Object Recognition

Chapter 8: Recognition of Object Categories

Prof. Dr. Johannes Maucher

HdM CSM

Version 1.3
May 26th, 2020

Document History

Version Nr.	Date	Changes
1.0		Initial Version
1.1	29.05.2018	Idea of visual words from BoW included
1.2	02.05.2019	Integrated Subsection Visual Words Design Choices
1.3	26.05.2020	Adaptations for SS 20

Chapter 8: Generic Object Recognition

Idea

- From document retrieval and document classification:
  - Documents are typically described as vectors, which count the occurence of each word in the document. These vectors are called Bag of words.
  - Rows of Document/Word matrix are the BoWs. The transpose of this matrix is an index, which can be applied for document retrieval.

	Word 1	Word 2	Word 3	$\cdots$	Word Z
Document 1	2	0	1	$\cdots$	0
Document 2	0	0	3	$\cdots$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Document N	2	2	0	$\cdots$	0

Adapt the idea from documents to images

Analogy:

- Instead of documents we now have images
- Instead of words, which describe the documents, we now have local features, which describe the images

Problem:

Adapt the idea from documents to images

Analogy:

- Instead of documents we now have images
- Instead of words, which describe the documents, we now have local features, which describe the images

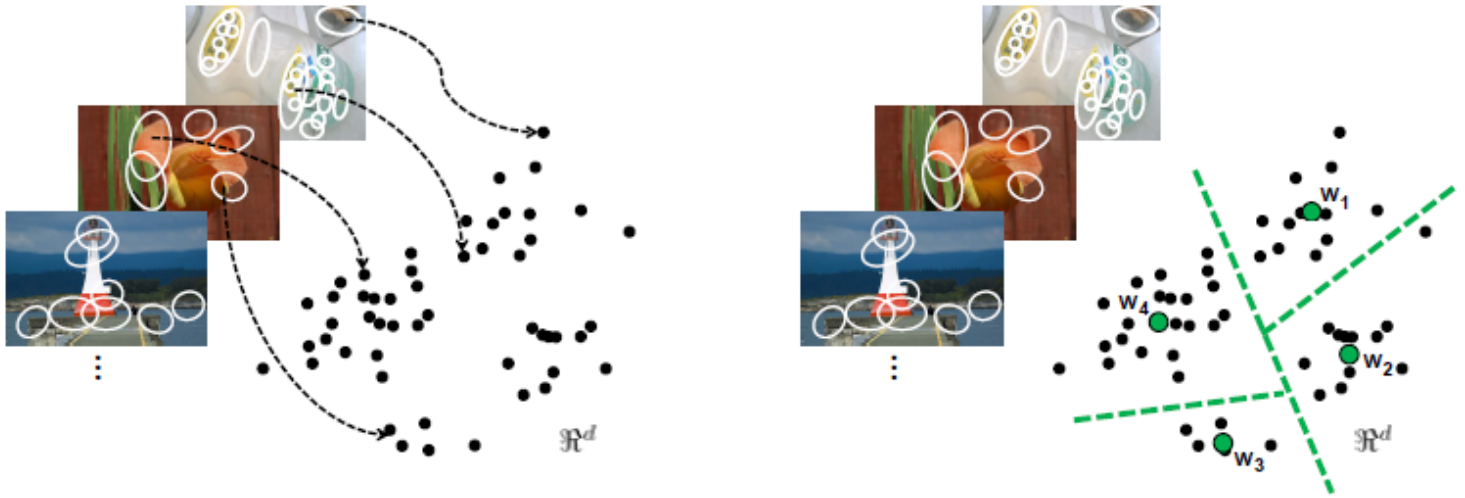
Problem:

- There exists a discrete set of words
- The set of possible local feature descriptors is infinite.

Solution:

- Quantize the infinite space of local feature descriptors into a set of discrete clusters.
- For each cluster assign a cluster representative (typical the cluster center).
- The discrete set of cluster centers constitute the visual vocabulary.

Calculate Visual Words



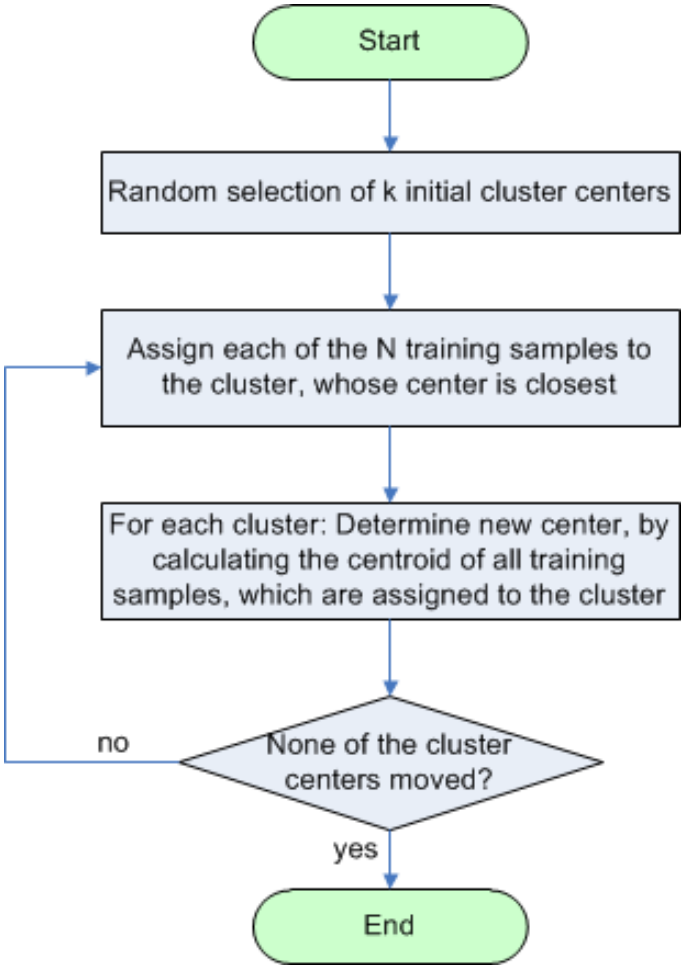
Source: [Grauman and Leibe, 2011]

- Local features are described by high dimensional descriptors (Typical: $d = 128$ dimensions).
- Each local feature descriptor is a point in the $d-$dimensional space $\mathcal{R}^d$.
- The $d-$dimensional space is partitioned into K cells (=clusters).
- Cluster centers are the visual words
- The set of all K cluster centers is the visual vocabulary

Creating a visual vocabulary

- How to quantize the $d-$dimensional space $\mathcal{R}^d$?
- Collect large sample of local feature descriptors from a representative set of images (Training Data).
- Quantize the space according to the statistics of this given set of local feature descriptors.
- Standard algorithm for this task: K-Means Clustering Algorithm.
- The cluster centers are the visual words. As soon as the visual words are computed, the local features of the training data can be discarded.

K-Means Clustering (=Vector Quantisation) Training



K-means finds local minimum of reconstruction error

- Assign Input $\mathbf{x}_p \in T$ to cluster C_i if cluster center $\mathbf{v}_i$ is the center which is closest to $\mathbf{x}_p$:

$$\|\mathbf{x}_p - \mathbf{v}_i\| = \min_{j \in 1 \ldots K} \|\mathbf{x}_p - \mathbf{v}_j\|, \tag{1}$$

  where K is the number of clusters and $\|\mathbf{x} - \mathbf{v}\|$ is the distance between $\mathbf{x}$ and $\mathbf{v}$.
- The cluster center $\mathbf{v}_i$ of cluster C_i is

$$\mathbf{v}_i = \frac{1}{n_i} \sum_{\forall \mathbf{x}_p \in C_i} \mathbf{x}_p, \tag{2}$$

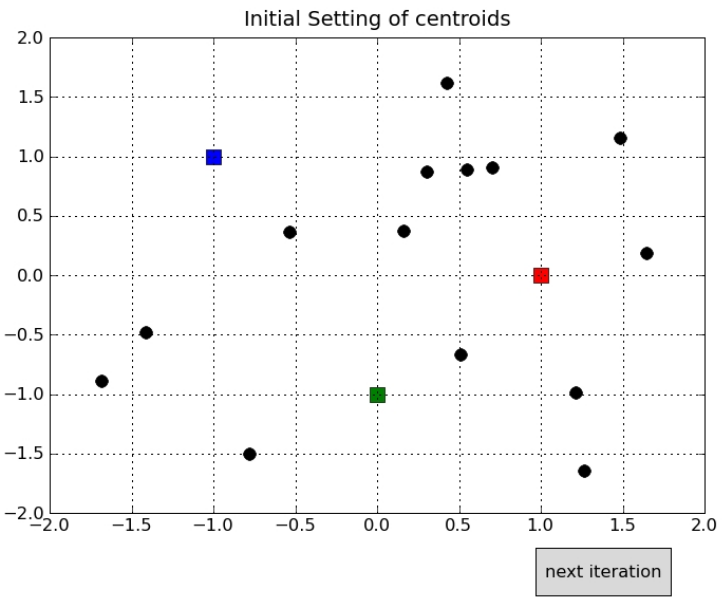
  where n_i is the number of vectors that are assigned to C_i.
- Reconstruction Error of the Clustering is the sum over all distances between input vectors and their corresponding cluster center:

$$E = \sum_{i=1}^{K} \sum_{\forall \mathbf{x}_p \in C_i} \|\mathbf{x}_p - \mathbf{v}_i\| \tag{3}$$

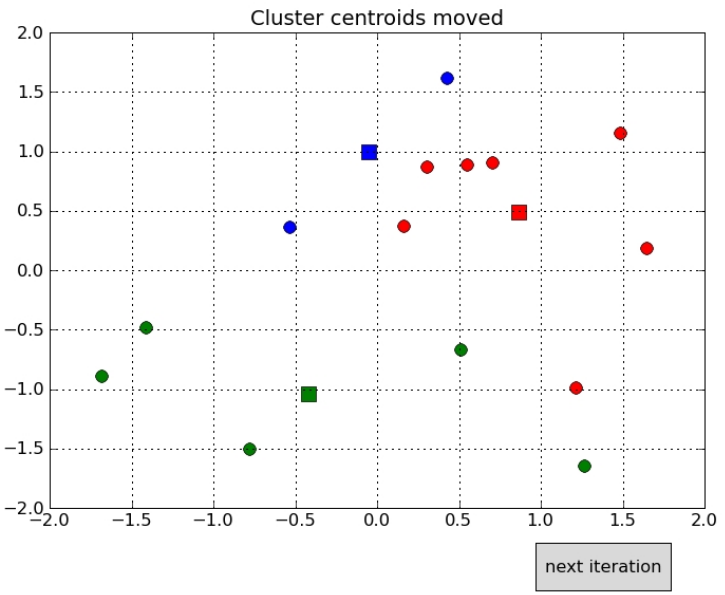
- K-Means Clustering terminates in a local minimum of the reconstruction error.

Beispiel k-means Clustering von 14 Objekten (1)

Initiale Konfiguration

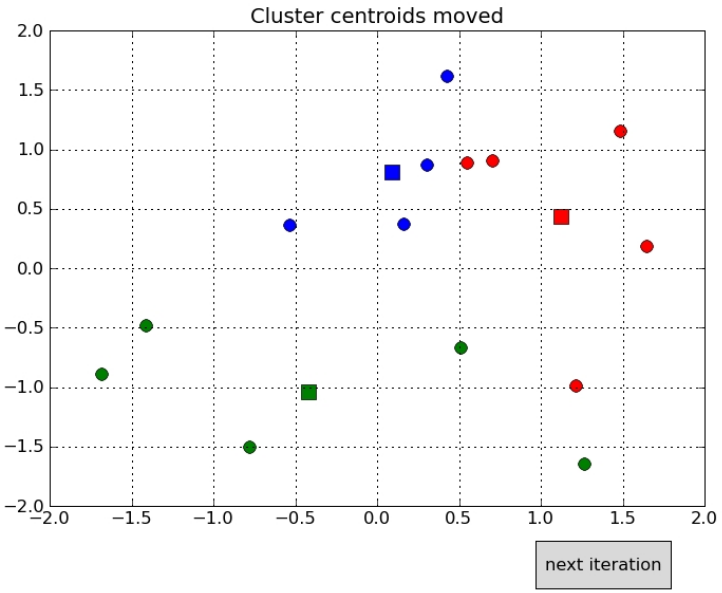


Iteration 1

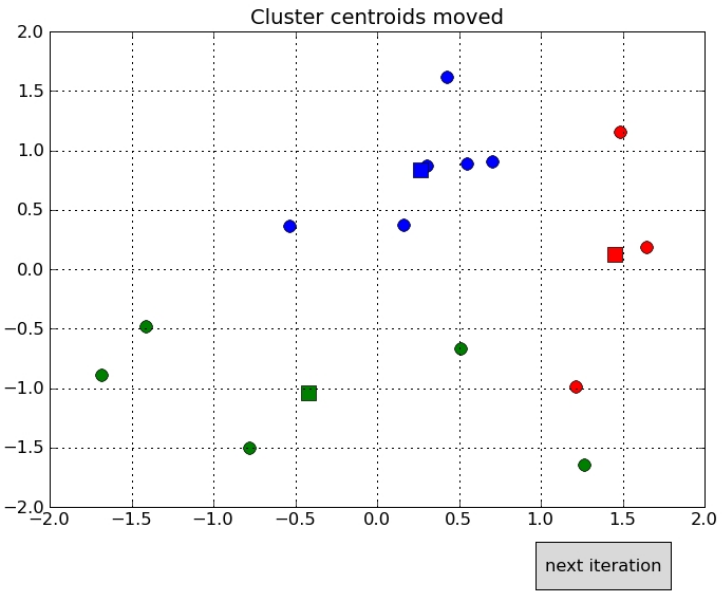


Beispiel k-means Clustering von 14 Objekten (2)

Iteration 2

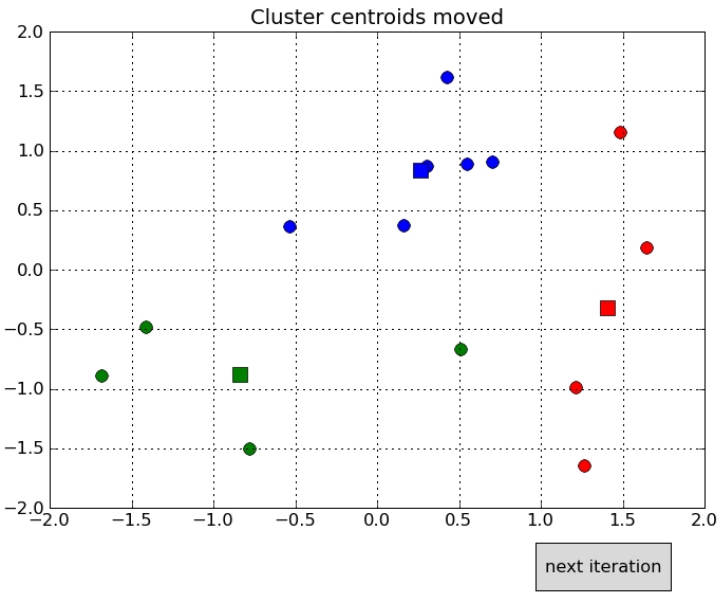


Iteration 3

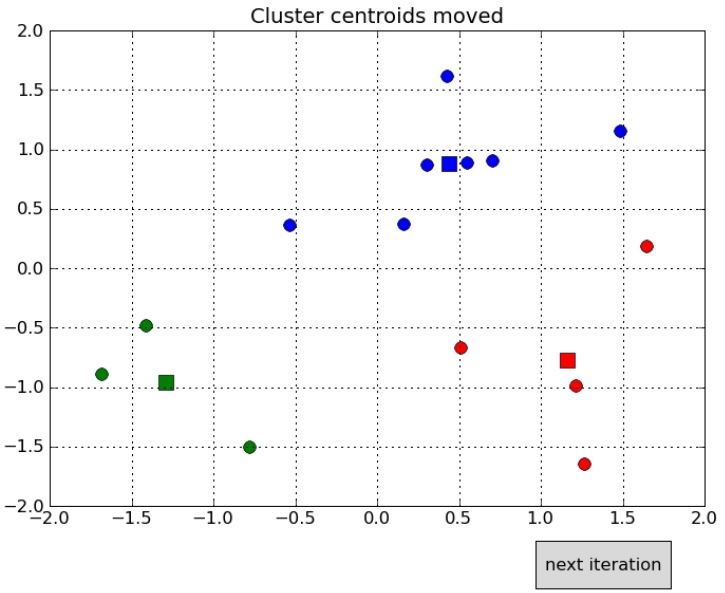


Beispiel k-means Clustering von 14 Objekten (3)

Iteration 4



Iteration 5 (Stabiler Endzustand)



Example: Similar local patches are mapped to the same visual word

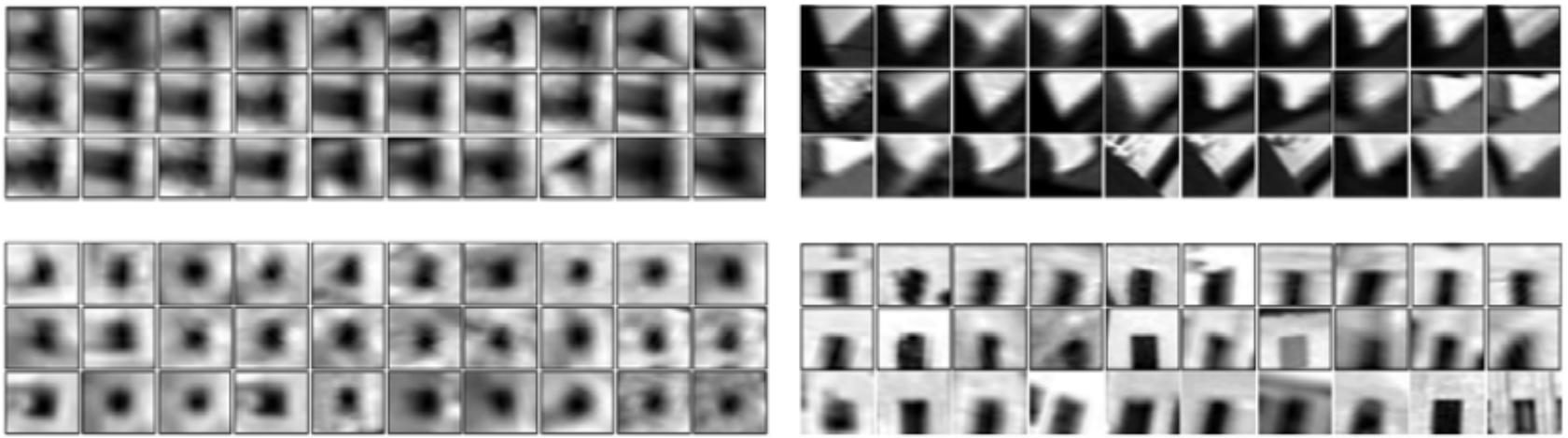
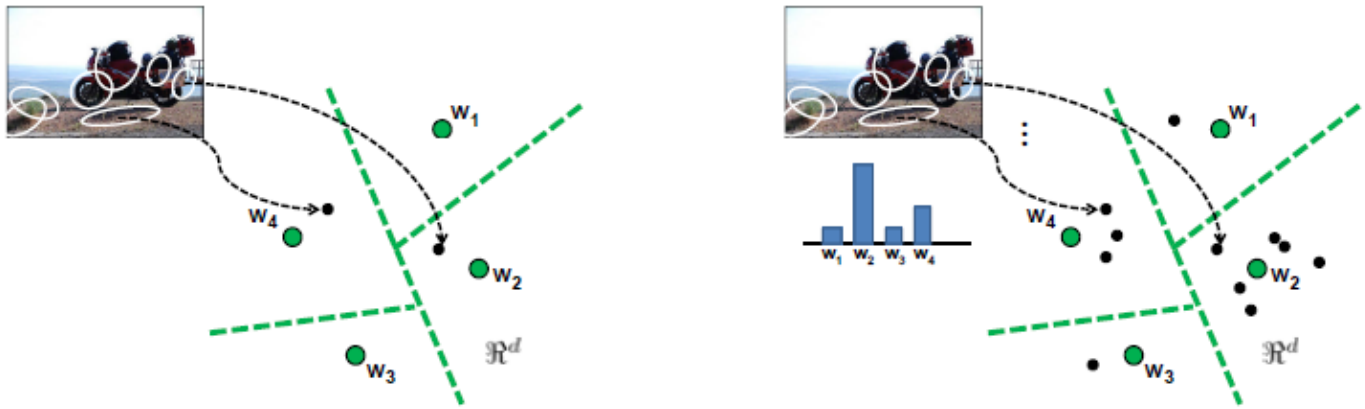


Figure: Each of the four groups displays patches, that are mapped to the same visual words. Source: [Sivic and Zisserman, 2003]

Mapping of local features to visual words



Source: [Grauman and Leibe, 2011]

- Local features of novel images can be mapped to clusters, i.e. to visual words, by determining the cluster center, which is closest to the local feature (Euclidean distance).
- A Bag of Word (BoW) Descriptor is a $K-$dimensional vector. The $i.th$ component of this vector describes how often the $i.th$ visual word appears in the image.

Visual Word representation of images

- Set of N images: $I = \{I_i\}_{i=1}^{N}$
- Set of K visual words $V = \{v_t\}_{t=1}^{K}$
- $N(t, i)$ is the number of times visual word v_t occurs in image I_i. $N(t, i)$ is the entry in row i, column t of the following matrix.

	v_1	v_2	v_3	$\cdots$	v_K
I_1	2	0	1	$\cdots$	0
I_2	0	0	3	$\cdots$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
I_N	2	2	0	$\cdots$	0

Once images are described in this form any Machine Learning classification algorithm can be applied for recognizing object category. E.g. Naive Bayes, k-Nearest Neighbour, SVMs, Neural Networks, ...

Problems with k-Means clustering

- Number of clusters k must be selected by the user. What is a good value of k?
- K-means finds only a local minimum of the reconstruction error. The result strongly depends on the initial setting of the cluster centers.

How to cope with this problem?

- [Csurka et al., 2004] proposes to find appropriate k and good vocabulary by
  - Iterate over a wide range of values for k
  - For each k restart k-means 10 times to obtain 10 different vocabularies
  - For each k and each vocabulary train and test a multiclass classifier (e.g. Naive Bayes)
  - Select k and vocabulary, which yields lowest classification error rate.

Naive Byes Classification

- Naive Bayes has already been introduced in chapter *Global Features*.
- Assume that a query image I_q shall be assigned to one of L object categories $C_1, C_2, \ldots, C_L$.
- The trained Naive Bayes Classifier calculates the a-posteriori probability

$$P(C_j|I_q) = \frac{\prod_{t=1}^{K} P(v_t|C_j)^{N(t,q)} P(C_j)}{P(I_q)} \tag{4}$$

for all classes C_j. The class which maximizes this expression is the most probable category.

- Maximizing

$$P^*(C_j|I_q) = \prod_{t=1}^{K} P(v_t|C_j)^{N(t,q)} P(C_j) \tag{5}$$

yields the same result as maximizing equation (4) and is easier to compute.

Naive Bayes Training

- Required training data: Set of N images, each labeled with the class C_j of the displayed object.
- For all classes C_j determine the a-priori probabilities

$$P(C_j) = \frac{|C_j|}{N}, \tag{6}$$

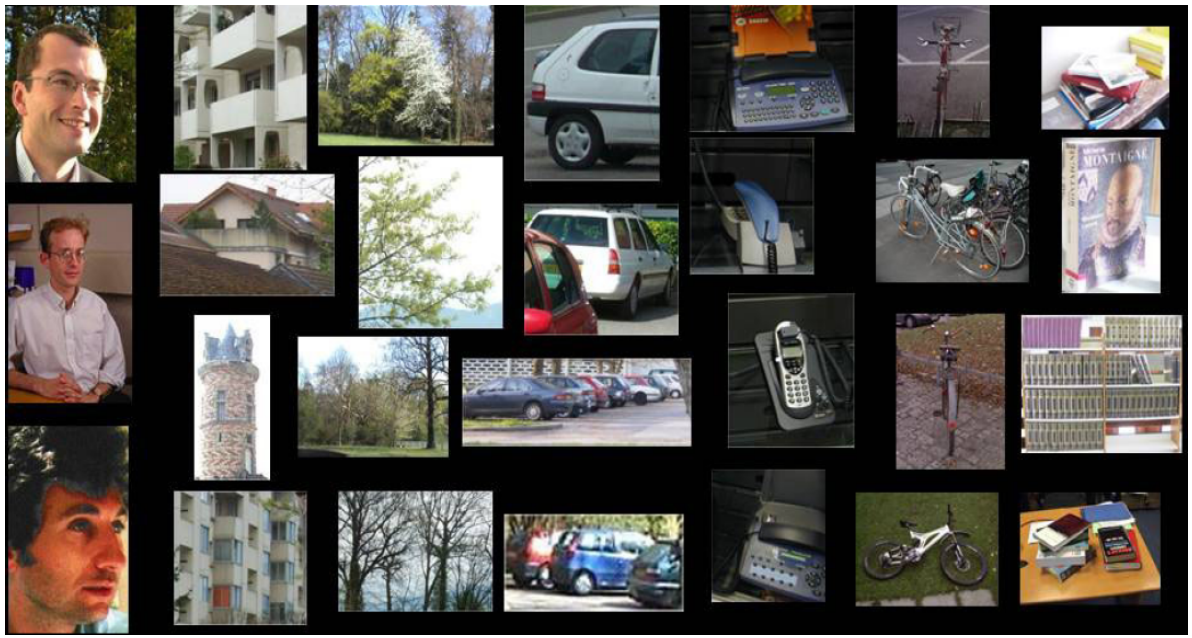
where $|C_j|$ is the number of images in the training corpus, that belong to class C_j.
- For all visual words v_t and all classes C_j determine the likelihood

$$P(v_t|C_j) = \frac{\sum_{I_i \in C_j} N(t,i)}{\sum_{s=1}^{K} \sum_{I_i \in C_j} N(s,i)} \tag{7}$$

- In order to avoid likelihoods of value 0 Laplace Smoothing is applied instead of the equation above:

$$P(v_t|C_j) = \frac{1 + \sum_{I_i \in C_j} N(t,i)}{K + \sum_{s=1}^{K} \sum_{I_i \in C_j} N(s,i)} \tag{8}$$

Experimental Results from [Csurka et al., 2004]: Dataset



- 1776 images, belonging to 7 different classes
- Highly variable poses, significant amount of background clutter
- Image resolution between 0.3 and 2.1 megapixels
- Color images, but only luminance channel is used

Experimental Results from [Csurka et al., 2004]: Performance Metrics

- Confusion Matrix

$$M_{i,j} = |\{I_k \in C_j \ : \ h(I_k) = C_i\}| \tag{9}$$

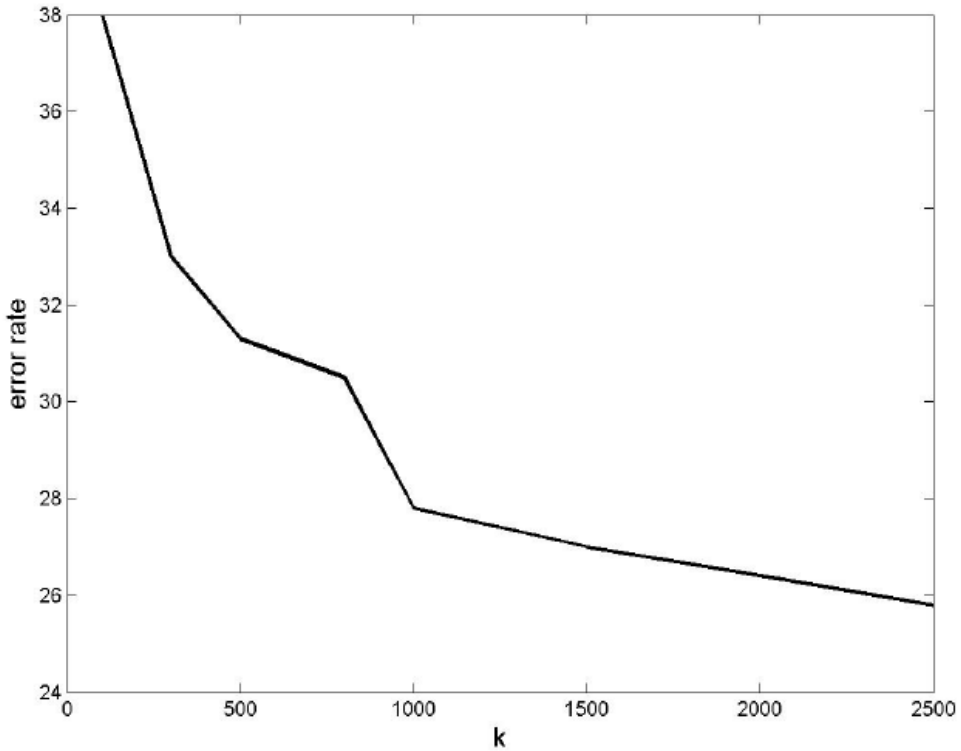
where $h(I_k)$ is the classifier's output for input image I_k.

- Overall Error Rate

$$R = 1 - \frac{\sum_{j=1}^{L} M_{j,j}}{\sum_{j=1}^{L} |C_j|} \tag{10}$$

- Mean Ranks: Mean position of the correct label, when labels are sorted in decreasing order w.r.t. the classifier score, defined in equation (5).

Experimental Results from [Csurka et al., 2004]: Error Rate vs. vocabulary size



Source: [Csurka et al., 2004] Only slight improvement for $k > 1000$.

Experimental Results from [Csurka et al., 2004]: Naive Bayes Result

Table 1. Confusion matrix and the mean rank for the best vocabulary (*k=1000*).

True classes →	*faces*	*buildings*	*trees*	*cars*	*phones*	*bikes*	*books*
faces	**76**	4	2	3	4	4	13
buildings	2	**44**	5	0	5	1	3
trees	3	2	**80**	0	0	5	0
cars	4	1	0	**75**	3	1	4
phones	9	15	1	16	**70**	14	11
bikes	2	15	12	0	8	**73**	0
books	4	19	0	6	7	2	**69**
Mean ranks	1.49	1.88	1.33	1.33	1.63	1.57	1.57

Experimental Results from [Csurka et al., 2004]: SVM Result

Table 2. Confusion matrix and mean rank for SVM (*k=1000, linear kernel*).

True classes →	*faces*	*buildings*	*trees*	*cars*	*phones*	*bikes*	*books*
faces	**98**	14	10	10	34	0	13
buildings	1	**63**	3	0	3	1	6
trees	1	10	**81**	1	0	6	0
cars	0	1	1	**85**	5	0	5
phones	0	5	4	3	**55**	2	3
bikes	0	4	1	0	1	**91**	0
books	0	3	0	1	2	0	**73**
Mean ranks	1.04	1.77	1.28	1.30	1.83	1.09	1.39

- Drop of error rate from 28% (Naive Bayes) to 15% (SVM).
- Same vocabulary as for Naive Bayes has been applied.
- C-Parameter of SVM: $C = 0.005$

Visual Vocabularies: Design Choices

- Which type of local features?
- How to sample local features?
- How to select training dataset, used for constructing the vocabulary?
- Supervised versus unsupervised training?
- How much visual words?
- What shall be counted in the Bag of Words? Word Frequency, TF-IDF, binary, ... ([Sivic and Zisserman, 2003]),...
- Algorithm to construct the vocabulary (training) (see [Coates and Ng, 2011])
- Algorithm to map new features to visual words (encoding) (see [Coates and Ng, 2011])

Sampling of local features

Figure: Sparse sampling (left), uniform dense sampling (center), random sampling (right).

- For identification (finding specific objects) sparse sampling often yields best performance.
- For object categorization, dense sampling offers better coverage (more local features).

Source:
`http://www.vision.rwth-aachen.de/teaching/lecture_computer_vision/winter-12-13/lectures/`

Choice of Training Data

- Best results if same images are applied for calculating the vocabulary as for the classification- or retrieval task.
- When training a recognition system for a particular set of categories, descriptors from training examples from all categories should be sampled [Grauman and Leibe, 2011].

Number of visual words, Count of Words

- In [Sivic and Zisserman, 2003] (Video Google) a vocabulary of 10000 visual words has been applied. Each video frame contained in the order of 1000 visual words.
- In the same work for the video google retrieval system tf-idf,shows better performance than tf (number of visual words in the image) and tf performs better than binary count.

Term Frequency $tf_{i,j}$ counts how often word i appears in document j

Inverse Document Frequency idf_i log of inverse ratio of documents that contain word i:

$$idf_i = \log\left(\frac{N}{n_i}\right)$$

where N is the number of documents, and n_i is the number of documents, that contain word i.

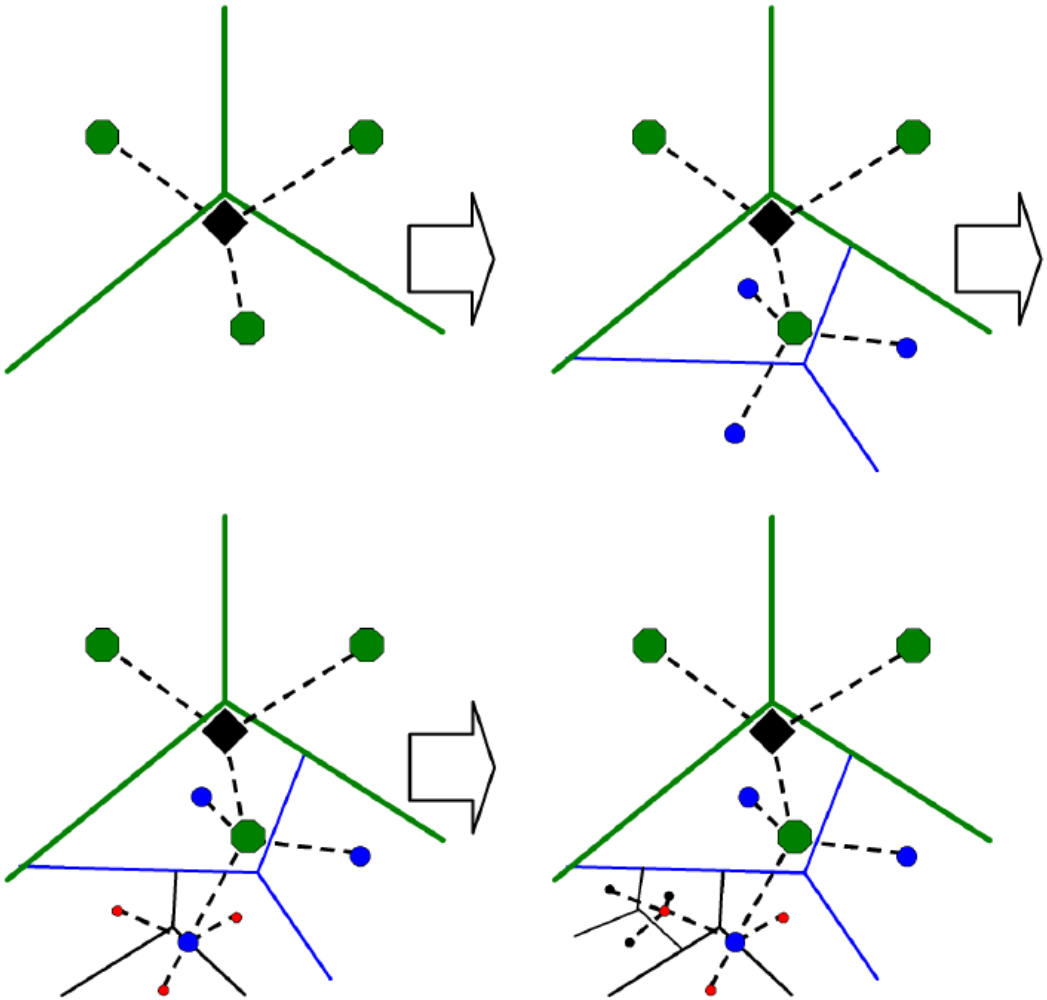
Term Frequency - Inverse Document Frequency:

$$tf\text{-}idf_{i,j} = tf_{i,j} \cdot idf_i$$

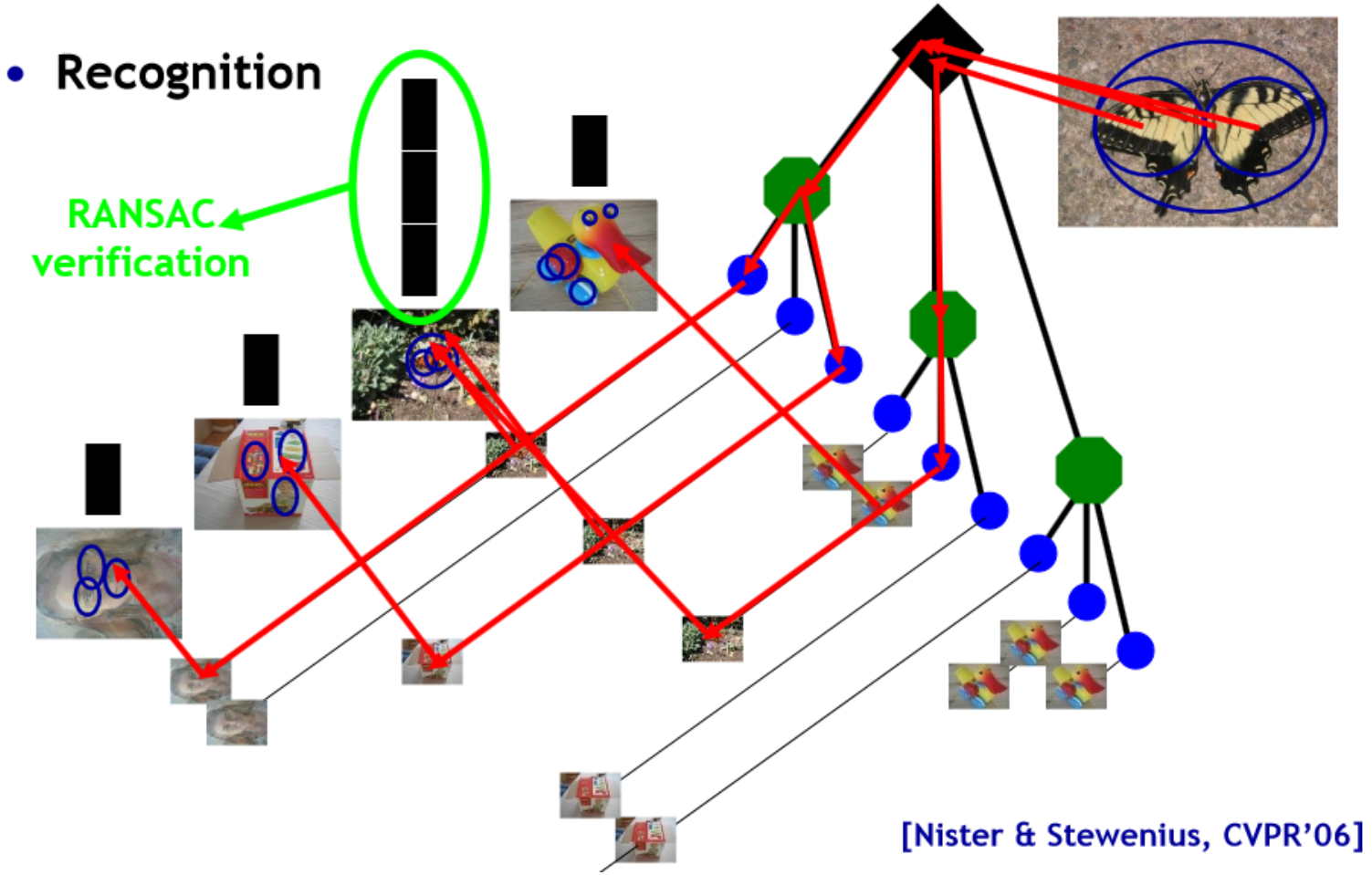
Algorithms for training and encoding: Vocabulary Trees

- Instead of applying k-means for flat clustering, hierarchical k-means can be applied to generate Vocabulary Trees (see [Nister and Stewenius, 2006]).
- Main Benefit: Computational cost of assigning new image features to words is logarithmic in the size of the vocabulary, instead of linear.
- Thus larger vocabularies can be applied, which offers better performance in specific object recognition.
- *Moreover, we have found that for a large range of vocabulary sizes (up to somewhere between 1 and 16 million leaf nodes), the retrieval performance increases with the number of leaf nodes.*[Nister and Stewenius, 2006].

Algorithms for training and encoding: Vocabulary Trees



Algorithms for training and encoding: Vocabulary Trees - Recognition



Source:

Algorithms for training and encoding: Vocabulary Trees - Performance

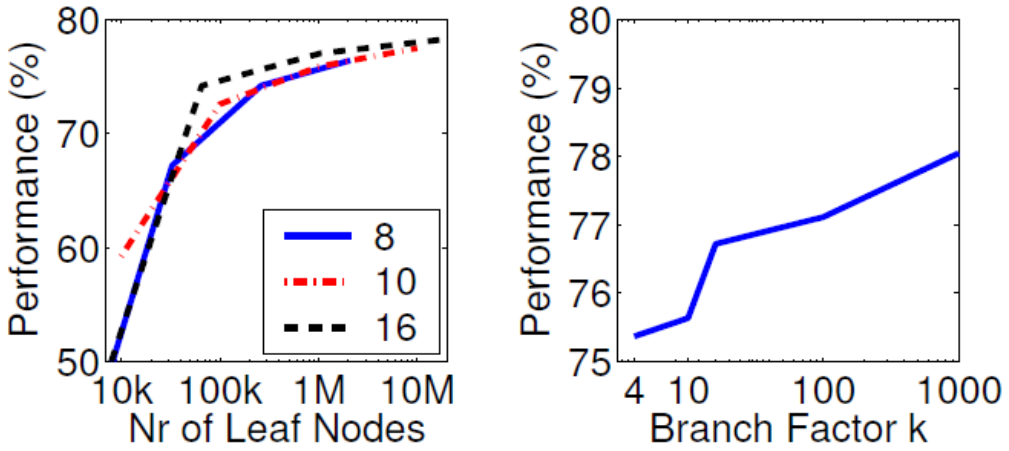


Figure 7. Vocabulary tree shapes tested on the 6376 ground truth image set. Left: Performance vs number of leaf nodes with branch factor $k = 8$, 10 and 16. Right: Performance vs k for $1M$ leaf nodes. Performance increases significantly with number of leaf nodes, and some, but not dramatically with the branch factor.

Source: [Nister and Stewenius, 2006]

Algorithms for training and encoding: Vocabulary Trees - Real Time Image Retrieval

Figure 10. A snapshot of the CD-cover recognition running. With 40000 images in the database, the retrieval is still real-time and robust to occlusion, specularities, viewpoint, rotation and scale changes. The camera is directly connected to the laptop via firewire. The captured frames are shown on the top left, and the top of the query is displayed on the bottom right. Some of the CD-covers are also connected to music that is played upon successful recognition.

Source: [Nister and Stewenius, 2006]

Algorithms for training and encoding: Sparse Coding

- Instead of applying k-means[1] recently sparse coding approaches, such as Restricted Boltzman Machines (RBM) show better performance.
- Comparison of k-means (vector quantization) and several sparse coding approaches can be found in [Coates and Ng, 2011].

[1] The application of k-means is also called vector quantization

Bag of Words: Pros and Cons

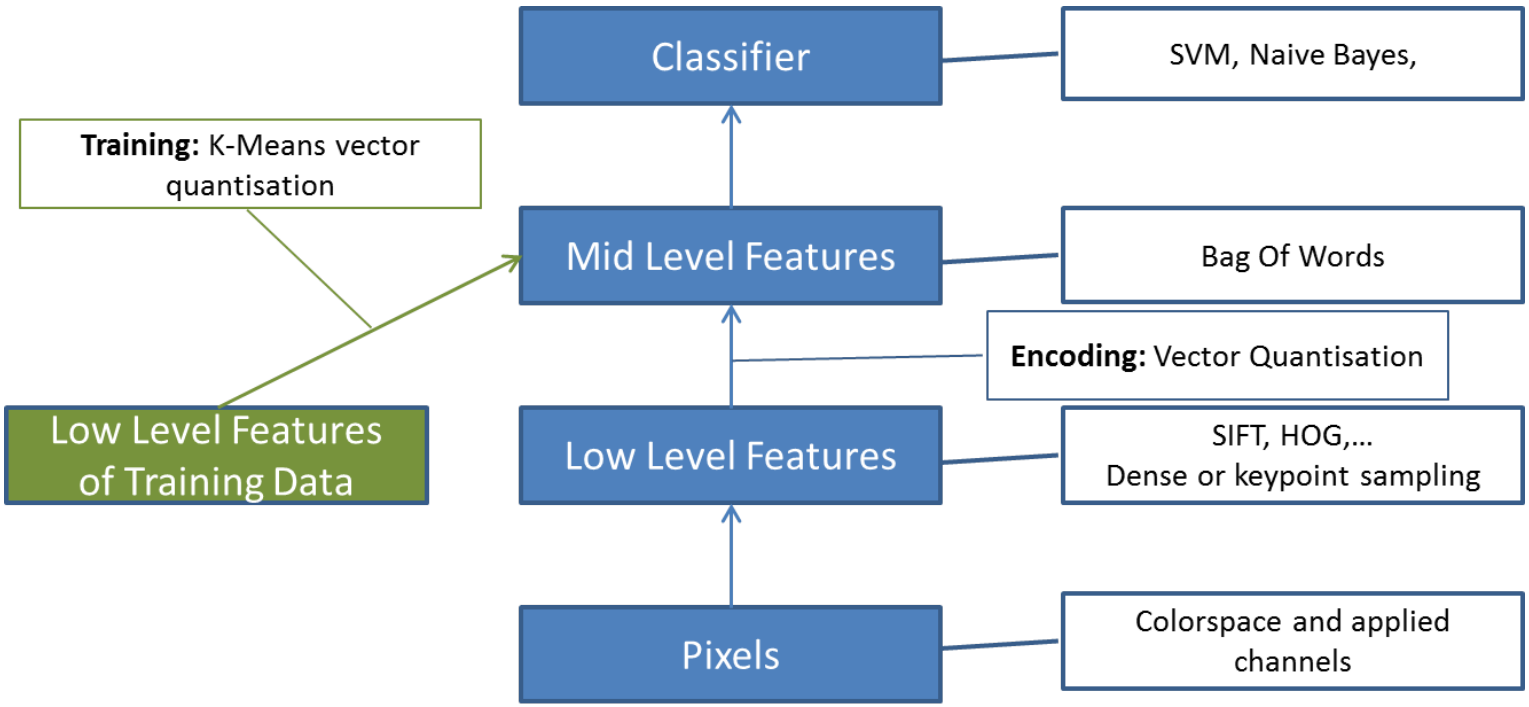
Pros:

- Robust w.r.t. background clutter
- BoW representation allows application of any machine learning classification algorithm

Cons and open questions:

- Bag of Word representation is totally orderless, information about the spatial layout is not contained
- How to find a good vocabulary?
  - What is the best k?
  - How to set the initial cluster centers such that final clustering corresponds to a good local minimum.
  - Is $k-$means clustering and the corresponding encoding from SIFT features to vocabulary the best approach?

Layered Schema of BoW Approach



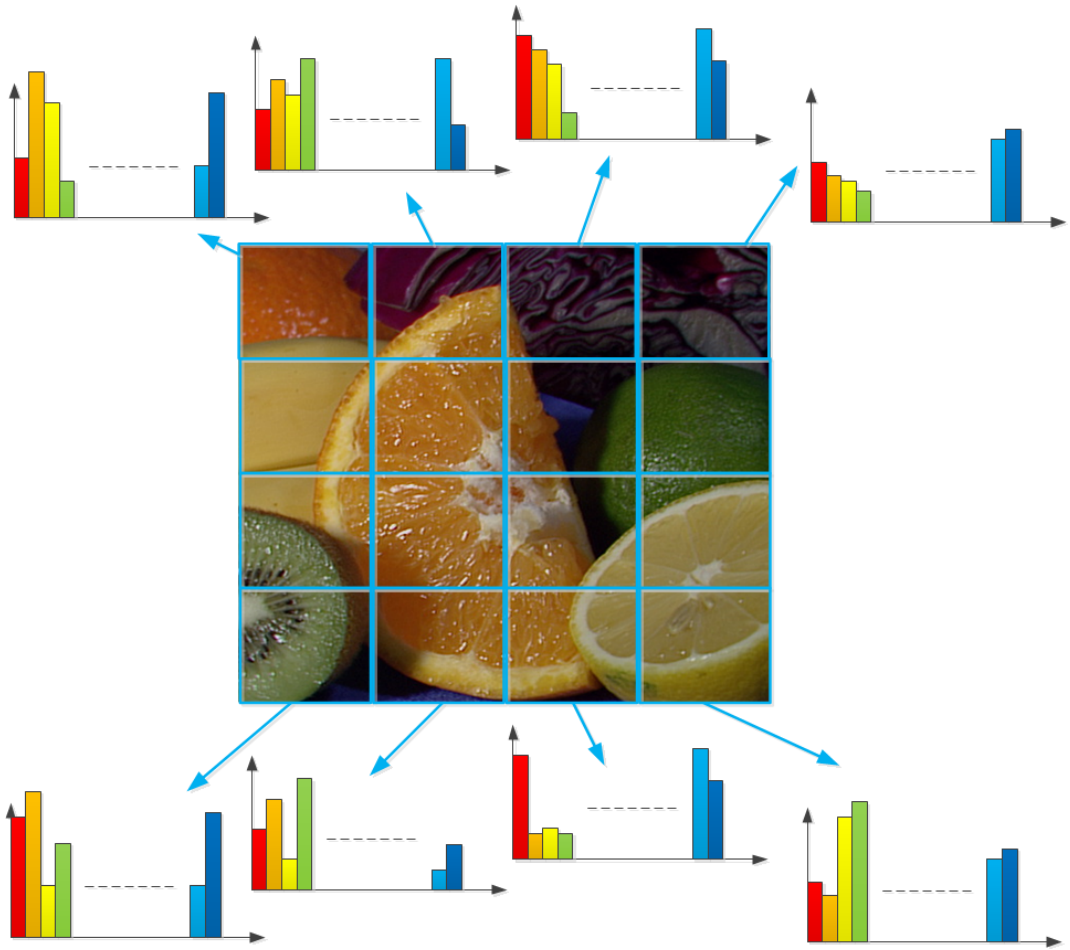
Approaches to solve the BoW Problems

- In [Lazebnik et al., 2006] the concept of Spatial Pyramid Matching has been introduced. It can be considered as an extension of the BoW concept, which integrates spatial information.
- Sparse Coding consistently yields better results, than k-means clustering and vector quantization [Coates and Ng, 2011], [Boureau et al., 2010], [Yang et al., 2009].

Spatial Pyramid Matching

- Spatial Pyramid Matching ([Lazebnik et al., 2006]) can be considered as a combination of Pyramid Match Kernels ([Grauman and Darrell, 2007]) and Bag Of Words ([Csurka et al., 2004]).
- Subdivide and Disorder Strategy:
  - Subdivide the image into subwindows
  - Calculate histogram of something within each subwindow.
  - Histograms are orderless, thus invariant w.r.t. affine transformations, rotations,...
  - The entire representation contains the histograms of all subwindows in a spatially ordered form.

Subdivide and Disorder

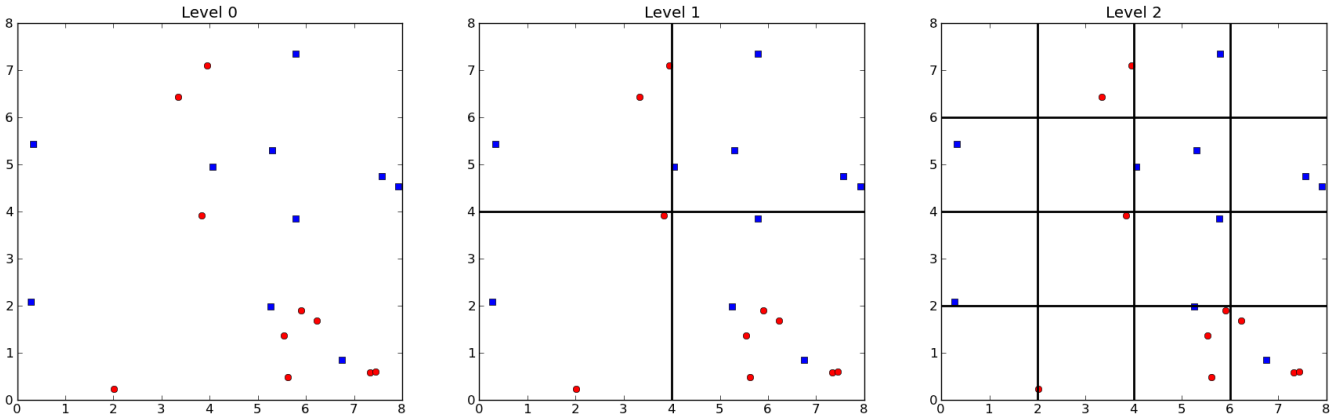


- The overall representation contains the histograms ordered according to the spatial order of the subwindows.

Pyramid Match Kernel [Grauman and Darrell, 2007]

- X and Y are two sets of vectors in $d-$dimensional feature space.
- Pyramide Matching approximately measures the correspondence of X and Y.
- Place a sequence of increasingly coarser grids over the feature space and take a weighted sum of the number of matches that occur at each level of resolution.

Idea of matching at different resolutions



- Note that this is d-dimensional feature space not image space.
- Actually sets of histograms are matched, not sets of points.

Pyramid Match Kernel

- Construct a sequence of $L+1$ grids, such that the grid at level $\ell \in \{0, \ldots, L\}$ has 2^ℓ cells along each dimension and $D = 2^{d\ell}$ cells in total.
- H_X^ℓ and H_Y^ℓ are the histograms of X and Y at this resolution, so that $H_X^\ell(i)$ and $H_Y^\ell(i)$ are the numbers of points from X and Y that fall into the $i.th$ cell of the grid. Then the number of matches at level ℓ is given by the histogram intersection function

$$\mathcal{I}^\ell = \mathcal{I}(H_X^\ell, H_Y^\ell) = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \tag{11}$$

Pyramid Match Kernel

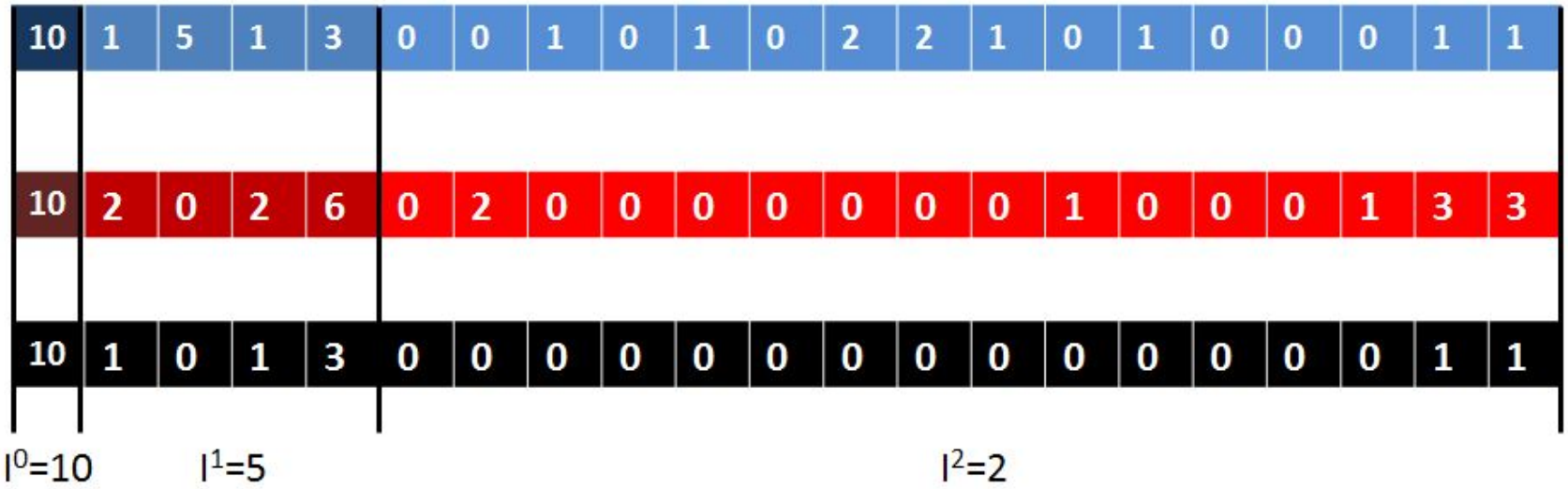
- The set of matches at level ℓ includes the set of matches at level $\ell + 1$. The number of new matches at level ℓ is therefore $\mathcal{I}^\ell - \mathcal{I}^{\ell+1}$.
- In order to calculate a total score of matches, the matches at finer resolution levels are weighted higher than matches at coarser resolutions.
- The weight associated with level ℓ is

$$\frac{1}{2^{L-\ell}}$$

- The Pyramid Match Kernel is then

$$\begin{aligned} \kappa^L(X, Y) &= \mathcal{I}^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (\mathcal{I}^\ell - \mathcal{I}^{\ell+1}) \\ &= \frac{1}{2^L} \mathcal{I}^0 + \sum_{\ell=1}^{L} \frac{1}{2^{L-\ell+1}} \mathcal{I}^\ell \end{aligned} \tag{12}$$

Pyramid Match Kernel Example



Pyramid Match Kernel: K2(blau,rot)=1/4*(10 + 5 + 4) =4.75

- First row: H_X: Histogram of blue features in example on page 40
- Second row: H_Y: Histogram of red features
- Third row: Histogram Intersections

Spatial Pyramid Matching

- Pyramid Match Kernel as introduced in [Grauman and Darrell, 2007] is a totally orderless representation, that discards all spatial information.
- Spatial Pyramid Matching performs 2-dimensional pyramid matching in the image space and clustering in feature space (BoW).
- Clustering quantizes all features into *M* different types.
- Only features of the same type can be matched.
- For each type $m \in \{1, \ldots, M\}$ two sets X_m and Y_m, of 2-dimensional vectors, each representing the 2-dimensional image-coordinate of a feature of type m, exists[2].
- Spatial Pyramid Matching Kernel:

$$K^L(X, Y) = \sum_{m=1}^{M} \kappa^L(X_m, Y_m) \tag{13}$$

- For $L = 0$ this is the same as if the BoW representations of X and Y are matched.

[2] As before X corresponds to one image and Y to the other image

Toy Example Spatial Pyramid Matching

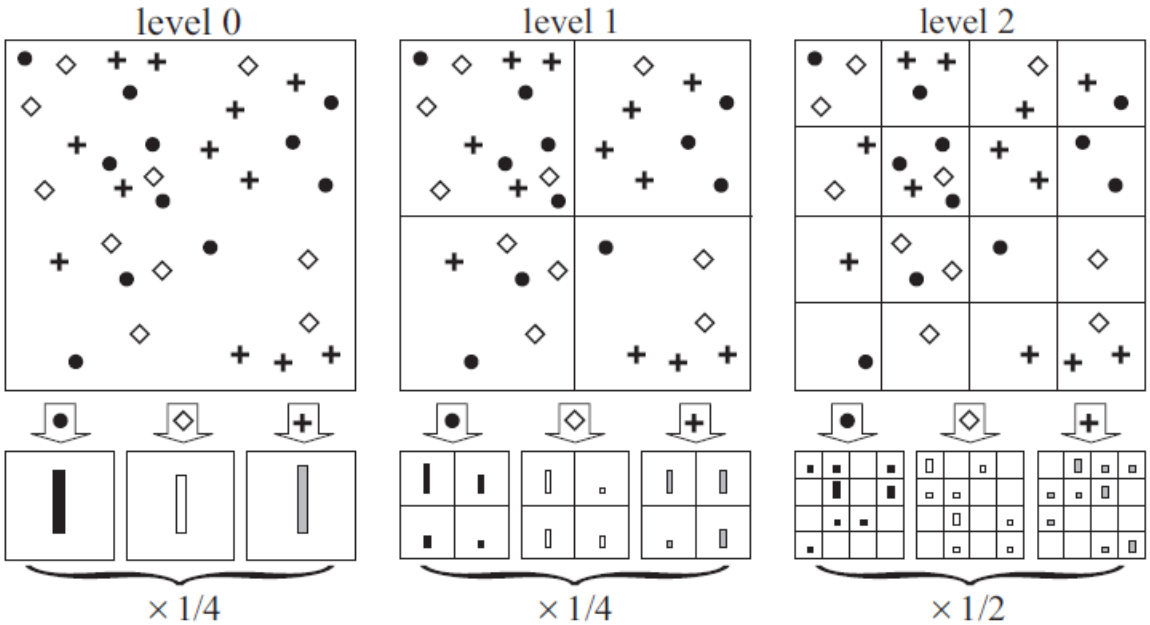


Figure: Toy example of constructing a three-level pyramid. The image has three feature types, indicated by circles, diamonds, and crosses. The image is subdivided at three different levels of resolution.For each level of resolution and each channel, the features that fall in each spatial bin are counted [Lazebnik et al., 2006].

Spatial Pyramid Matching Kernel in a single Histogram

- Since the pyramid match kernel (equation 12) is a weighted sum of histogram intersections and for positive numbers

$$w \min(a, b) = \min(wa, wb),$$

the spatial pyramid matching K^L can be represented as a long vector of histogram intersections, formed by concatenating the appropriately weighted histograms of all types m at all resolutions ℓ.

- For L levels and M types the resulting vector has

$$M \sum_{\ell=0}^{L} 4^{\ell} = \frac{M(4^{L+1} - 1)}{3} \tag{14}$$

components.

- The vectors are extremely sparse. Computational complexity of the kernel is linear in the number of the features.
- Typical values: $M = 200, L = 2$ yields vector of length 4200.

Histogram Normalization

- Comparison of histograms requires that they are in a normalized form.
- All histograms are normalized by the total weight of all features in the image.
- Effect: Total number of all features is the same for all images.

Feature Extraction (as proposed in [Lazebnik et al., 2006])

- Dense sampling of SIFT descriptors
- One SIFT descriptor for a patch of (16×16) pixels.
- Sampled over a grid with spacing of 8 pixels.
- Since the number of features per image may be large, a random subset of features is used as input for k-means clustering.
- Dense sampling is particularly recommended for scene classification. E.g. sparse sampling would not find any keypoints in large heterogenous areas like sky or calm water.

Binary SVM Classifier in general

- In general a binary SVM classifier learns a decision function

$$f(\mathbf{z}) = \sum_{i=1}^{N} \alpha_i y_i \kappa(\mathbf{z}, \mathbf{z}_i) + b, \tag{15}$$

  where $\{(\mathbf{z}_i, y_i)\}_{i=1}^{N}$ is the set of N labeled training vectors[3]. The label $y_i \in \{-1, +1\}$ indicates the class of input $\mathbf{z}_i$.
- $\kappa(\mathbf{z}, \mathbf{z}_i)$ is an arbitrary kernel-function.
- For a test vector $\mathbf{z}$ the value $f(\mathbf{z}) > 0$ indicates that $\mathbf{z}$ is assigned to the class labeled by $y = +1$. For $f(\mathbf{z}) \leq 0$ the other class is assigned.

[3] See also equation (14) of SVM slides of Machine Learning Lecture, J. Maucher

SVM with Spatial Pyramid Matching Kernel

- Multi-class SVM is applied according to one-versus-all-rule: For each class a single binary SVM-classifier is learned, which separates images of the respective class (label $y = +1$) from the rest (label $y = -1$).
- For a single class label all training images, which belong to this class by $y = +1$, all other images are labeled by $y = -1$
- Use this set of labeled training images to learn a binary classifier for the given class. The classifier is a function

$$f(Z) = \sum_{i=1}^{N} \alpha_i y_i \kappa(Z, Z_i) + b, \tag{16}$$

where $\kappa(Z, Z_i) = K^L(Z, Z_i)$ is the Spatial Pyramid Matching Kernel as defined in equation (13) and Z is the corresponding descriptor set of an image.

Experiment Configuration from [Lazebnik et al., 2006]

- Applied Image Datasets:
  - Fifteen scene categories
  - Caltech 101
  - Graz
- All experiments are repeated 10 times with different randomly selected training and test images.

Experimental Results from [Lazebnik et al., 2006] on scene categorization

Figure 2. Example images from the scene category database. The starred categories originate from Oliva and Torralba [13].

Experimental Results from [Lazebnik et al., 2006] on scene categorization and Caltech 101

	Weak features ($M = 16$)		Strong features ($M = 200$)		Strong features ($M = 400$)	
L	Single-level	Pyramid	Single-level	Pyramid	Single-level	Pyramid
0 (1×1)	45.3 ±0.5		72.2 ±0.6		74.8 ±0.3	
1 (2×2)	53.6 ±0.3	56.2 ±0.6	77.9 ±0.6	79.0 ±0.5	78.8 ±0.4	80.1 ±0.5
2 (4×4)	61.7 ±0.6	64.7 ±0.7	79.4 ±0.3	**81.1** ±0.3	79.7 ±0.5	**81.4** ±0.5
3 (8×8)	63.3 ±0.8	**66.8** ±0.6	77.2 ±0.4	80.7 ±0.3	77.2 ±0.5	81.1 ±0.6

	Weak features		Strong features (200)	
L	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7

- Weak features mean sparse, strong features mean dense sampling of SIFT descriptors
- Case $L = 0$ is identical to BoW approach.

Spatial Pyramid Matching: Pros and Cons

Pros:

- Better performance than BoW approach
- Widely applied and basis for large amount of future work

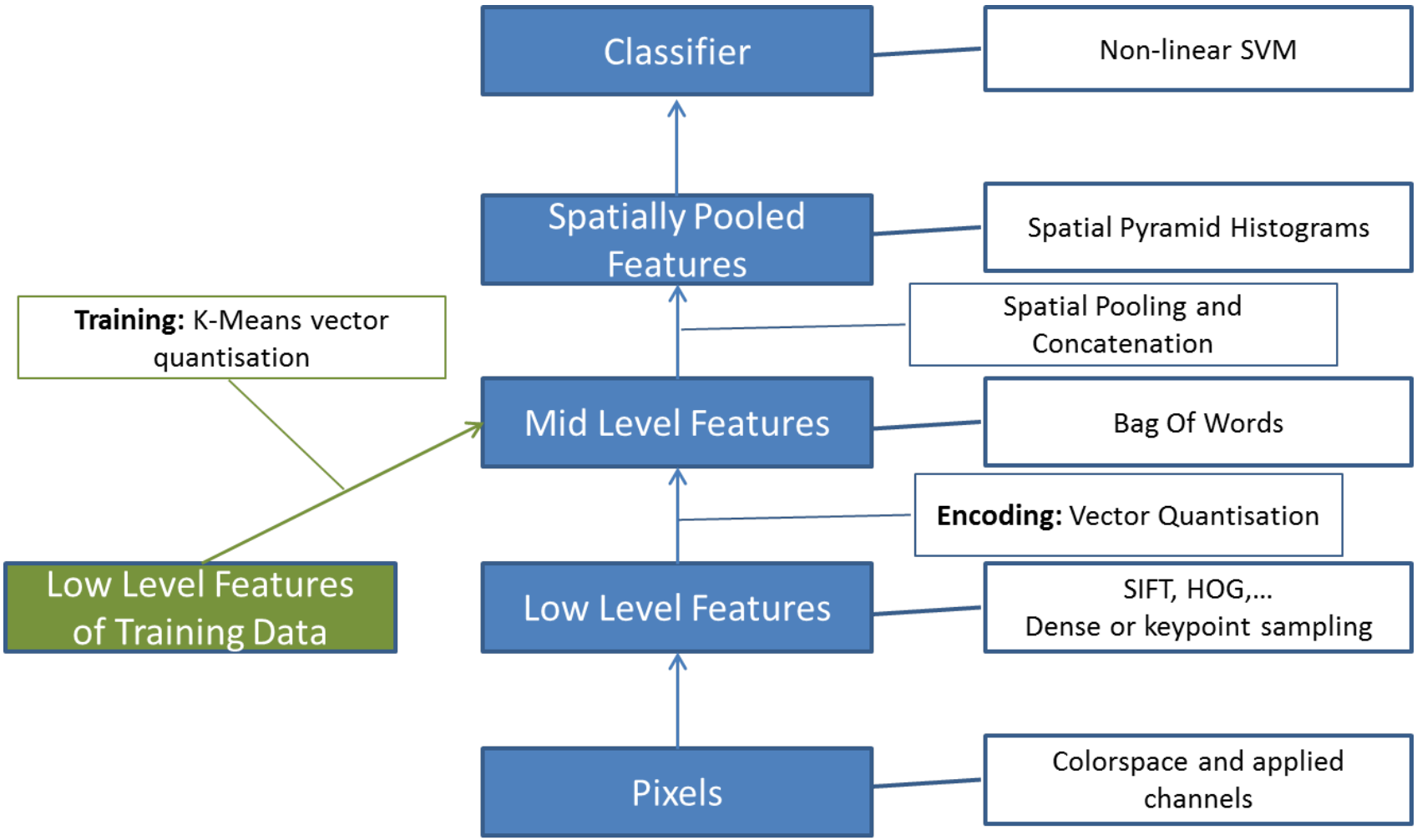
Cons and open questions:

- The SVM Kernel applied in equation (16), is non-linear. For non-linear SVMs the complexity is
  - $\mathcal{O}(N^3)$ (computational)
  - $\mathcal{O}(N^2)$ (memory)
  - $\mathcal{O}(N)$ (testing)

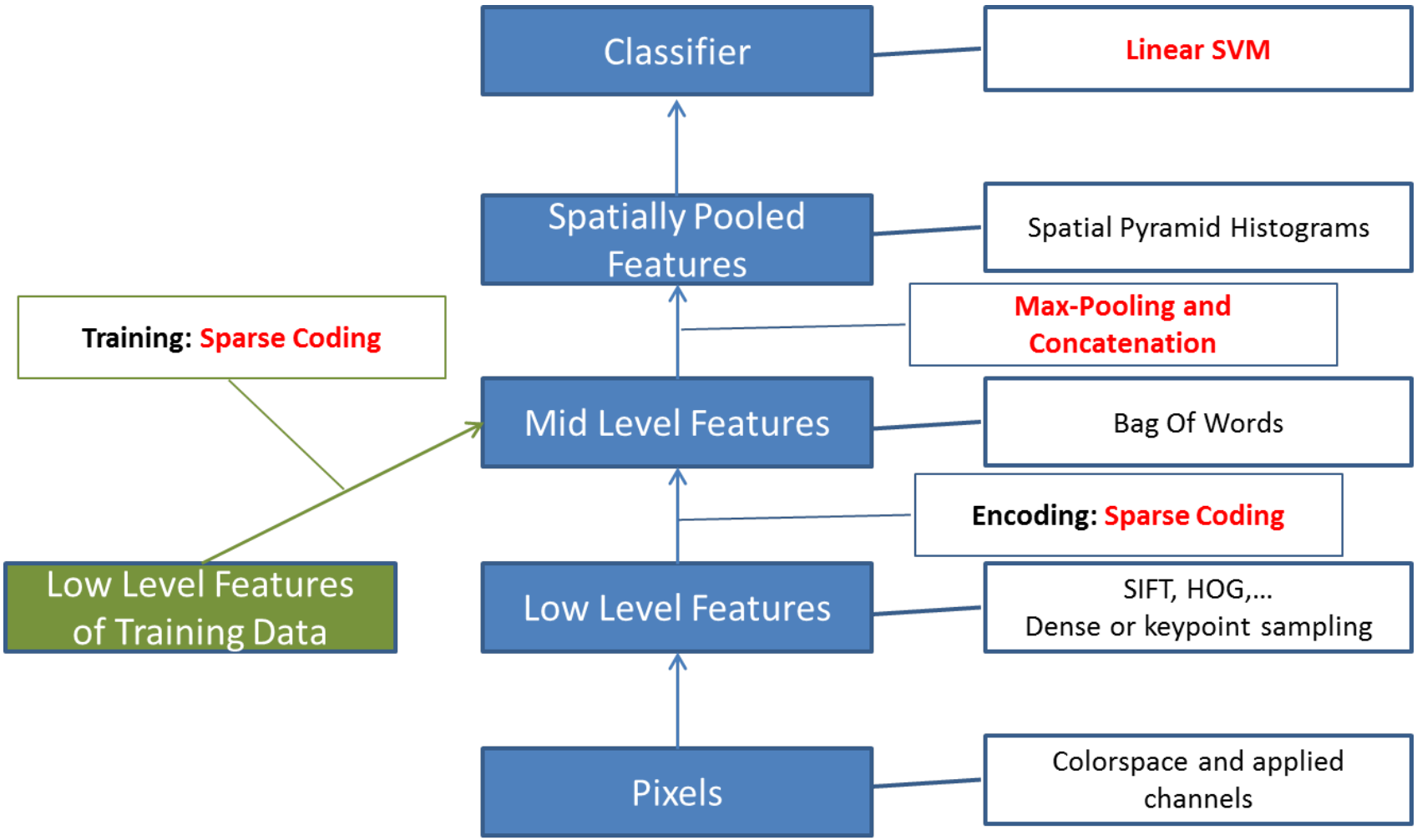
  This limits real-world application with typical training sizes far beyond thousands of images.

In [Yang et al., 2009] an extension has been developed, that allows linear SVMs. This approach not only decreases complexity but increases classification accuracy. Key element is the use of sparse coding instead of vector quantisation.

Layered Schema of Spatial Pyramid Matching

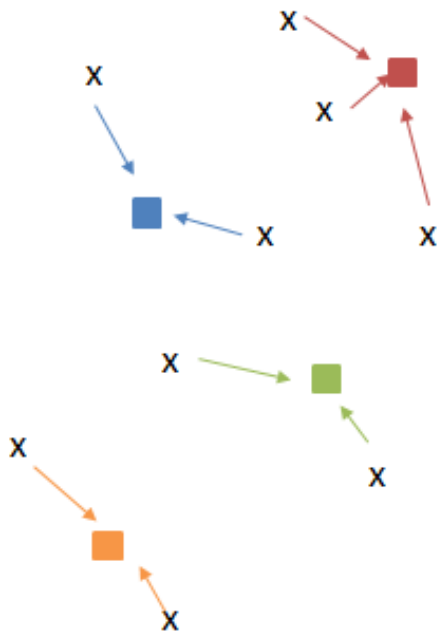


Outlook: Layered Schema of Linear Spatial Pyramid Matching



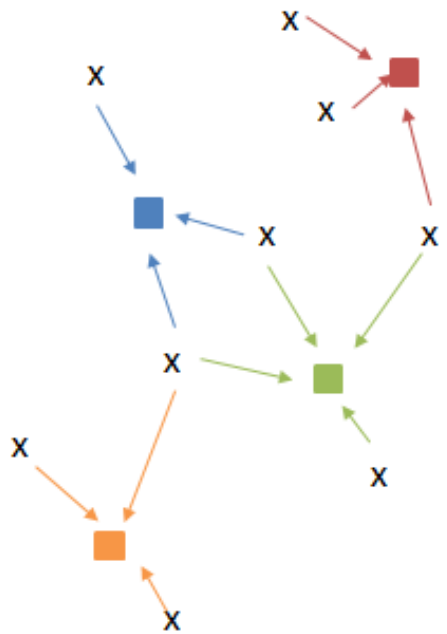
From Vector Quantisation to Sparse Coding

Vector Quantisation:
Each lower level feature is mapped to **exactly one** higher level feature



x : lower level features
■ : higher level features

Sparse Coding:
Each lower level feature is mapped to a **small number** of higher level feature



x : lower level features
■ : higher level features

Notation

- The N training samples are the rows of the matrix

$$X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T \tag{17}$$

- The K cluster centers are the rows of the matrix

$$V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K]^T \tag{18}$$

  Matrix V is also called codebook.
- The rows of the $N \times K$ matrix

$$U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T \tag{19}$$

  define the cluster membership of the training sampels, i.e. if the $i.th$ training vector $\mathbf{x}_i$ belongs to cluster j, than the $j.th$ component of $\mathbf{u}_i$ is 1 and all other components are 0.

Notation

- L2-Norm of vector $\mathbf{x}$:

$$\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{d} x_i^2} \tag{20}$$

- L1-Norm of vector $\mathbf{x}$:

$$|\mathbf{x}| = \sum_{i=1}^{d} |x_i| \tag{21}$$

K-Means as optimisation problem

- K-means clustering finds the codebook V such that the reconstruction error (equation 3) is minimized, i.e. k-means solves the following optimisation problem

$$\min_{V} \sum_{m=1}^{N} \min_{k=1,\ldots,K} \|\mathbf{x}_m - \mathbf{v}_k\|^2 \tag{22}$$

- This optimisation problem can be re-formulated in the following matrix factorisation problem

$$\min_{U,V} \sum_{m=1}^{N} \|\mathbf{x}_m - \mathbf{u}_m V\|^2 \quad \text{subject to} \quad Card(\mathbf{u}_m) = 1, |\mathbf{u}_m| = 1, \mathbf{u}_m \succeq 0 \tag{23}$$

where
  - $Card(\mathbf{u}_m) = 1$ means that only one element of $\mathbf{u}_m$ is nonzero
  - $|\mathbf{u}_m| = 1$ means that the $L1$-norm is 1.
  - $\mathbf{u}_m \succeq 0$ means that all elements in $\mathbf{u}_m$ are nonnegative

K-Means as optimisation problem

- In the k-means training phase the optimisation problem (23) is solved w.r.t. U and V.
- In the coding phase, the learned codebook V is fixed and equation(23) will be solved w.r.t. U for a new dataset X.

Relaxing the optimisation constraint

- In equation(23) the constraint $Card(\mathbf{u}_m) = 1$ may be too restrictive.
- Sparse Coding: Relax the constraint, that each input is mapped to exactly one cluster, by replacing the cardinality constrained by a $L1$-norm regularization on $\mathbf{u}_m$.

$$\min_{U,V} \sum_{m=1}^{N} \|\mathbf{x}_m - \mathbf{u}_m V\|^2 + \lambda |\mathbf{u}_m| \quad \text{subject to } \|\mathbf{v}_k\| = 1 \ \forall k \tag{24}$$

- The $L1$-norm regularization enforce $\mathbf{u}_m$ to have a small number of nonzero elements.
- The constraint $\|\mathbf{v}_k\| = 1$ provides normalisation of the codebook.
- The sparse optimisation problem can be solved e.g. by the Coordinate Descent Algorithm

Coordinate Descent Optimisation

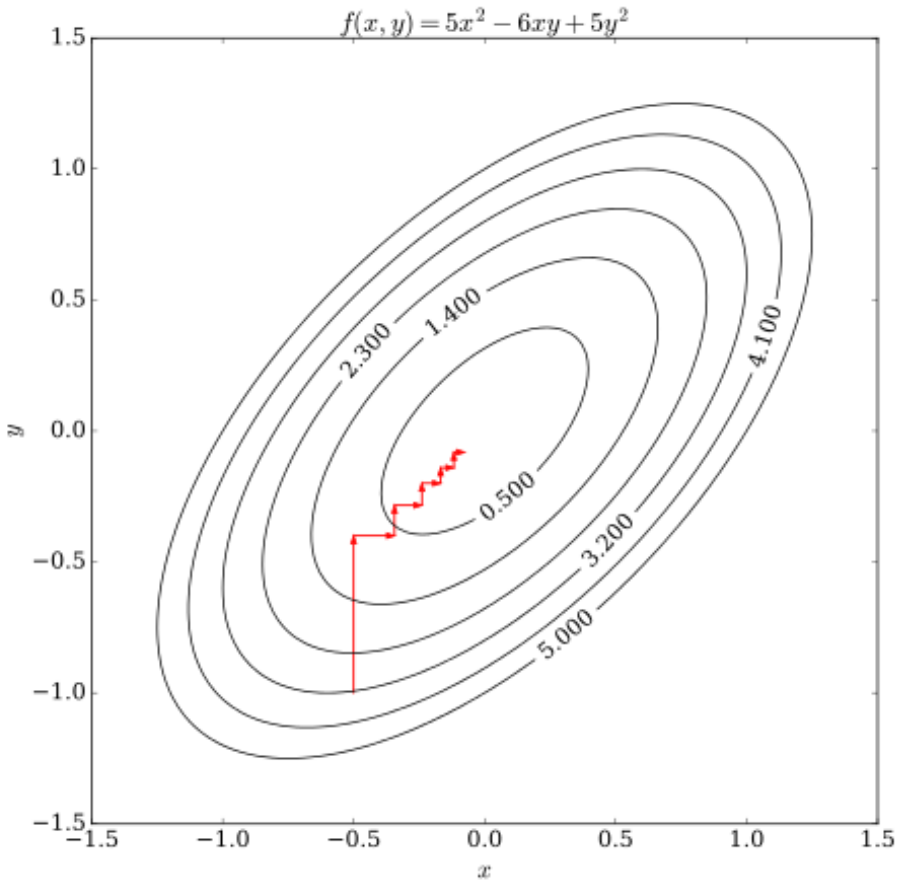
- Goal: Minimisation of multivariate function $f(\mathbf{x} = f(x_1, x_2, \ldots, x_n)$
- Start at an arbitrary point $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ and minimize in each iteration the function along only one coordinate-axis.
- In round $k+1$ successively adapt all x_i as follows:

$$x_i^{k+1} = \arg\min_{y \in \mathcal{R}} f(x_1^{k+1}, x_2^{k+1}, \ldots, x_{i-1}^{k+1}, y, x_{i+1}^k, \ldots, x_n^k)$$

- Function decreases in each round until stationary point is reached:

$$f(\mathbf{x}^0) \geq f(\mathbf{x}^1) \geq f(\mathbf{x}^2) \geq \ldots$$

Coordinate Descent Optimisation



Sparse Neural Nets

- Other popular sparse coding techniques are e.g. sparse Restricted Boltzman Machines (RBM) and sparse Autoencoders
- RBMs and autoencoders can be represented as neural networks.

Representation Types

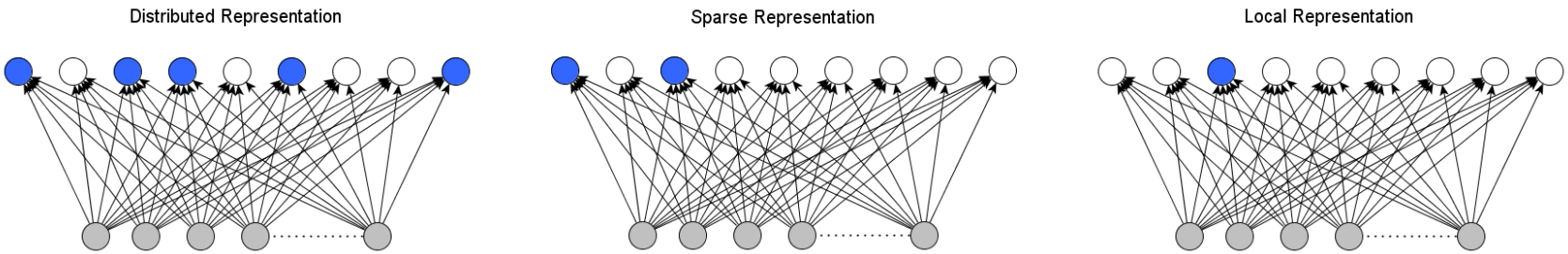
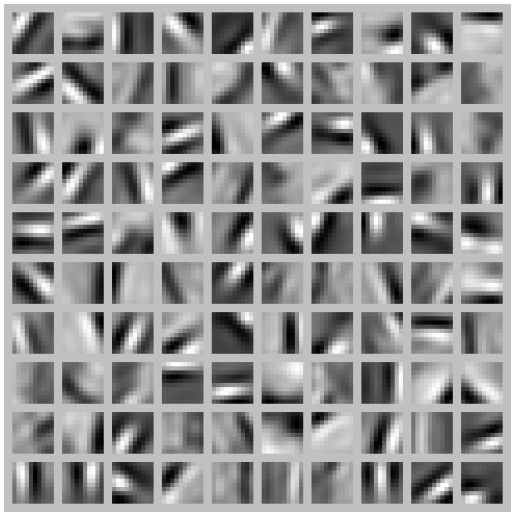


Figure: Distributed, sparse and local representation

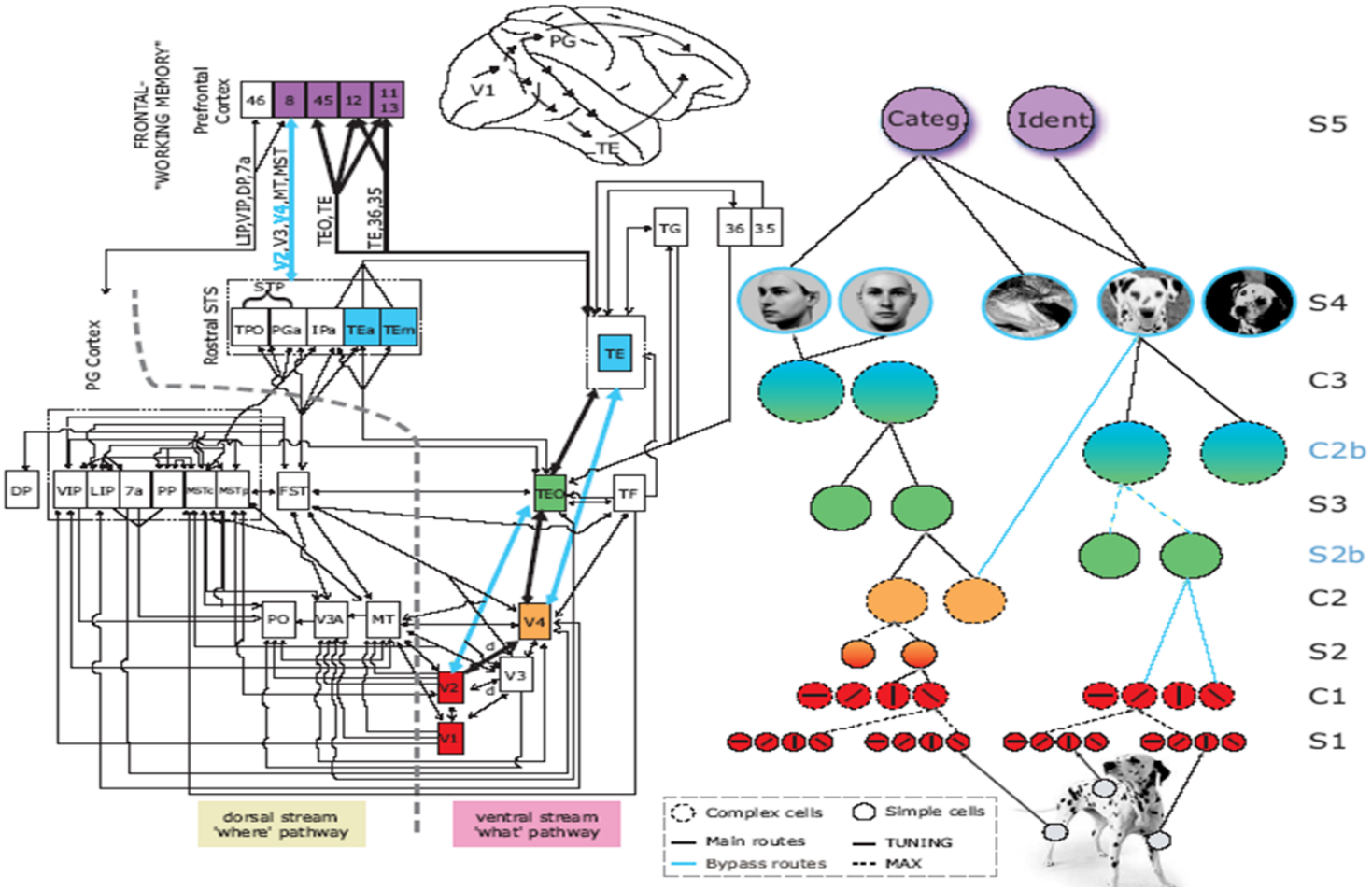
- Distributed representation requires fewer units to encode information
- Sparse representation: Each unit encodes a specific higher-level feature. Only the units of features which occur in the current input data are active.
- Human Brain is sparse: Only about 1-4 % of neurons are active simultaneously

Simple Cells in the visual cortex are sparse

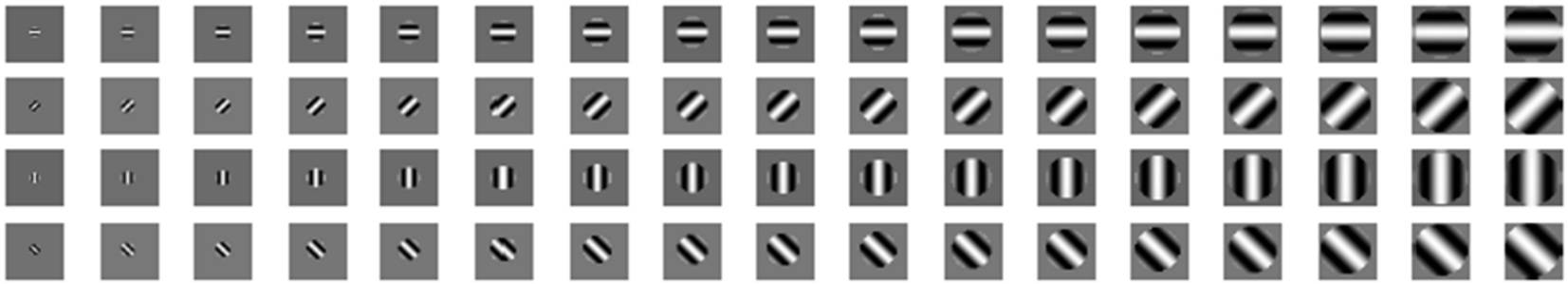
- Sparse autoencoder with 100 hidden units trained on image patches of size (10×10) pixels
- Each of the neurons corresponds to a filter with a filter response as displayed in the image below.
- These filter responses resemple the receptive fields of the simple cells (S1) of the visual cortex.



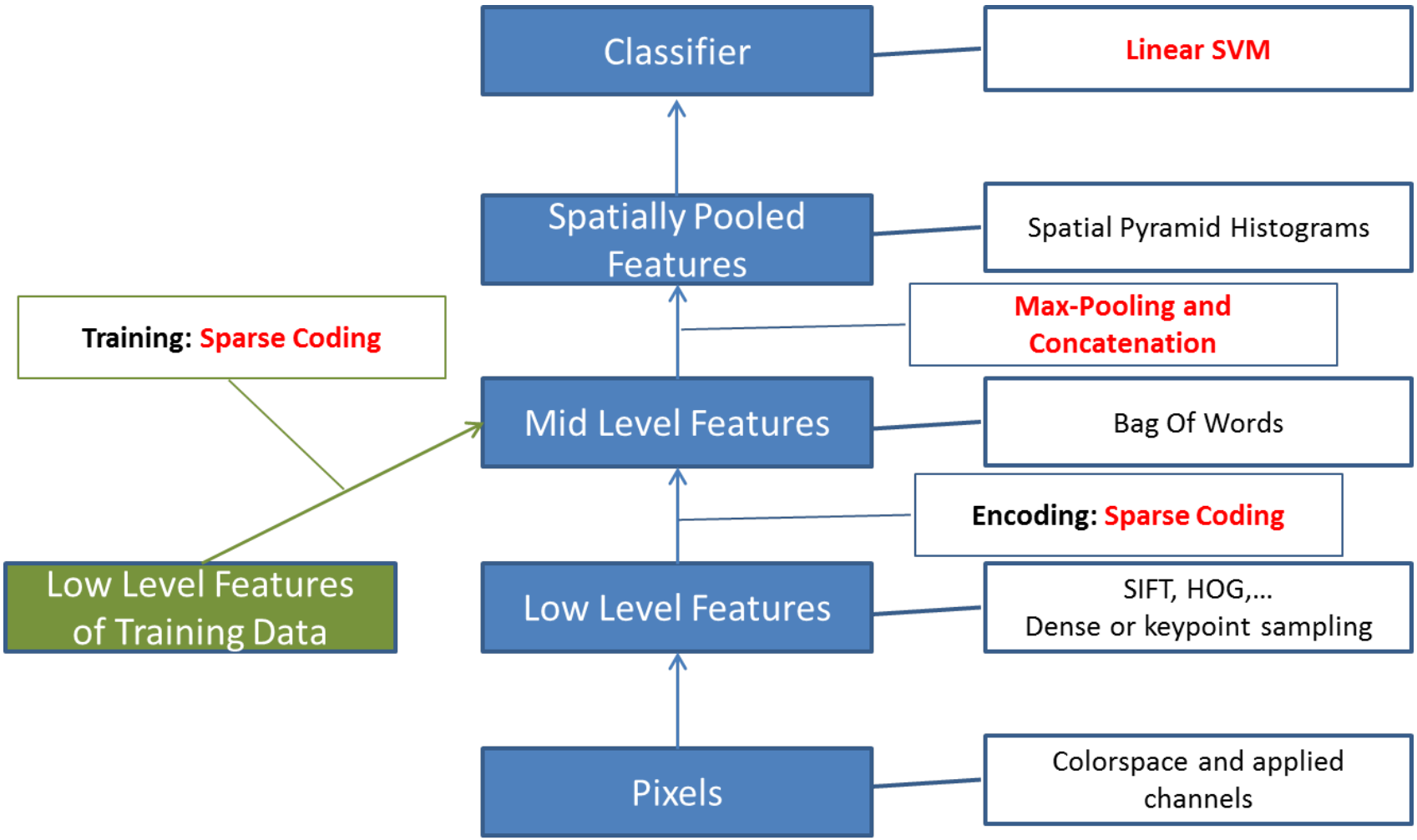
Gabor Filter responses of S1 cells in the visual cortex



Model of visual cortex



Layered Schema of Linear Spatial Pyramid Matching [Yang et al., 2009]



Use Sparse Coding instead of Vector Quantisation

Training:

- Let X be the set of N training descriptors, as defined in equation (17)
- Solve equation (24) w.r.t. U and V for the given training images, where U and V are defined as in equation (19) and (18), respectively.

Test:

- Let X be the matrix of M descriptors of the new image.
- V is the codebook determined in the training phase.
- Solve equation (24) w.r.t. U.
- The $i.th$ row of U determines which codewords (mid level features) of the codebook V are activated by the $i.th$ descriptor in X.
- The $j.th$ column determines the responses of all descriptors in X to one specific mid level feature (codeword) in V.

Max Pooling

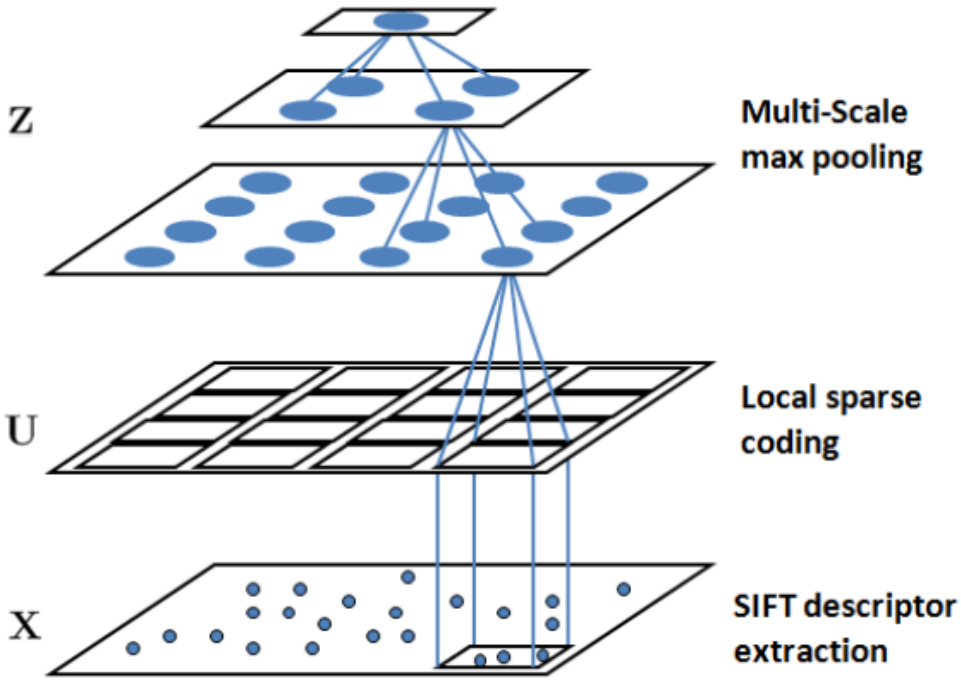
- Let U be the sparse code of descriptor set X.
- For each of the K columns in matrix U calculate the $j.th$ component of vector $\mathbf{Z}$ as

$$z_j = \max\{|u_{1,j}|, |u_{2,j}|, \ldots, |u_{M,j}|\} \tag{25}$$

- The value of z_j describes the presence of mid-level feature $\mathbf{v}_j$ in the corresponding spatial region of the image, described by X.
- Note, that in the original Spatial Pyramid Matching approach by [Lazebnik et al., 2006] the representation z is a histogram, which counts the frequency of a mid-level feature $\mathbf{v}_j$ in the corresponding spatial region of the image.
- Max Pooling is also applied in the visual cortex by the *C1* cells.

Max-Pooling Spatial Pyramids

- Similar to the construction of histograms in SPM, Max Pooling Spatial Pyramids are constructed by applying max pooling across different locations and over different spatial scales.
- Max-Pooled features from different locations and scales are then concatenated as in SPM.



Linear SVM Classification

- Let $\mathbf{z}_i$ be the max pooling representation of image I_i.
- The linear kernel in the SVM is

$$\kappa(\mathbf{z}_i, \mathbf{z}_j) = \mathbf{z}_i^T \mathbf{z}_j = \sum_{\ell=0}^{2} \sum_{s=0}^{2^\ell} \sum_{t=0}^{2^\ell} \mathbf{z}_i^T(\ell, s, t)\mathbf{z}_j(\ell, s, t), \tag{26}$$

  where $\mathbf{z}_i^T(\ell, s, t)$ is the max-pooling output of the spatial segment at position (s, t) at level ℓ.
- The binary decision function of the SVM is then

$$f(\mathbf{z}) = \left(\sum_{i=1}^{N} \alpha_i y_i \mathbf{z}_i \right)^T \mathbf{z} + b = \mathbf{w}^T \mathbf{z} + b \tag{27}$$

Performance, as claimed in [Yang et al., 2009]

- Computational training cost is $\mathcal{O}(N)$, testing cost is constant (independent of N).
- *Linear SPM kernel based on sparse coding statistics always achieves excellent classification accuracy, because*
  1. *Sparse Coding has much less quantization errors than VQ*
  2. *It is well known that image patches are sparse in nature, and thus sparse coding is particularly suitable for image data*
  3. *The computed statistics by max pooling are more salient and robust to local translations*
- On the Caltech-101 dataset the classification accuracy of this approach is about 10% better, than the SPM approach of [Lazebnik et al., 2006]

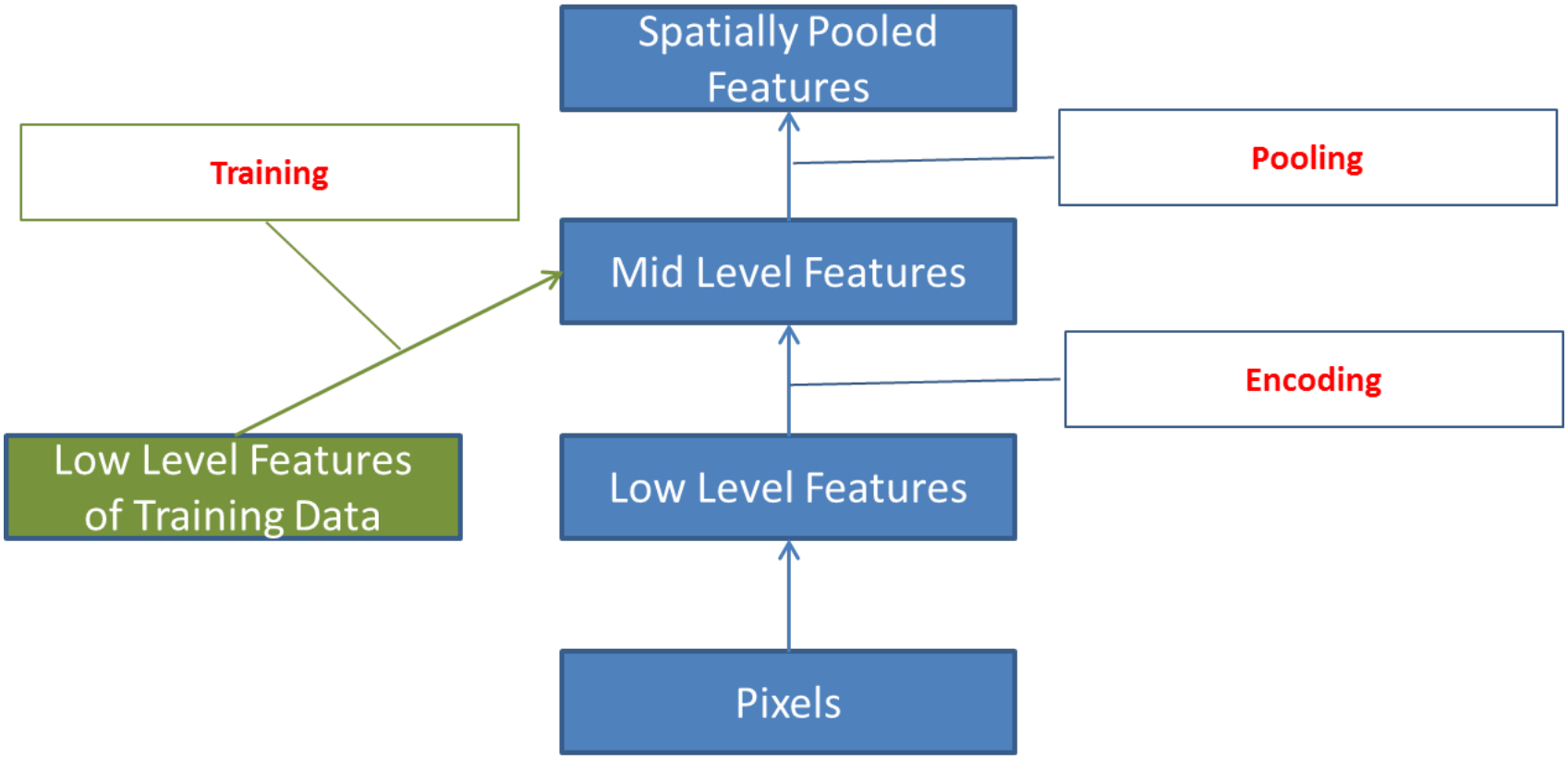
Performance [Yang et al., 2009]

	Caltech-101		Caltech-256		15 Scenes
Training Images per category	15	30	15	30	100
KSPM	56.44	63.99	23.34	29.51	76.73
LSPM	53.23	58.81	13.2	15.45	65.32
ScSPM	67.00	73.2	27.73	34.02	80.28

- KSPM: Spatial Pyramid Matching with K-means, histogram pooling and non-linear SVM[Lazebnik et al., 2006]
- LSPM: Spatial Pyramid Matching with K-means, histogram pooling and linear SVM
- ScSPM: Spatial Pyramid Matching with Sparse Coding, maxpooling and linear SVM [Yang et al., 2009].

Further Research Results

Training, Encoding, Pooling: How to choose these factors to obtain maximal classification accuracy?

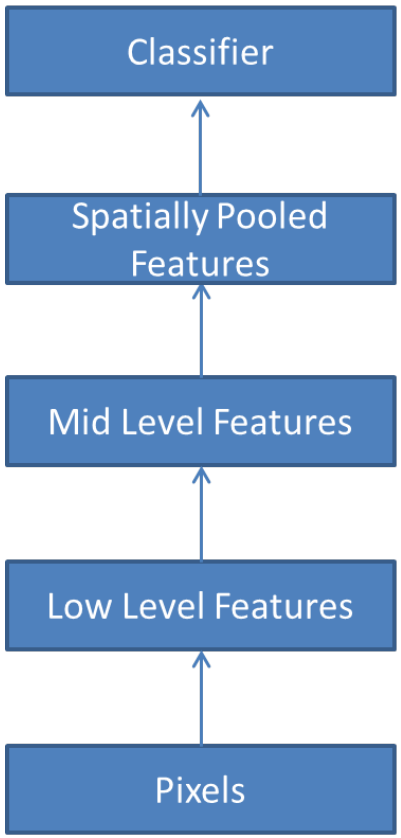


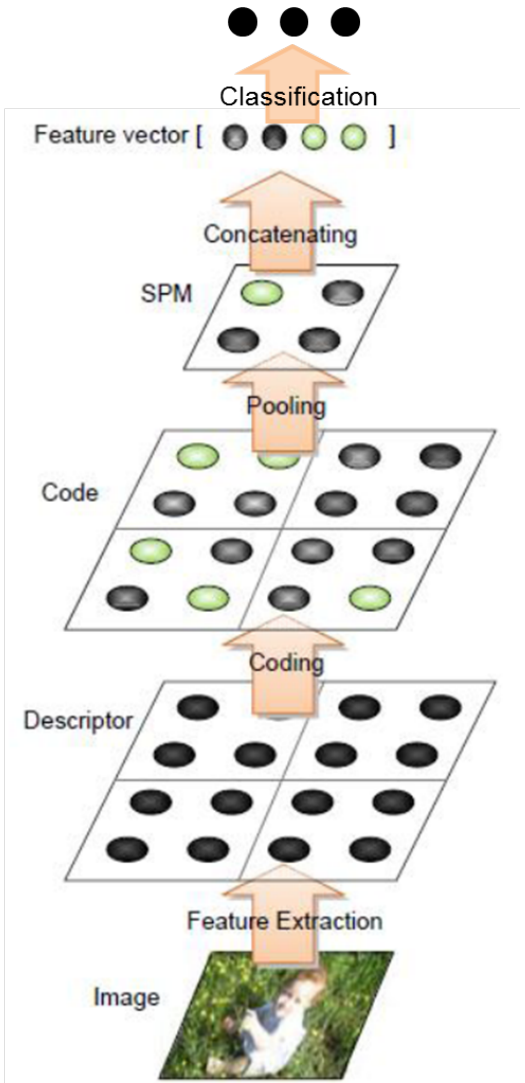
Further Research Results

- In [Coates and Ng, 2011] Coates and Ng decoupled training and encoding.
- For example it is possible to use vector quantisation for training and sparse coding for encoding or vice versa.
- Their main result: *When using sparse coding as encoder, virtually any training algorithm can be used to create the dictionary (training)*.

- In [Boureau et al., 2010] Boureau et al investigated the influence and combination of encoding and pooling. Their results are:
  - Sparse coding outperforms other coding modules, irrespective of the pooling module.
  - Max-pooling dramatically improves linear classification performance, irrespective of the coding module.

Linear SPM as Neural Network





References I

Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. (2010).

Learning mid-level features for recognition.

In *CVPR*, pages 2559–2566.

Coates, A. and Ng, A. (2011).

The importance of encoding versus training with sparse coding and vector quantization.

In Getoor, L. and Scheffer, T., editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 921–928, New York, NY, USA. ACM.

Csurka, G., Dance, C. R., Fan, L., Willamowski, J., and Bray, C. (2004).

Visual categorization with bags of keypoints.

In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22.

Grauman, K. and Darrell, T. (2007).

The pyramid match kernel: Efficient learning with sets of features.

J. Mach. Learn. Res., 8:725–760.

References II

Grauman, K. and Leibe, B. (2011).

Visual Object Recognition.

Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Lazebnik, S., Schmid, C., and Ponce, J. (2006).

Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories.

In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 2169–2178, Washington, DC, USA. IEEE Computer Society.

Nister, D. and Stewenius, H. (2006).

Scalable recognition with a vocabulary tree.

In *CVPR*, pages 2161–2168.

References III

Sivic, J. and Zisserman, A. (2003).

Video google: A text retrieval approach to object matching in videos.

In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1470–, Washington, DC, USA. IEEE Computer Society.

Yang, J., Yu, K., Gong, Y., and T.Huang (2009).

Linear spatial pyramid matching using sparse coding for image classification.

In *in IEEE Conference on Computer Vision and Pattern Recognition(CVPR*.