IMAGE REPRESENTATION

Xinyi Fan

COS598c Spring2014

IMAGE REPRESENTATION

Xinyi Fan

COS598c Spring2014

APPROACHES

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding

APPROACHES

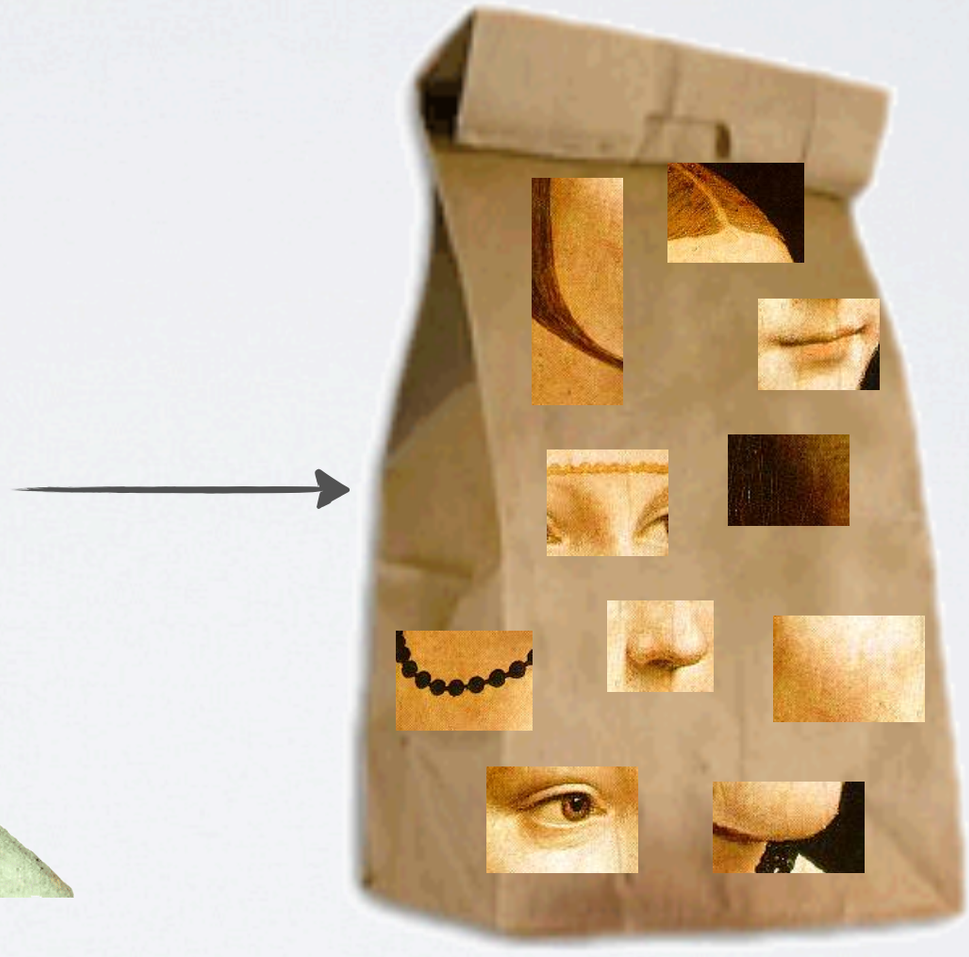
- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding

Object

Object ⟶ Feature Words

Object

Bag of
Feature Words



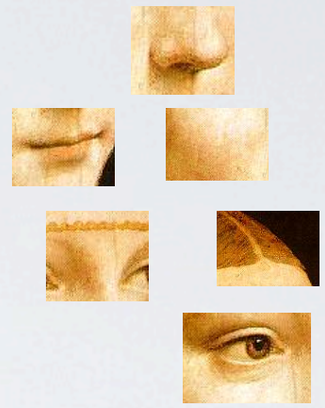
Adapted from slides by Fei-Fei Li

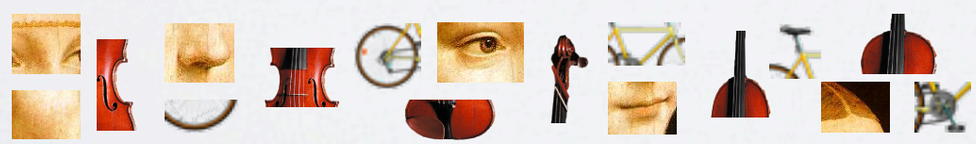
BAG OF WORDS

- Definition
  - Independent, orderless features

BAG OF WORDS

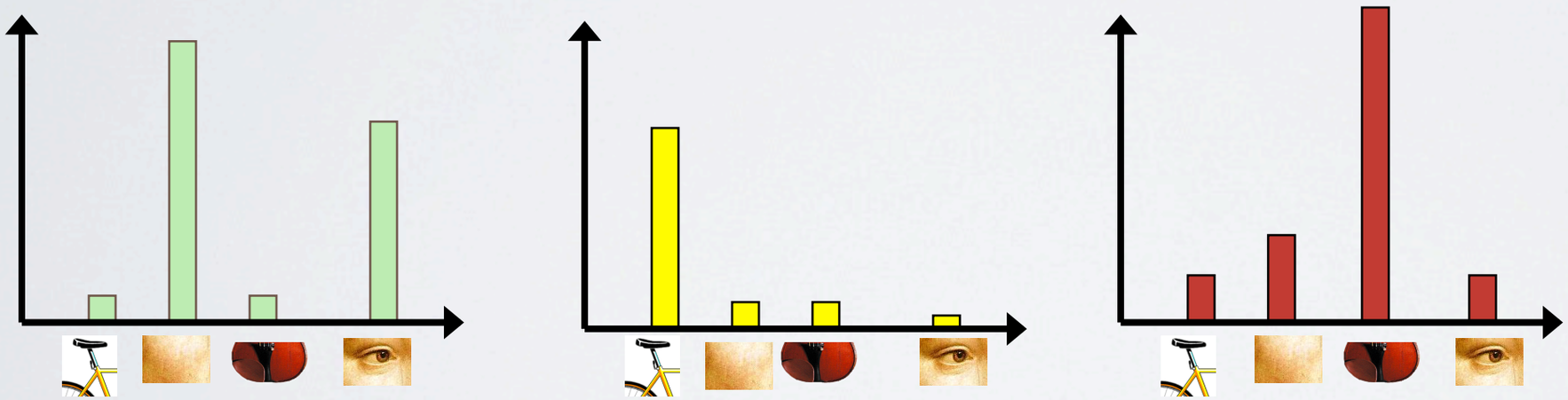
- Definition
  - Independent, orderless features





BAG OF WORDS

- Definition
  - Independent, orderless features
  - Histogram representation

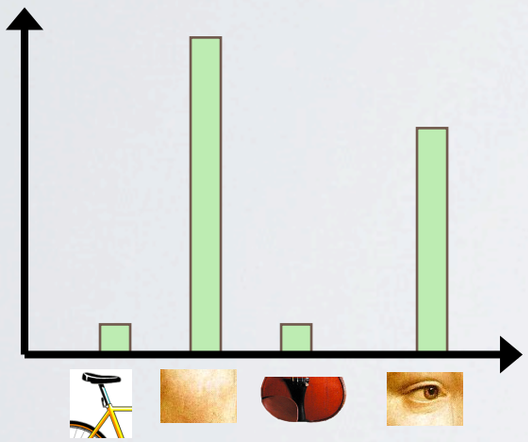


Adapted from slides by Fei-Fei Li

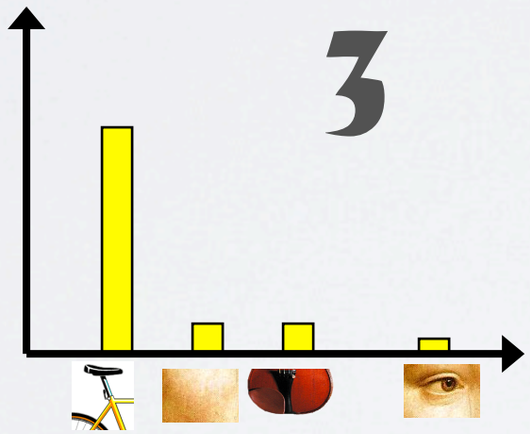
BAG OF WORDS

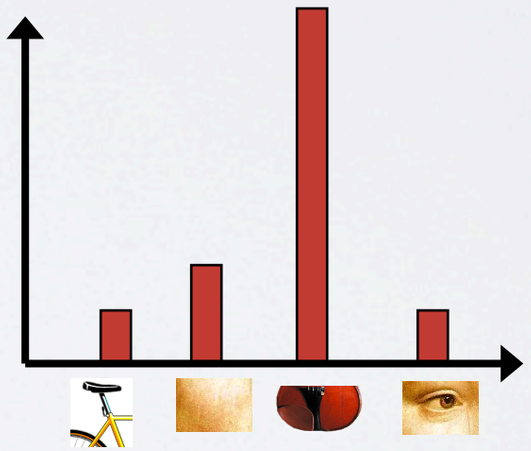
- Definition
  - Independent, orderless features
  - Histogram representation

1

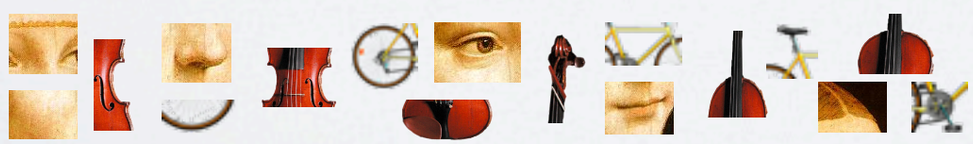


3





2

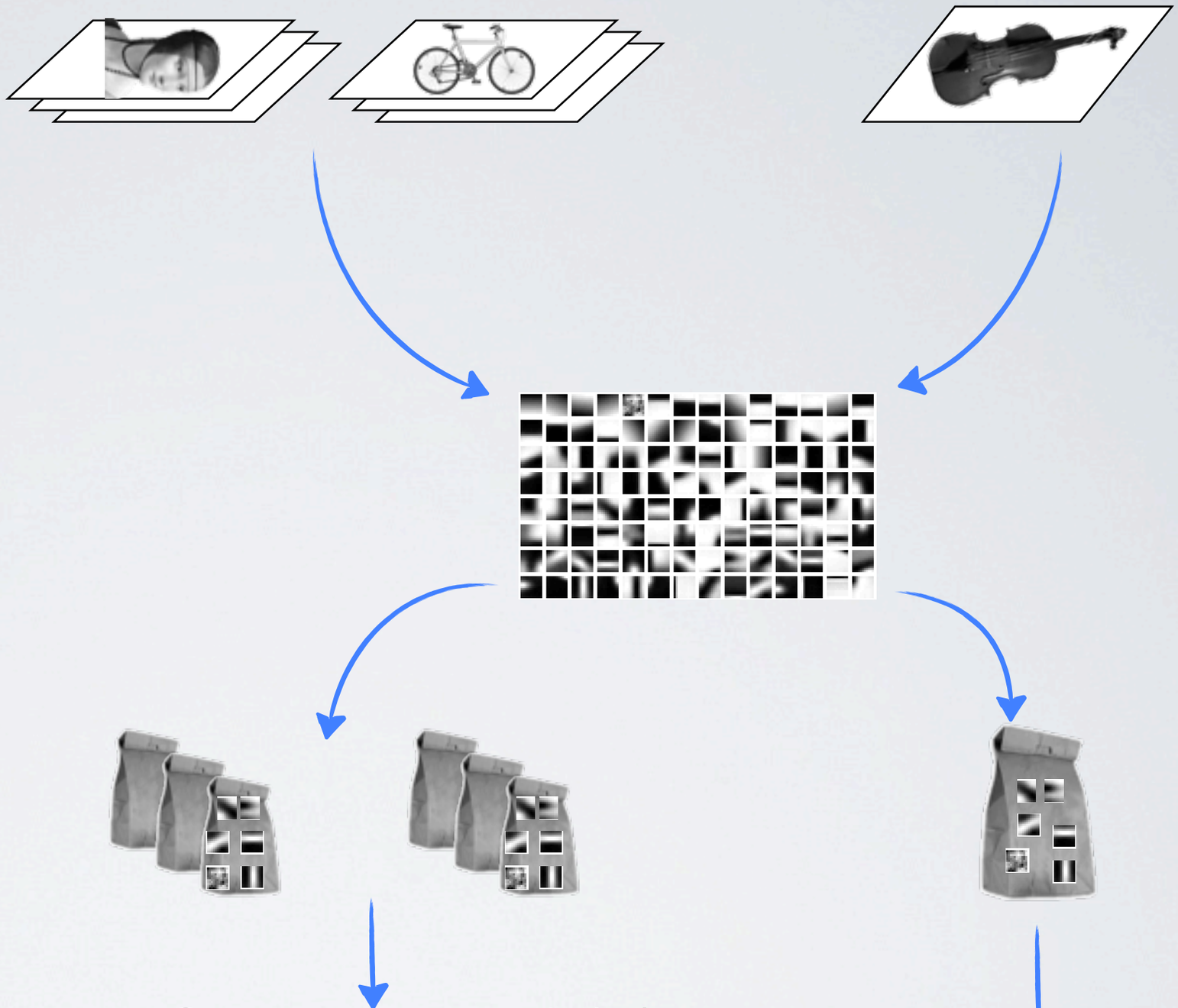


Adapted from slides by Fei-Fei Li

Feature detection & representation

Codewords dictionary formation

Image representation

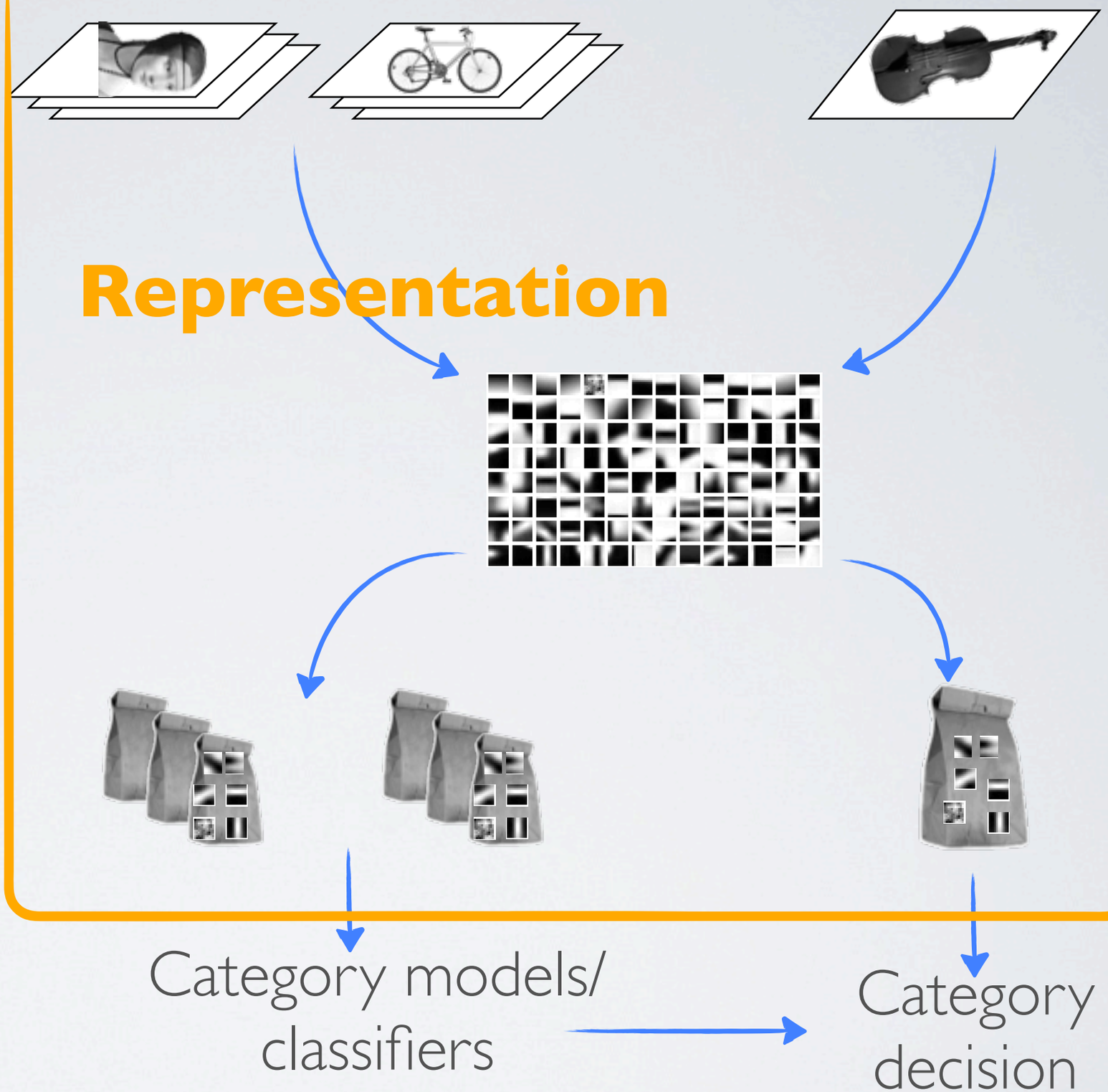


Adapted from slides by Fei-Fei Li

Feature detection & representation

Codewords dictionary formation

Image representation

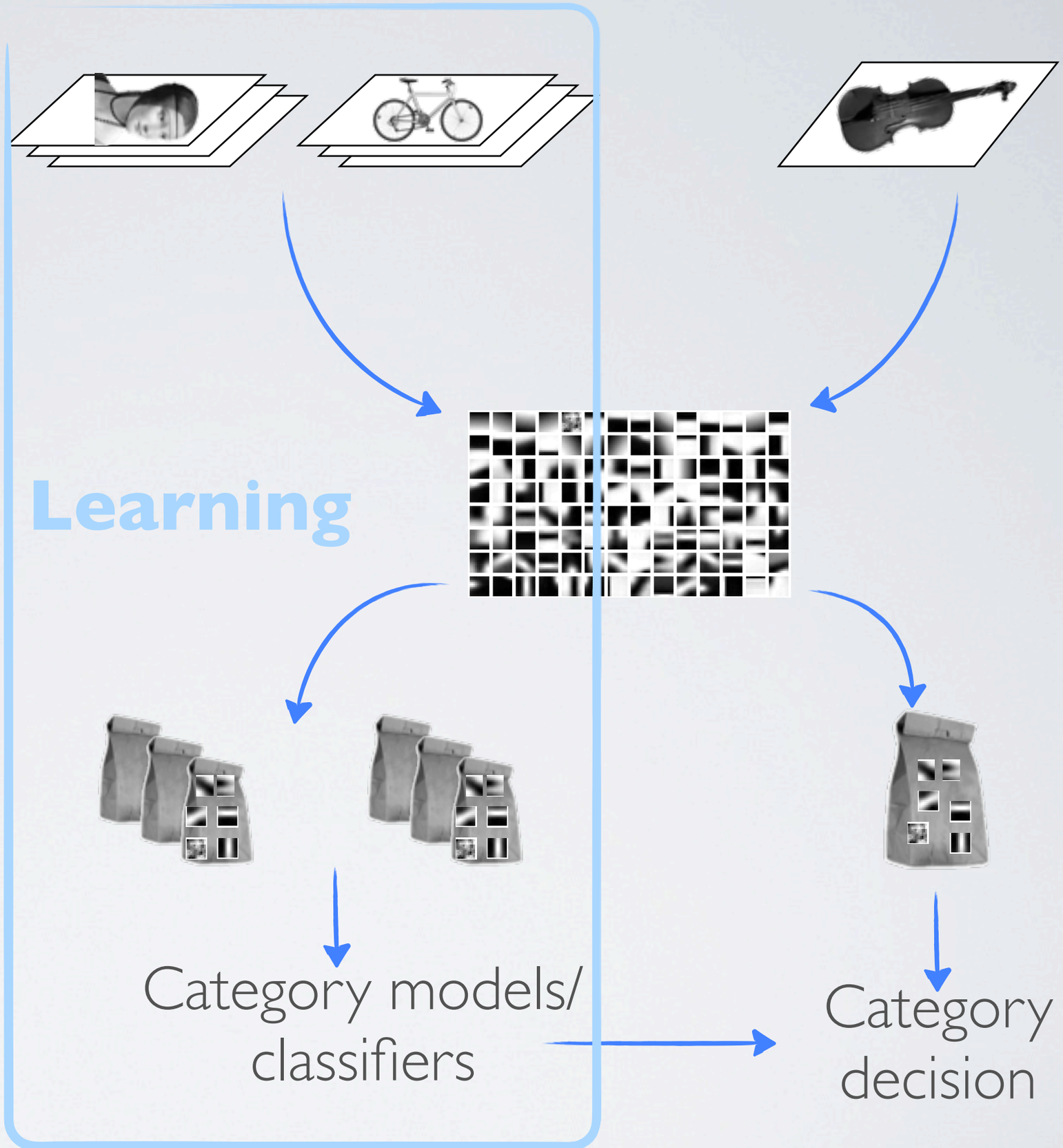


Adapted from slides by Fei-Fei Li

Feature detection & representation

Codewords dictionary formation

Image representation

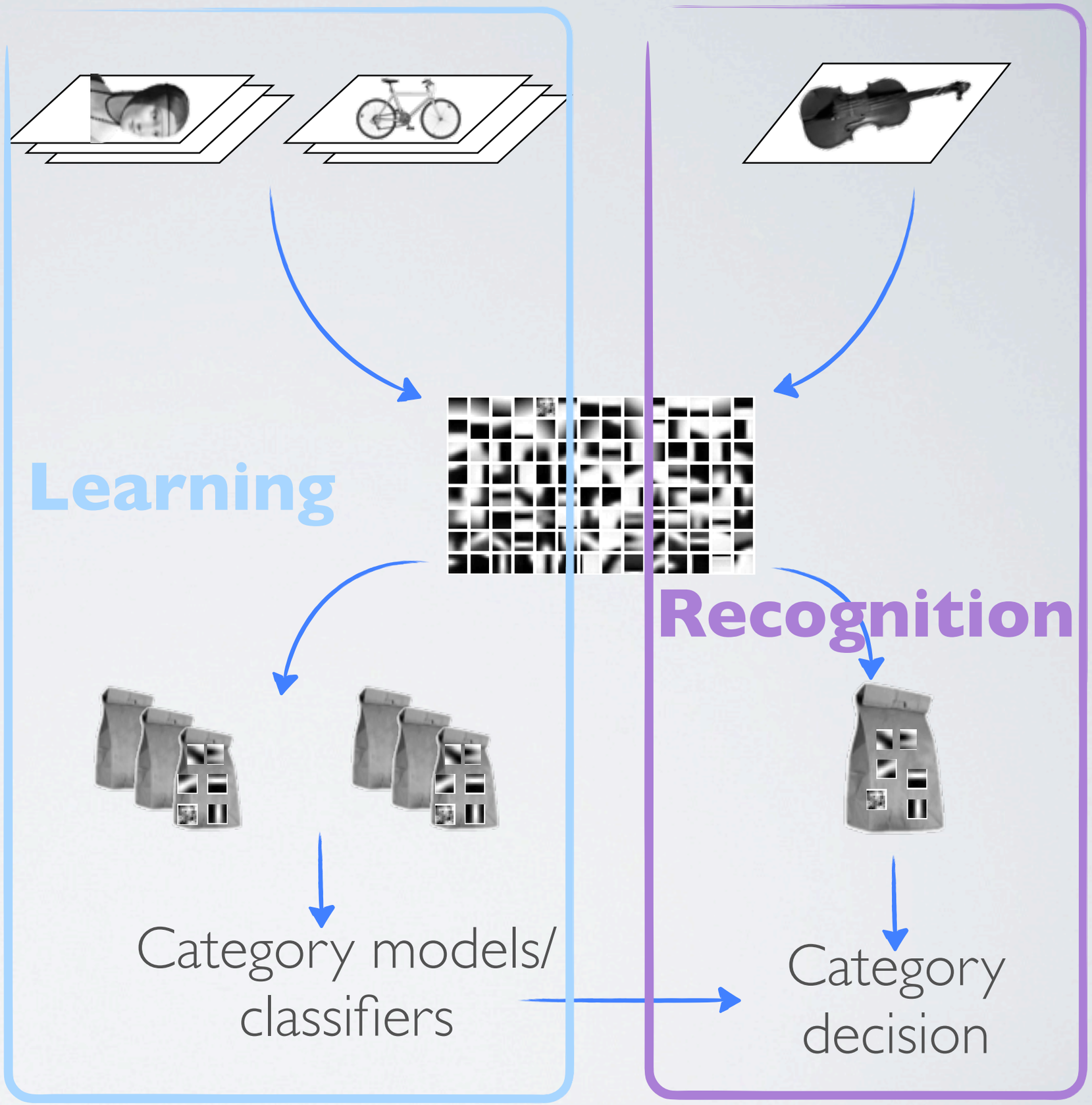


Adapted from slides by Fei-Fei Li

Feature detection & representation

Codewords dictionary formation

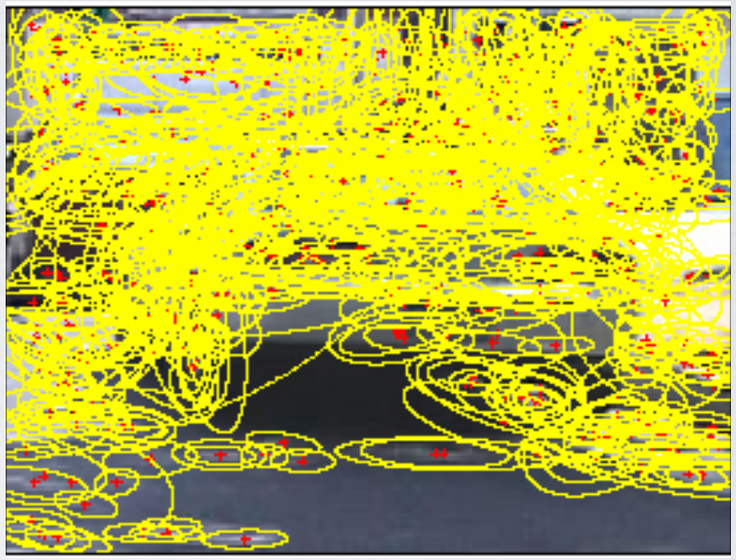
Image representation



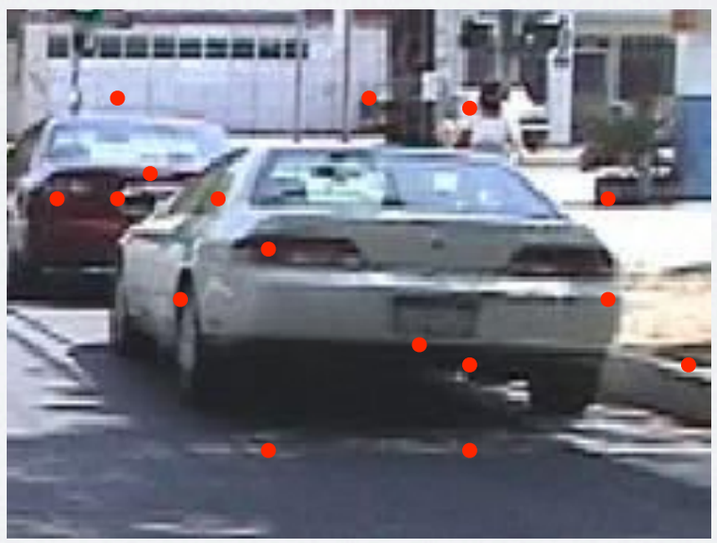
Adapted from slides by Fei-Fei Li

Feature Detection & Representation

dense regular grids

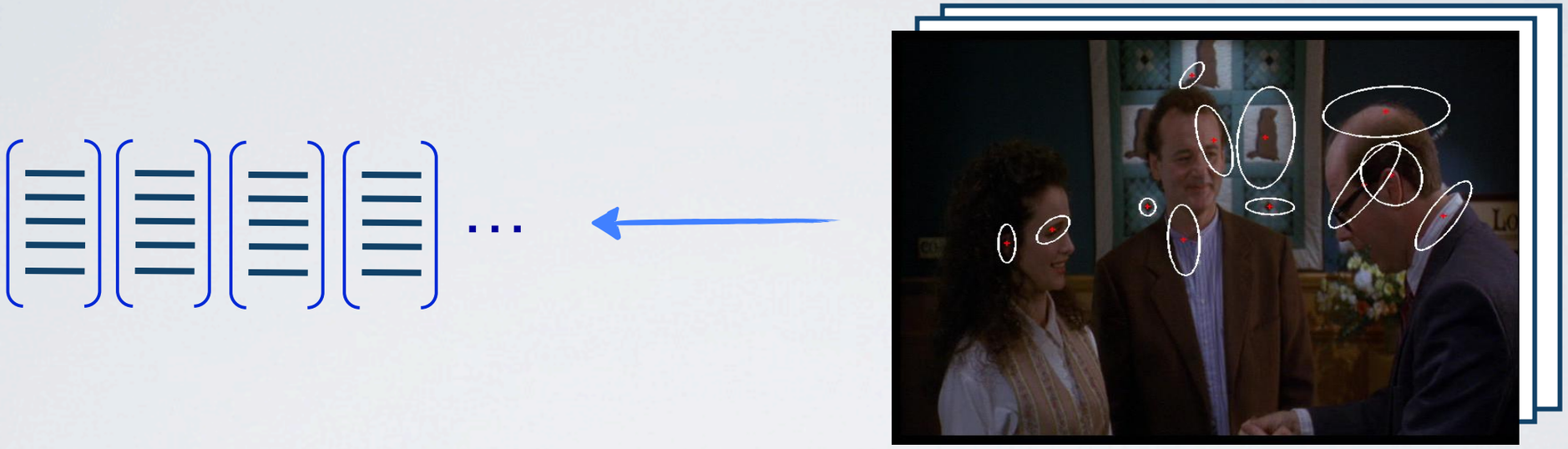


points of interest

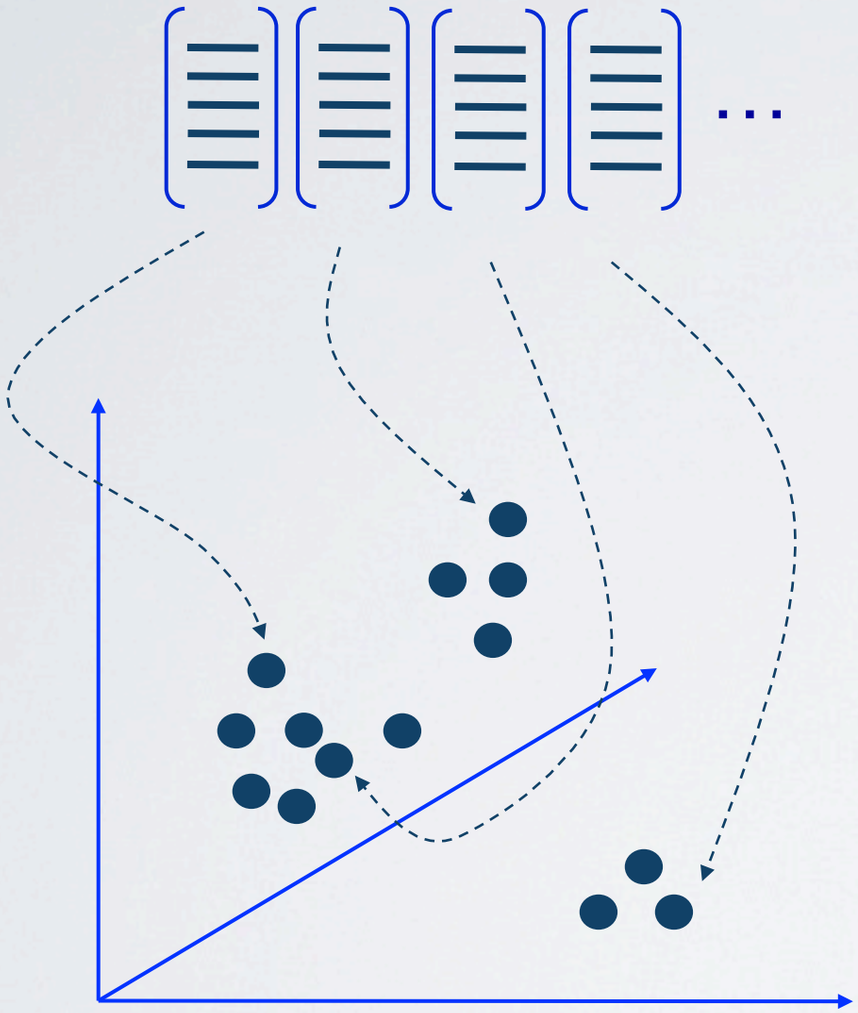


random

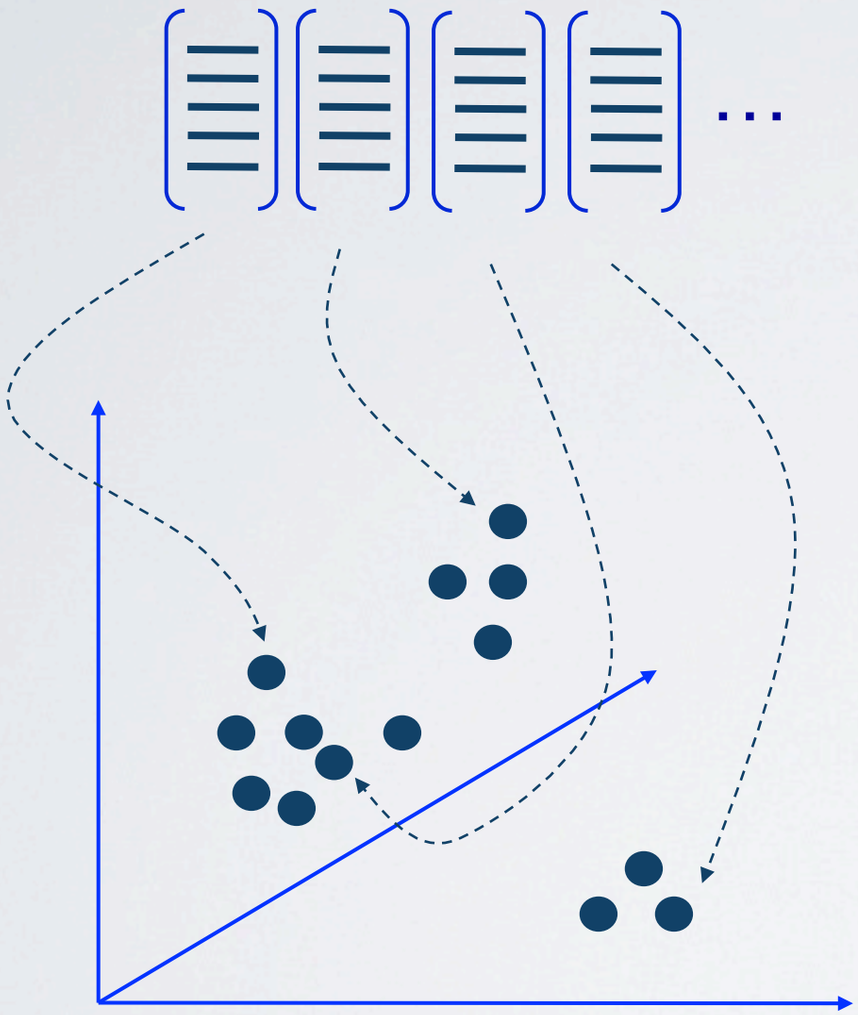
Feature Detection & Representation



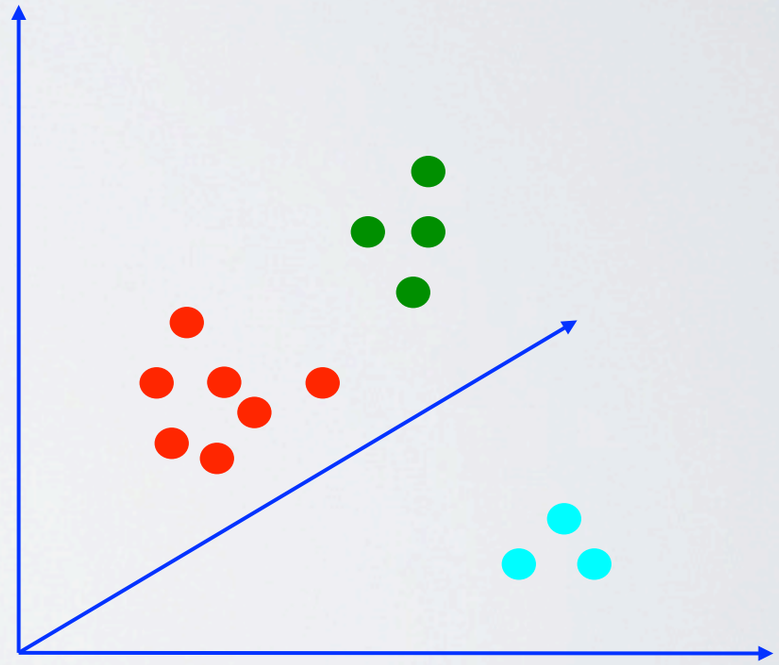
Codewords Dictionary Formation



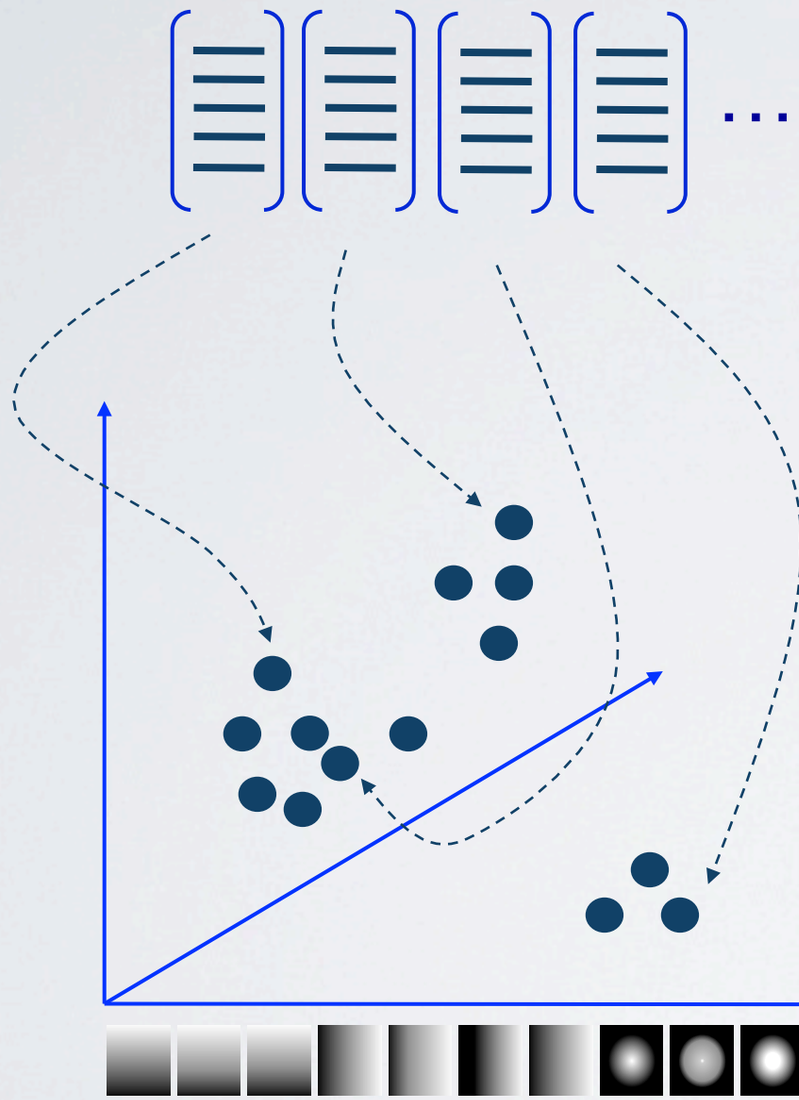
Codewords Dictionary Formation



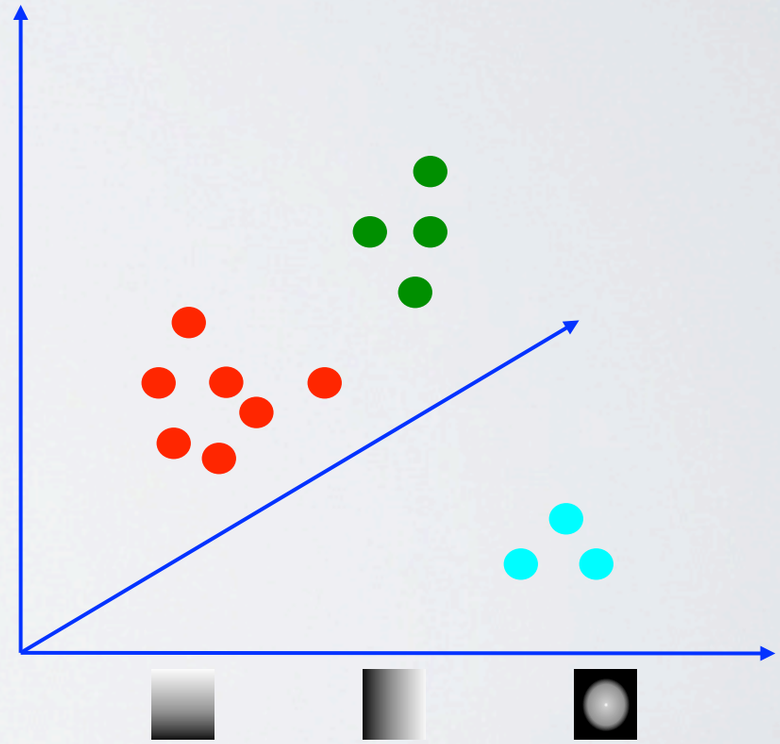
Vector quantization



Codewords Dictionary Formation



Vector quantization



Codewords Dictionary Formation

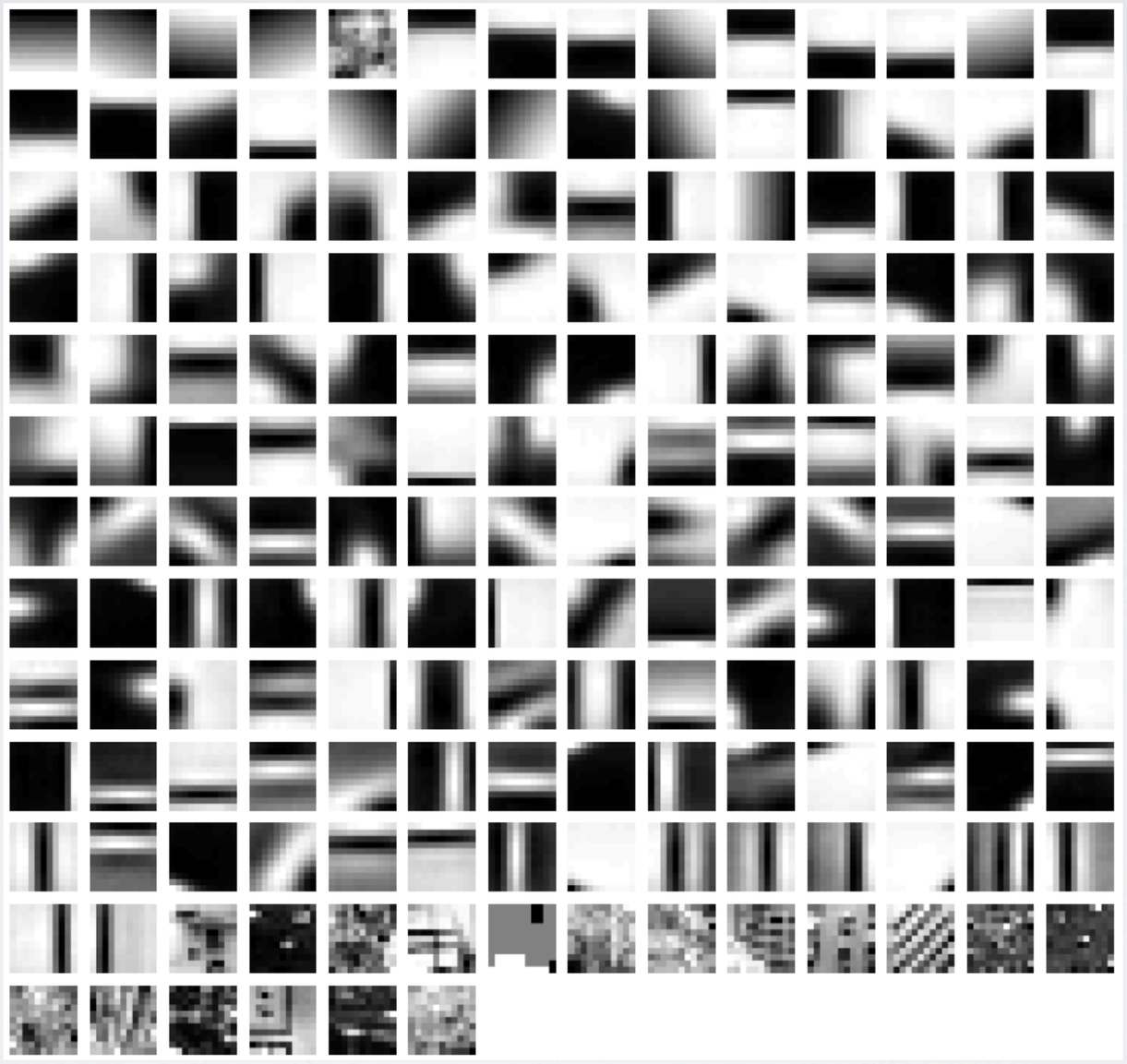
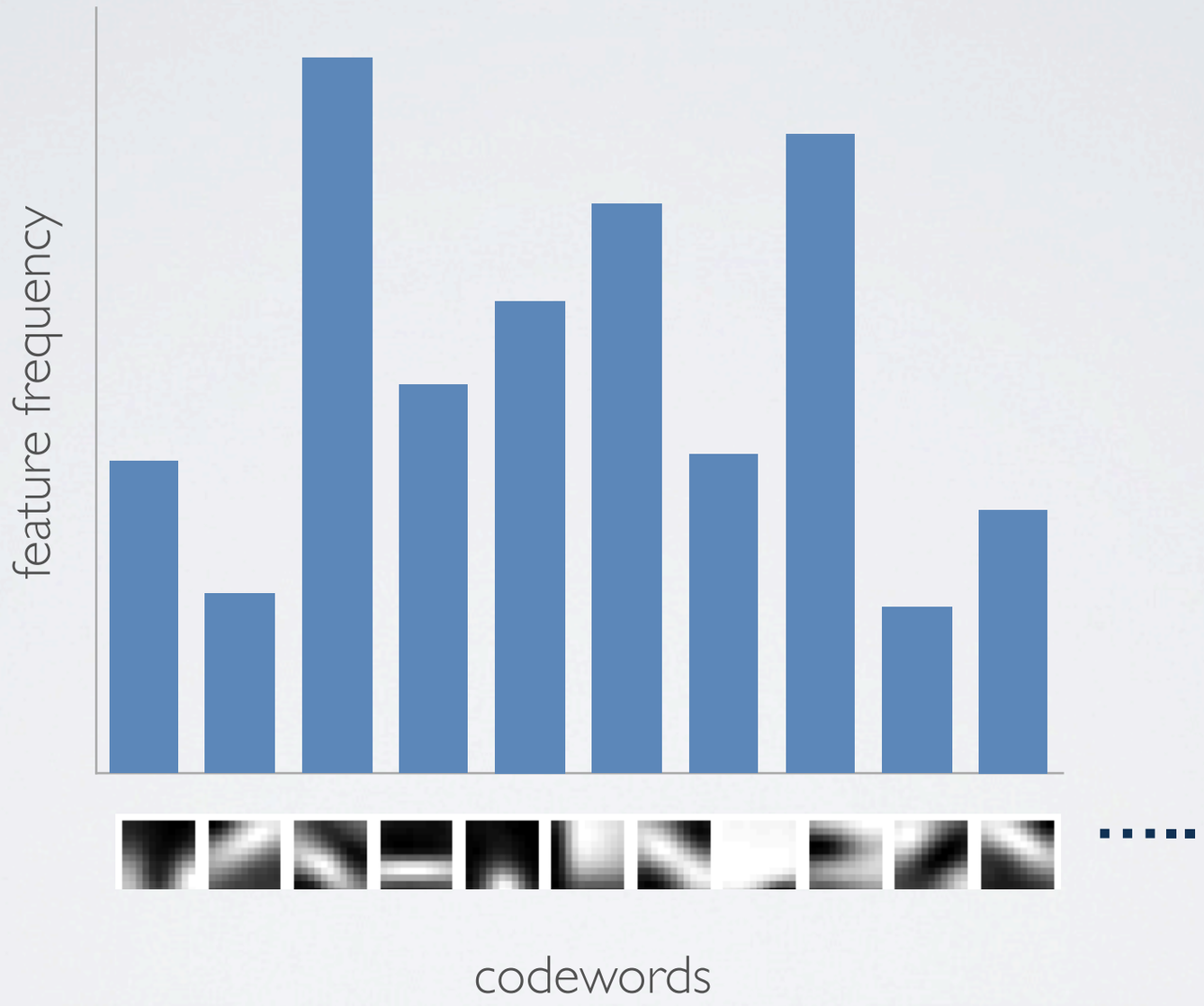


Image Representation



Using BoW Representation

Use BoW as feature vector for standard classifier

-Naive Bayesian

-SVM

Cluster BoW vectors over image collection

-Category classification (supervised)

-Object discovery (unsupervised)

Use BoW to build hierarchical models

- Decompose scene/object

BoW Summary

Issues

- Sampling strategy
  dense uniform, interest points, random...
- Codebook learning
  supervised/unsupervised, size...
- Similarity measurement
  SVM, Pyramid Matching
- Spatial information
- Scalability

BoW Summary

Issues

- Sampling strategy
  dense uniform, interest points, random...
- Codebook learning
  supervised/unsupervised, size...
- Similarity measurement
  SVM, Pyramid Matching
- Spatial information
- Scalability

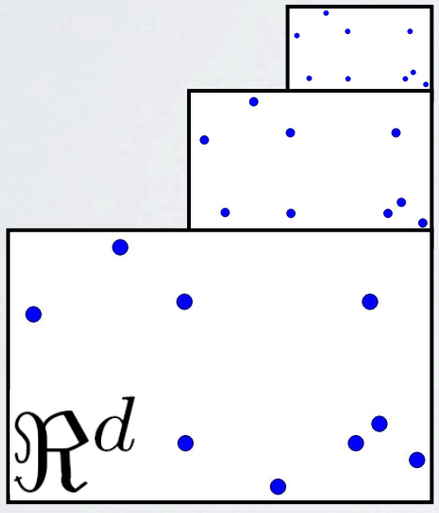
BoW Summary

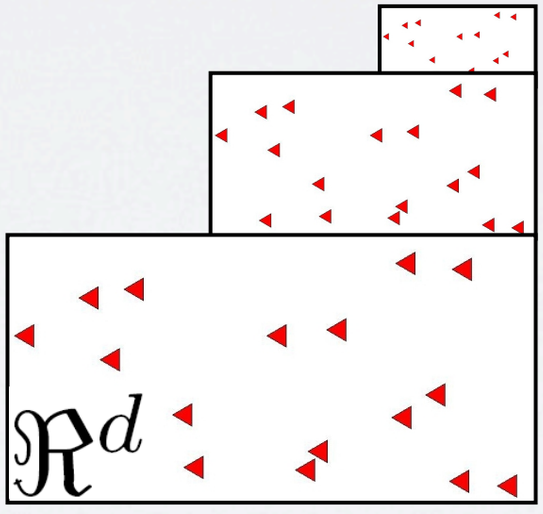
Issues

- Sampling strategy
  dense uniform, interest points, random...
- Codebook learning
  supervised/unsupervised, size...
- Similarity measurement
  SVM, Pyramid Matching
- Spatial information
- Scalability

Pyramid Matching

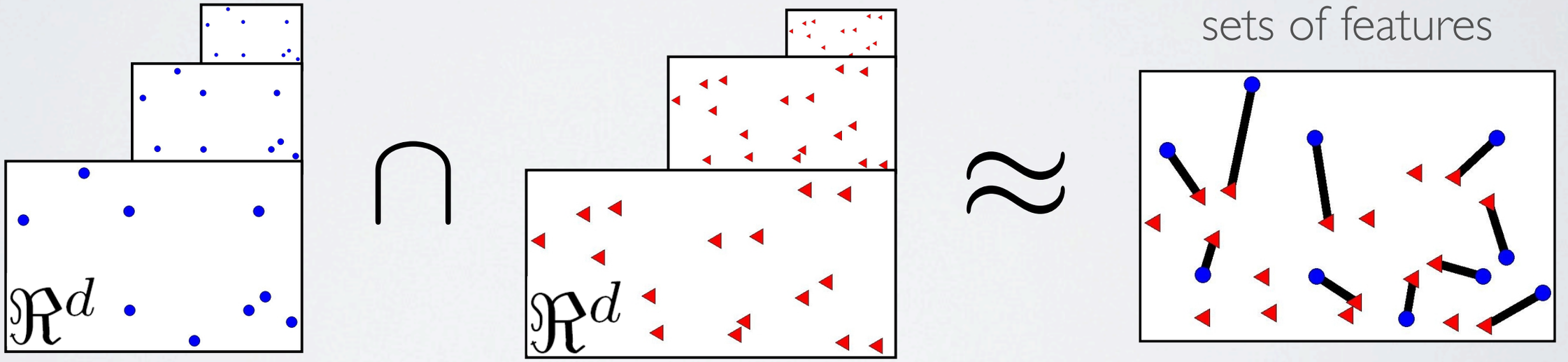
Pyramid Matching





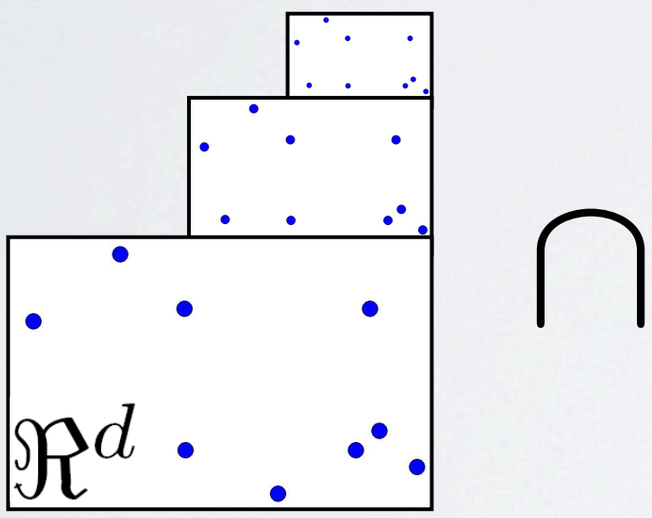
Pyramid Matching

optimal partial matching between sets of features

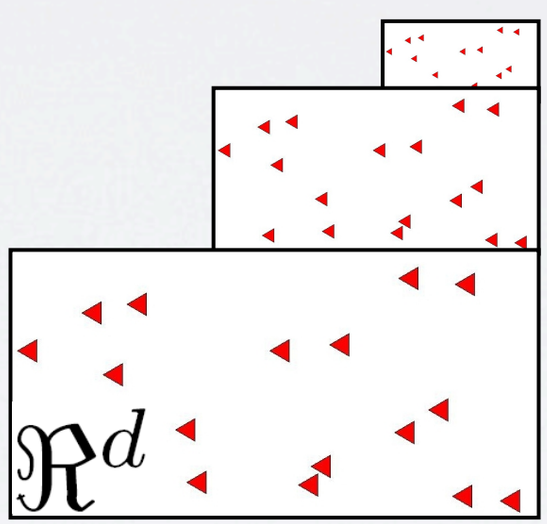


Pyramid Matching

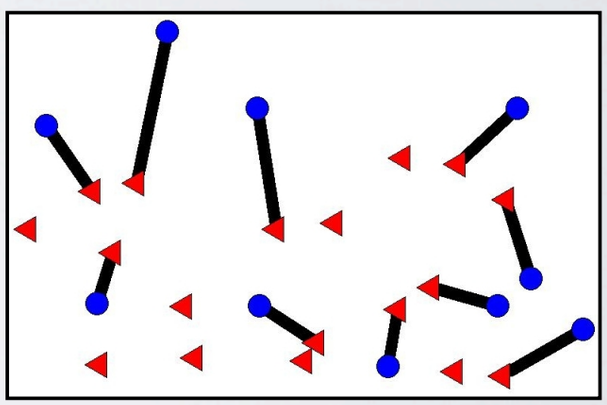
optimal partial matching between sets of features



$\cap$



$\approx$



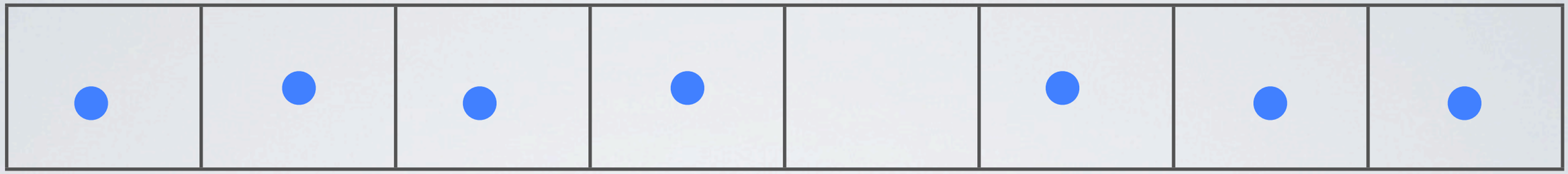
$X = \{\vec{x}_1, \ldots, \vec{x}_m\}$ $\quad Y = \{\vec{y}_1, \ldots, \vec{y}_n\}$

$\vec{x}_i \in \Re^d$ $\quad \vec{y}_i \in \Re^d$

Adapted from slides by Grauman and Darrell

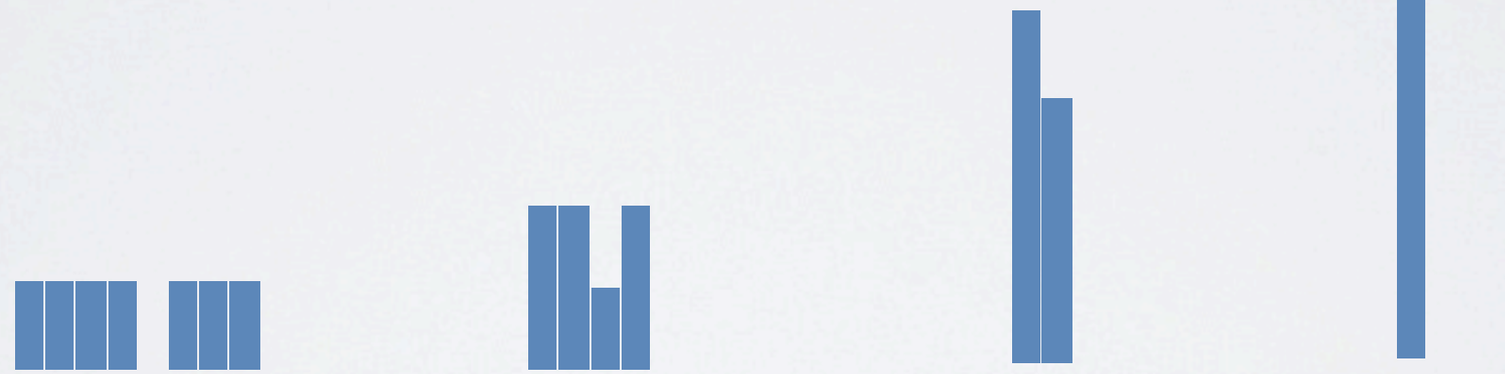
Feature Extraction

$$X = \{\vec{x}_1, \ldots, \vec{x}_m\} \quad \vec{x}_i \in \Re^d$$



d = 1, L = 3

Histogram pyramid:
level ℓ has bins of size 2^ℓ

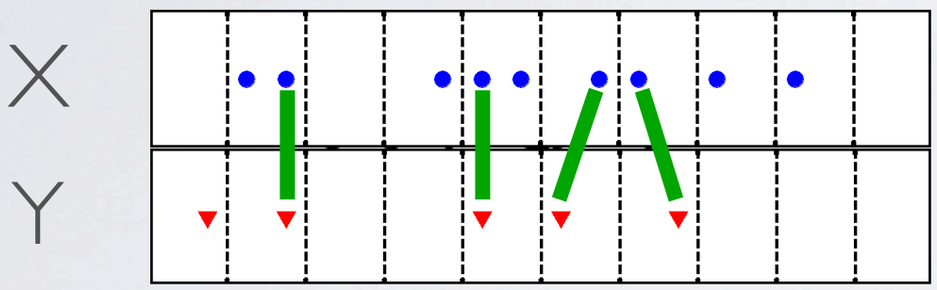


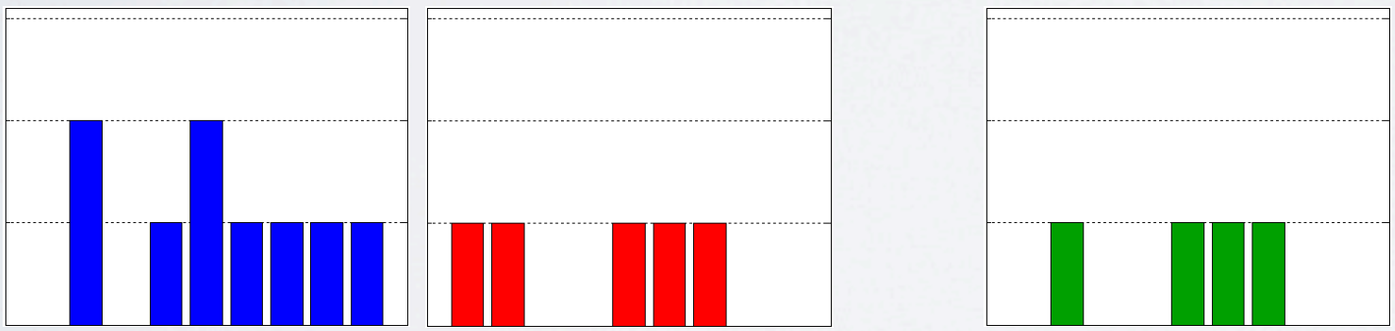
$H_0(X)$ $H_1(X)$ $H_2(X)$ $H_3(X)$

$$\Psi(X) = [H_0(X), \ldots, H_L(X)]$$

Counting Matches

Histogram intersection: $\mathcal{I}(H(X), H(Y)) = \sum_{j=1}^{r} \min(H(X)_j, H(Y)_j)$





$H(X)$ $H(Y)$ $\mathcal{I}(H(X), H(Y)) = 4$

Adapted from slides by Grauman and Darrell

Counting New Matches

Histogram intersection: $\mathcal{I}(H(X), H(Y)) = \sum_{j=1}^{r} \min(H(X)_j, H(Y)_j)$

matches at current level — matches at previous level

$$N_\ell = \mathcal{I}(H_\ell(X), H_\ell(Y)) - \mathcal{I}(H_{\ell-1}(X), H_{\ell-1}(Y))$$

Difference in histogram intersections across levels counts **number of new pairs matched**

Adapted from slides by Grauman and Darrell

Pyramid Match Kernel

histogram pyramids

$$K_\Delta(\Psi(X), \Psi(Y)) =$$

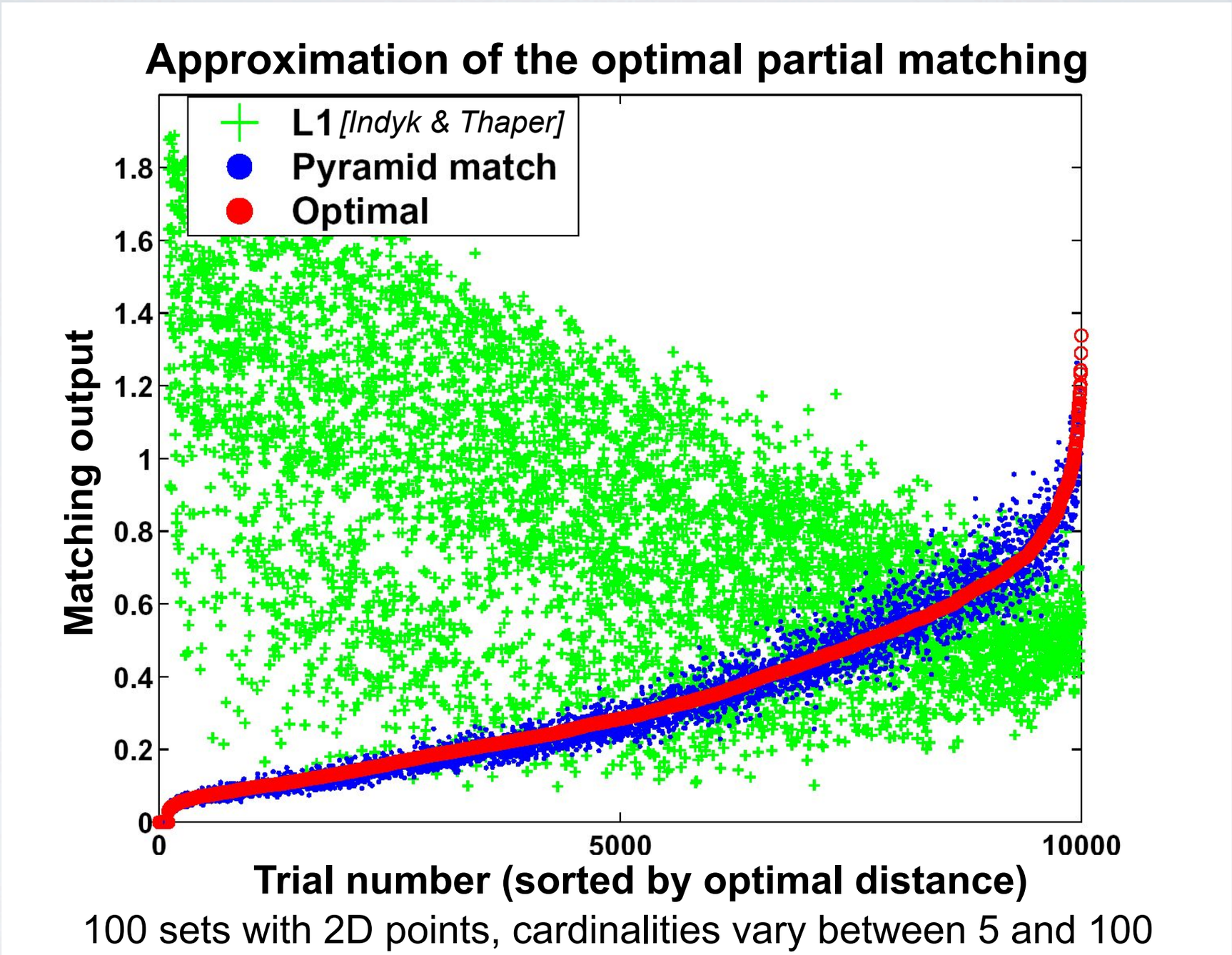
$$\sum_{\ell=0}^{L} \frac{1}{2^\ell} (\mathcal{I}(H_\ell(X), H_\ell(Y)) - \mathcal{I}(H_{\ell-1}(X), H_{\ell-1}(Y)))$$

number of newly matched pairs at level ℓ

measure of difficulty of a match at level ℓ

Adapted from slides by Grauman and Darrell

Approximation of Optimal Partial Matching



100 sets with 2D points, cardinalities vary between 5 and 100

Adapted from slides by Grauman and Darrell

Building a Classifier

- Train SVM by computing kernel values between all labeled training examples

- Classify novel examples by computing kernel values against support vectors

- One-versus-all for multi-class classification

Adapted from slides by Grauman and Darrell

BoW Summary

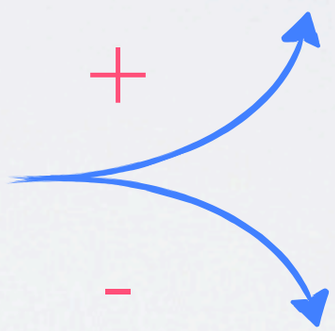
Issues

- Sampling strategy
  dense uniform, interest points, random...
- Codebook learning
  supervised/unsupervised, size...
- Similarity measurement
  SVM, Pyramid Matching
- Spatial information
- Scalability

BoW Summary

Spatial information

BoW removes spatial layout



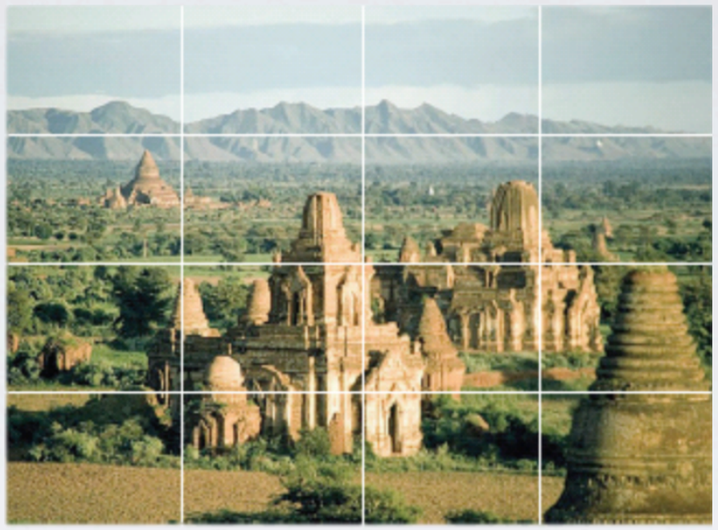
Increases the invariance to scale/ translation/deformation

Sacrifices discriminative power

APPROACHES

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding

Spatial Pyramid Matching



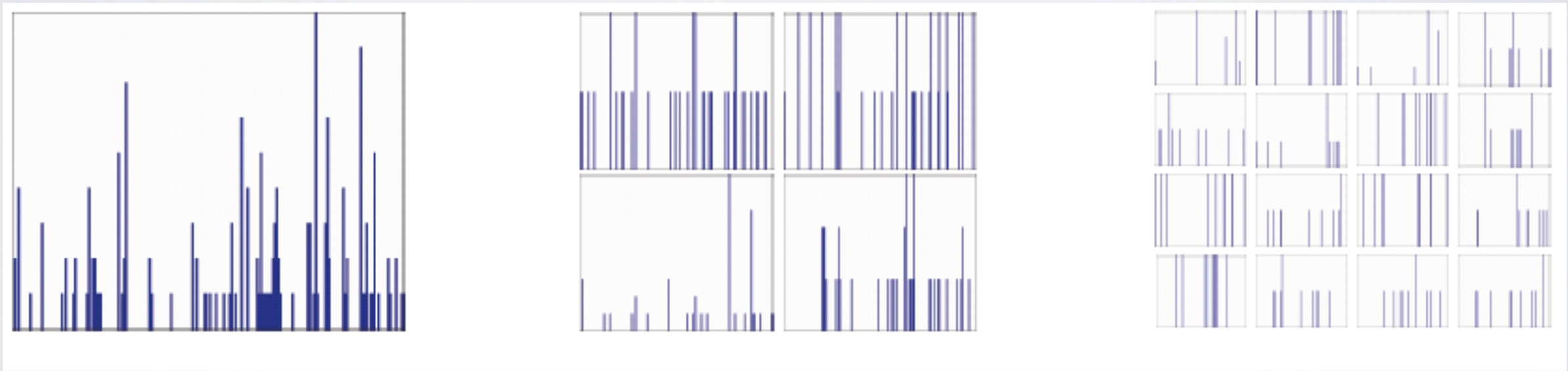
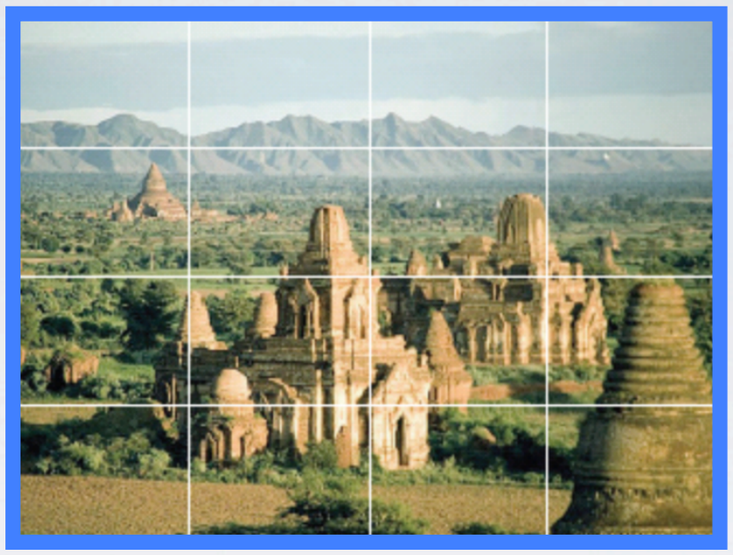
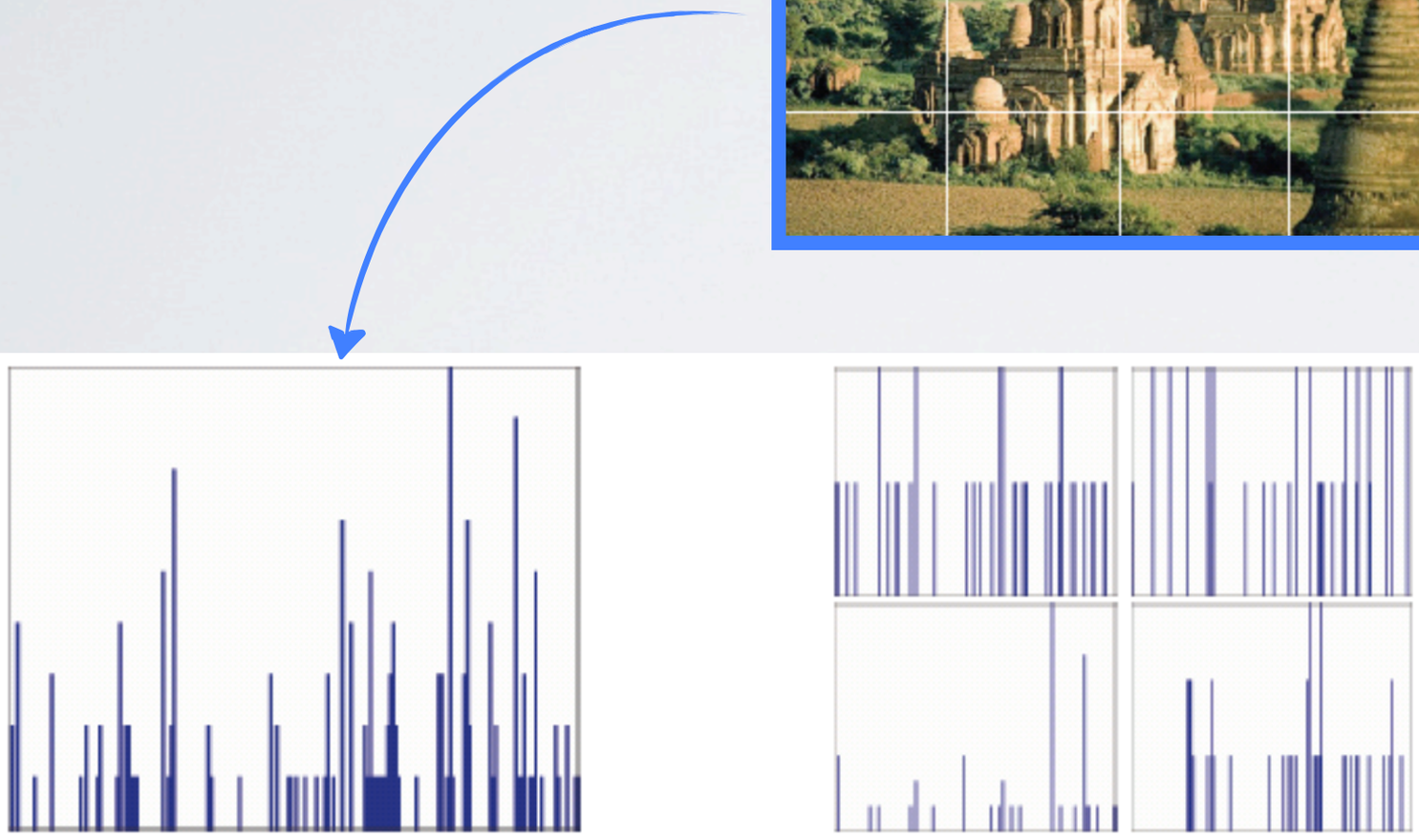


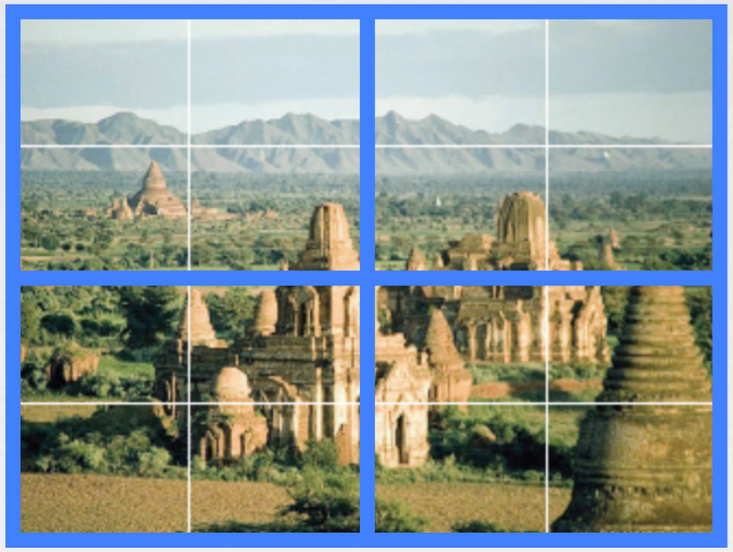
Image credit: Lazebnik et al

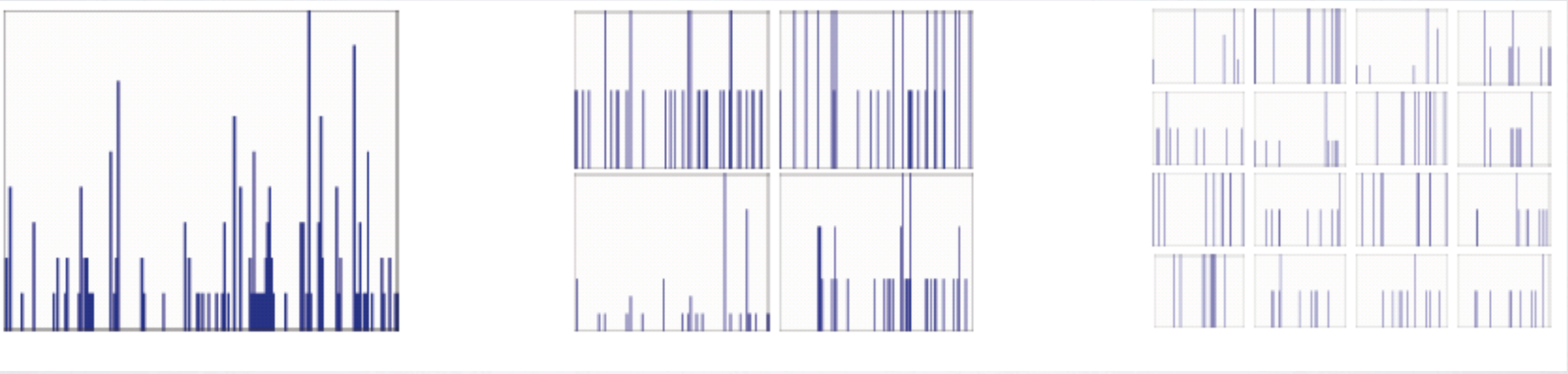
Spatial Pyramid Matching



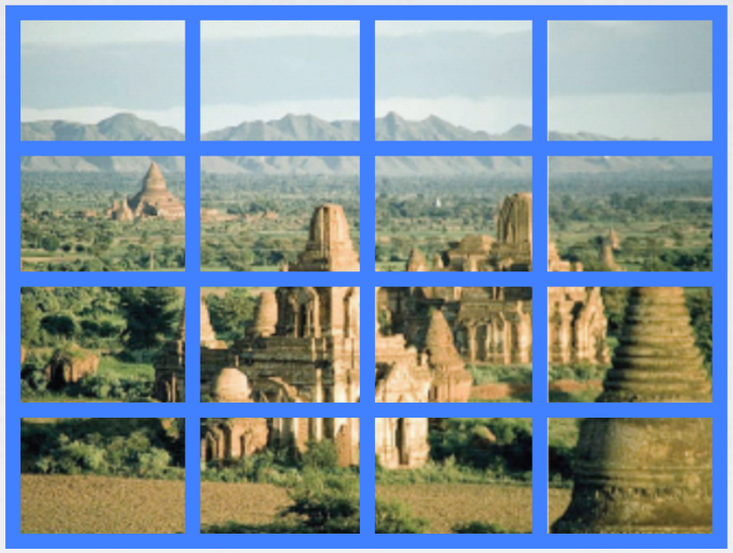


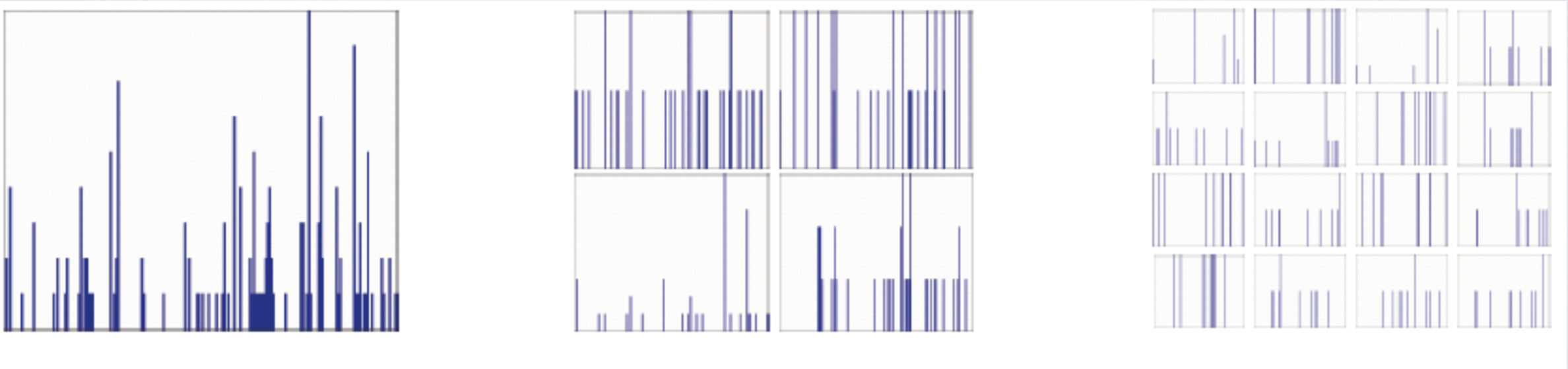
Spatial Pyramid Matching





Spatial Pyramid Matching

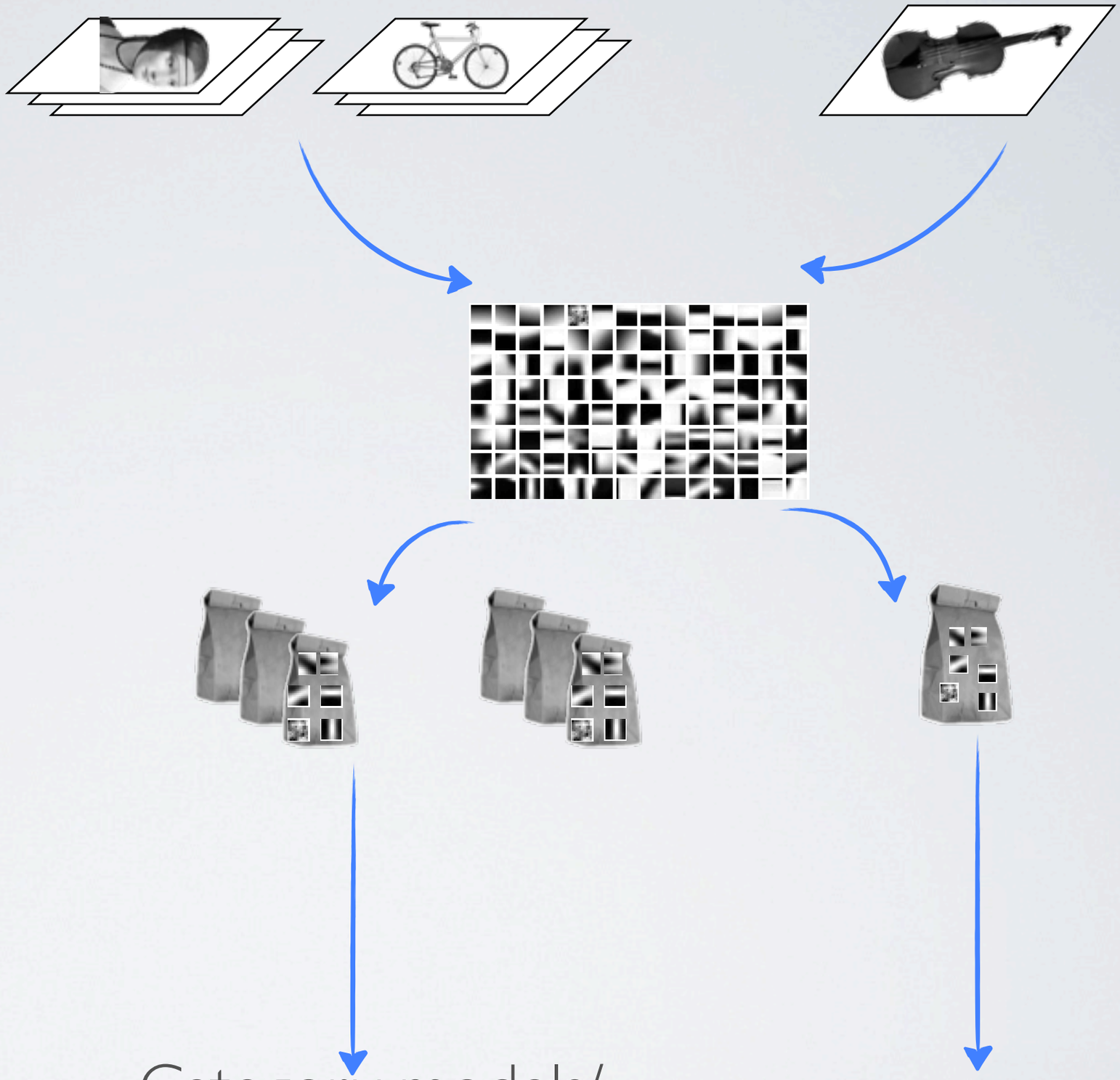




Feature detection & representation

Codewords dictionary formation

Image representation



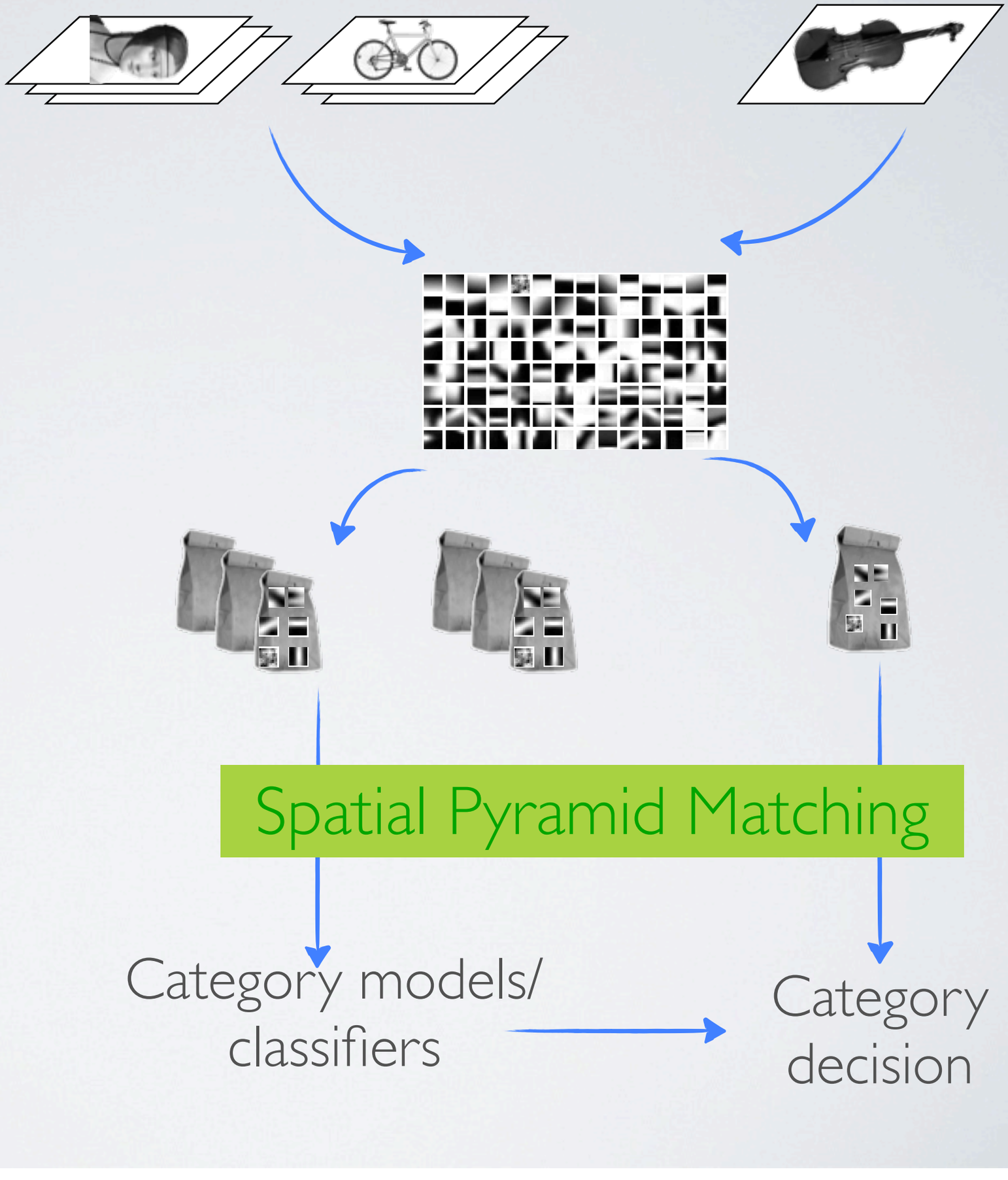
Category models/ classifiers

Category decision

Feature detection & representation

Codewords dictionary formation

Image representation



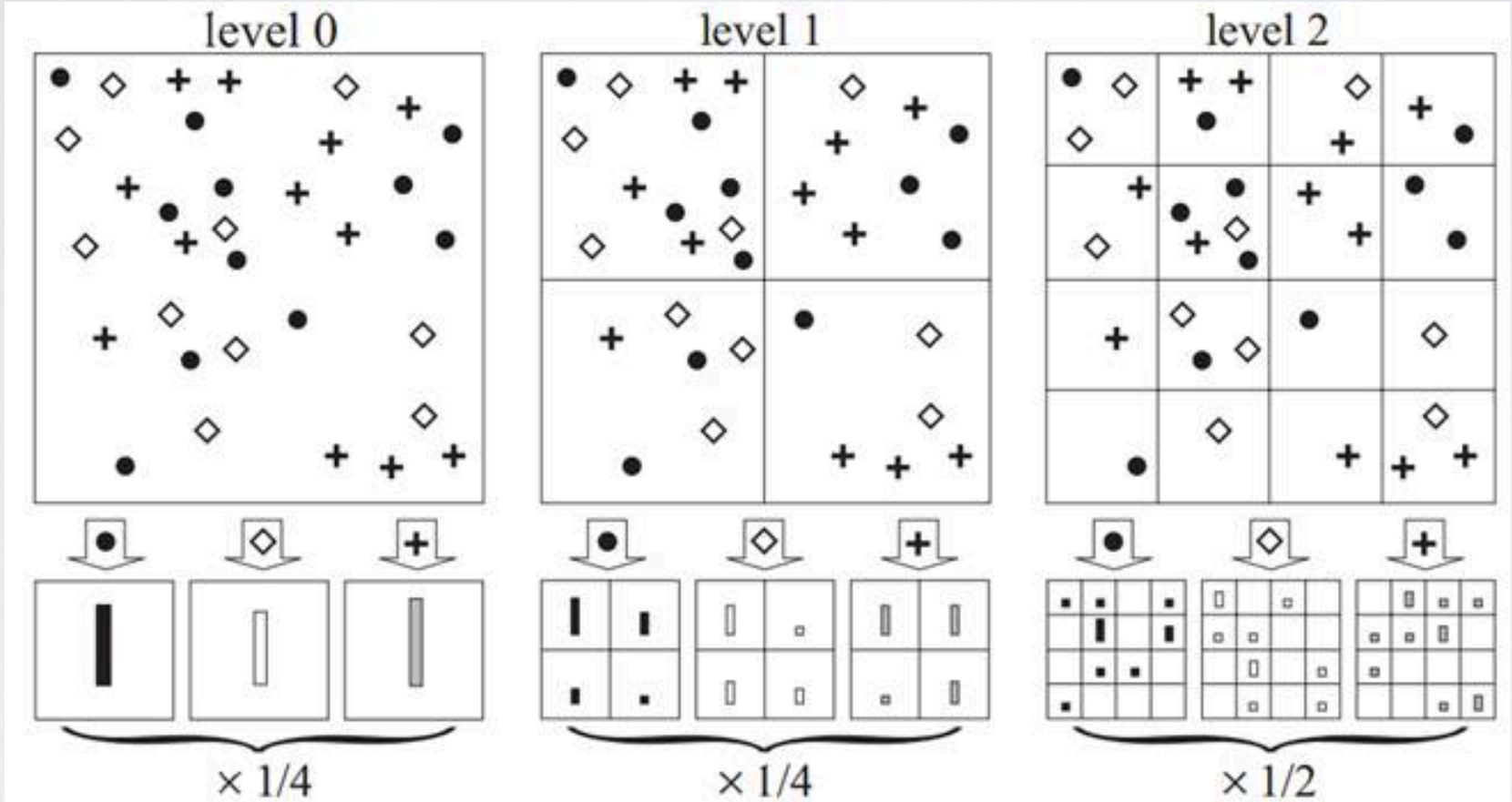
Spatial Pyramid Matching

- Quantize feature vectors into M discrete types

- Perform pyramid matching in 2D image space for each channel m = 1, ..., M

- Assume features of the same type m can be matched to each other

Spatial Pyramid Matching



$$\sum_{m=1}^{M} \kappa^L(X_m, Y_m)$$

Image credit: Lazebnik et al 2006

Pyramid Match Kernel

histogram pyramids

$$K_\Delta(\Psi(X), \Psi(Y)) =$$

$$\sum_{\ell=0}^{L} \frac{1}{2^\ell} (\mathcal{I}(H_\ell(X), H_\ell(Y)) - \mathcal{I}(H_{\ell-1}(X), H_{\ell-1}(Y)))$$

number of newly matched pairs at level ℓ

measure of difficulty of a match at level ℓ

Adapted from slides by Grauman and Darrell

Pyramid Match Kernel

$$\kappa^L(X,Y) = K_\Delta(\Psi(X),\Psi(Y)) =$$

$$\sum_{\ell=0}^{L} \frac{1}{2^\ell}(\mathcal{I}(H_\ell(X),H_\ell(Y)) - \mathcal{I}(H_{\ell-1}(X),H_{\ell-1}(Y)))$$

The final kernel is sum of the separate channel kernels:

$$\sum_{m=1}^{M} \kappa^L(X_m,Y_m)$$

Adapted from slides by Grauman and Darrell

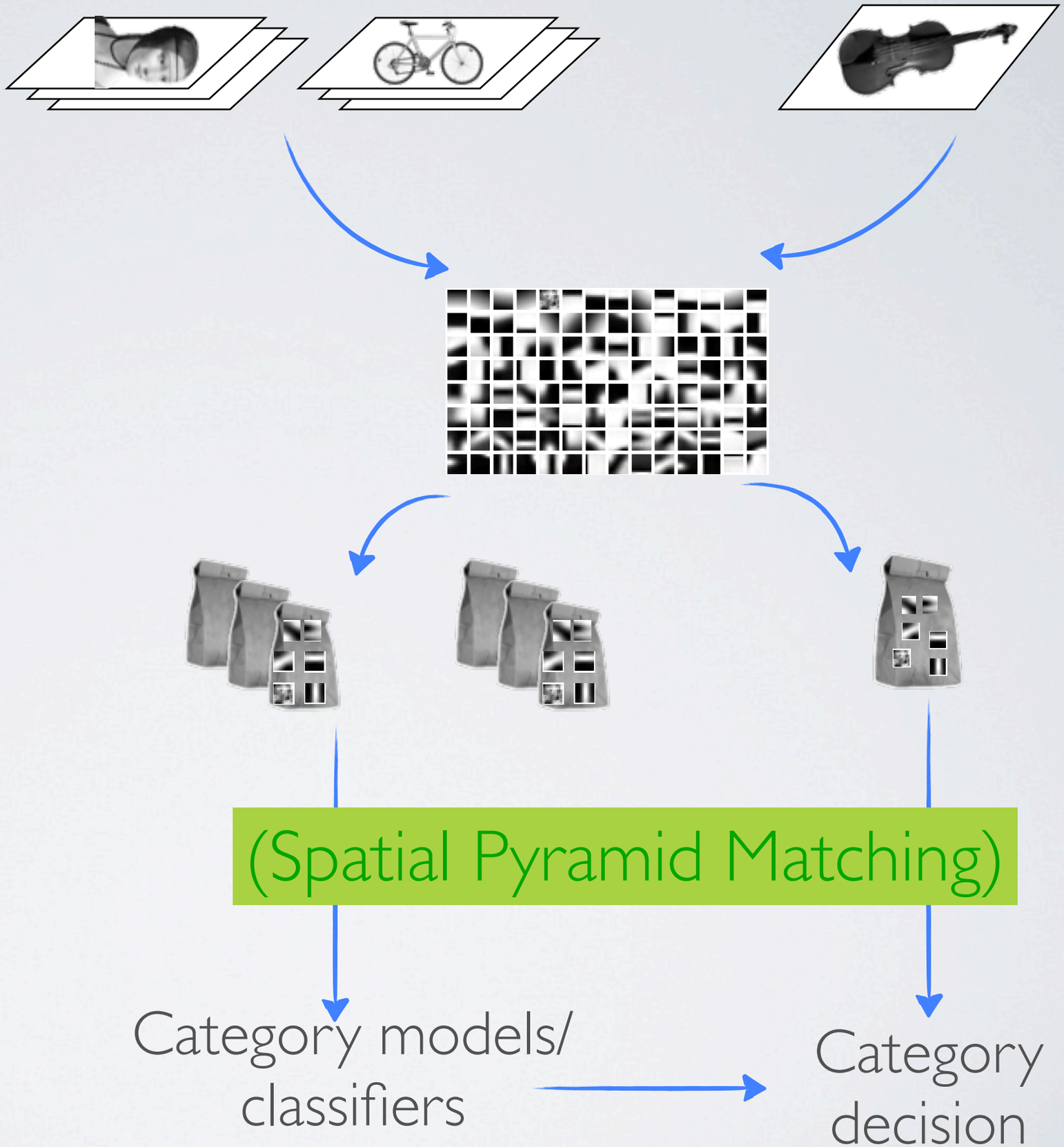
APPROACHES

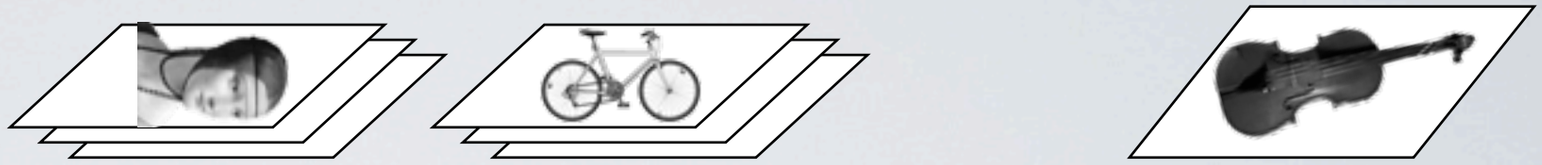
- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding
  - Linear SPM using Sparse Coding
  - Locality-constrained Linear Coding
  - Fisher Vector

Feature detection & representation

Codewords dictionary formation

Image representation





Feature detection & representation

Other ways of encoding local feature descriptors

Codewords dictionary formation

Image representation

For better performance

(Spatial Pyramid Matching)

Category models/ classifiers

Category decision

APPROACHES

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding
  - Linear SPM using Sparse Coding
  - Locality-constrained Linear Coding
  - Fisher Vector

Linear SPM using Sparse Coding

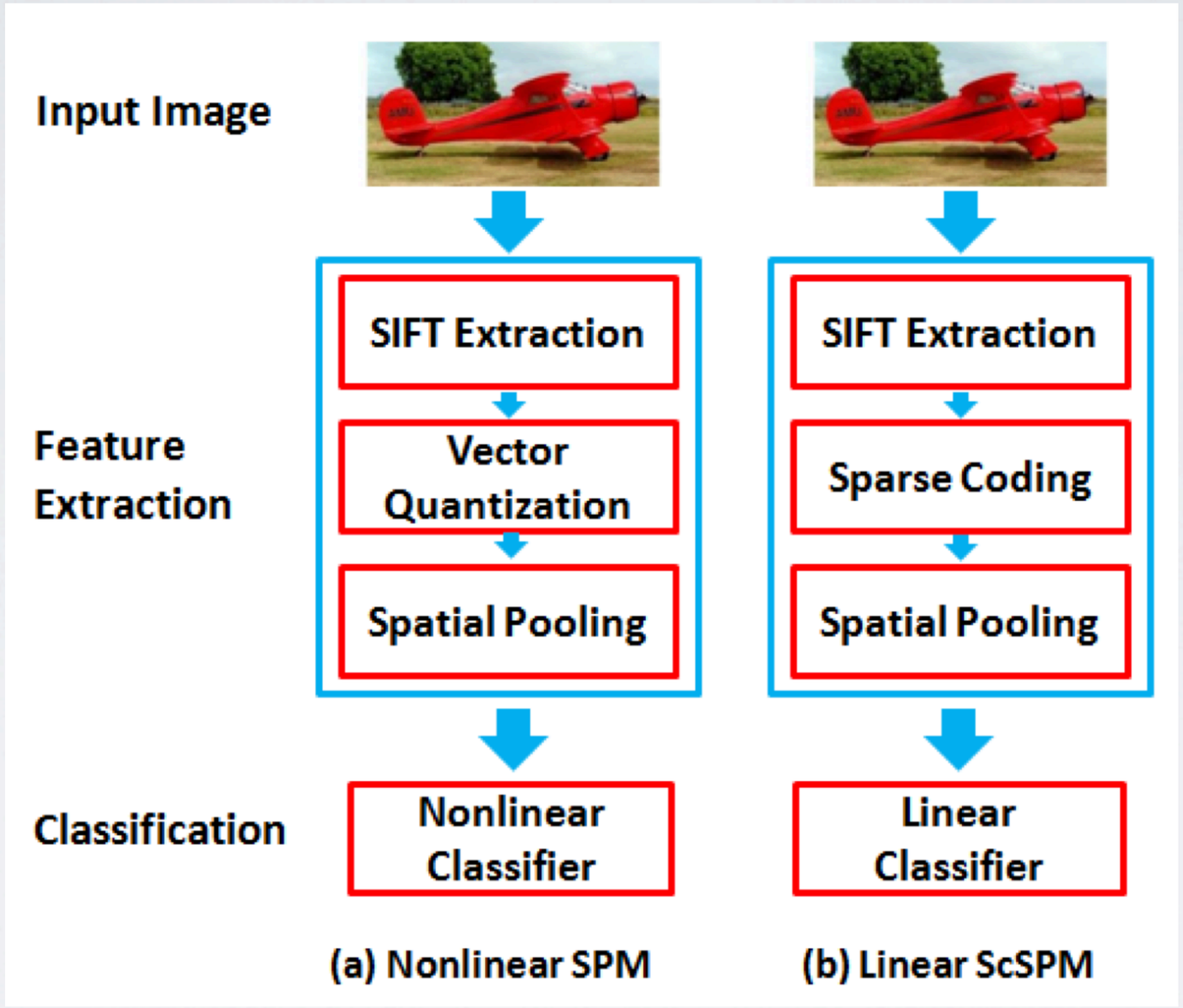


Image credit: Yang et al 2009

Linear SPM using Sparse Coding

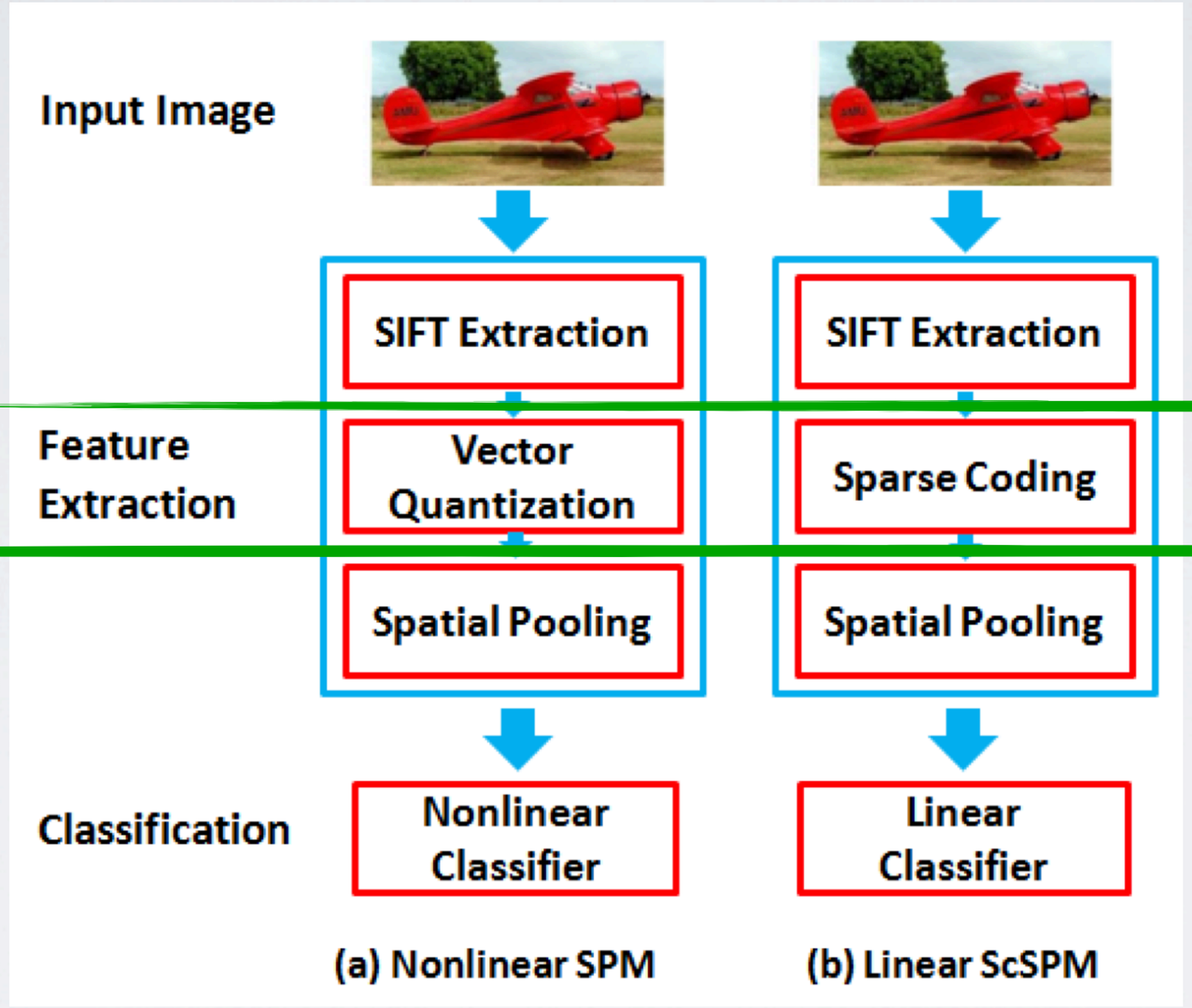


Image credit: Yang et al 2009

Encoding SIFT: From VQ to SC

$$\min_{\mathbf{V}} \quad \sum_{m=1}^{M} \min_{k=1,\ldots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2$$

$$\text{codebook:} \quad \mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_K]^\top$$

Encoding SIFT: From VQ to SC

$$\min_{\mathbf{V}} \quad \sum_{m=1}^{M} \min_{k=1,\ldots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m \mathbf{V}\|^2$$

$$\text{s.t.} \quad Card(\mathbf{u}_m) = 1, |\mathbf{u}_m| = 1, \mathbf{u}_m \succeq 0, \forall m$$

$$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]^\top$$

Encoding SIFT: From VQ to SC

$$\min_{\mathbf{V}} \quad \sum_{m=1}^{M} \min_{k=1,\dots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2$$

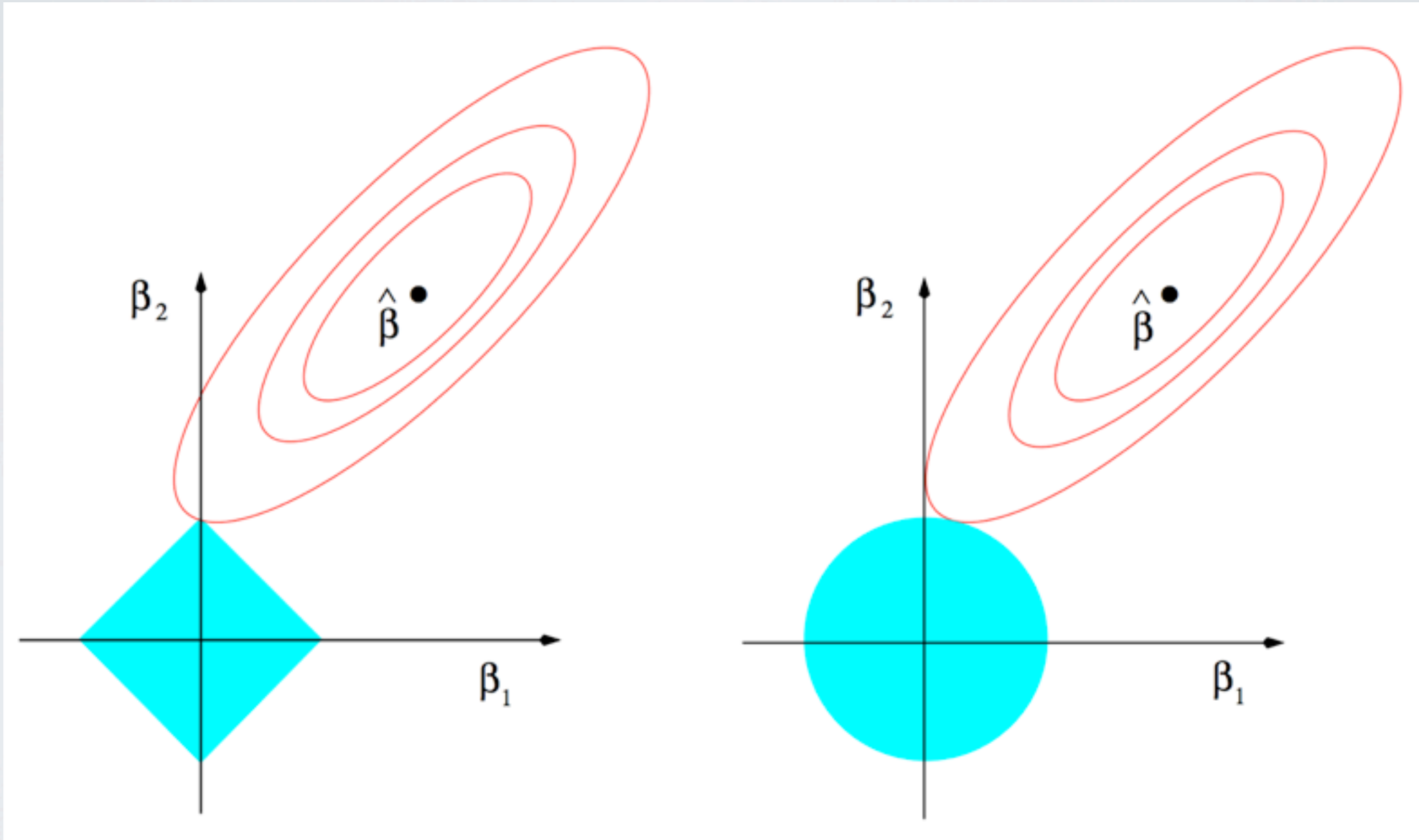
$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2$$

$$\text{s.t.} \quad Card(\mathbf{u}_m) = 1, |\mathbf{u}_m| = 1, \mathbf{u}_m \succeq 0, \forall m$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$

$$\text{s.t.} \quad \|\mathbf{v}_k\| \leq 1, \quad \forall k$$

Why L1 encourages sparsity



$$\min \|\mathbf{x} - \mathbf{uV}\|_2^2 + \lambda \|\mathbf{u}\|_1 \qquad \min \|\mathbf{x} - \mathbf{uV}\|_2^2 + \lambda \|\mathbf{u}\|_2$$

Encoding SIFT: From VQ to SC

$$\min_{\mathbf{V}} \quad \sum_{m=1}^{M} \min_{k=1,\dots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2$$
$$\text{s.t.} \quad Card(\mathbf{u}_m) = 1, |\mathbf{u}_m| = 1, \mathbf{u}_m \succeq 0, \forall m$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$
$$\text{s.t.} \quad \|\mathbf{v}_k\| \le 1, \quad \forall k$$

Encoding SIFT: From VQ to SC

$$\min_{\mathbf{V}} \quad \sum_{m=1}^{M} \min_{k=1,\dots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2$$
$$\text{s.t.} \quad Card(\mathbf{u}_m) = 1, |\mathbf{u}_m| = 1, \mathbf{u}_m \succeq 0, \forall m$$

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$
$$\text{s.t.} \quad \|\mathbf{v}_k\| \le 1, \quad \forall k$$

Implementation:
feature-sign search algorithm [Lee et al 2006]
http://ai.stanford.edu/~hllee/softwares/nips06-sparsecoding.htm

Algorithm Architecture

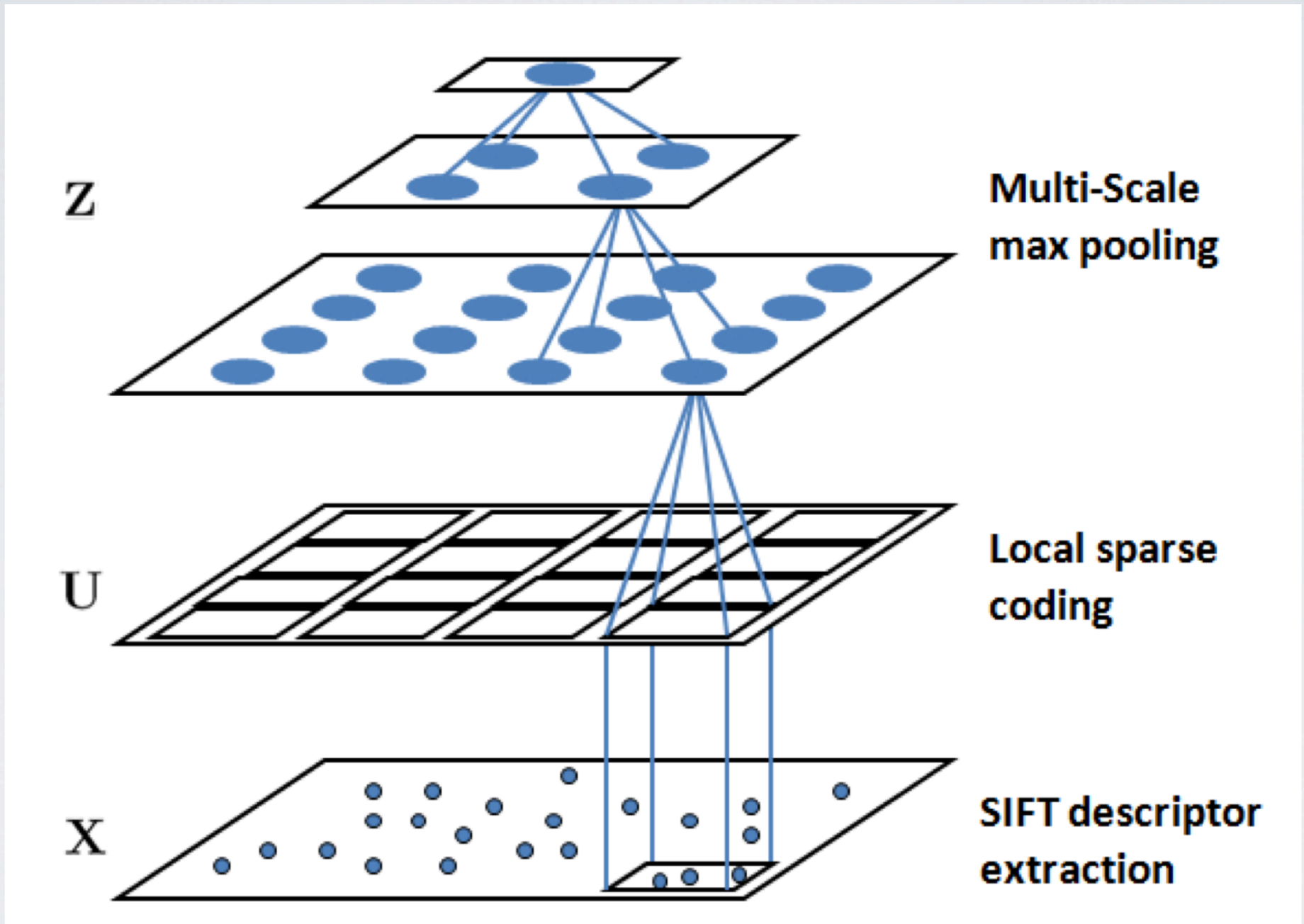


Image credit: Yang et al 2009

Linear SPM

$$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]^\top$$

$$\mathbf{z} = \mathcal{F}(\mathbf{U})$$

define $\mathcal{F}$ as:

$$z_j = \max\{|u_{1j}|, |u_{2j}, \ldots, |u_{Mj}||\}$$

$$\kappa(\mathbf{z}_i, \mathbf{z}_j) = \mathbf{z}_i^\top \mathbf{z}_j = \sum_{l=0}^{2} \sum_{s=1}^{2^l} \sum_{t=1}^{2^l} \langle \mathbf{z}_i^l(s,t), \mathbf{z}_j^l(s,t) \rangle$$

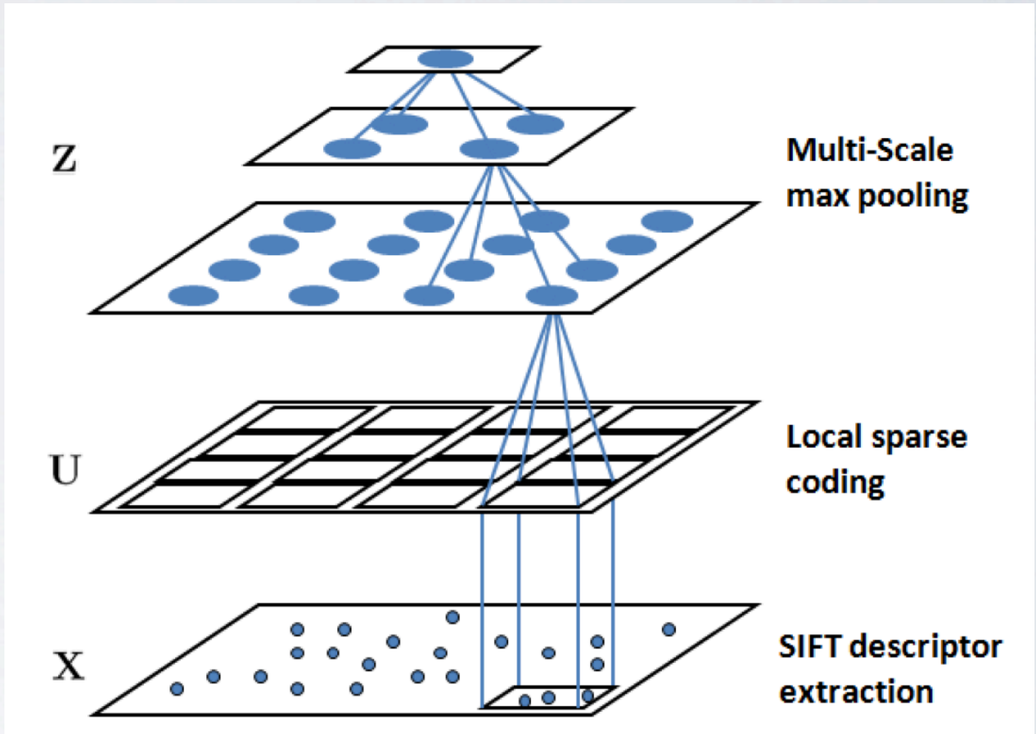
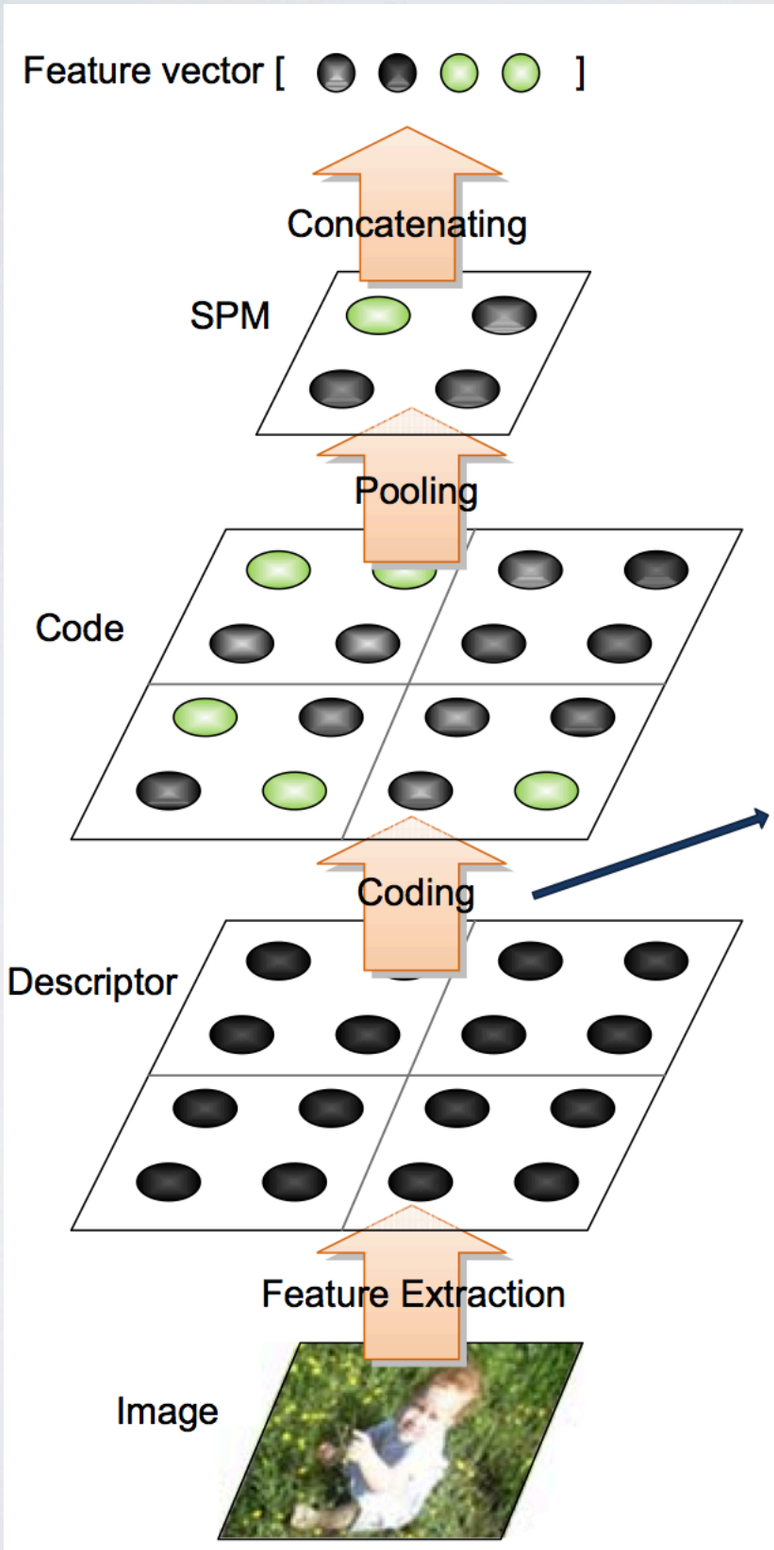


Image credit: Yang et al 2009

APPROACHES

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding
  - Linear SPM using Sparse Coding
  - Locality-constrained Linear Coding
  - Fisher Vector

Locality-constrained Linear Coding



LLC Coding process

Step 3:
c_i is an Mx1 vector with K non-zero elements whose values are the corresponding c^* of step 2

input: x_i → code: c_i

Step 2:
Reconstruct x_i using B_i

$$c^* = \underset{c}{\operatorname{argmin}} \| x_i - c_i^T B_i \|^2$$

$$\text{st. } \sum_j c_j = 1$$

input: x_i

codebook: $B=\{b_j\}_{j=1,\ldots,M}$

Step 1:
Find K-Nearest Neighbors of x_i, denoted as B_i

Image credit: Wang et al 2010

Locality-constrained Linear Coding

SC

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$

$$\text{s.t.} \quad \|\mathbf{v}_k\| \leq 1, \quad \forall k$$

Locality-constrained Linear Coding

SC

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$

$$\text{s.t.} \quad \|\mathbf{v}_k\| \leq 1, \quad \forall k$$

LLC

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda\|\mathbf{d}_m \odot \mathbf{u}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathbf{u}_m = 1, \quad \forall m$$

Locality-constrained Linear Coding

SC

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda|\mathbf{u}_m|$$

$$\text{s.t.} \quad \|\mathbf{v}_k\| \leq 1, \quad \forall k$$

LLC

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda\|\mathbf{d}_m \odot \mathbf{u}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathbf{u}_m = 1, \quad \forall m$$

$$\text{locality adaptor: } \mathbf{d}_m = \exp\left(\frac{\text{dist}(\mathbf{x}_m, \mathbf{V})}{\sigma}\right)$$

where

$\text{dist}(\mathbf{x}_m, \mathbf{V}) = [\text{dist}(\mathbf{x}_m, \mathbf{v}_1), \ldots, \text{dist}(\mathbf{x}_m, \mathbf{v}_K)]^\top$

$\text{dist}(\mathbf{x}_m, \mathbf{v}_k)$ is the Euclidean distance between $\mathbf{x}_m$ and $\mathbf{v}_k$

σ is for adjusting the decay speed

Properties of LLC

- Better reconstruction
- Local smooth sparsity
- Analytical solution

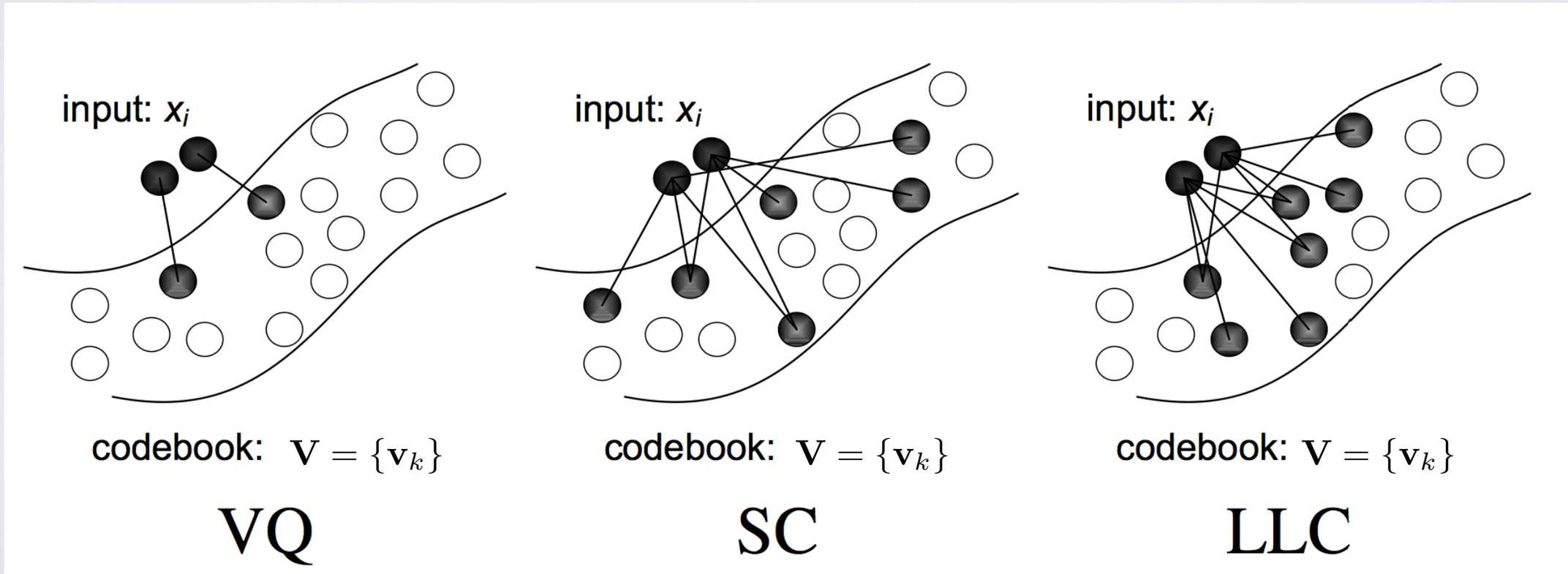


Image credit: Wang et al 2010

Properties of LLC

- Better reconstruction
- Local smooth sparsity
- Analytical solution

$$\tilde{\mathbf{u}}_m = \left((\mathbf{V} - \mathbf{1}\mathbf{x}_m^\top)(\mathbf{V} - \mathbf{1}\mathbf{x}_m^\top)^\top + \lambda \mathrm{diag}(\mathbf{d})\right) \backslash \mathbf{1}$$
$$\mathbf{u}_m = \tilde{\mathbf{u}}_m / \mathbf{1}^\top \tilde{\mathbf{u}}_m$$

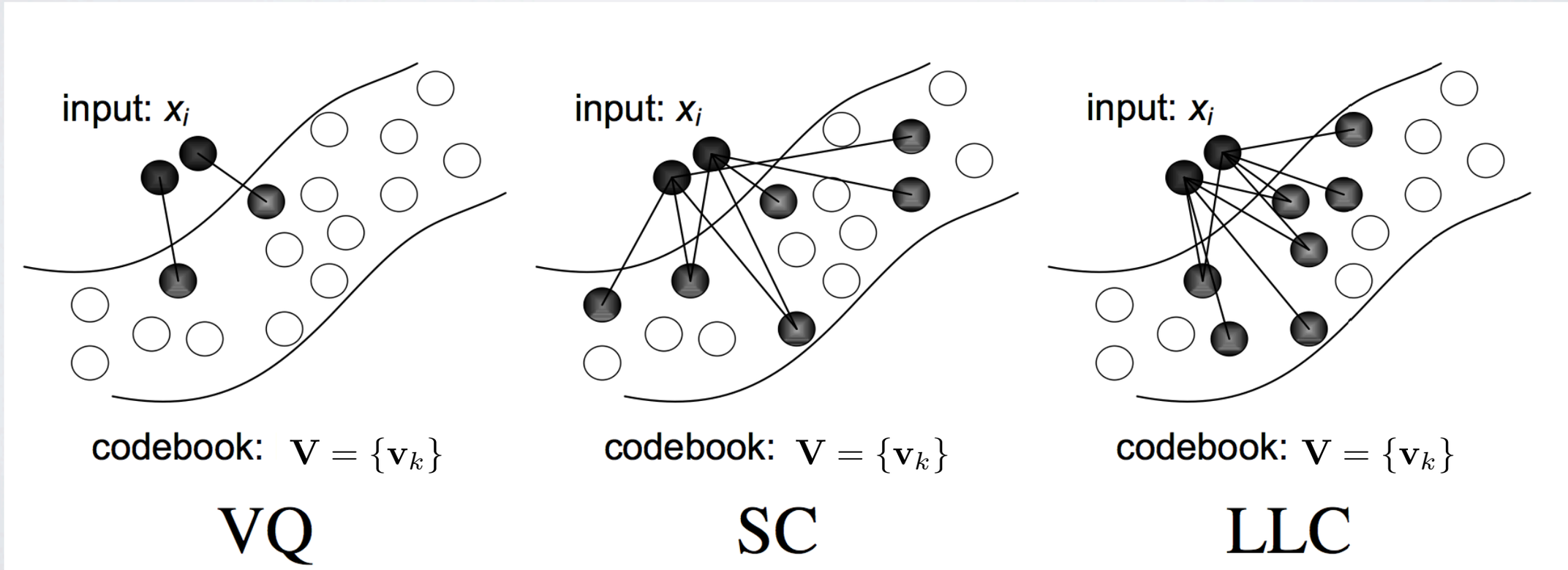


Image credit: Wang et al 2010

Approximated LLC for Fast Encoding

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda\|\mathbf{d}_m \odot \mathbf{u}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathbf{u}_m = 1, \quad \forall m$$

Approximated LLC for Fast Encoding

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda\|\mathbf{d}_m \odot \mathbf{u}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathbf{u}_m = 1, \quad \forall m$$

Select ***local bases*** of each descriptor
to form a ***local coordinate system***

$$\min_{\tilde{\mathbf{U}}} \sum_{m=1}^{M} \|\mathbf{x}_m - \tilde{\mathbf{u}}_m\mathbf{V}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \tilde{\mathbf{u}}_m = 1, \quad \forall m$$

Approximated LLC for Fast Encoding

$$\min_{\mathbf{V},\mathbf{U}} \quad \sum_{m=1}^{M} \|\mathbf{x}_m - \mathbf{u}_m\mathbf{V}\|^2 + \lambda\|\mathbf{d}_m \odot \mathbf{u}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \mathbf{u}_m = 1, \quad \forall m$$

Select ***local bases*** of each descriptor to form a ***local coordinate system***

the K nearest neighbors of $\mathbf{x}_m$ forms the local basis $\mathbf{V}_m$

$$\min_{\tilde{\mathbf{U}}} \sum_{m=1}^{M} \|\mathbf{x}_m - \tilde{\mathbf{u}}_m\mathbf{V}_m\|^2$$

$$\text{s.t.} \quad \mathbf{1}^\top \tilde{\mathbf{u}}_m = 1, \quad \forall m$$

APPROACHES

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding
  - Linear SPM using Sparse Coding
  - Locality-constrained Linear Coding
  - Fisher Vector

Fisher Kernel

$X = \{x_1, \ldots, x_T\}$: a sample of T observations $x_t \in \mathcal{X}$

u_λ: the *pdf*

$\lambda = [\lambda_1, \ldots, \lambda_M]^\top \in \mathbb{R}^M$

score function:

$$G_\lambda^X = \nabla_\lambda \log u_\lambda(X)$$

similarity measurement: [Jaakkola and Haussler, 1998]

$$K_{FK}(X, Y) = G_\lambda^{X\top} F_\lambda^{-1} G_\lambda^Y$$

Fisher Kernel

similarity measurement:

$$K_{FK}(X,Y) = G_\lambda^{X\top} F_\lambda^{-1} G_\lambda^Y$$

Fisher Information Matrix:

$$F_\lambda = E_{x\sim u_\lambda}(G_\lambda^X G_\lambda^{X\top}) \qquad F_\lambda^{-1} = L_\lambda^\top L_\lambda$$

Fisher Kernel

similarity measurement:

$$K_{FK}(X,Y) = G_\lambda^{X\top} F_\lambda^{-1} G_\lambda^Y$$

Fisher Information Matrix:

$$F_\lambda = E_{x\sim u_\lambda}(G_\lambda^X G_\lambda^{X\top}) \qquad F_\lambda^{-1} = L_\lambda^\top L_\lambda$$

Fisher Kernel re-written as:

$$K_{FK}(X,Y) = \mathcal{G}_\lambda^{X\top} \mathcal{G}_\lambda^Y$$

$$\text{where } \mathcal{G}_\lambda^X = L_\lambda \nabla_\lambda \log u_\lambda(X)$$

Fisher Vector

Fisher Vector on Images

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

GMM:

$$u_\lambda(x) = \sum_{k=1}^{K} w_k u_k(x)$$

$$\text{where } u_k(x) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k)\right\}$$

Fisher Vector on Images

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

GMM:

$$u_\lambda(x) = \sum_{k=1}^{K} w_k u_k(x)$$

$$\text{where } u_k(x) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k)\right\}$$

EM algorithm to estimate the parameters:

$$\lambda = \{w_k, \mu_k, \Sigma_k, k = 1, \ldots, K\}$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Gradients:

$$\nabla_{\alpha_k} \log u_\lambda(x_t) = \gamma_t(k) - w_k$$

$$\nabla_{\mu_k} \log u_\lambda(x_t) = \gamma_t(k) \left(\frac{x_t - \mu_t}{\sigma_k^2} \right)$$

$$\nabla_{\sigma_k} \log u_\lambda(x_t) = \gamma_t(k) \left[\frac{(x_t - \mu_k)^2}{\sigma_k^4} - \frac{1}{\sigma_k} \right]$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Gradients:

$$\nabla_{\alpha_k} \log u_\lambda(x_t) = \gamma_t(k) - w_k$$

$$\nabla_{\mu_k} \log u_\lambda(x_t) = \gamma_t(k) \left(\frac{x_t - \mu_t}{\sigma_k^2} \right)$$

$$\nabla_{\sigma_k} \log u_\lambda(x_t) = \gamma_t(k) \left[\frac{(x_t - \mu_k)^2}{\sigma_k^4} - \frac{1}{\sigma_k} \right]$$

Posterior Probability:

$$\gamma_t(k) = \frac{w_k u_k(x_k)}{\sum_{j=1}^{K} w_j u_j(x_t)}$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Gradients:

$$\nabla_{\alpha_k} \log u_\lambda(x_t) = \gamma_t(k) - w_k$$

$$w_k = \frac{\exp(\alpha_k)}{\sum_{j=1}^{K} \exp(\alpha_j)}$$

$$\nabla_{\mu_k} \log u_\lambda(x_t) = \gamma_t(k) \left(\frac{x_t - \mu_t}{\sigma_k^2} \right)$$

$$\nabla_{\sigma_k} \log u_\lambda(x_t) = \gamma_t(k) \left[\frac{(x_t - \mu_k)^2}{\sigma_k^4} - \frac{1}{\sigma_k} \right]$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Gradients:

$$\nabla_{\alpha_k} \log u_\lambda(x_t) = \gamma_t(k) - w_k$$

BoW

$$\nabla_{\mu_k} \log u_\lambda(x_t) = \gamma_t(k) \left(\frac{x_t - \mu_t}{\sigma_k^2} \right)$$

$$\nabla_{\sigma_k} \log u_\lambda(x_t) = \gamma_t(k) \left[\frac{(x_t - \mu_k)^2}{\sigma_k^4} - \frac{1}{\sigma_k} \right]$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Fisher Information Matrix:

$$F_\lambda = E_{x \sim u_\lambda}(G_\lambda^X G_\lambda^{X\top}) \qquad F_\lambda^{-1} = L_\lambda^\top L_\lambda$$

Soft Assignment

Fisher Vector:

$$\mathcal{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$$

Fisher Information Matrix:

$$F_\lambda = E_{x \sim u_\lambda}(G_\lambda^X G_\lambda^{X\top}) \qquad F_\lambda^{-1} = L_\lambda^\top L_\lambda$$

Assume almost hard assignment → FIM diagonal → coordinate-wise normalization on gradient vectors

Fisher Vector

Assume almost hard assignment → FIM diagonal → coordinate-wise normalization on gradient vectors

Normalized Gradients:

$$\mathcal{G}^X_{\alpha_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} (\gamma_t(k) - w_k)$$

$$\mathcal{G}^X_{\mu_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \left(\frac{x_t - \mu_k}{\sigma_k} \right)$$

$$\mathcal{G}^X_{\sigma_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \frac{1}{\sqrt{2}} \left[\frac{(x_t - \mu_k)^2}{\sigma_k^2} - 1 \right]$$

Concatenate the gradient vectors:

Dimension = (2D+1)K

Fisher Vector

Assume almost hard assignment → FIM diagonal → coordinate-wise normalization on gradient vectors

Normalized Gradients:

$$\mathcal{G}^X_{\alpha_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} (\gamma_t(k) - w_k)$$

$$\mathcal{G}^X_{\mu_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \left(\frac{x_t - \mu_k}{\sigma_k} \right)$$

$$\mathcal{G}^X_{\sigma_k} = \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \frac{1}{\sqrt{2}} \left[\frac{(x_t - \mu_k)^2}{\sigma_k^2} - 1 \right]$$

Concatenate the gradient vectors:

Dimension = (2D+1)K

$$\mathcal{G}^X_\lambda \leftarrow \frac{1}{T} \mathcal{G}^X_\lambda$$

T : patch size

Algorithm 1 Compute Fisher vector from local descriptors

Input:

- Local image descriptors $X = \{x_t \in \mathbb{R}^D, t = 1, \ldots, T\}$,
- Gaussian mixture model parameters $\lambda = \{w_k, \mu_k, \sigma_k, k = 1, \ldots, K\}$

Output:

- normalized Fisher Vector representation $\mathcal{G}_\lambda^X \in \mathbb{R}^{K(2D+1)}$

1. **Compute statistics**
   - For $k = 1, \ldots, K$ initialize accumulators
     - $S_k^0 \leftarrow 0, \quad S_k^1 \leftarrow 0, \quad S_k^2 \leftarrow 0$
   - For $t = 1, \ldots T$
     - Compute $\gamma_t(k)$ using equation (15)
     - For $k = 1, \ldots, K$:
       - $S_k^0 \leftarrow S_k^0 + \gamma_t(k)$,
       - $S_k^1 \leftarrow S_k^1 + \gamma_t(k) x_t$,
       - $S_k^2 \leftarrow S_k^2 + \gamma_t(k) x_t^2$

2. **Compute the Fisher vector signature**
   - For $k = 1, \ldots, K$:

$$\mathcal{G}_{\alpha_k}^X = \left(S_k^0 - T w_k\right) / \sqrt{w_k} \tag{31}$$

$$\mathcal{G}_{\mu_k}^X = \left(S_k^1 - \mu_k S_k^0\right) / \left(\sqrt{w_k}\sigma_k\right) \tag{32}$$

$$\mathcal{G}_{\sigma_k}^X = \left(S_k^2 - 2\mu_k S_k^1 + (\mu_k^2 - \sigma_k^2) S_k^0\right) / \left(\sqrt{2w_k}\sigma_k^2\right) \tag{33}$$

   - Concatenate all Fisher vector components into one vector
     $\mathcal{G}_\lambda^X = \left(\mathcal{G}_{\alpha_1}^X, \ldots, \mathcal{G}_{\alpha_K}^X, \mathcal{G}_{\mu_1}^{X\prime}, \ldots, \mathcal{G}_{\mu_K}^{X\prime}, \mathcal{G}_{\sigma_1}^{X\prime}, \ldots, \mathcal{G}_{\sigma_K}^{X\prime}\right)'$

3. **Apply normalizations**
   - For $i = 1, \ldots, K(2D+1)$ apply power normalization
     - $[\mathcal{G}_\lambda^X]_i \leftarrow sign\left([\mathcal{G}_\lambda^X]_i\right) \sqrt{|[\mathcal{G}_\lambda^X]_i|}$
   - Apply ℓ_2-normalization:
     $\mathcal{G}_\lambda^X = \mathcal{G}_\lambda^X / \sqrt{\mathcal{G}_\lambda^{X\prime} \mathcal{G}_\lambda^X}$

$$\gamma_t(k) = \frac{w_k u_k(x_k)}{\sum_{j=1}^{K} w_j u_j(x_t)}$$

To make it work with linear classifier

Image credit: Sanchez et al 2013

Extension on FV

- Spatial Pyramid

  [Sanchez et al 2013]

- Deep Fisher Networks

  [Simonyan et al 2013]

- Other methods account scene geometry in FV framework

  [Krapac et al, 2011, Sanchez et al, 2012]

Extension on FV

- Spatial Pyramid

  [Sanchez et al 2013]

- Deep Fisher Networks

  [Simonyan et al 2013]

- Other methods account scene geometry in FV framework

  [Krapac et al, 2011, Sanchez et al, 2012]

Extension on FV

- Spatial Pyramid

  [Sanchez et al 2013]

- Deep Fisher Networks

  [Simonyan et al 2013]

- Other methods account scene geometry in FV framework

  [Krapac et al, 2011, Sanchez et al, 2012]

Deep Fisher Networks

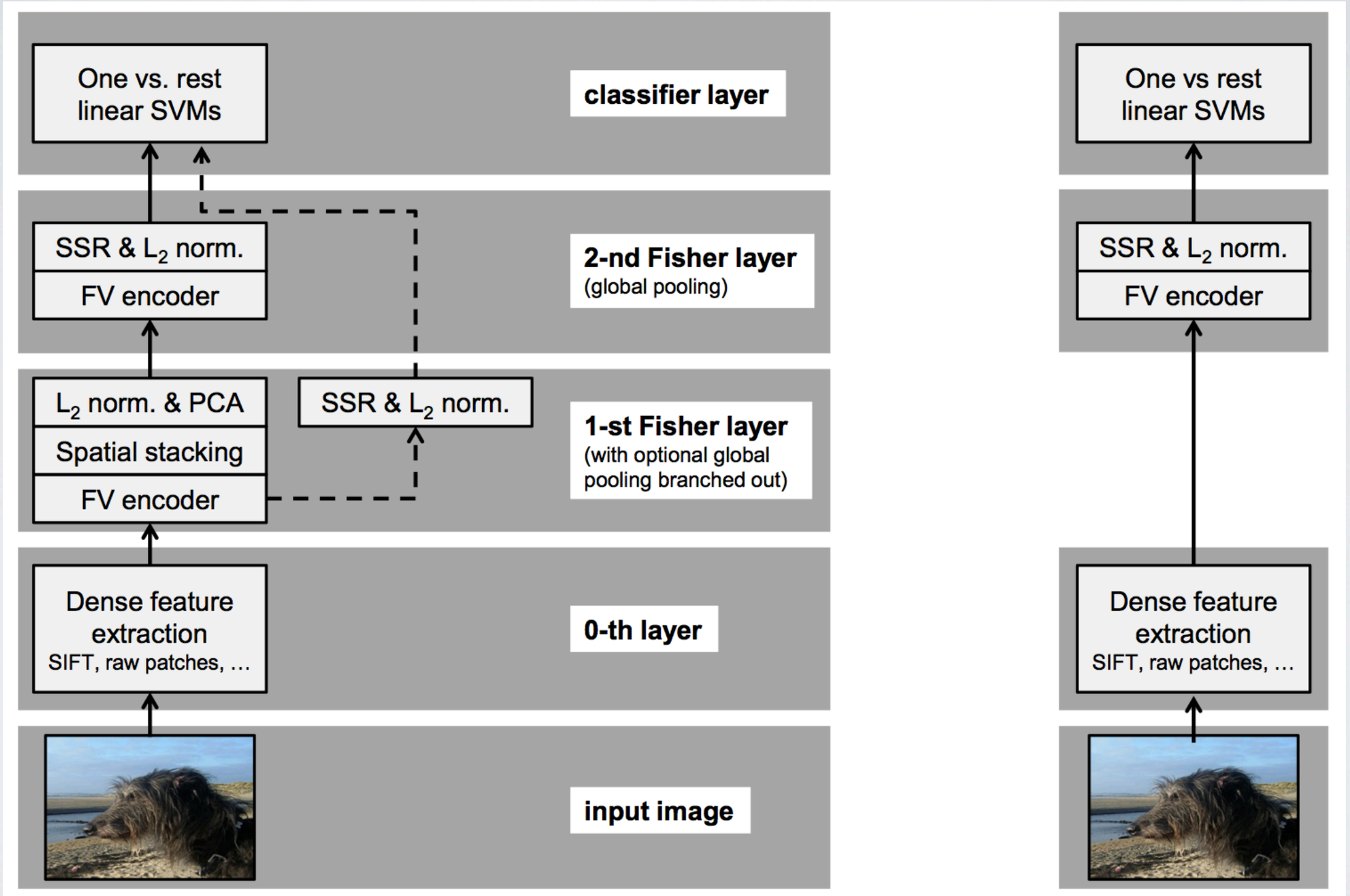
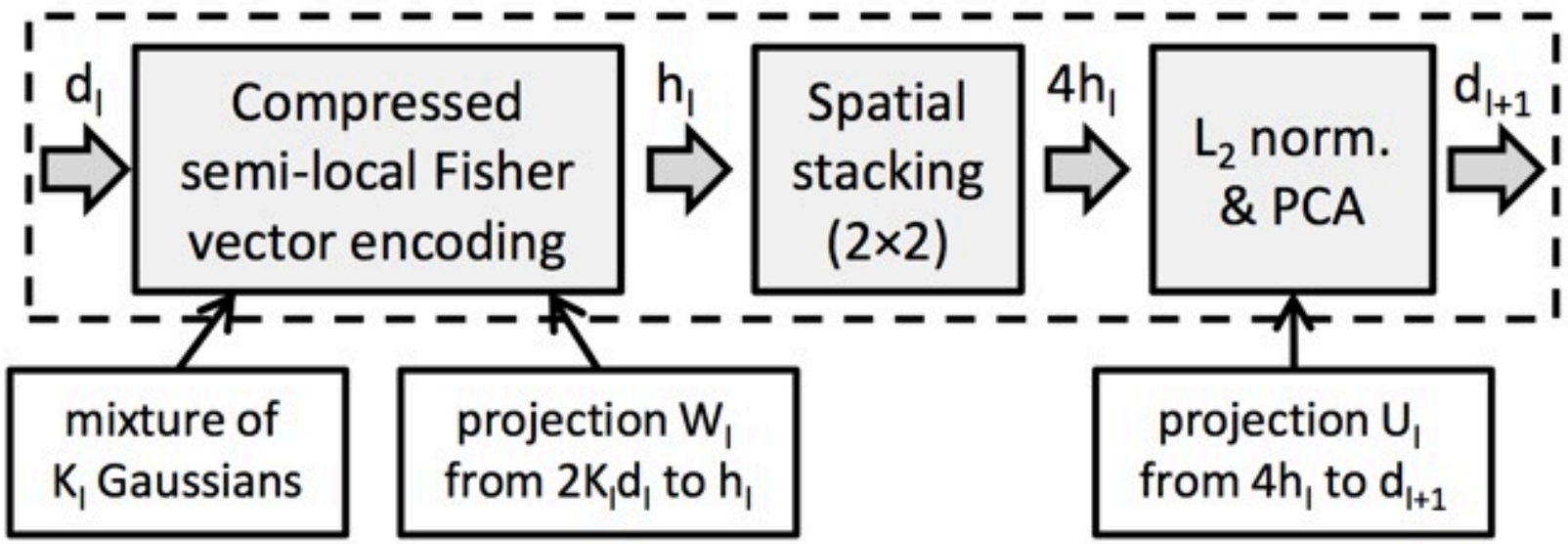


Image credit: Simonyan et al 2013

Single Fisher Layer



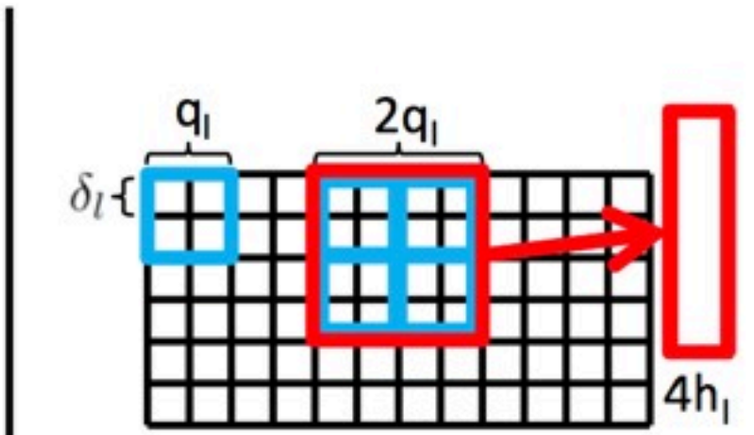
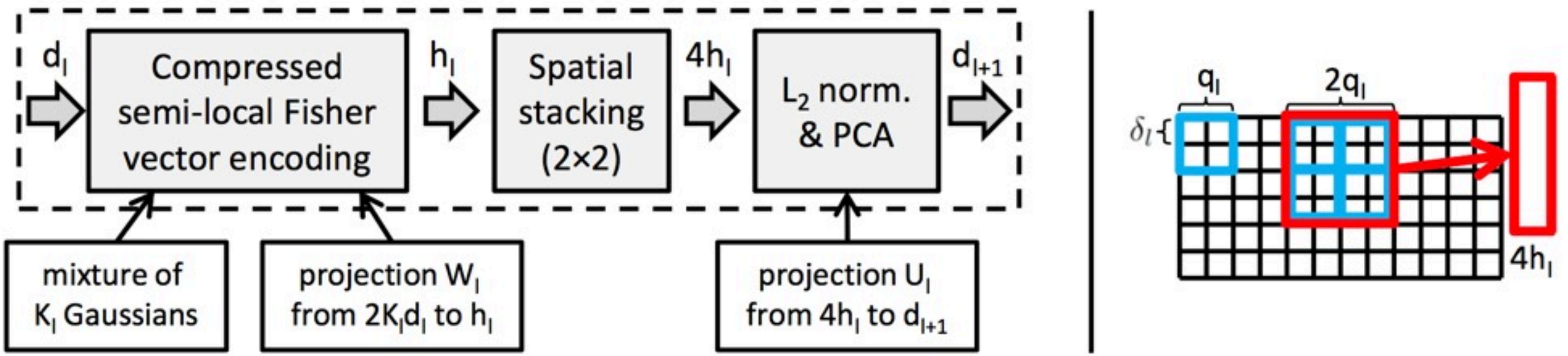


Image credit: Simonyan et al 2013

Single Fisher Layer



more details: Deep Fisher Networks for Large-Scale Image Classification
http://www.robots.ox.ac.uk/~vgg/publications/2013/Simonyan13b/simonyan13b.pdf

Image credit: Simonyan et al 2013

Evaluations - on Pascal VOC 2007

Method				mAP								
(a) FK	Lin	ss3	256	**61.69**	78.97	67.43	51.94	70.92	30.79	72.18	79.94	61.35
(b) SV	Lin	ss3	1024	**58.16**	74.32	63.79	47.02	69.44	29.06	66.46	77.31	60.18
(c) LLC	Lin	ss2	25k	**57.60**	71.05	62.85	47.40	67.67	25.21	62.70	77.02	59.59
(d) LLC-F	Lin	ss2	25k	**59.32**	74.10	64.92	51.48	68.33	27.18	62.89	78.44	61.39
(e) VQ	Chi	ss2	25k	**56.07**	70.00	58.90	42.86	66.75	26.59	62.27	75.67	57.09
(f) LLC	Lin	ss3	25k	**57.27**	71.35	62.65	46.12	68.98	26.04	63.92	76.98	59.71
(g) LLC	Sqr	ss3	25k	**56.71**	71.24	61.75	42.73	68.21	25.85	62.33	76.40	59.31
(h) LLC	Chi	ss3	25k	**57.66**	72.41	62.19	47.30	68.91	25.78	63.95	77.27	59.83
(i) LLC-F	Lin	ss3	25k	**59.74**	74.17	65.39	51.15	69.69	28.67	64.40	78.48	63.00
(j) VQ	Chi	ss3	25k	**55.30**	70.10	59.24	44.14	66.34	26.79	60.88	75.62	55.42
(k) KCB	Chi	ss3	25k	**56.26**	70.83	60.60	44.50	66.52	27.02	62.07	76.29	57.61
(l) LLC	Lin	ss5	25k	**56.96**	69.82	61.63	46.71	68.27	25.66	63.78	76.32	59.83
(m) LLC-F	Lin	ss5	25k	**58.70**	73.44	62.90	50.22	67.90	27.85	64.35	77.91	62.44
(n) VQ	Chi	ss5	25k	**53.87**	68.74	57.14	41.24	64.54	25.20	61.12	74.06	53.22
(o) LLC	Lin	ss3	14k	**56.18**	70.71	59.67	44.81	67.20	26.03	60.99	76.25	58.54
(p) VQ	Chi	ss3	14k	**54.82**	69.09	58.61	41.27	66.30	26.49	61.46	75.42	55.77
(q) LLC	Lin	ss3	10k	**56.01**	69.66	60.44	44.21	67.78	24.66	61.84	75.42	57.70
(r) VQ	Chi	ss3	10k	**54.98**	69.56	57.97	42.86	65.84	23.52	61.06	75.89	55.55
(s) LLC	Lin	ss3	4k	**53.79**	69.83	57.63	42.04	66.46	22.44	55.62	72.77	56.98
(t) LLC	Sqr	ss3	4k	**52.07**	68.52	54.62	40.14	65.34	21.53	51.89	71.54	55.19
(u) LLC	Chi	ss3	4k	**53.47**	70.17	56.20	42.73	65.27	22.23	55.18	72.78	56.95
(v) LLC-1	Lin	ss3	4k	**36.06**	53.39	43.20	22.47	46.32	11.40	29.48	64.66	45.41
(w) LLC-F	Lin	ss3	4k	**55.87**	72.27	61.41	44.08	67.85	24.97	57.92	75.40	59.44
(x) VQ	Sqr	ss3	4k	**51.97**	67.29	55.22	36.58	64.42	21.89	56.31	72.90	52.11
(y) VQ	Chi	ss3	4k	**53.42**	68.65	57.04	39.86	64.59	21.96	58.79	73.89	53.77
(z) VQ	Lin	ss3	4k	**46.54**	60.63	48.80	32.76	58.54	16.26	50.44	68.42	45.97
(α) KCB	Chi	ss3	4k	**54.60**	69.82	59.20	41.97	64.85	23.90	59.02	74.98	54.63

Table 1: Image classification results using Pascal VOC 2007 dataset *(continued on next page)*
VQ – baseline method; **FK** – Fisher kernel; **SV** – super vector coding; **KCB** – kernel codebook; **LLC** – locally-constrained linear coding; **LLC-F** – LLC encoding with original+left-right flipped training images; **LLC-1** – L1-normalized LLC encoding; **Lin/Sqr/Chi** – linear/hellinger/χ^2 kernel map; third column: SIFT sampling density; fourth column: visual vocabulary size

Results from: Chatfield et al 2011

Evaluations - on Pascal VOC 2007

(a) FK	55.98	49.61	58.40	44.77	78.84	70.81	84.96	31.72	51.00	56.41	80.24	57.46
(b) SV	50.19	46.46	51.86	44.07	77.85	67.12	83.07	27.56	48.50	51.10	75.50	52.26
(c) LLC	54.24	45.27	51.56	44.24	77.52	67.05	83.29	27.57	45.73	53.62	76.01	52.32
(d) LLC-F	54.41	47.24	52.75	44.55	78.10	68.48	83.73	29.90	50.95	55.48	78.55	53.62
(e) VQ	53.80	41.35	51.55	42.57	76.33	65.11	82.98	26.70	43.47	52.18	74.27	50.87
(f) LLC	53.96	46.34	52.10	42.39	77.17	67.15	83.36	23.11	44.45	52.12	75.36	52.21
(g) LLC	53.91	45.63	49.61	42.91	75.50	65.93	83.03	27.10	43.82	52.28	74.71	51.86
(h) LLC	54.25	45.97	51.11	43.22	76.74	67.07	83.49	27.73	44.77	52.77	76.00	52.47
(i) LLC-F	55.31	49.52	53.03	44.60	78.52	68.85	84.24	28.44	50.47	55.18	77.56	54.17
(j) VQ	53.12	40.40	51.27	39.16	76.87	63.09	82.16	20.96	42.96	51.40	75.13	50.94
(k) KCB	54.31	41.64	51.91	41.05	76.40	66.06	82.97	23.86	43.66	51.32	75.10	51.40
(l) LLC	53.17	44.89	51.03	40.70	76.88	66.84	82.78	26.19	45.46	52.17	75.11	52.03
(m) LLC-F	54.12	47.33	52.31	42.90	77.69	68.39	83.28	27.43	50.06	54.41	77.36	51.73
(n) VQ	52.08	38.96	48.66	38.64	74.84	62.40	81.24	25.08	40.46	49.35	72.19	48.22
(o) LLC	52.62	44.06	49.17	38.65	75.90	66.61	82.30	27.19	44.63	52.02	74.55	51.64
(p) VQ	52.68	40.18	50.58	38.14	75.25	63.87	82.23	21.36	42.19	51.72	74.10	49.64
(q) LLC	52.90	45.11	49.78	39.89	75.57	65.29	82.02	25.96	43.20	51.23	74.82	52.64
(r) VQ	52.97	41.09	49.64	41.28	76.33	64.24	81.94	21.49	41.91	52.95	74.06	49.51
(s) LLC	51.72	42.82	46.14	39.47	74.08	62.04	80.92	24.50	38.81	49.35	71.24	51.03
(t) LLC	50.83	40.10	42.63	39.47	71.69	58.57	79.74	22.89	38.98	47.78	70.87	49.09
(u) LLC	52.13	41.51	43.94	40.71	73.24	60.58	81.09	24.28	38.90	48.56	71.99	50.96
(v) LLC-1	32.84	19.06	21.34	22.72	53.28	33.59	68.74	11.38	6.17	37.97	60.44	37.30
(w) LLC-F	53.02	43.92	47.68	39.90	75.95	62.77	81.93	25.25	45.23	51.69	74.07	52.70
(x) VQ	51.51	38.23	46.50	34.99	74.62	60.71	80.05	18.79	37.13	50.22	71.71	48.32
(y) VQ	52.40	38.57	49.20	36.85	75.59	61.59	81.63	20.47	40.05	50.92	73.39	49.21
(z) VQ	48.25	28.31	41.73	30.21	70.51	55.24	77.79	14.79	31.37	39.73	68.27	42.73
(α) KCB	52.49	40.48	50.53	37.98	76.03	63.73	82.47	22.31	43.08	50.96	73.97	49.68

Results from: Chatfield et al 2011

Evaluations - on SUN 397

Table 5 Comparison of the FV with the state-of-the-art on SUN 397

Method	ntrain=5	ntrain=10	ntrain=20	ntrain=50
Xiao et al. (2010)	14.5	20.9	28.1	38.0
FV (SIFT)	19.2 (0.4)	26.6 (0.4)	34.2 (0.3)	43.3 (0.2)
FV (SIFT + LCS)	21.1 (0.3)	29.1 (0.3)	37.4 (0.3)	47.2 (0.2)

Image credit: Sanchez et al 2013

BoW Summary

Issues

- Sampling strategy
  dense uniform, interest points, random...
- Codebook learning
  supervised/unsupervised, size...
- Similarity measurement
  SVM, Pyramid Matching
- Spatial information
- Scalability

From Vectors to Codes

Given global image representation, want to learn compact binary codes for image retrieval task on large dataset

From Vectors to Codes

Motivation

- Tractable memory usage
- Constant lookup time
- Similarity preserved by hamming distance

From Vectors to Codes

Problem definition:

Given a database of images $\{x_i\}$ and a distance function

$D(i,j)$ seek a binary feature vector $y_i = f(x_i)$ that

preserves the nearest neighbor relationships using a

Hamming distance.

From Vectors to Codes

What makes a good code

- Easily computed for a novel input
- Requires a small number of bits to code the full dataset
- Maps similar items to similar binary codewords

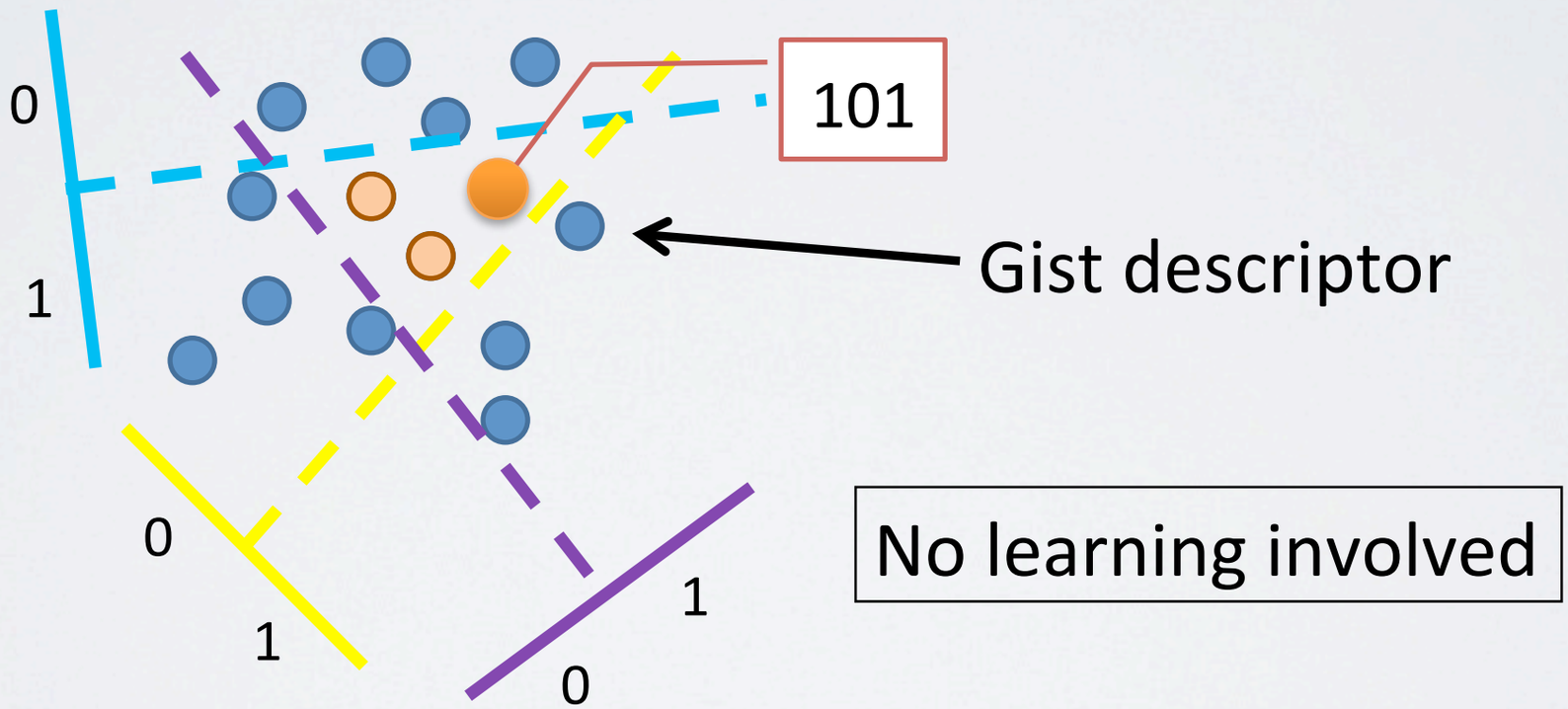
From Vectors to Codes

Approaches

- Locality Sensitive Hashing (LSH) [Andoni and Indyk]
- Boosting [Torralba et al 2008]
- Restricted Boltzmann Machines (RBM) [Torralba et al 2008]
- Spectral Hashing [Weiss et al 2008]
- Multidimensional Spectral Hashing [Weiss et al 2012]

Locality Sensitive Hashing

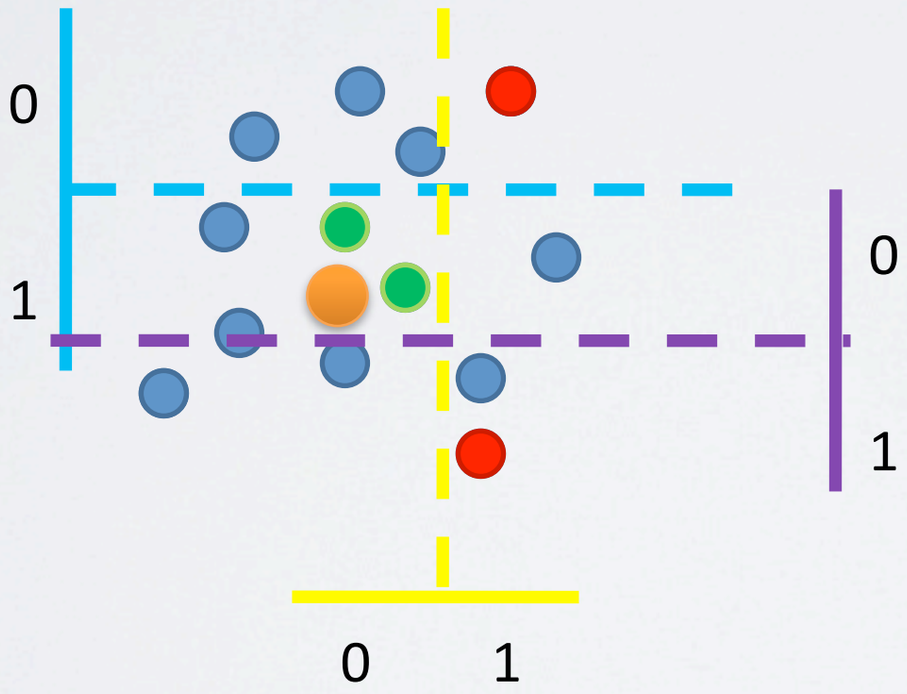
- Take random projections of data
- Quantize each projection with few bits



Adapted from slides by Weiss et al

Boosting

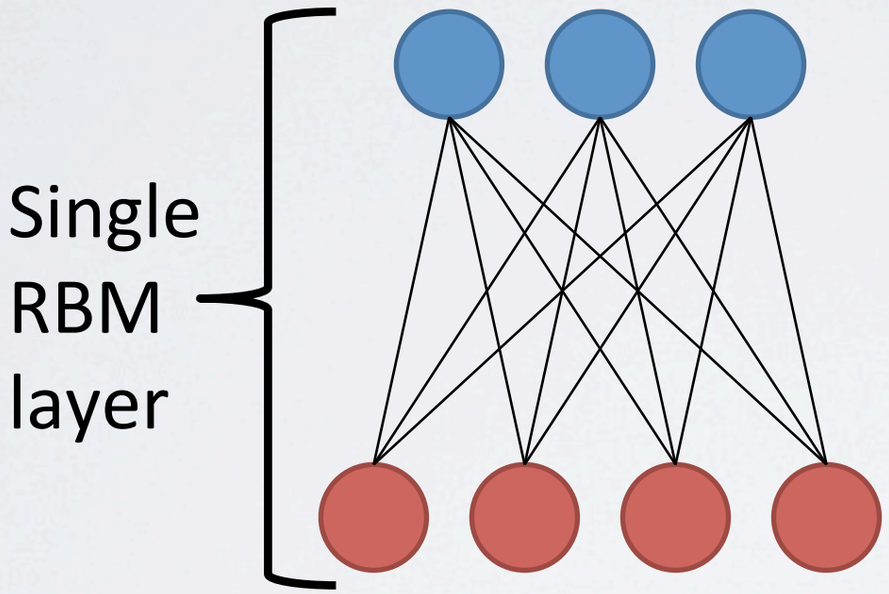
- Positive examples are pairs of similar images
- Negative examples are pairs of unrelated images



Learn threshold & dimension for each bit (weak classifier)

Adapted from slides by Weiss et al

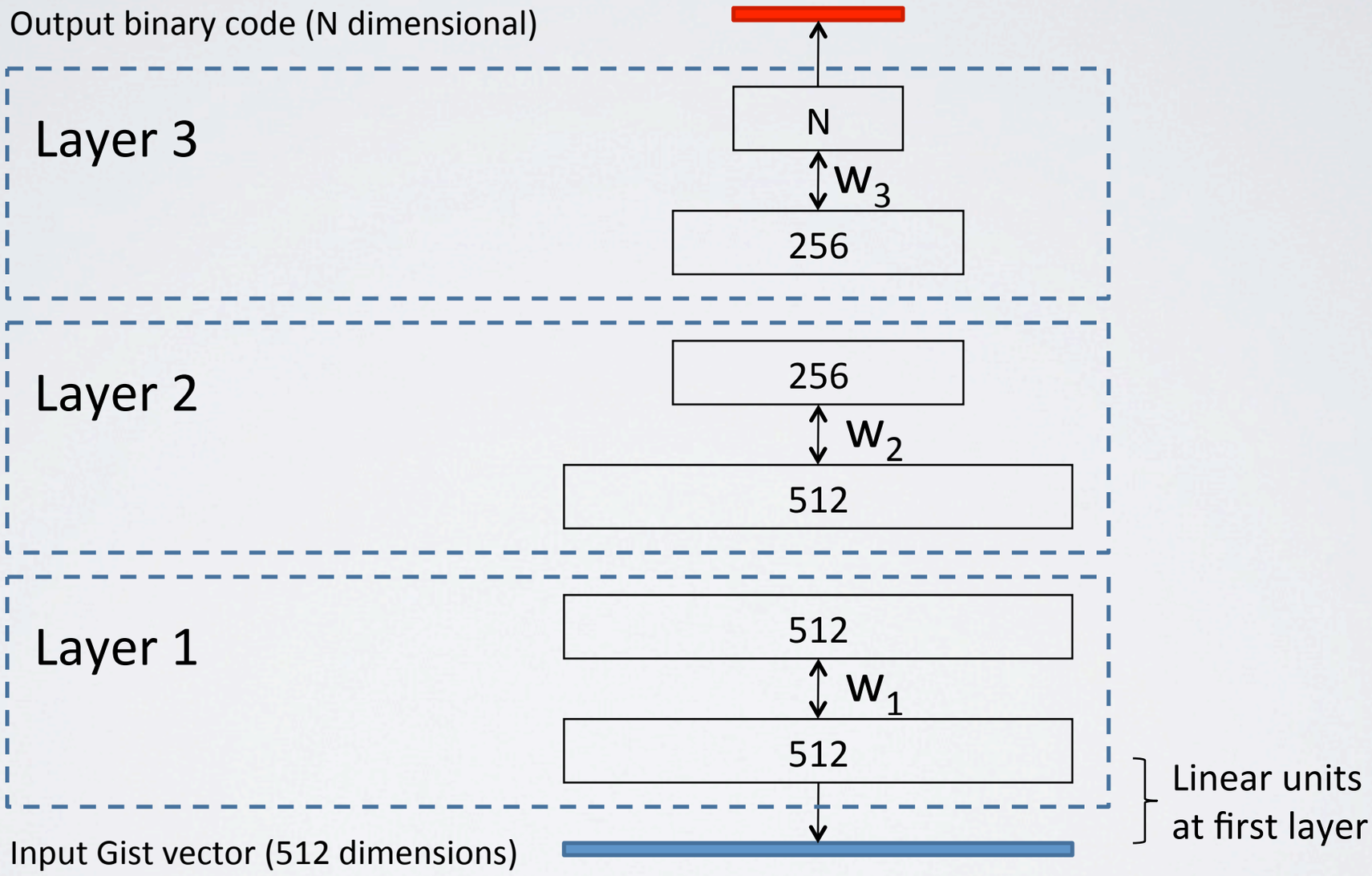
Restricted Boltzmann machines



Units are binary & stochastic

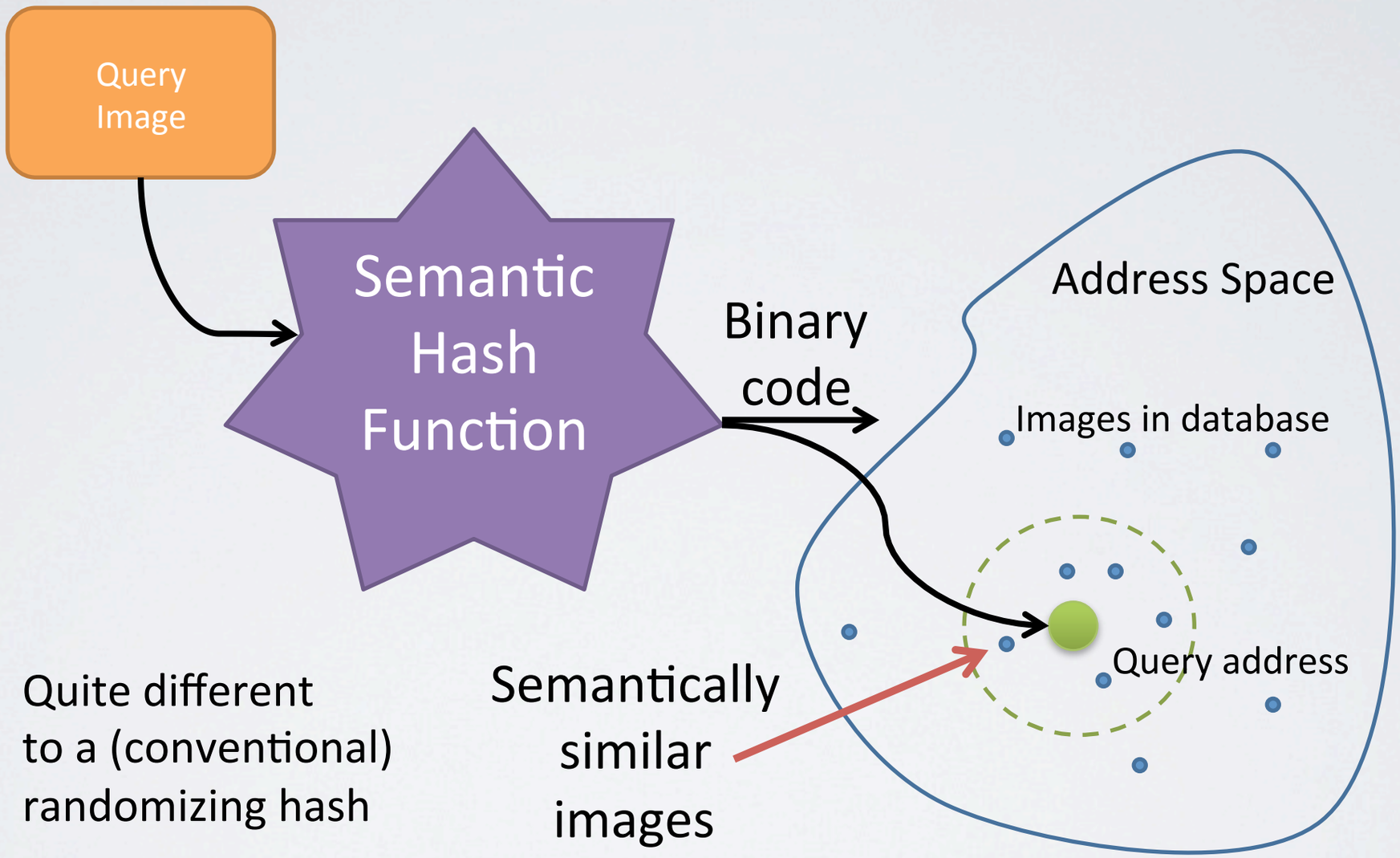
Adapted from slides by Weiss et al

Restricted Boltzmann machines



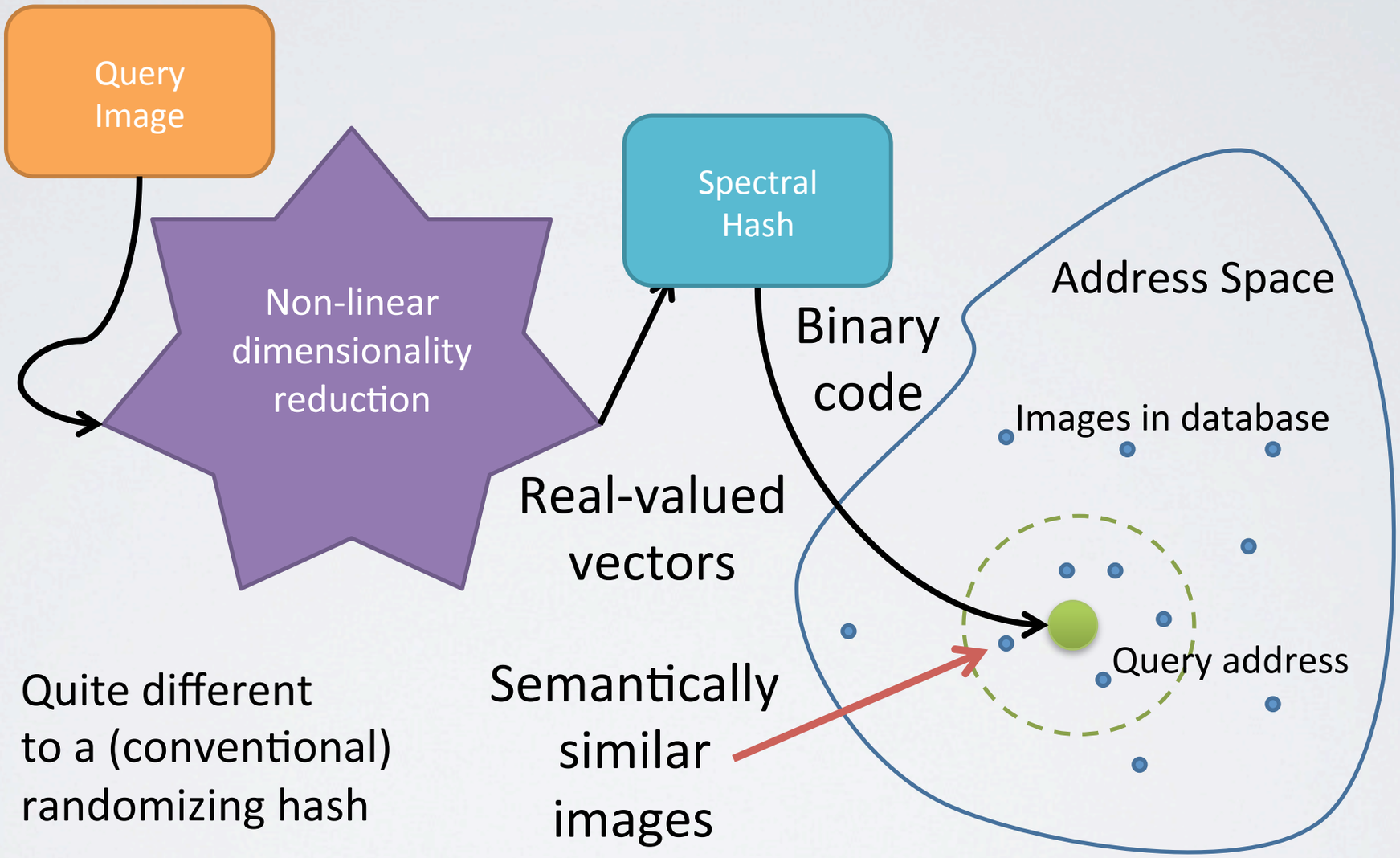
Adapted from slides by Weiss et al

Retrieval Algorithm: Semantic Hashing



Adapted from slides by Weiss et al

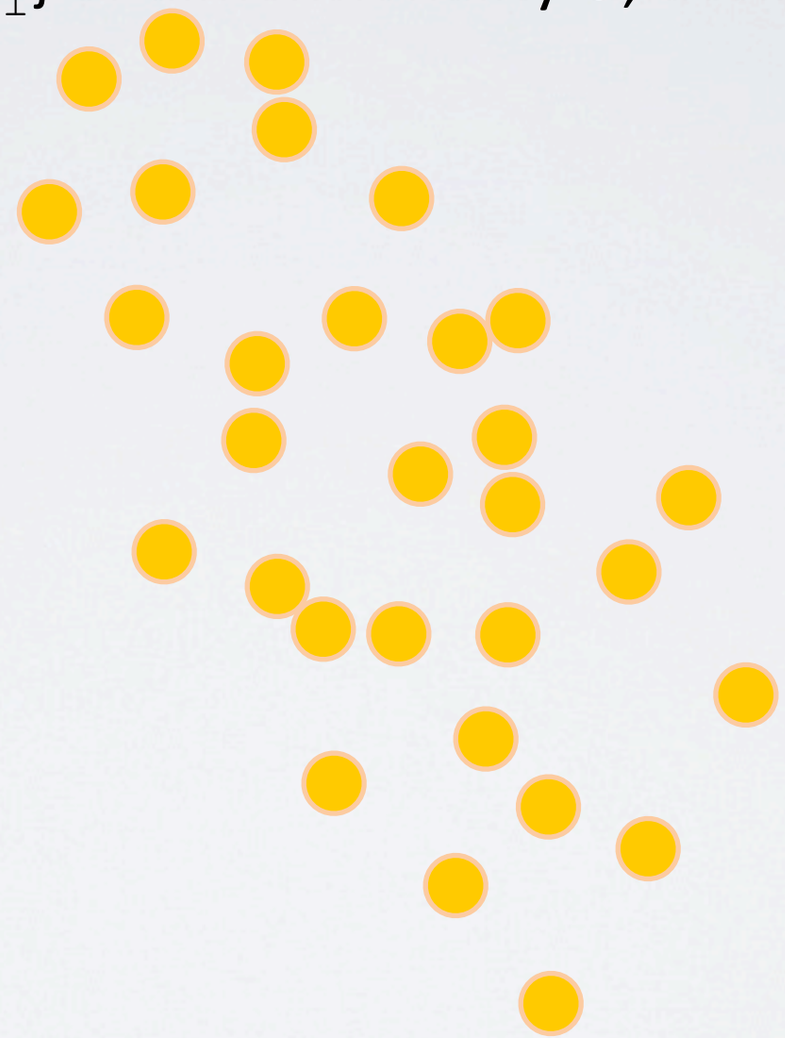
Spectral Hashing



Adapted from slides by Weiss et al

The Algorithm

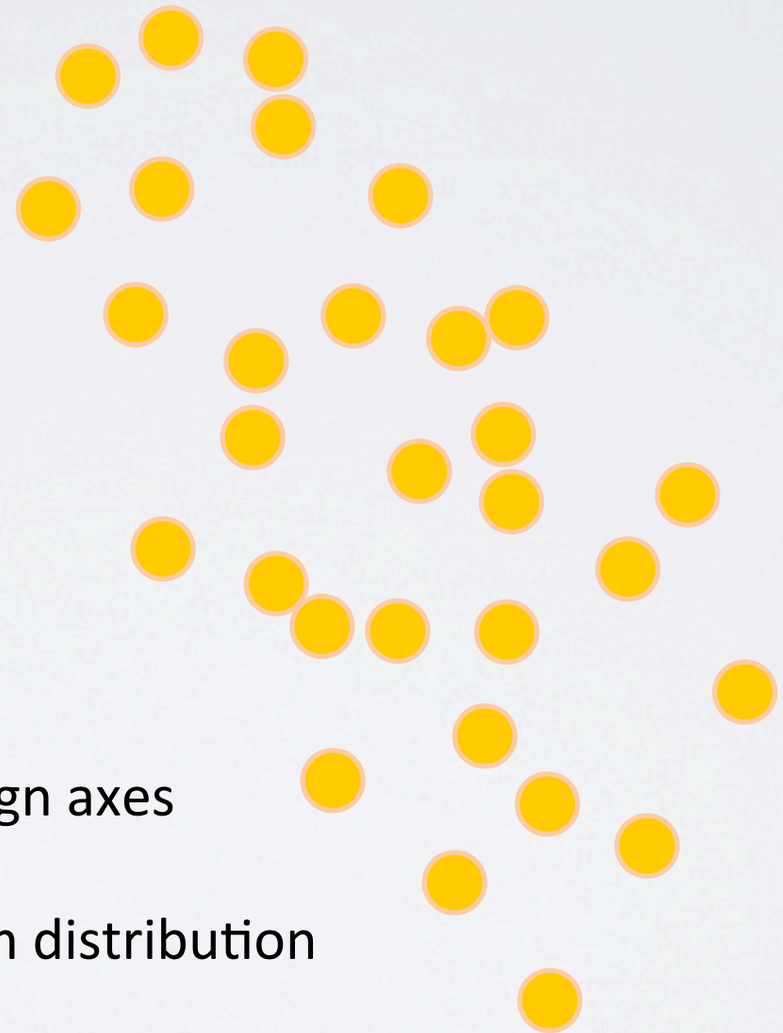
Input: Data $\{x_i\}$ of dimensionality d; desired # bits, k



Adapted from slides by Weiss et al

The Algorithm

- Fit multi-dimensional rectangle

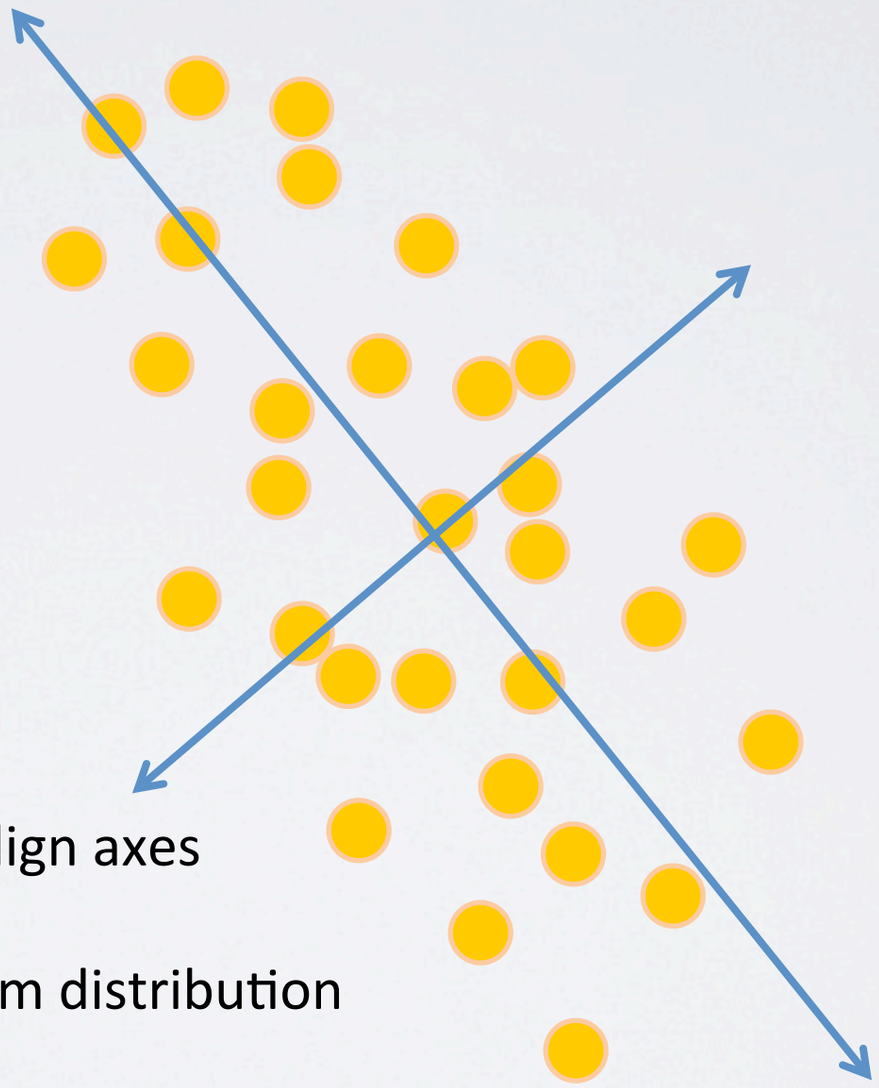


- Run PCA to align axes
- Bound uniform distribution

Adapted from slides by Weiss et al

The Algorithm

- Fit multi-dimensional rectangle

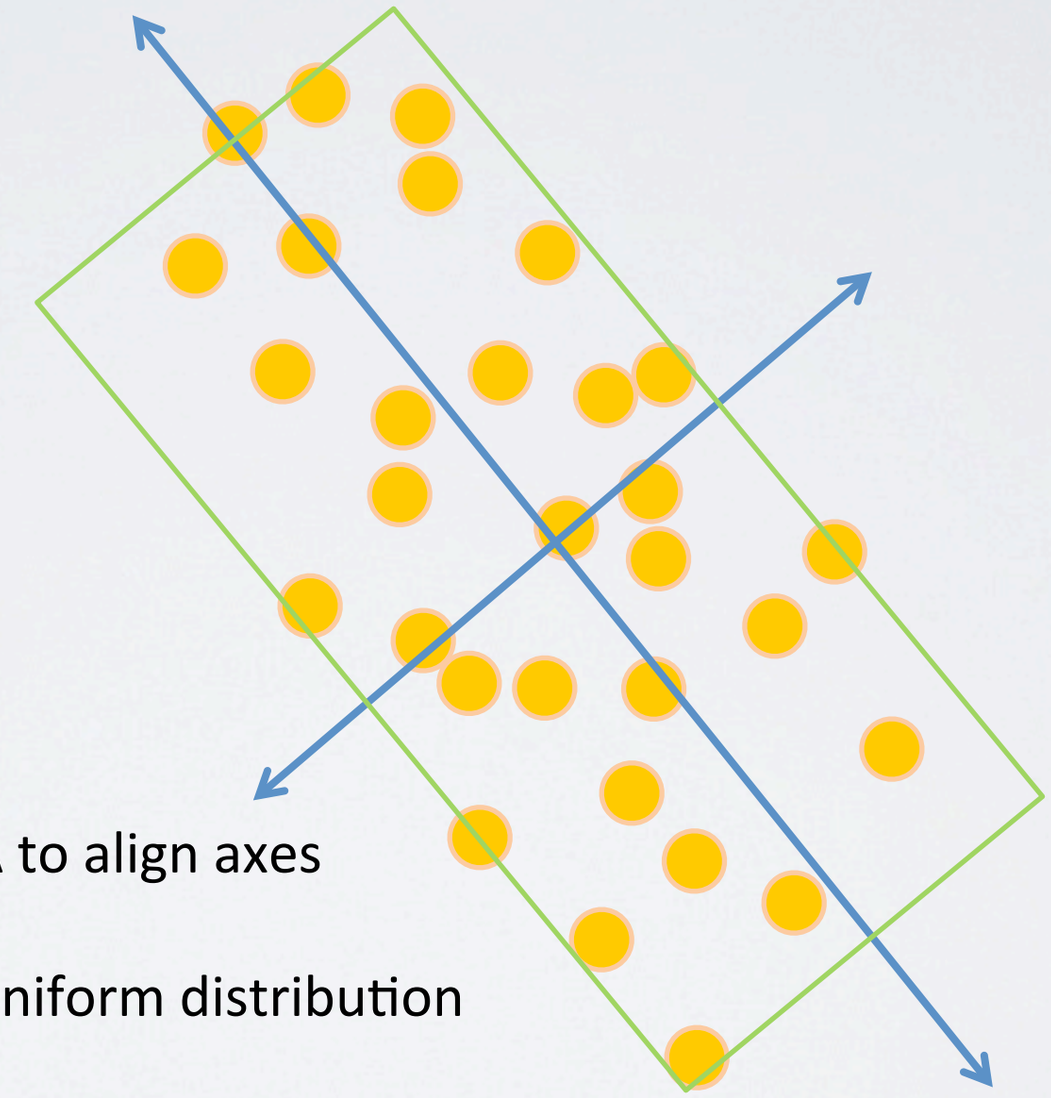


- Run PCA to align axes
- Bound uniform distribution

Adapted from slides by Weiss et al

The Algorithm

- Fit multi-dimensional rectangle

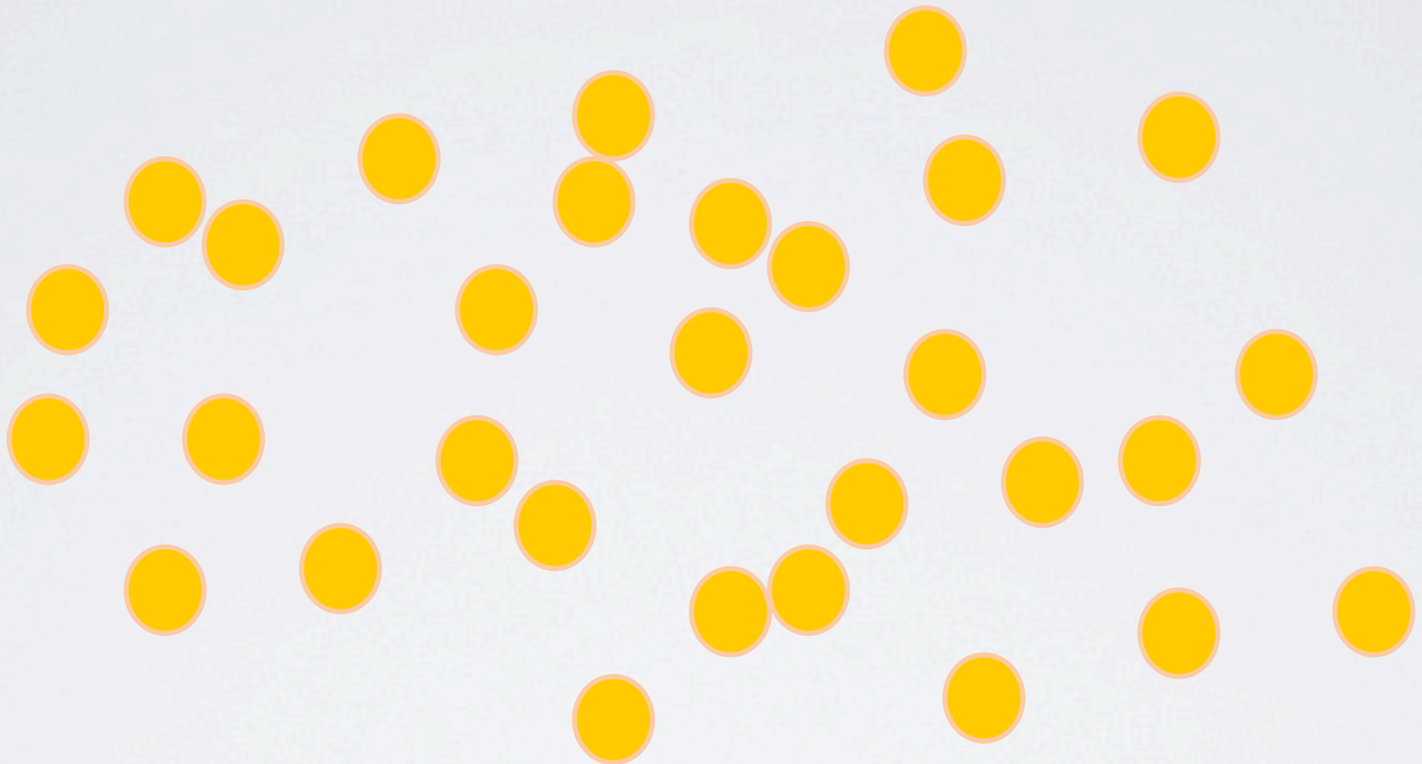


- Run PCA to align axes
- Bound uniform distribution

Adapted from slides by Weiss et al

The Algorithm

- Calculate Eigenfunctions



Adapted from slides by Weiss et al

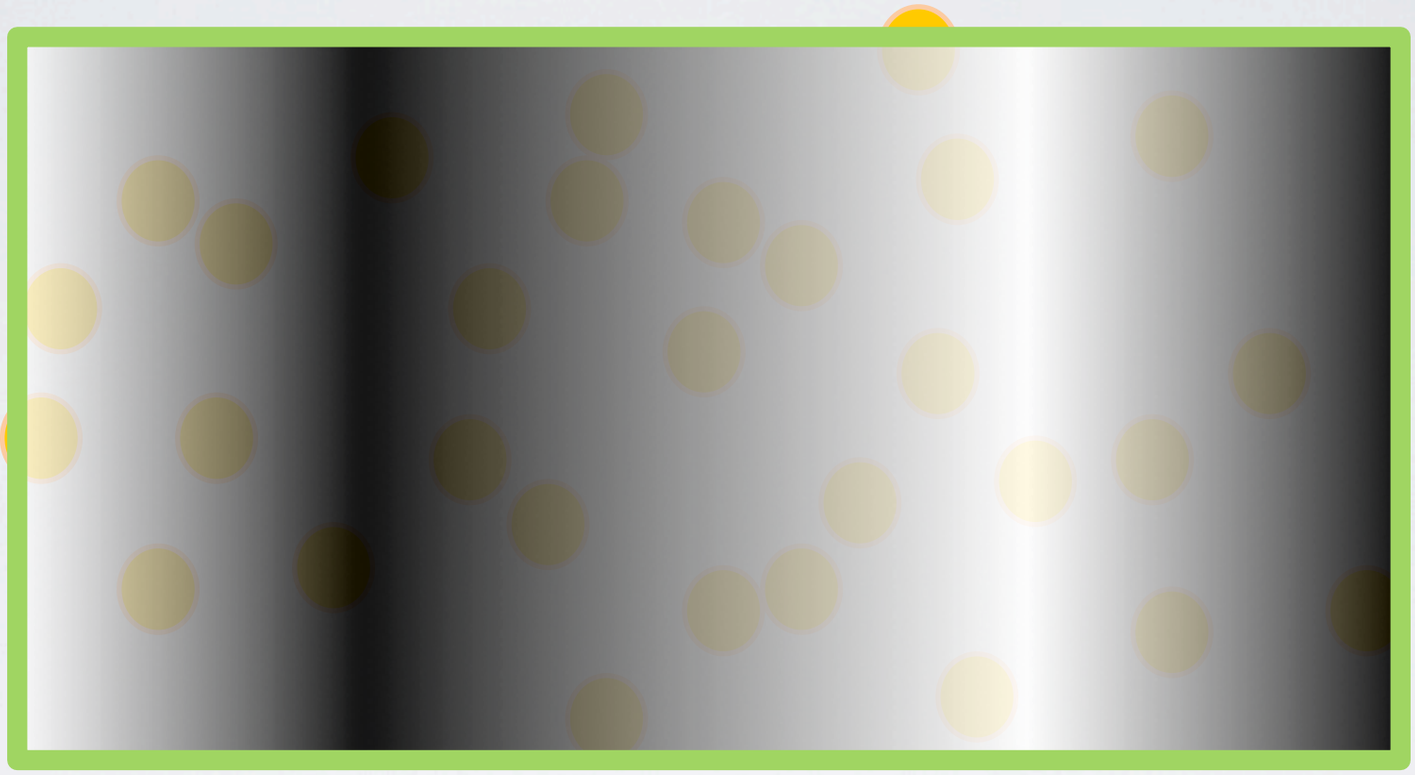
The Algorithm

- Calculate Eigenfunctions

Adapted from slides by Weiss et al

The Algorithm

- Calculate Eigenfunctions



Adapted from slides by Weiss et al

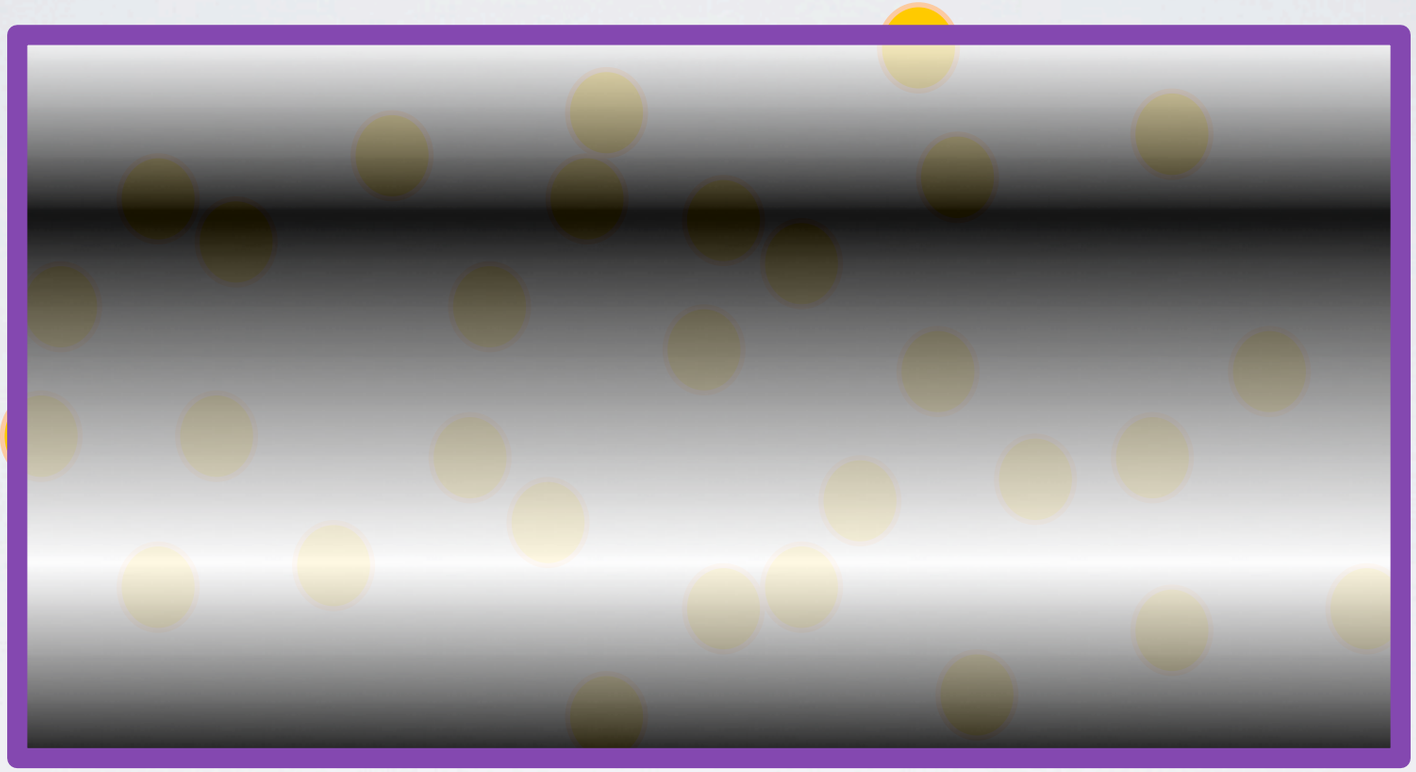
The Algorithm

- Calculate Eigenfunctions

Adapted from slides by Weiss et al

The Algorithm

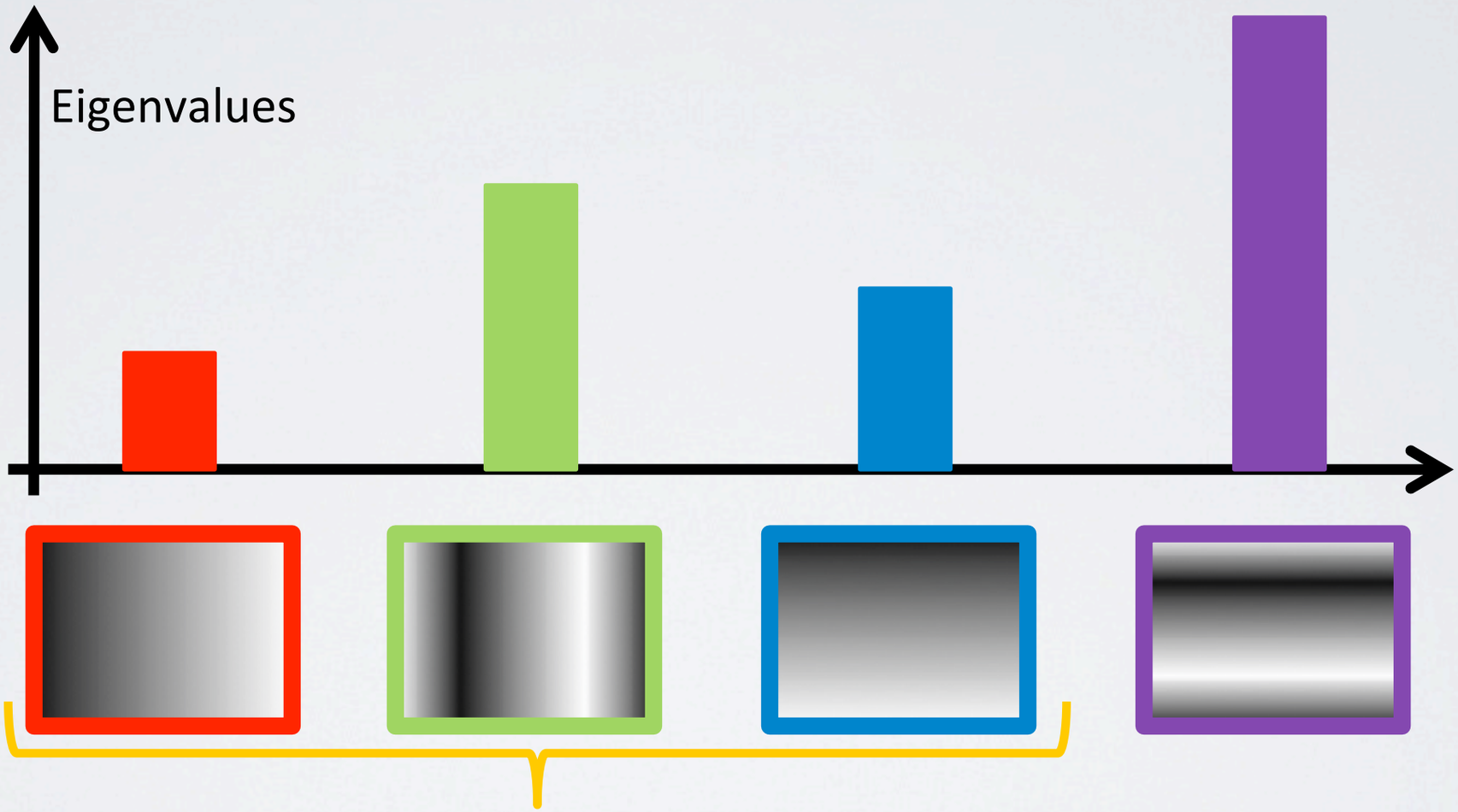
- Calculate Eigenfunctions



Adapted from slides by Weiss et al

The Algorithm

- Pick the *k* smallest Eigenfunctions

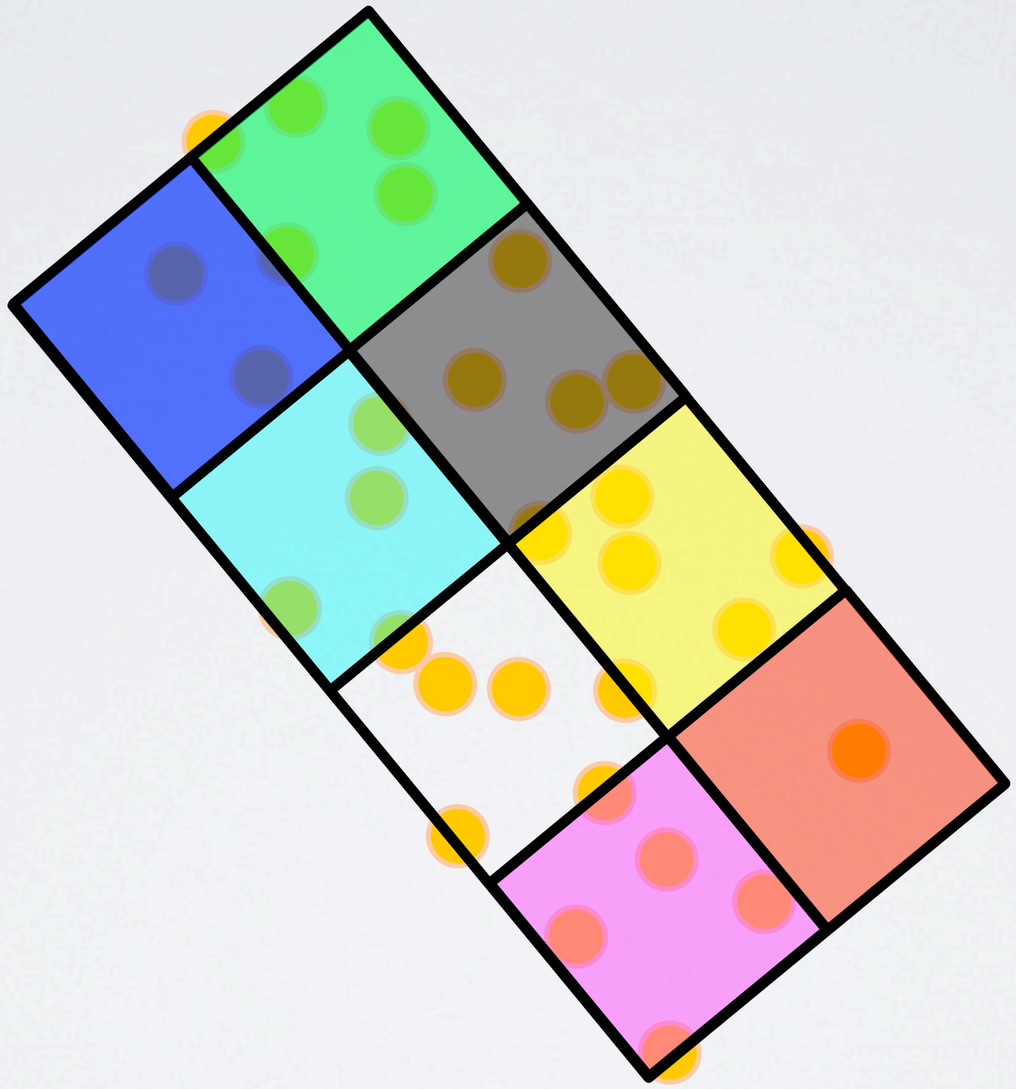


e.g. k=3

Adapted from slides by Weiss et al

The Algorithm

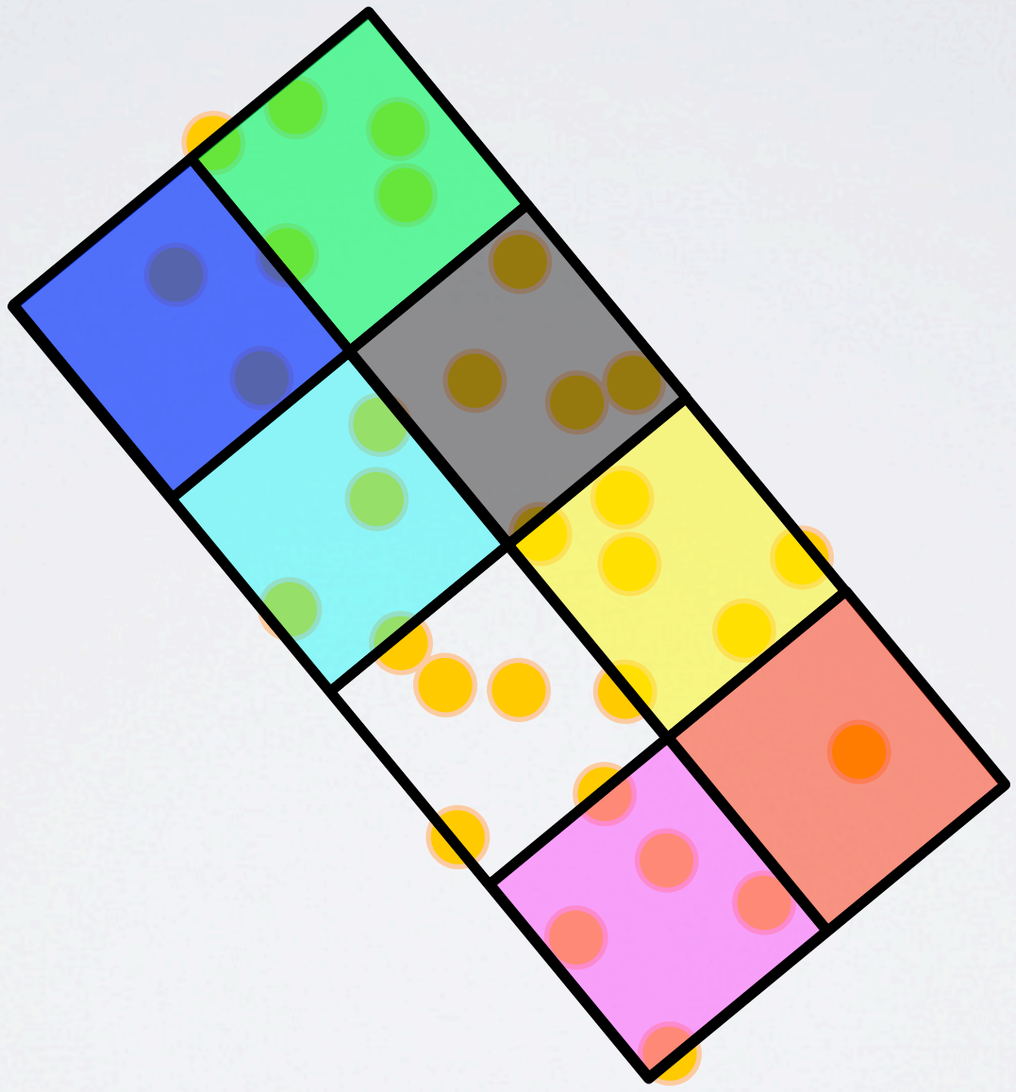
- Threshold the chosen Eigenfunctions



Adapted from slides by Weiss et al

The Algorithm

- Threshold the chosen Eigenfunctions



more details: Spectral Hashing, Yair Weiss, Antonio Torralba, Rob Fergus
http://people.csail.mit.edu/torralba/publications/spectralhashing.pdf

Summary

Image representation

- Bag of Words
- Spatial Pyramid Matching
- Descriptor Encoding
  - Sparse Coding
  - Locality-constrained Linear Coding
  - Fisher Vector
- Binary Code

Thank you