

Fisher kernels with application to image representation

Advanced Learning Models

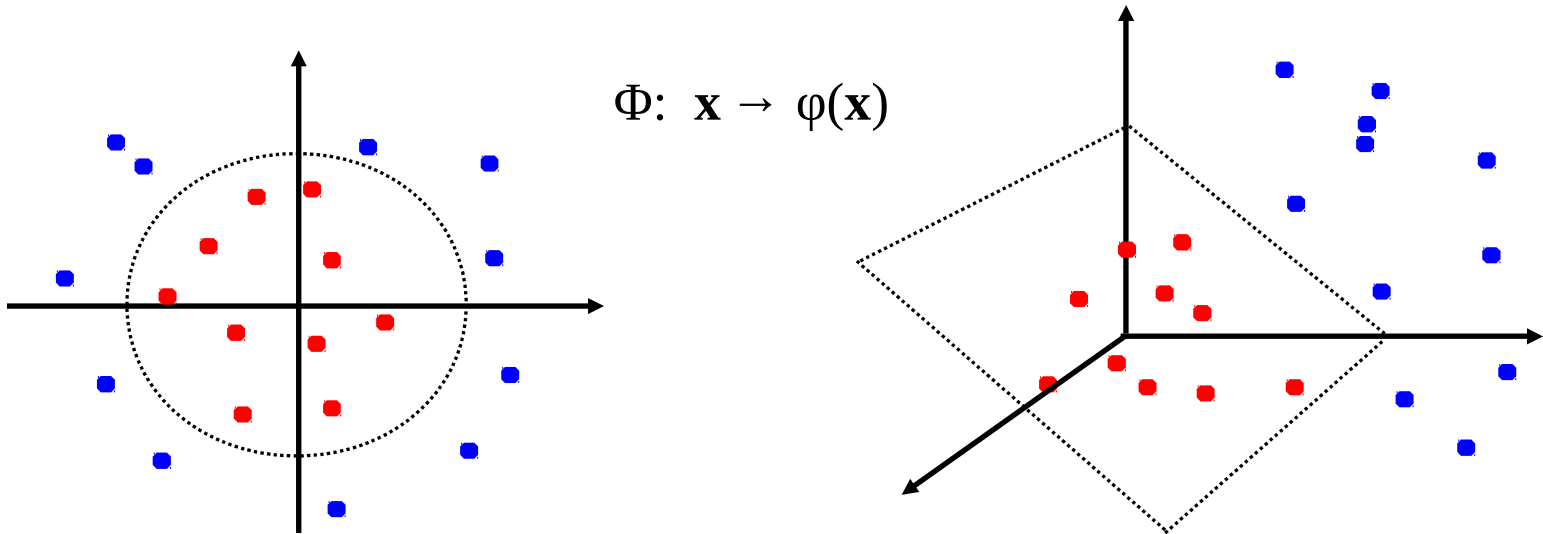
Jakob Verbeek

<http://lear.inrialpes.fr/people/mairal/teaching/2016-2017/MSIAM>

A brief recap on kernel methods

- A way to achieve non-linear classification (or other data analysis) by using a kernel that computes inner products of data after non-linear transformation
 - ▶ Given the transformation, we can derive the kernel function.
- Conversely, if a kernel is positive definite, it is known to compute a dot-product in a (not necessarily finite dimensional) feature space
 - ▶ Given the kernel, we can determine the feature mapping function.

$$k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$$



A brief recap on kernel methods

- Most often we start with data in a vector space, and map it to another feature space to allow for non-linear classification in the original space, using linear classification in the feature space
- Kernels can also be used to represent non-vectorial data, and to make them amenable to linear classification (or other linear data analysis) techniques
- For example, suppose we want to classify sets of points in a vector space, where the size of each set may vary

$$X = \{x_1, x_2, \dots, x_N\} \quad \text{with} \quad x_i \in \mathbb{R}^d$$

- We can define a representation of sets by concatenating the mean and variance of the set in each dimension

$$\phi(X) = \begin{pmatrix} \text{mean}(X) \\ \text{var}(X) \end{pmatrix}$$

- ▶ Fixed size representation of sets in 2d dimensions
- ▶ Use kernel to compare different sets:

$$k(X_1, X_2) = \langle \phi(X_1), \phi(X_2) \rangle$$

Fisher kernels

- Proposed by Jaakkola & Haussler, “Exploiting generative models in discriminative classifiers”, In *Advances in Neural Information Processing Systems* 11, 1998
- Motivated by the need to represent variably sized objects in a vector space, such as sequences, sets, trees, graphs, etc., such that they become amenable to be used with linear classifiers, and other data analysis tools
- A generic method to define kernels over arbitrary data types based on generative statistical models.
 - ▶ Define a probability distribution over the items we want to represent

$$p(x; \theta), \quad x \in X, \quad \theta \in R^D$$

Fisher kernels

- Given a generative data model $p(x; \theta)$, $x \in X$, $\theta \in R^D$
- Represent data x in X by means of the gradient of the data log-likelihood, or “Fisher score”:

$$g(x) = \nabla_{\theta} \ln p(x),$$
$$g(x) \in R^D$$

- Define a kernel over X by taking the scaled inner product between the Fisher score vectors:

$$k(x, y) = g(x)^T F^{-1} g(y)$$

- Where F is the Fisher information matrix F :

$$F = \mathbf{E}_{p(x)} [g(x) g(x)^T]$$

- F is positive definite since

$$\alpha^T F \alpha = \mathbf{E}_{p(x)} [(g(x)^T \alpha)^2] > 0$$

Fisher kernels

- The Fisher score has zero mean under the generative model

$$\begin{aligned} E_{p(x)}[g(x)] &= \int_x p(x) \frac{\partial}{\partial \theta} \ln p(x) \\ &= \int_x p(x) \frac{1}{p(x)} \frac{\partial}{\partial \theta} p(x) \\ &= \int_x \frac{\partial}{\partial \theta} p(x) \\ &= \frac{\partial}{\partial \theta} \int_x p(x) \\ &= \frac{\partial}{\partial \theta} 1 \\ &= 0 \end{aligned}$$

- Therefore, the Fisher information matrix is the covariance matrix of the Fisher score under the generative model

$$F = \mathbf{E}_{p(x)}[g(x)g(x)^T]$$

Fisher vector

- Since F is positive definite we can decompose its inverse as

$$F^{-1} = L^T L$$

- Therefore, we can write the kernel as

$$k(x_i, x_j) = g(x_i)^T F^{-1} g(x_j) = \phi(x_i)^T \phi(x_j)$$

- ▶ Where ϕ is known as the **Fisher vector**

$$\phi(x_i) = L g(x_i)$$

- From this explicit finite-dimensional data embedding it follows immediately that the Fisher kernel is a positive-semidefinite
- Since F is covariance of Fisher score, normalization by L makes the Fisher vector have unit covariance matrix under $p(x)$

Normalization with inverse Fisher information matrix

- Gradient of log-likelihood w.r.t. parameters $g(x) = \nabla_{\theta} \ln p(x)$
- Fisher information matrix $F_{\theta} = \int g(x) g(x)^T p(x) dx$
- Normalized Fisher kernel $k(x_1, x_2) = g(x_1)^T F_{\theta}^{-1} g(x_2)$
 - ▶ Renders Fisher kernel invariant for parametrization
- Consider different parametrization given by some invertible function $\lambda = f(\theta)$
- Jacobian matrix relating the parametrizations $[J]_{ij} = \frac{\partial \theta_j}{\partial \lambda_i}$
- Gradient of log-likelihood w.r.t. new parameters
$$h(x) = \nabla_{\lambda} \ln p(x) = J \nabla_{\theta} \ln p(x) = J g(x)$$
- Fisher information matrix $F_{\lambda} = \int h(x) h(x)^T p(x) dx = J F_{\theta} J^T$
- Normalized Fisher kernel
$$\begin{aligned} h(x_1)^T F_{\lambda}^{-1} h(x_2) &= g(x_1)^T J^T (J F_{\theta} J^T)^{-1} J g(x_2) \\ &= g(x_1)^T J^T J^{-T} F_{\theta}^{-1} J^{-1} J g(x_2) \\ &= g(x_1)^T F_{\theta}^{-1} g(x_2) \\ &= k(x_1, x_2) \end{aligned}$$

Data-adaptive kernel design

- Fisher vector given by linear projection of gradient

$$\phi(x) = F^{-1/2} \nabla_{\theta} \ln p(x)$$

- Fisher kernel is dot-product over Fisher vectors

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

- Parameters of the model $p(x)$ estimated from data
 - ▶ Structure typically determined manually in advance
- Data characteristics captured by Fisher vector depend on data used to train the generative model
- Semi-automatic data-driven kernel design instead of predominant completely manual design

Fisher kernels: example with Gaussian data model

- Let λ be the inverse variance, i.e. precision, parameter

$$p(x) = N(x; \mu, \lambda) = \sqrt{\lambda/(2\pi)} \exp\left[-\frac{1}{2}\lambda(x-\mu)^2\right]$$

$$\ln p(x) = \frac{1}{2} \ln \lambda - \frac{1}{2} \ln(2\pi) - \frac{1}{2} \lambda (x-\mu)^2$$

$$\theta = (\mu, \lambda)^T$$

- The partial derivatives are found to be

$$\frac{\partial \ln p(x)}{\partial \mu} = \lambda(x-\mu)$$

$$\frac{\partial \ln p(x)}{\partial \lambda} = \frac{1}{2} [\lambda^{-1} - (x-\mu)^2]$$

Fisher kernels: example with Gaussian data model

- The partial derivatives are found to be

$$\frac{\partial \ln p(x)}{\partial \mu} = \lambda(x - \mu)$$

$$\frac{\partial \ln p(x)}{\partial \lambda} = \frac{1}{2} [\lambda^{-1} - (x - \mu)^2]$$

- Using central 3rd and 4th Gaussian moment, we get Fisher Information matrix

$$E_p[(x - \mu)^3] = 0$$

$$E_p[(x - \mu)^4] = 3\sigma^4$$

$$F = \begin{pmatrix} \lambda & 0 \\ 0 & \frac{1}{2}\lambda^{-2} \end{pmatrix}$$

- The Fisher vector is then

$$\phi(x) = \begin{pmatrix} (x - \mu)/\sigma \\ \frac{1}{\sqrt{2}} (1 - (x - \mu)^2/\sigma^2) \end{pmatrix}$$

Fisher kernels: example with Gaussian data model

- Now suppose an i.i.d. data model over a set of data points

$$p(x) = N(x; \mu, \lambda) = \sqrt{\lambda/(2\pi)} \exp\left[-\frac{1}{2}\lambda(x-\mu)^2\right]$$

$$p(X) = p(x_1, \dots, x_N) = \prod_{i=1}^N p(x_i)$$

- Then the Fisher vector is given by the sum of Fisher vectors of the points
 - ▶ Encodes the discrepancy in the first and second order moment of the data w.r.t. those of the model

$$\phi(X) = \sum_{i=1}^N \phi(x_i) = N \begin{pmatrix} (\hat{\mu} - \mu)/\sigma \\ (\sigma^2 - \hat{\sigma}^2)/(\sigma^2 \sqrt{2}) \end{pmatrix}$$

- ▶ Where

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^N x_i$$

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

Fisher kernels – relation to generative classification

- Suppose we make use of generative model for classification via Bayes' rule
 - ▶ Where x is the data to be classified, and y is the discrete class label

$$p(y|x) = p(x|y) p(y) / p(x),$$
$$p(x) = \sum_{k=1}^K p(y=k) p(x|y=k)$$

and

$$p(x|y) = p(x; \theta_y),$$
$$p(y=k) = \pi_k = \frac{\exp(\alpha_k)}{\sum_{k'=1}^K \exp(\alpha_{k'})}$$

- Classification with the Fisher kernel obtained using the marginal distribution $p(x)$ is at least as powerful as classification with Bayes' rule.
- This becomes useful when the class conditional models are poorly estimated, either due to bias or variance type of errors.
- In practice often used without class-conditional models, but direct generative model for the marginal distribution on X .

Fisher kernels – relation to generative classification

- Consider the Fisher score vector with respect to the marginal distribution on X

$$\begin{aligned}\nabla_{\theta} \ln p(x) &= \frac{1}{p(x)} \nabla_{\theta} \sum_{k=1}^K p(x, y=k) \\ &= \frac{1}{p(x)} \sum_{k=1}^K p(x, y=k) \nabla_{\theta} \ln p(x, y=k) \\ &= \sum_{k=1}^K p(y=k|x) [\nabla_{\theta} \ln p(y=k) + \nabla_{\theta} \ln p(x|y=k)]\end{aligned}$$

- In particular for the alpha that model the class prior probabilities we have

$$\frac{\partial \ln p(x)}{\partial \alpha_k} = p(y=k|x) - \pi_k$$

Fisher kernels – relation to generative classification

- First K elements in Fisher score given by class posteriors minus a constant

$$g(x) = \nabla_{\theta} \ln p(x) = (p(y=1|x) - \pi_1, \dots, p(y=K|x) - \pi_K, \dots)$$

- Consider discriminative multi-class classifier, for the k-th class
 - ▶ Let the weight vector be zero, except for the k-th position where it is one
 - ▶ Let the bias term be equal to the prior probability of that class
- Then

$$f_k(x) = w_k^T g(x) + b_k = p(y=k|x)$$

and thus

$$\operatorname{argmax}_k f_k(x) = \operatorname{argmax}_k p(y=k|x)$$

- Thus the Fisher kernel based classifier can implement classification via Bayes' rule, and generalizes it to other classification functions.

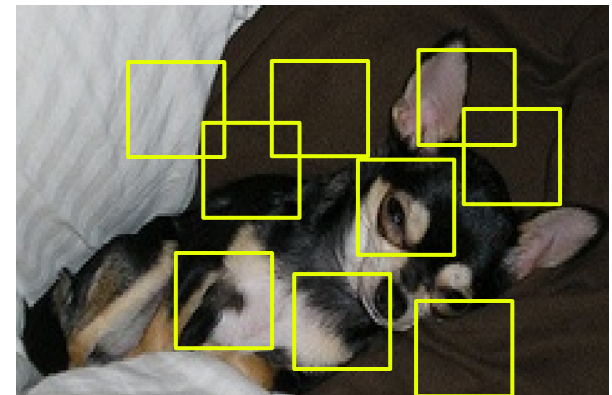
Challenging factors in object recognition

- Intra-class appearance variation
 - ▶ Objects deformation due to pose
 - ▶ Transparency: e.g. bottles
 - ▶ Sub-categories: boat = ferry + yacht +...
- Scene composition
 - ▶ Heavy occlusions: e.g. tables and chairs
 - ▶ Clutter: coincidental image content present
- Imaging conditions
 - ▶ viewpoint, scale, illumination



Representing images as “bags of features”

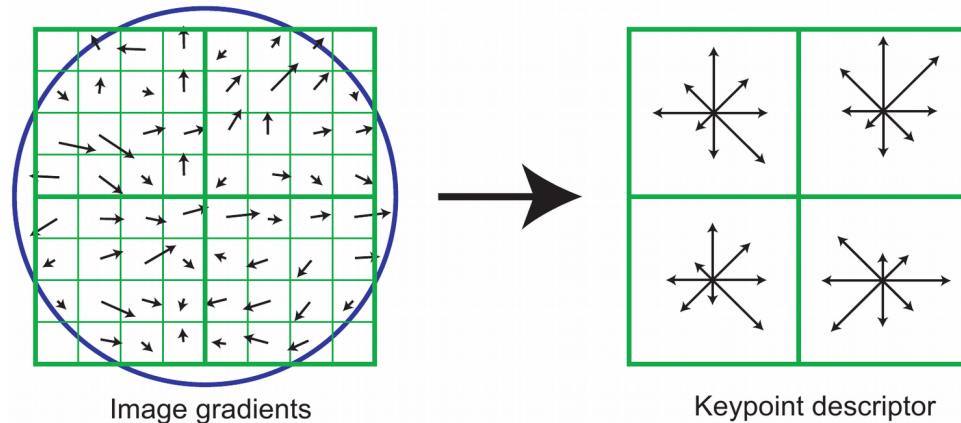
- Global rigid representation likely to be affected by nuisance factors such as deformation, (self-)occlusion, clutter, etc.
- Instead consider local image regions, or “patches”, on which some representation is computed that is (partially) invariant to imaging conditions such as viewpoint, illumination, scale, etc.
 - ▶ Local patterns more likely to be preserved, or at least some of them
- Patch extraction and description stage
 - ▶ Patch sampling from image on dense multi-scale grid, or interest points
 - ▶ Descriptor computation: SIFT, HOG, LBP, color names, ...
- Set of local descriptors characterizes the image (or video, or speech, or ...)
- Feature aggregation stage
 - ▶ Global image signature computed
 - ▶ Can be classified or used for matching
- See e.g. Schmid & Mohr, PAMI, 1997.



Local descriptor based image representations

- SIFT patch description most popular
 - ▶ 4x4 spatial grid
 - ▶ 8 bin orientation histogram
 - ▶ Lowe, IJCV, 2004

$$X = \{x_1, \dots, x_N\}$$

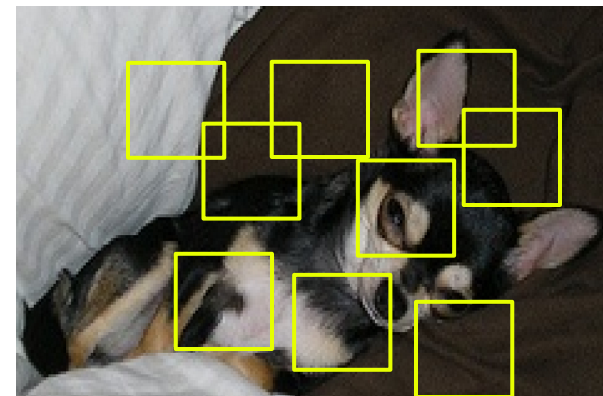


- Coding stage: embed local descriptors, typically in higher dimensional space
 - ▶ For example: assignment to cluster indices

$$\phi(x_i)$$

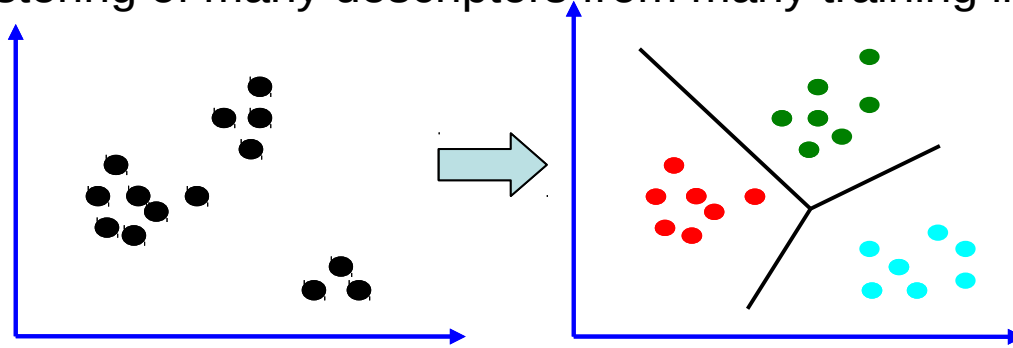
- Pooling stage: aggregate per-patch embeddings
 - ▶ For example: sum pooling

$$\Phi(X) = \sum_{i=1}^N \phi(x_i)$$



The “bag of visual words” representation

- Offline clustering of many descriptors from many training images

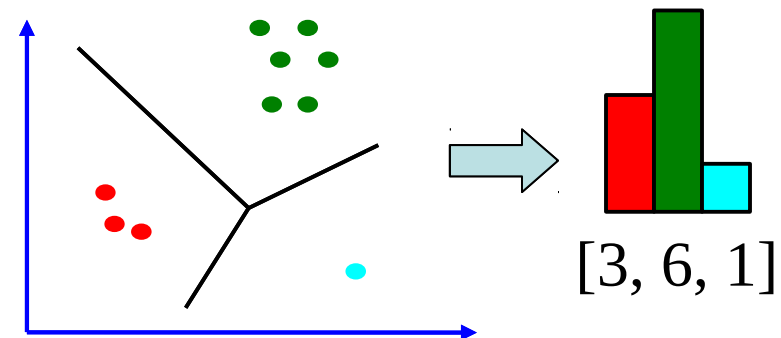
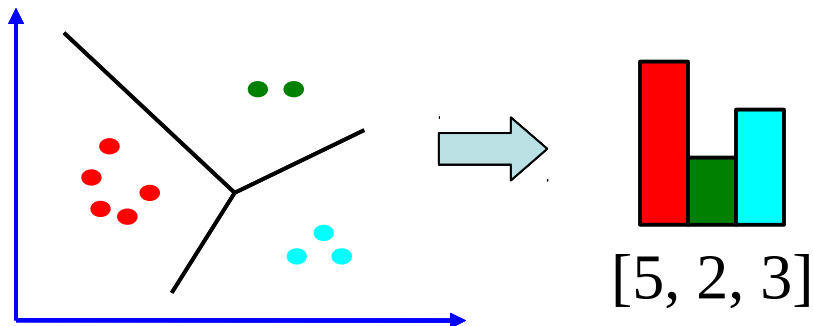


- Encoding a new image:
 - Compute local descriptors, assign to cluster
 - Count histogram of descriptors in each cluster

- Sum pooling of “1-hot encoding” of local descriptors

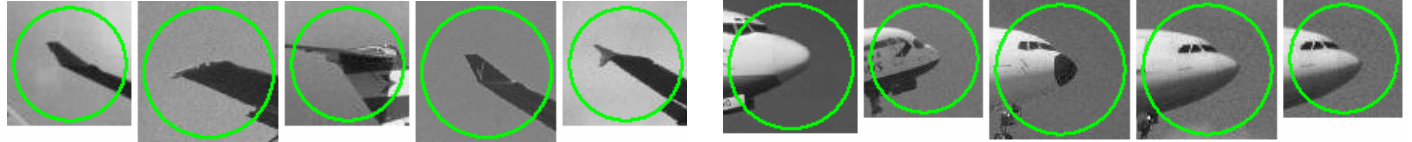
$$\phi(x_i) = [0, \dots, 0, 1, 0, \dots, 0]$$

$$h = \sum_i \phi(x_i)$$

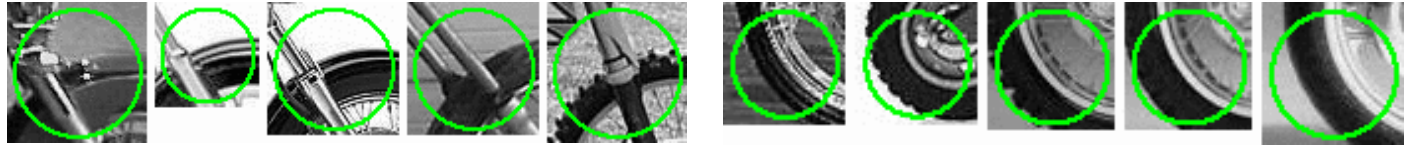


Example visual words found by clustering

Airplanes



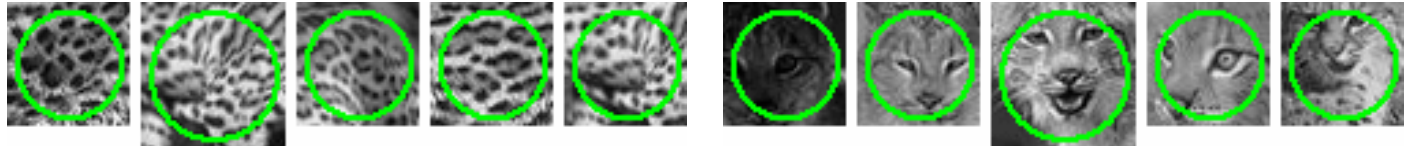
Motorbikes



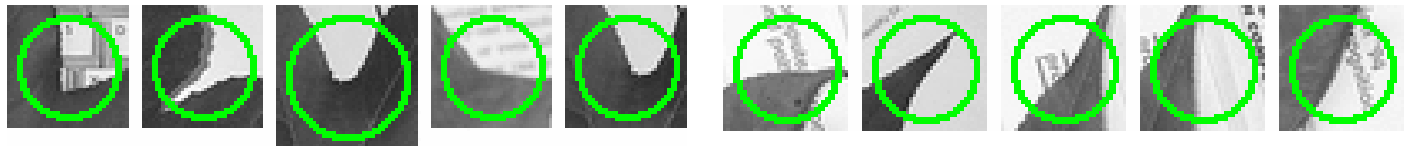
Faces



Wild Cats



Leafs



People



Bikes



Application of FV for bag-of-words image-representation

- Bag of word (BoW) representation

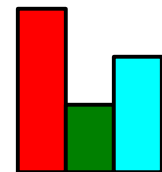
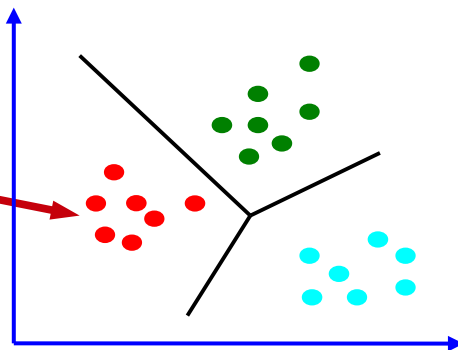
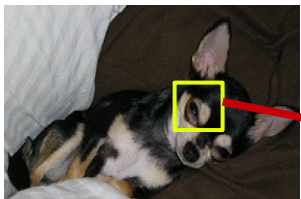
- ▶ Map every descriptor to a cluster / visual word index $w_i \in \{1, \dots, K\}$

- Model visual word indices with i.i.d. multinomial $p(w_i = k) = \frac{\exp \alpha_k}{\sum_{k'} \exp \alpha_{k'}} = \pi_k$

- ▶ Likelihood of N i.i.d. indices: $p(w_{1:N}) = \prod_{i=1}^N p(w_i)$

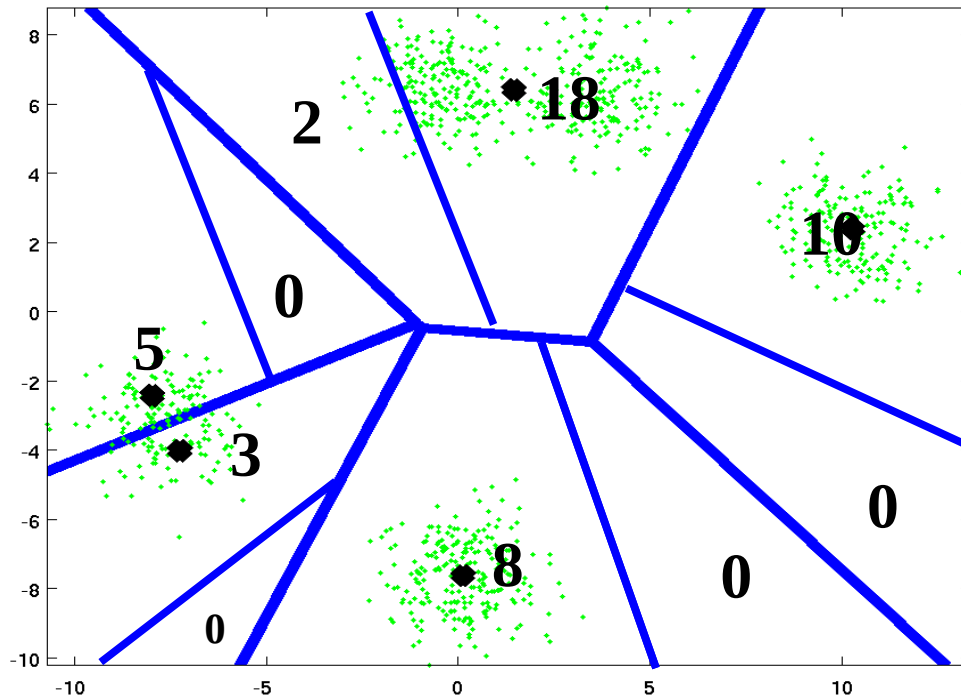
- ▶ Fisher vector given by gradient
 - i.e. BoW histogram + constant

$$\frac{\partial \ln p(w_{1:N})}{\partial \alpha_k} = \sum_{i=1}^N \frac{\partial \ln p(w_i)}{\partial \alpha_k} = h_k - N \pi_k$$



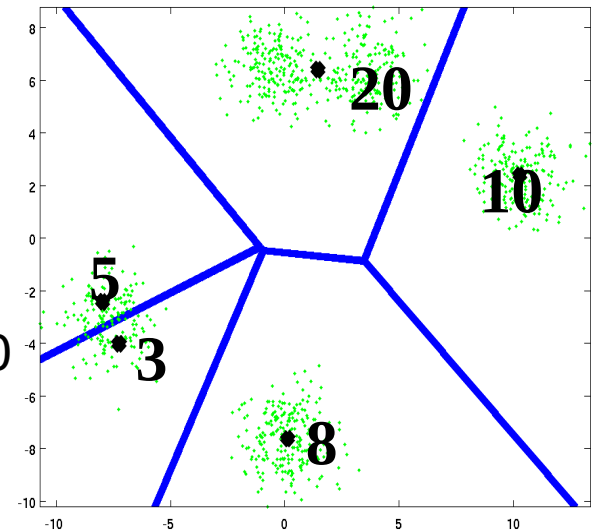
Fisher vector GMM representation: Motivation

- Suppose we want to refine a given visual vocabulary to obtain a richer image representation
- Bag-of-words histogram stores # patches assigned to each word
 - Need more words to refine the representation
 - But this directly increases the computational cost
 - And leads to many empty bins: redundancy



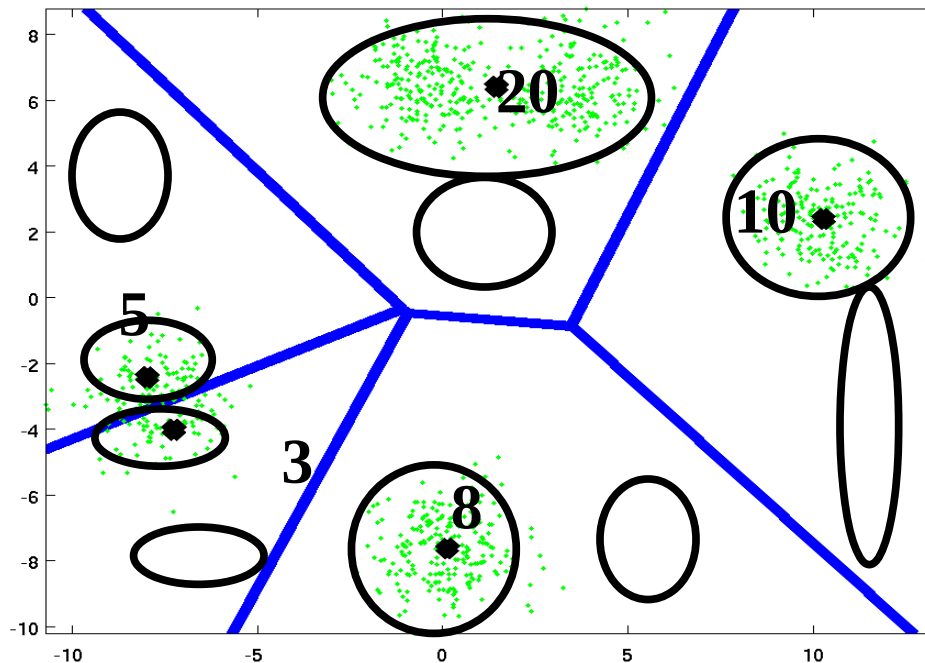
Fisher vector GMM representation: Motivation

- Feature vector quantization is computationally expensive
- To extract visual word histogram for a new image
 - Compute distance of each local descriptor to each k-means center
 - run-time $O(NKD)$: linear in
 - N: nr. of feature vectors $\sim 10^4$ per image
 - K: nr. of clusters $\sim 10^3$ for recognition
 - D: nr. of dimensions $\sim 10^2$ (SIFT)
- So in total in the order of 10^9 multiplications per image to obtain a histogram of size 1000
- Can this be done more efficiently ?!
 - Yes, extract more than just a visual word histogram from a given clustering



Fisher vector representation in a nutshell

- Instead, the Fisher Vector for GMM also records the mean and variance of the points per dimension in each cell
 - More information for same # visual words
 - Does not increase computational time significantly
 - Leads to high-dimensional feature vectors
- Even when the counts are the same, the position and variance of the points in the cell can vary



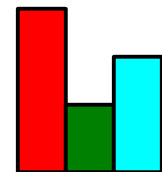
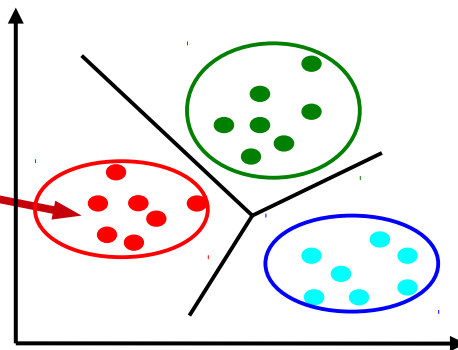
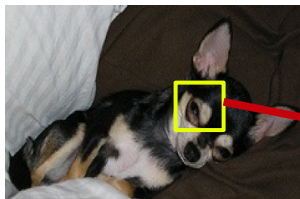
Application of FV for Gaussian mixture model of local features

- Gaussian mixture models for local image descriptors
[Perronnin & Dance, CVPR 2007]
 - ▶ State-of-the-art feature pooling for image/video classification/retrieval

- Offline: Train k-component GMM on collection of local features

$$p(x) = \sum_{k=1}^K \pi_k N(x; \mu_k, \sigma_k)$$

- Each mixture component corresponds to a visual word
 - ▶ Parameters of each component: mean, variance, mixing weight
 - ▶ We use diagonal covariance matrix for simplicity
 - Coordinates assumed independent, locally per Gaussian



Application of FV for Gaussian mixture model of local features

- Model local image features with Gaussian mixture model

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k N(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)$$

- Fisher vector representation: gradient of log-likelihood
 - ▶ For the means and variances we have:

$$F^{-1/2} \nabla_{\boldsymbol{\mu}_k} \ln p(\mathbf{x}_{1:N}) = \frac{1}{\sqrt{\pi_k}} \sum_{n=1}^N p(k|x_n) \frac{(\mathbf{x}_n - \boldsymbol{\mu}_k)}{\boldsymbol{\sigma}_k}$$

$$F^{-1/2} \nabla_{\boldsymbol{\sigma}_k} \ln p(\mathbf{x}_{1:N}) = \frac{1}{\sqrt{2\pi_k}} \sum_{n=1}^N p(k|x_n) \left\{ \frac{(\mathbf{x}_n - \boldsymbol{\mu}_k)^2}{\boldsymbol{\sigma}_k^2} - 1 \right\}$$

- ▶ Soft-assignments given by component posteriors

$$p(k|x_n) = \frac{\pi_k N(\mathbf{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)}{p(\mathbf{x}_n)}$$

Application of FV for Gaussian mixture model of local features

- Fisher vector components give the difference between the data mean predicted by the model and observed in the data, and similar for variance.

- For the gradient w.r.t. the mean

$$F^{-1/2} \nabla_{\mu_k} \ln p(x_{1:N}) = \frac{1}{\sqrt{\pi_k}} \sum_{n=1}^N p(k|x_n) \frac{(x_n - \mu_k)}{\sigma_k} = \frac{n_k}{\sigma_k \sqrt{\pi_k}} (\hat{\mu}_k - \mu_k)$$

▶ where $n_k = \sum_{n=1}^N p(k|x_n)$ $\hat{\mu}_k = n_k^{-1} \sum_{n=1}^N p(k|x_n) x_n$

- Similar for the gradient w.r.t. the variance

$$F^{-1/2} \nabla_{\sigma_k} \ln p(x_{1:N}) = \frac{1}{\sqrt{2\pi_k}} \sum_{n=1}^N p(k|x_n) \left\{ \frac{(x_n - \mu_k)^2}{\sigma_k^2} - 1 \right\} = \frac{n_k}{\sigma_k^2 \sqrt{2\pi_k}} (\hat{\sigma}_k^2 - \sigma_k^2)$$

▶ where $\hat{\sigma}_k^2 = n_k^{-1} \sum_{n=1}^N p(k|x_n) (x_n - \mu_k)^2$

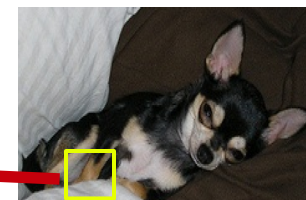
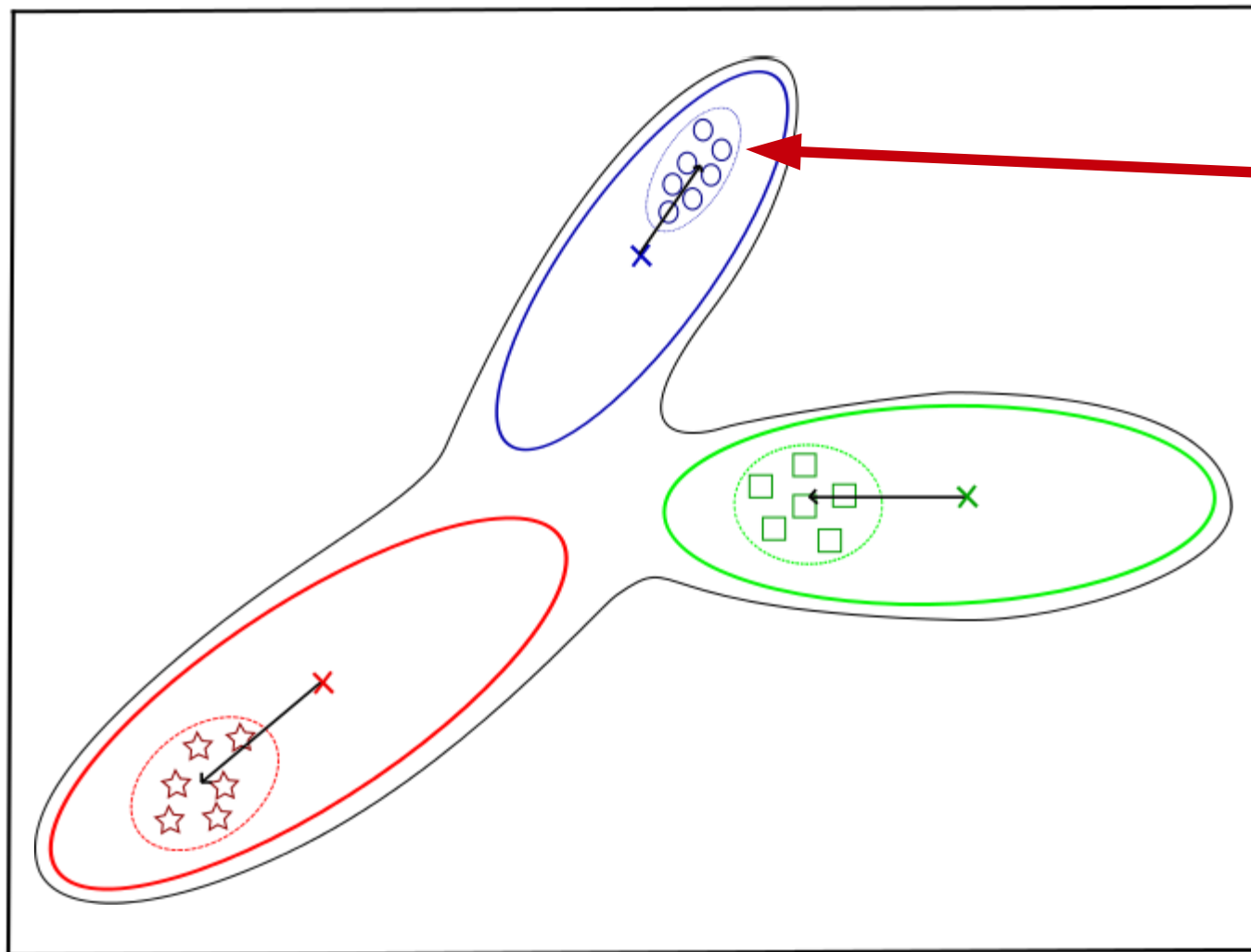
Image representation using Fisher kernels

- Data representation

$$G(X, \Theta) = F^{-1/2} \left(\frac{\partial L}{\partial \alpha_1}, \dots, \frac{\partial L}{\partial \alpha_K}, \nabla_{\mu_1} L, \dots, \nabla_{\mu_K} L, \nabla_{\sigma_1} L, \dots, \nabla_{\sigma_K} L \right)^T$$

- In total $K(1+2D)$ dimensional representation, since for each visual word / Gaussian we have
 - ▶ Mixing weight (1 scalar)
 - ▶ Mean (D dimensions)
 - ▶ Variances (D dimensions, since single variance per dimension)
- Gradient with respect to mixing weights often dropped in practice since it adds little discriminative information for classification.
 - ▶ Results in $2KD$ dimensional image descriptor

Illustration of gradient w.r.t. means of Gaussians



New Data Points

BoW and FV from a function approximation viewpoint

- Let us consider uni-dimensional descriptors: vocabulary quantizes real line
- For both BoW and FV the representation of an image is obtained by sum-pooling the representations of descriptors.
 - ▶ Ensemble of descriptors sampled in an image $X = \{x_1, \dots, x_N\}$
 - ▶ Representation of single descriptor

- One-of-k encoding for BoW $\phi(x_i) = [0, \dots, 0, 1, 0, \dots, 0]$
- For FV concatenate per-visual word gradients of form

$$\phi(x_i) = \left(\dots, p(k|x_i) \left[1 \quad \frac{(x_i - \mu_k)}{\sigma_k} \quad \frac{(x_i - \mu_k)^2 - \sigma_k^2}{\sigma_k^2} \right], \dots \right)$$

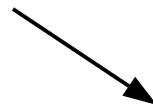
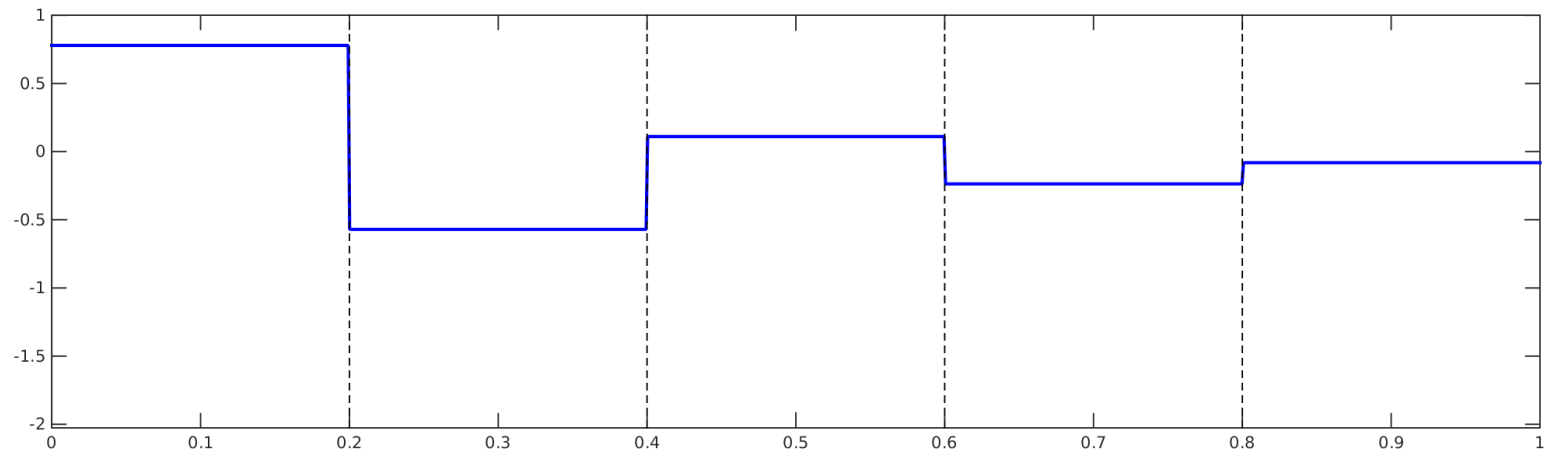
- Linear function of sum-pooled descriptor encodings is a sum of linear functions of individual descriptor encodings:

$$\Phi(X) = \sum_{i=1}^N \phi(x_i)$$

$$w^T \Phi(X) = \sum_{i=1}^N w^T \phi(x_i)$$

From a function approximation viewpoint

- Consider the score of a single descriptor for BoW
 - ▶ If assigned to k-th visual word then $w^T \phi(x_i) = w_k$
 - ▶ Thus: constant score for all descriptors assigned to a visual word



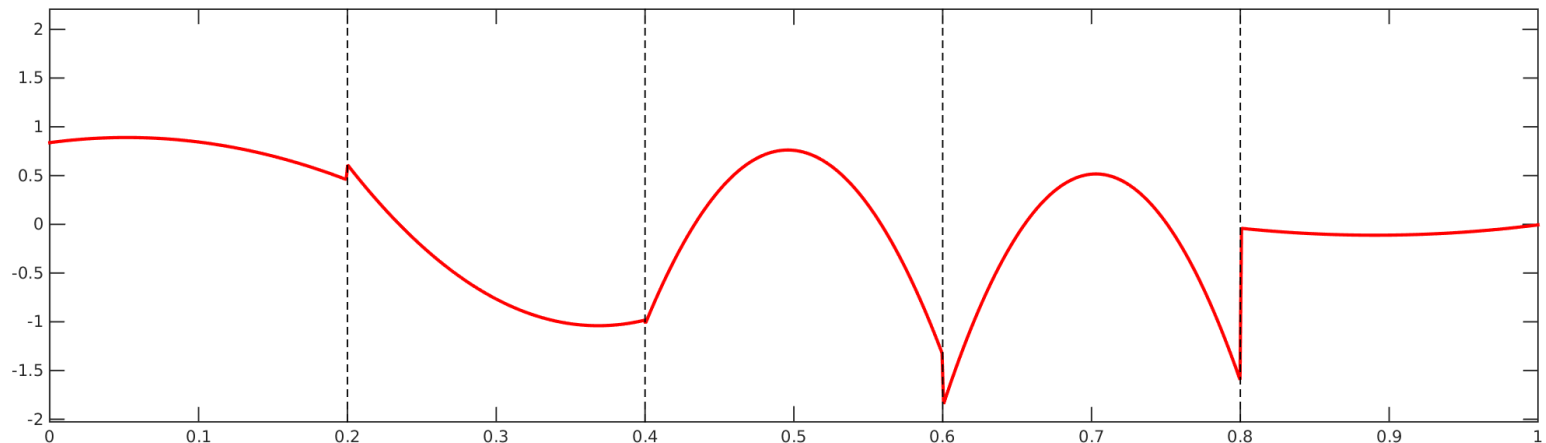
Each cell corresponds to a visual word

From a function approximation viewpoint

- Consider the same for FV, and assume soft-assignment is “hard”
 - ▶ Thus: assume for one value of k we have $p(k|x_i) \approx 1$
 - ▶ If assigned to the k -th visual word:

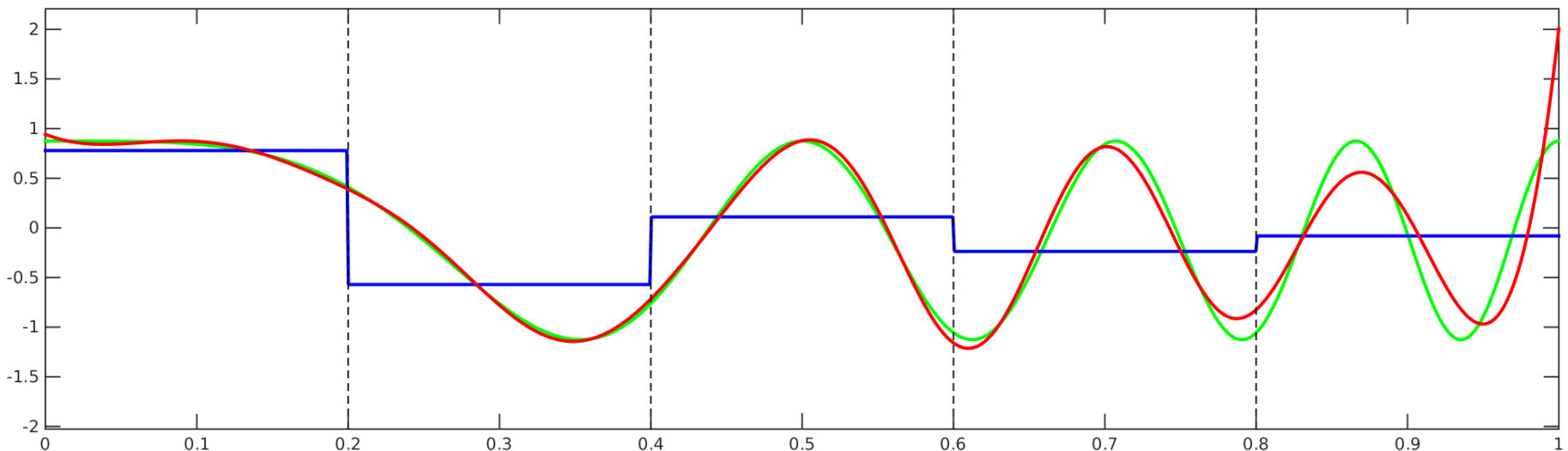
$$w^T \phi(x_i) = w_k^T \begin{bmatrix} 1 & \frac{(x_i - \mu_k)}{\sigma_k} & \frac{(x_i - \mu_k)^2 - \sigma_k^2}{\sigma_k^2} \end{bmatrix}$$

- Note that w_k is no longer a scalar but a vector
- ▶ Thus: score is a second-order polynomial of the descriptor x , for descriptors assigned to a given visual word.



From a function approximation viewpoint

- Consider that we want to approximate a **true classification function (green)** based on either **BoW (blue)** or **FV (red)** representation
 - ▶ Weights for BoW and FV representation fitted by least squares to optimally match the target function
- Better approximation with FV
 - ▶ Local second order approximation, instead of local zero-order
 - ▶ Smooth transition from one visual word to the next



Fisher vectors: classification performance VOC'07

- Yearly evaluation from 2005 to 2012 for image classification

Bicycle



Bus



Car



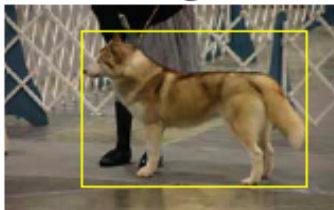
Cat



Cow



Dog



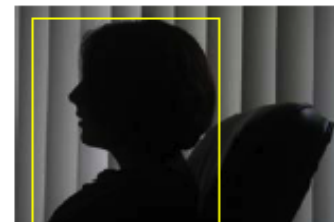
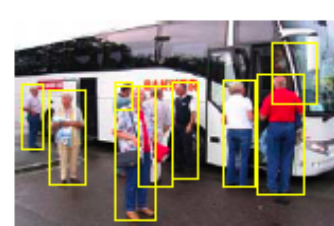
Horse



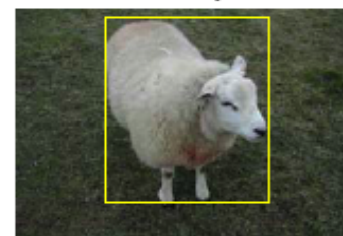
Motorbike



Person

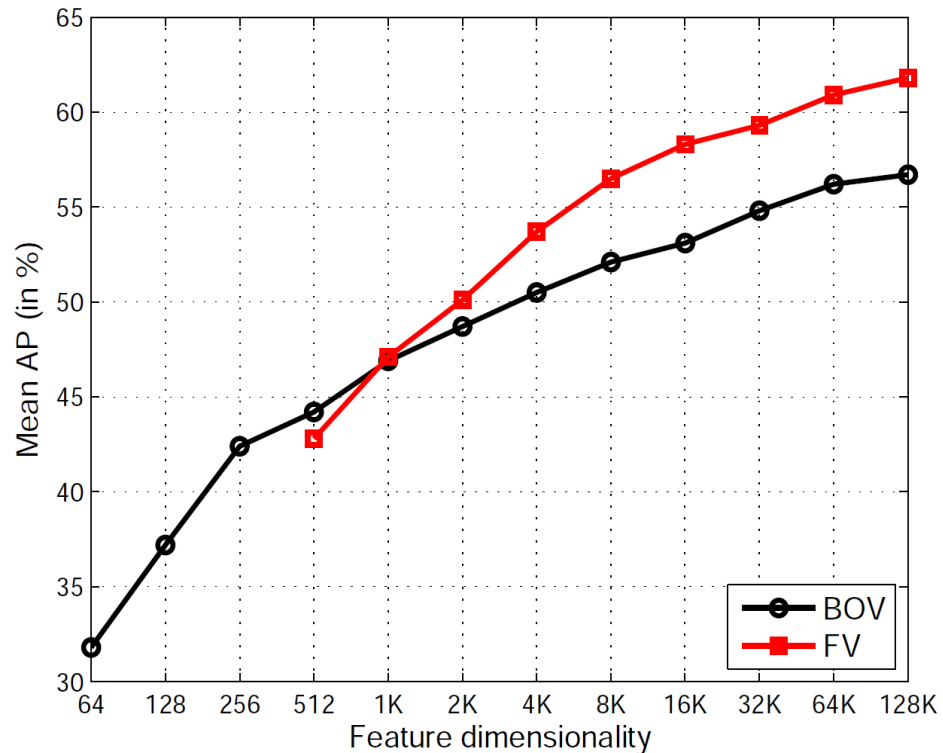
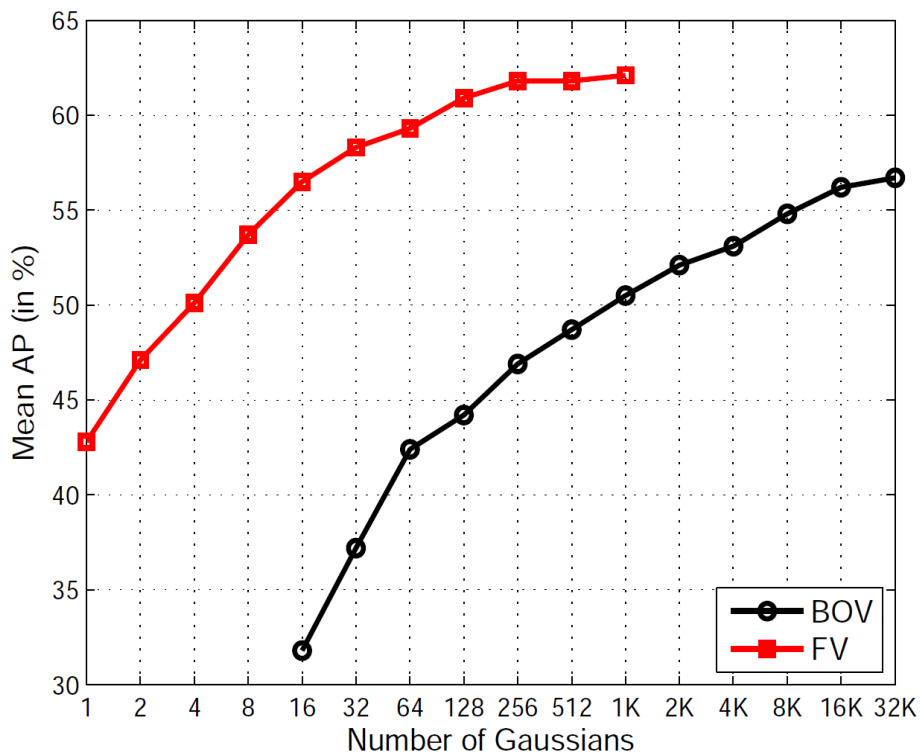


Sheep



Fisher vectors: classification performance VOC'07

- Fisher vector representation yields better performance for a given number of Gaussians / visual words than Bag-of-words.
- For a fixed dimensionality Fisher vectors perform better, and are more efficient to compute



Normalization of the Fisher vector

- Inverse Fisher information matrix F
 - ▶ Renders FV invariant for re-parametrization
 - ▶ Linear projection, analytical approximation for MoG gives diagonal matrix
[Sanchez, Perronnin, Mensink, Verbeek IJCV'13]
$$F = E[g(x)g(x)^T]$$
$$f(x) = F^{-1/2}g(x)$$
- Power-normalization, applied independently per dimension $f(x) \leftarrow \text{sign}(f(x))|f(x)|^\rho$
 - ▶ Renders Fisher vector less sparse
[Perronnin, Sanchez, Mensink, ECCV'10]
 - ▶ Corrects for poor independence assumption on local descriptors
[Cinbis, Verbeek, Schmid, PAMI'15]
$$0 < \rho < 1$$
- L2-normalization
 - ▶ Makes representation invariant to number of local features
 - ▶ Among other Lp norms the most effective with linear classifier
[Sanchez, Perronnin, Mensink, Verbeek IJCV'13]
$$f(x) \leftarrow \frac{f(x)}{\sqrt{f(x)^T f(x)}}$$

Effect of power and L2 normalization in practice

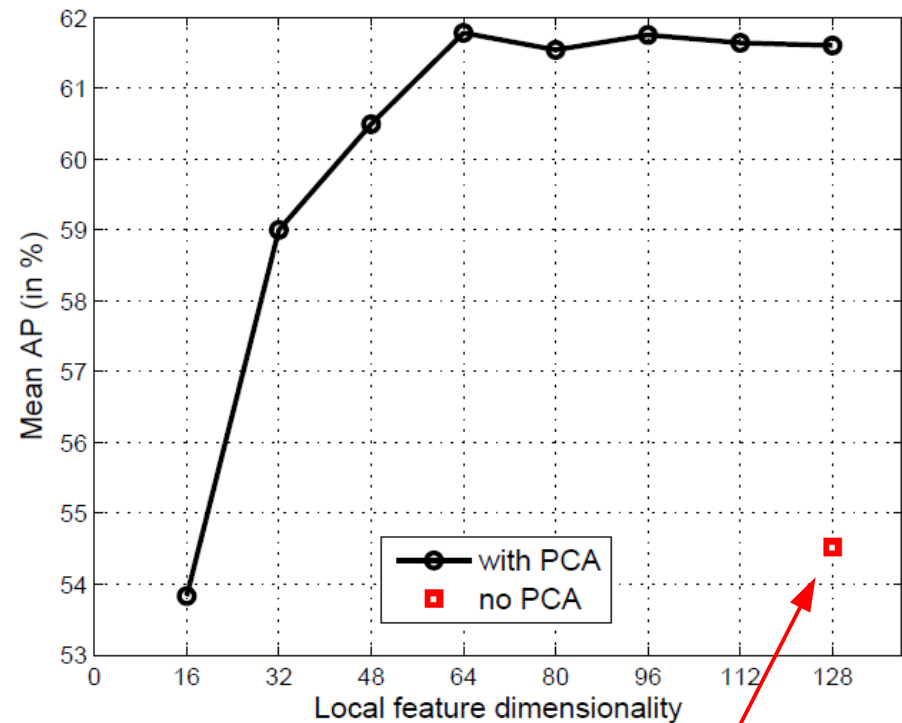
- Classification results on the PASCAL VOC 2007 benchmark dataset.
- Regular dense sampling of local SIFT descriptors in the image
 - ▶ PCA projected to 64 dimensions to de-correlate and compress
- Using mixture of 256 Gaussians over the SIFT descriptors
 - ▶ FV dimensionality: $2 \times 64 \times 256 = 32 \times 1024$

Power Normalization	L2 normalization	Performance (mAP)	Improvement over baseline
No	No	51.5	0
Yes	No	59.8	8.3
No	Yes	57.3	5.8
Yes	Yes	61.8	10.3

PCA dimension reduction of local descriptors

- We use diagonal covariance model
- Dimensions might be correlated
- Apply PCA projection to
 - ▶ De-correlate features
 - ▶ Reduce dimension of final FV
- FV with 256 Gaussians over local SIFT descriptors of dimension 128

Results on PASCAL VOC'07:



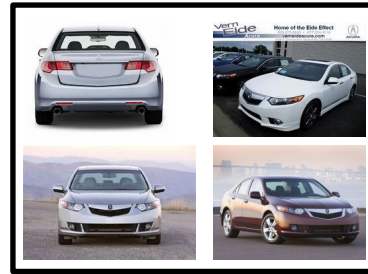
Example applications: Fine-grained classification



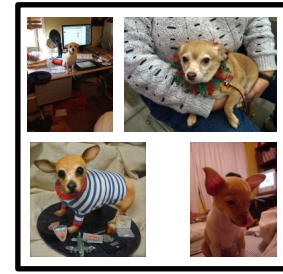
aircraft (100)



birds (83)



cars (196)



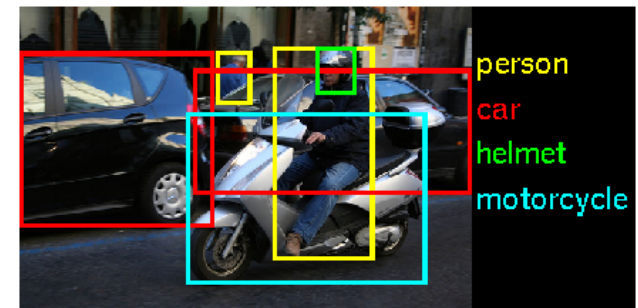
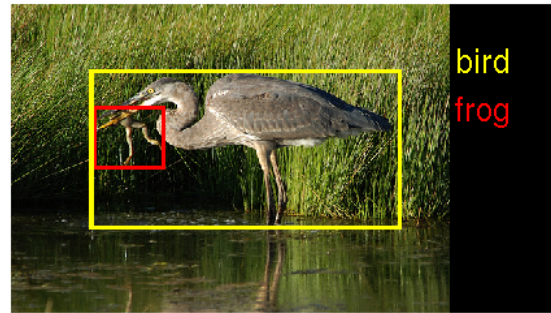
dogs (120)



shoes (70)

- Winning INRIA+Xerox system at FGComp'13:
<http://sites.google.com/site/fgcomp2013>
 - ▶ multiple low-level descriptors: SIFT, color, etc.
 - ▶ Fisher Vector embedding
[Gosselin, Murray, Jégou, Perronnin, “Revisiting the Fisher vector for fine-grained classification”, PRL'14.]
- Many other successful uses of FVs for fine-grained recognition
 - ▶ Rodriguez and Larlus, “Predicting an object location using a global image representation”, ICCV'13.
 - ▶ Gavves, Fernando, Snoek, Smeulders, Tuytelaars, “Fine-Grained Categorization by Alignments”, ICCV'13
 - ▶ Murray, Perronnin, “Generalized Max Pooling”, CVPR'14.

Example applications: object detection



- ImageNet'13 detection: <http://www.image-net.org/challenges/LSVRC/2013/>
- Winning system by University of Amsterdam
 - ▶ region proposals with selective search
 - ▶ Fisher Vector embedding
 - ▶ Fast Local Area Independent Representation (FLAIR)

Van de Sande, Snoek, Smeulders, “Fisher and VLAD with FLAIR”, CVPR'14.

Example applications: face verification



- Face track description:
 - ▶ track face
 - ▶ extract SIFT descriptors
 - ▶ encode using Fisher vectors
 - ▶ pool at face track level

Parkhi, Simonyan, Veldaldi, Zisserman, “A compact and discriminative face track descriptor”, CVPR’14.

- New state-of-the-art results on the YouTube faces dataset

	Method	Accuracy	AUC	EER
1	MGBS & SVM- [37]	78.9 ± 1.9	86.9	21.2
2	APEM FUSION [20]	79.1 ± 1.5	86.6	21.4
3	STFRD & PMML [11]	79.5 ± 2.5	88.6	19.9
4	VSOFF & OSS (Adaboost) [22]	79.7 ± 1.8	89.4	20.0
5	Our VF ² (restricted)	83.5 ± 2.3	92.0	16.1
6	Our VF ² (restricted & flip)	84.7 ± 1.4	93.0	14.9
7	Our VF ² (unrestricted & flip)	83.5 ± 2.1	94.0	13.0
8	Our VF ² (unrestricted & jitt. pool.)	83.8 ± 1.6	95.0	12.3

Example: action recognition and localization



- THUMOS action recognition challenge 2013 & 2014

<http://crcv.ucf.edu/ICCV13-Action-Workshop>

- Winning systems by INRIA-LEAR
 - ▶ improved dense trajectory video features
 - ▶ Fisher Vector embedding

Wang, Oneata, Verbeek and Schmid, "A robust and efficient video representation for action recognition", IJCV'15.

Bag-of-words vs. Fisher vector representation

- Bag-of-words image representation
 - ▶ k-means clustering
 - ▶ histogram of visual word counts, K dimensions
- Fisher vector image representation
 - ▶ GMM clustering
 - ▶ Local first and second order moments, 2KD dimensions
- For a given dimension of the representation
 - ▶ FV needs less clusters, and is faster to compute
 - ▶ FV gives better performance since it is a smoother function of the local descriptors.
- Power and L2 normalizations effective for both representations