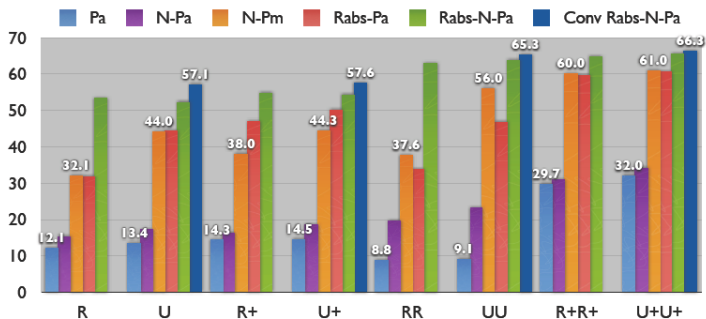
[Source: Y. LeCun]

Training Deep Networks with PSD

- Train layer wise [Hinton, 06]
 - $C(X, K^1)$
 - $C(f(K^1), K^2)$
 - ...
- Each layer is trained on the output $f(x)$ produced from previous layer.
- f is a series of non-linearity and pooling operations

[Source: Y. LeCun]

Object Recognition - Caltech 101



	R	RR	U	UU
Unsupervised	✗	✗	✓	✓
Random	✓	✓	✗	✗
Supervised	R+	R+ R+	U+	U+ U+

[Source: Y. LeCun]

Problems with sparse coding

- Sparse coding produces filters that are shifted version of each other
- It ignores that it's going to be used in a convolutional fashion
- Inference in all overlapping patches independently

Problems with sparse coding

- 1) the representations are redundant, as the training and inference are done at the patch level
- 2) inference for the whole image is computationally expensive

Solutions via Convolutional Nets

Problems

- 1) the representations are redundant, as the training and inference are done at the patch level
- 2) inference for the whole image is computationally expensive

Solutions

- 1) Apply sparse coding to the entire image at once, with the dictionary as a convolutional filter bank

$$\ell(x, z, D) = \frac{1}{2} \|x - \sum_{k=1}^K D_k * z_k\|_2^2 + |z|_1$$

with D_k a $s \times s$ 2D filter, x is $w \times h$ image and z_k a 2D $(w + s - 1) \times (h + s - 1)$ feature.

- 2) Use feed-forward, non-linear encoders to produce a fast approx. to the sparse code

$$\ell(x, z, D, W) = \frac{1}{2} \|x - \sum_{k=1}^K D_k * z_k\|_2^2 + \sum_{k=1}^K \|z_k - f(W^k * x)\|_2^2 + |z|_1$$

Convolutional Nets

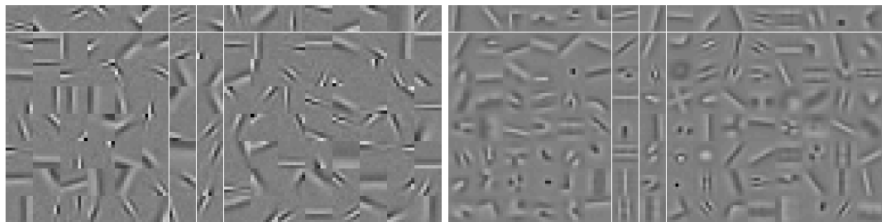


Figure: Image from Kavukcuoglu et al. 10. (left) Dictionary from sparse coding. (right) Dictionary learned from conv. nets.

Stacking

- One or more additional stages can be stacked on top of the first one.
- Each stage then takes the output of its preceding stage as input and processes it using the same series of operations with different architectural parameters like size and connections.

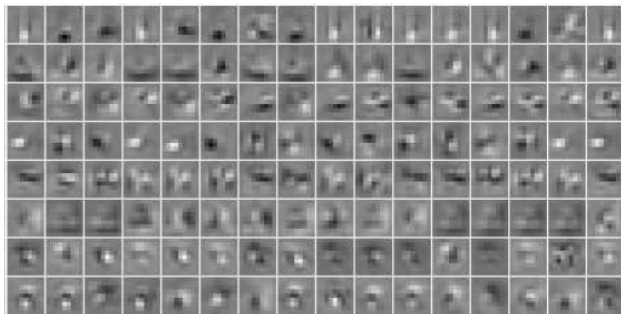
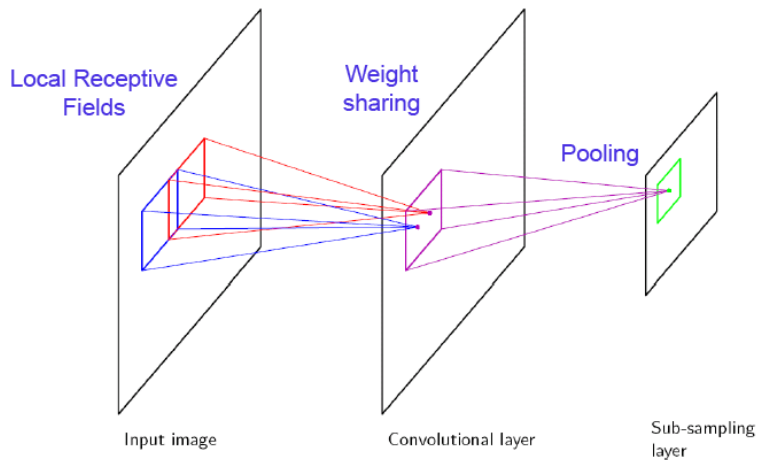


Figure: Image from Kavukcuoglu et al. 10. Second layer.

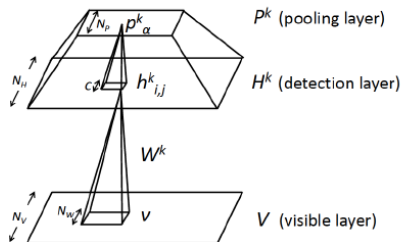
Convolutional Nets



[Source: Y. LeCun]

Convolutional Nets

Convolutional RBM: Generative training of convolutional structures (with probabilistic max-pooling)



faces, cars, airplanes, motorbikes



(Lee et al, 2009; Desjardins and Bengio, 2008; Norouzi et al., 2009)

[Source: H. Lee]

INRIA pedestrians

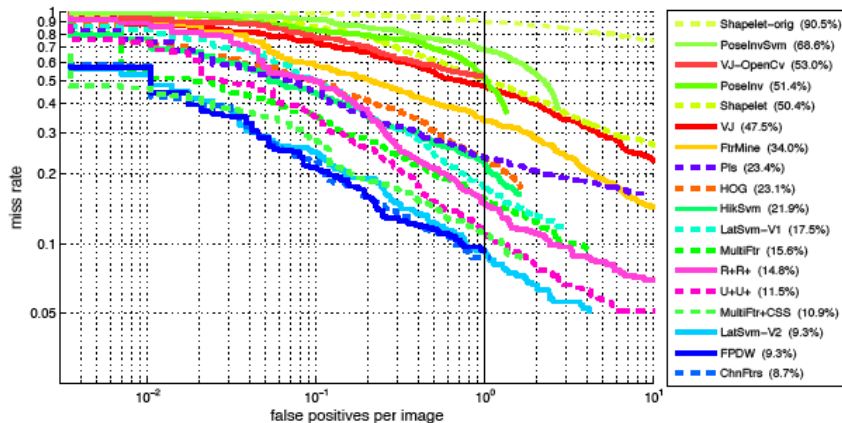


Figure: Image from Kavukcuoglu et al. 10. Second layer.

Many more deep architectures

- Restricted Boltzmann machines (RBMs)
- Deep belief nets
- Deep Boltzmann machines
- Stacked denoising auto-encoders
- ...

Topic Models

Topic Models

- Topic models have become a powerful **unsupervised learning** tool for text summarization, document classification, information retrieval, link prediction, and collaborative filtering.
- Topic modeling introduces latent topic variables in text and image that reveal the underlying structure.
- Topics are a group of related words

	Topics
1	learn kernel model reinforc algorithm machin classif
2	model learn network neural bayesian time visual
3	retriev inform base model text queri system
4	model imag motion track recognit object estim
5	imag base model recognit segment object detect
1	cell protein express gene activ mutat signal
2	pcr assai detect dna method probe specif
3	apo diseas allele alzheim associ onset gene
4	cell express gene tumor apoptosi protein cancer
5	mutat gene apc cancer famili protein diseas

[Source: J. Zen]

Probabilistic Topic Modeling

- Treat data as observations that arise from a generative model composed of latent variables.
 - The latent variables reflect the semantic structure of the document.
- Infer the latent structure using posterior inference (MAP):
 - What are the topics that summarize the document network?
- Predict new data by the estimated topic model
 - How the new data fit into the estimated topic structures?

[Source: J. Zen]

ABSTRACT

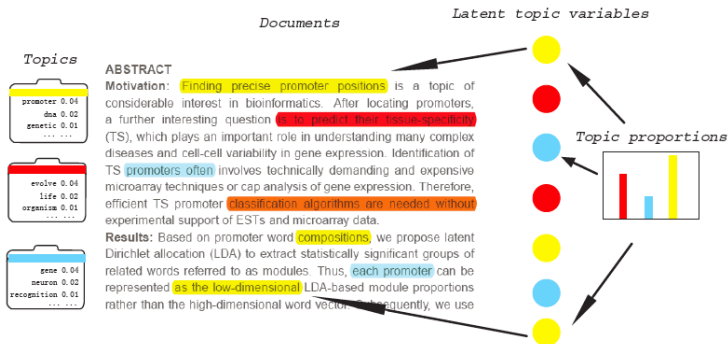
Motivation: Finding precise promoter positions is a topic of considerable interest in bioinformatics. After locating promoters, a further interesting question is to predict their tissue-specificity (TS), which plays an important role in understanding many complex diseases and cell-cell variability in gene expression. Identification of TS promoters often involves technically demanding and expensive microarray techniques or cap analysis of gene expression. Therefore, efficient TS promoter classification algorithms are needed without experimental support of ESTs and microarray data.

Results: Based on promoter word compositions, we propose latent Dirichlet allocation (LDA) to extract statistically significant groups of related words referred to as modules. Thus, each promoter can be represented as the low-dimensional LDA-based module proportions rather than the high-dimensional word vector. Subsequently, we use

- Simple intuition: a document exhibits multiple topics

[Source: J. Zen]

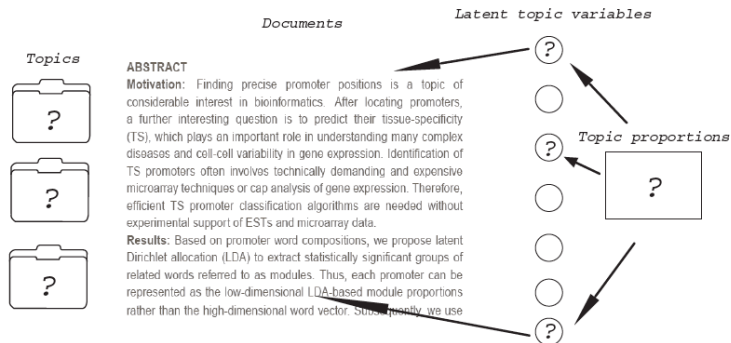
Generative models



- Each document is a mixture of corpus-wide topics.
- Each word is drawn from one of those topics.

[Source: J. Zen]

The posterior distribution (inference)

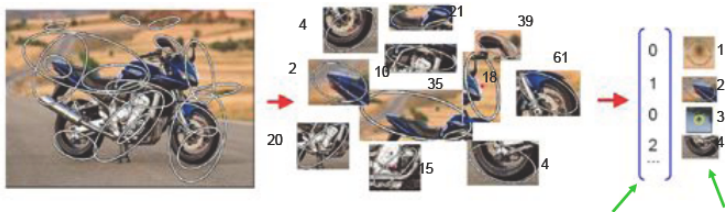


- Observations include documents and their words. Our goal is to infer the underlying topic structures marked by ? from observations.

[Source: J. Zen]

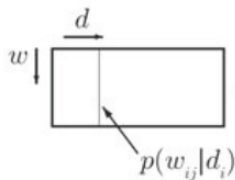
What does it mean with visual words

LDA model assumes exchangeability
Order of words does not matter



Stack visual word histograms
as columns in matrix

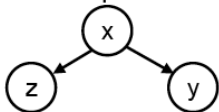
Throw away spatial information!



[Source: B. Schiele]

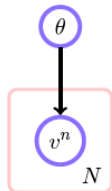
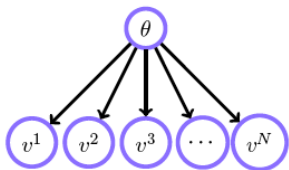
How to read Graphical Models

- Encoded independence assumption (all the arrows you don't see)

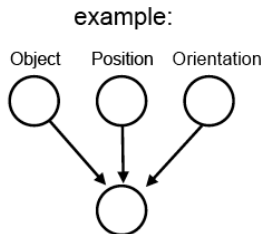


$$p(x, y, z) = p(z|x)p(y|x)p(x)$$

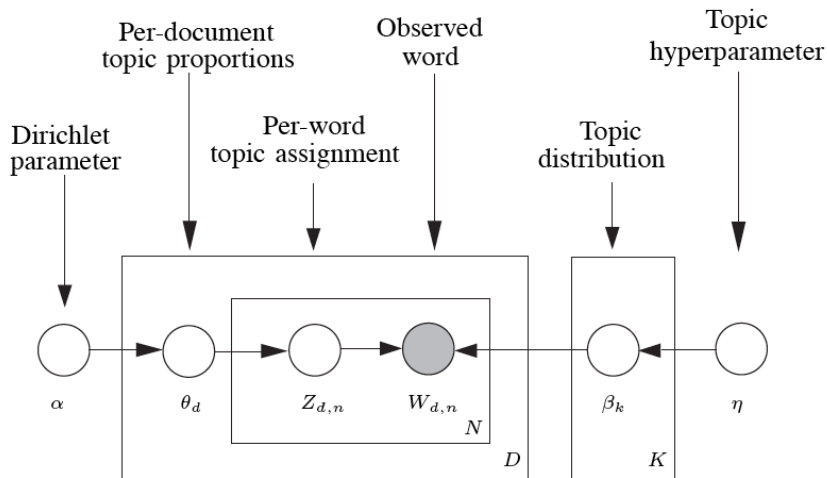
- Plate notation



- Observed vs. Unobserved

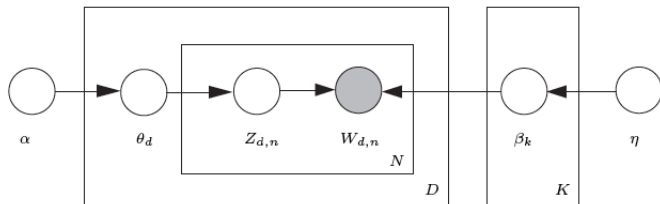


[Source: B. Schiele]



[Source: J. Zen]

LDA generative process

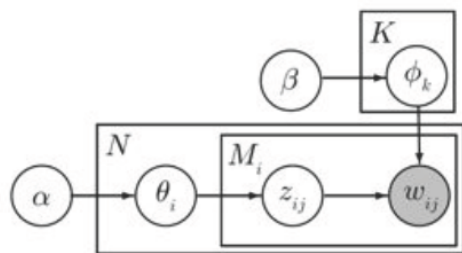


- Draw each topic $\beta_k \sim \text{Dir}(\eta)$, for $k \in \{1, \dots, K\}$.
- For each document:
 - ▶ Draw topic proportions $\theta_d \sim \text{Dir}(\alpha)$.
 - ▶ For each word:
 - ★ Draw $Z_{d,n} \sim \text{Mult}(\theta_d)$.
 - ★ Draw $W_{d,n} \sim \text{Mult}(\beta_{Z_{d,n}})$.

[Source: J. Zen]

LDA as matrix factorization

Blei, et al. 2003



w_{ij} - words

z_{ij} topic assignments

θ_i topic mixing weights

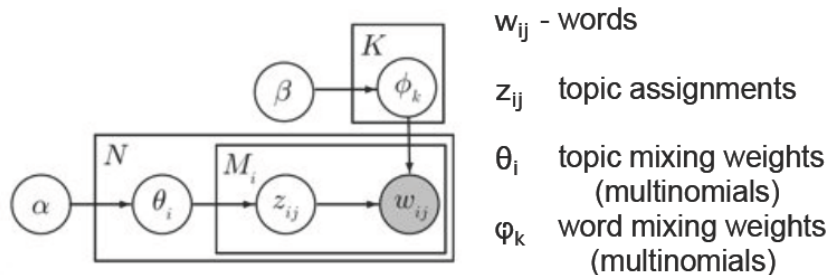
ϕ_k word mixing weights

$$p(w_{ij}) \propto \sum_{k=1}^K p(w_{ij}|z_{ij} = k, \phi_k) p(z_{ij} = k|\theta_i)$$

[Source: B. Schiele]

LDA as matrix factorization

Blei, et al. 2003



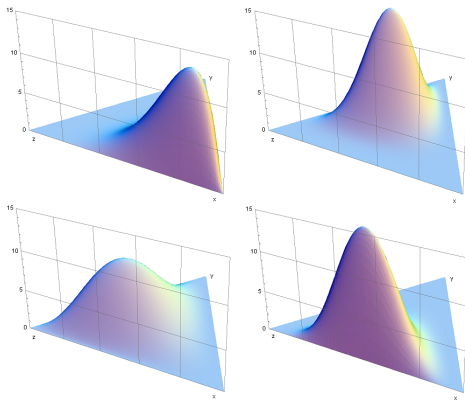
$$z_{ij} | \theta_i \sim \theta_i \quad \theta_i | \alpha \sim \text{Dirichlet}(\alpha)$$

$$w_{ij} | z_{ij} = k, \phi \sim \phi_k \quad \phi_k | \beta \sim \text{Dirichlet}(\beta)$$

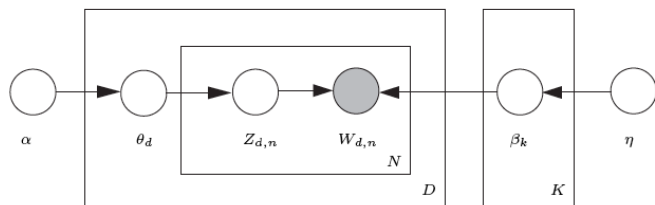
[Source: B. Schiele]

Dirichlet Distribution

$$\text{Dir}(\boldsymbol{\pi} | \boldsymbol{\theta}_c) = \frac{\Gamma\left(\sum_{i=1}^K \theta_{ci}\right)}{\prod_{i=1}^K \Gamma(\theta_{ci})} \pi_1^{(\theta_{c1}-1)} \dots \pi_K^{(\theta_{cK}-1)}$$



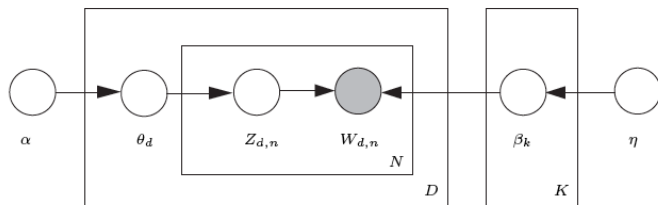
LDA inference



- From a collection of documents, infer
 - ▶ Per-document topic assignment $Z_{d,n}$.
 - ▶ Per-document topic proportions θ_d .
 - ▶ Per-corpus topic distribution β_k .
- Use posterior expectations $Z_{d,n}$, θ_d , and β_k to perform document classification and information retrieval.

[Source: J. Zen]

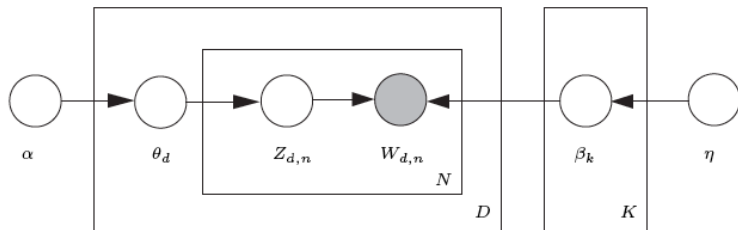
LDA: intractable exact inference



- $P(Z, \theta, \beta | W, \alpha, \eta) = P(W, Z, \theta, \beta | \alpha, \eta) / P(W | \alpha, \eta)$.
- $P(W | \alpha, \eta) = P(\beta | \eta) \int_{\theta} P(W | \theta, \beta) P(\theta | \alpha)$.
- Because θ and β are coupling, the integration is intractable.

[Source: J. Zen]

LDA: approximate inference

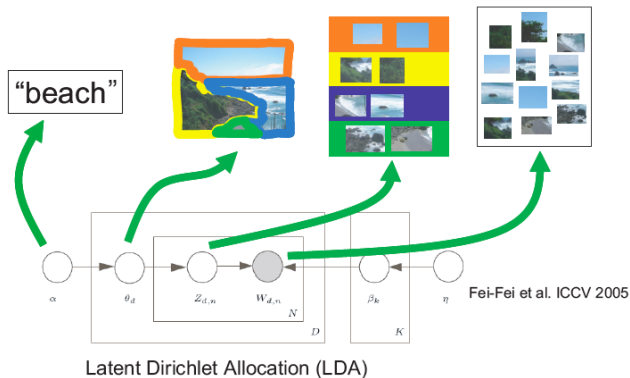


- Collapsed Gibbs sampling [Griffiths and Steyvers, 2004].
- Mean field variational inference [Blei et al., 2003].
- Expectation propagation [Minka and Lafferty, 2002].
- Collapsed variational inference [Teh et al., 2007].

[Source: J. Zen]

- Natural Scene Categorization [Fei-Fei and Perona, 05]
- Object Recognition [Sivic et al. 05]
- Object Recognition to model distribution on patches [Fritz et al.]
- Activity Recognition [Niebles et al.]
- Text and image [Saenko et al.]
- ...

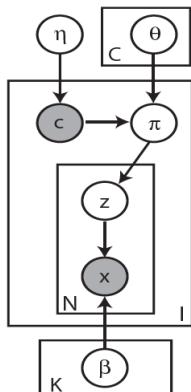
Natural Scene Categorization



- A different model is learned for each class.

[Source: J. Zen]

Natural Scene Categorization



$$p(x, z, \pi, c | \theta, \eta, \beta) = p(c | \eta) p(\pi | c, \theta) \cdot$$

$$\prod_{n=1}^N p(z_n | \pi) p(x_n | z_n, \beta)$$

$$p(c | \eta) = \text{Mult}(c | \eta)$$

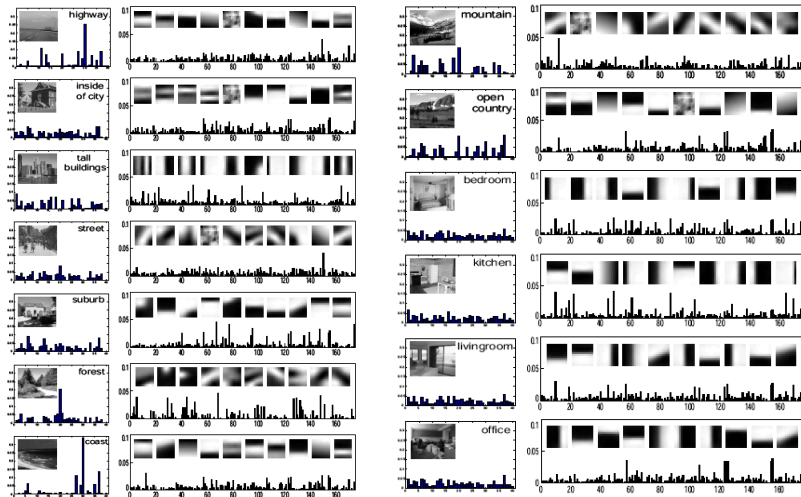
$$p(\pi | c, \theta) = \prod_{j=1}^C \text{Dir}(\pi | \theta_j)^{\delta(c, j)}$$

$$p(z_n | \pi) = \text{Mult}(z_n | \pi)$$

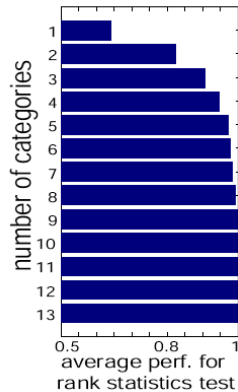
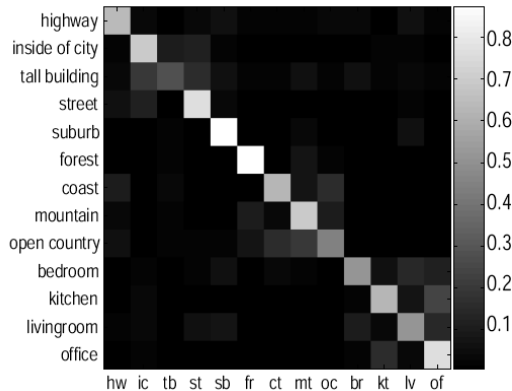
$$p(x_n | z_n, \beta) = \prod_{k=1}^K p(x_n | \beta_k)^{\delta(z_n^k, 1)}$$

- In reality is a bit more complicated as it reasons about class [Fei-Fei et al., 05].

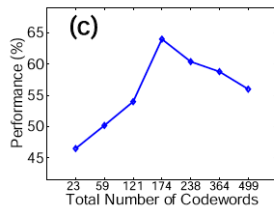
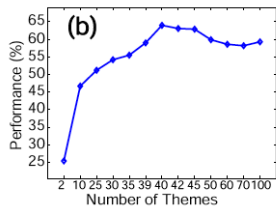
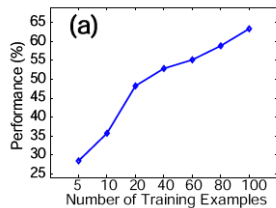
Natural Scene Categorization



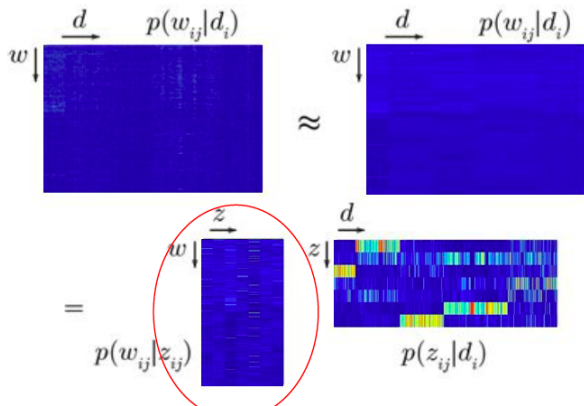
Classification Results



More Results



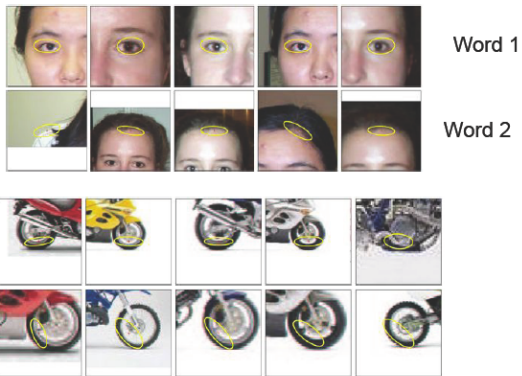
Object Recognition [Sivic et al. 05]



- Matrix Factorization view

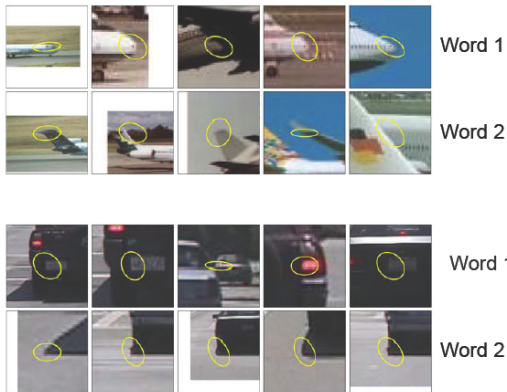
[Source: B. Schiele]

Most likely words given topic



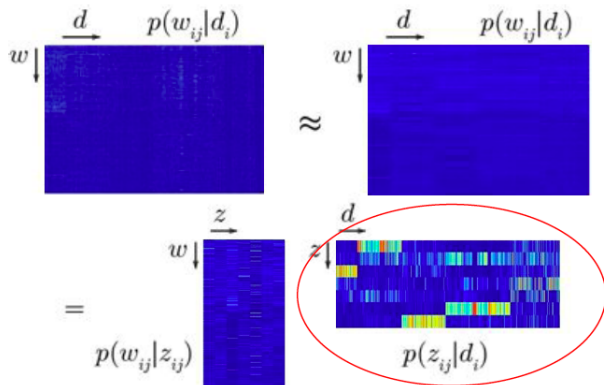
[Source: B. Schiele]

Most likely words given topic



[Source: B. Schiele]

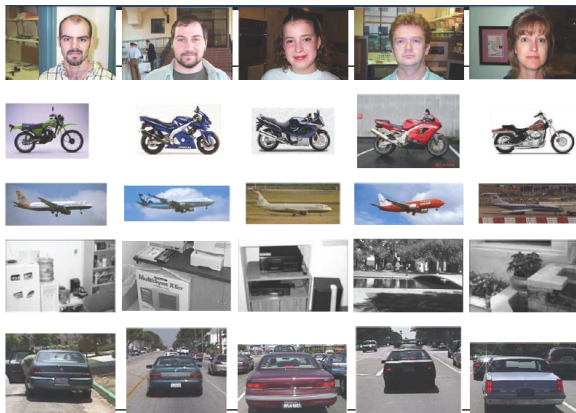
Object Recognition [Sivic et al. 05]



- Matrix Factorization view

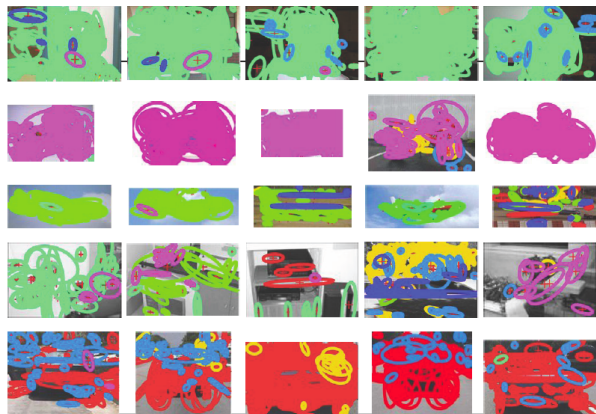
[Source: B. Schiele]

Object Recognition [Sivic et al. 05]



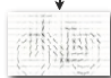
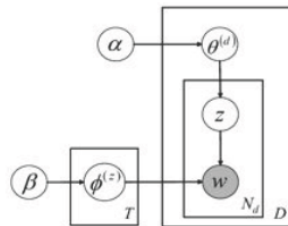
[Source: B. Schiele]

Object Recognition [Sivic et al. 05]



[Source: B. Schiele]

Object Recognition [Fritz et al.]



$$= \sum_{j=1}^T \theta_j * \text{[Image } j] + \dots$$

sparsity prior $Dirichlet(\alpha)$

sparsity prior $Dirichlet(\beta)$

[Source: B. Schiele]

Learning of Generative Decompositions [Fritz et al.]

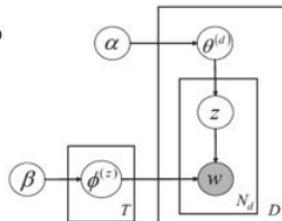
- Probabilistic Topic Model

- ▶ Document is composed (contains) a mixture of top
- ▶ Each topics generates a mixture of words
- ▶ (Latent-Dirichlet-Allocation Model [Blei'03] estimated by Gibbs Sampling [Griffiths'04])

$$P(w_i) = \sum_{j=1}^T P(w_i | z_i = j) P(z_i = j)$$

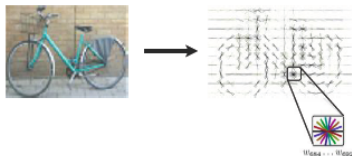
topic distribution for document d: $\theta^{(d)} = P(z)$

topic-word distribution for topic j: $\phi^{(j)} = P(w_i | z = j)$



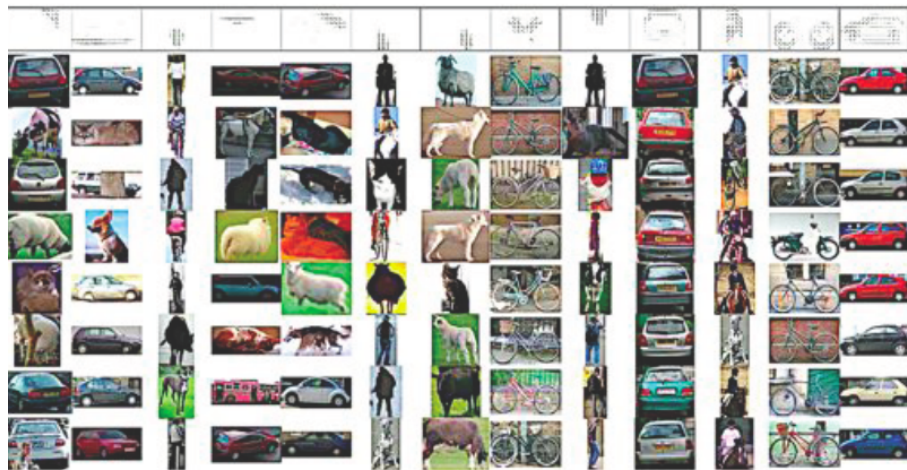
- Dense Object Representation:

- ▶ Document = Image of an Object
- ▶ Word = localized edge orientation



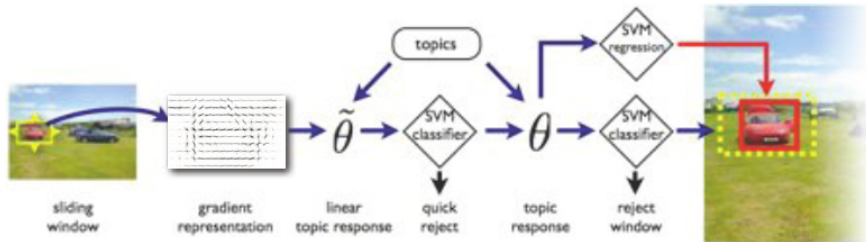
[Source: B. Schiele]

Topics [Fritz et al.]



[Source: B. Schiele]

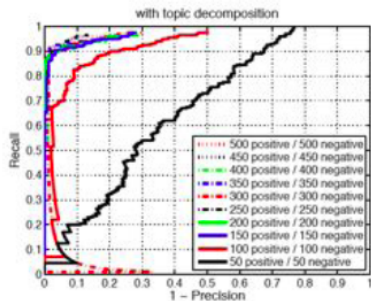
Supervised Classification [Fritz et al.]



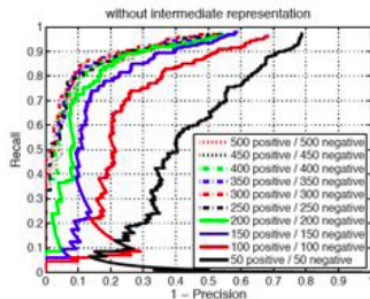
- **Generative/Discriminative Training**
 - ▶ **generative model:**
 - topic model decomposes objects into a mixture of topics
 - objects represented via topic distribution vector
 - ▶ **discriminative training**
 - SVM-classifier (RBF-kernel/Chi-Square kernel) using topic distribution vector

[Source: B. Schiele]

- SVM-Training using intermediate topic-representation:
 - ▶ maximal performance reached with 150 training samples



- SVM trained directly on oriented gradient histograms
 - ▶ maximal performance reached with 250 training samples
 - ▶ performance overall lower



[Source: B. Schiele]

- Introduce supervision, e.g., discriminative LDA, supervised LDA
- Introduce spatial information
- Multi-view approaches, e.g., correlated LDA, Pachenko
- Use infinite mixtures, e.g., HDP, PY processes to model heavy tails.
- And many more.