

XRCE@ILSVRC2011

Compressed Fisher vectors for LSVR

Florent Perronnin and Jorge Sánchez*
Xerox Research Centre Europe (XRCE)



*Now with CIII, Cordoba University, Argentina



Our system in a nutshell

High-dimensional image signatures: Fisher Vectors (FV)

Perronnin, Sánchez and Mensink, “Improving the Fisher kernel for large-scale image classification”, ECCV’10.

+

Compression: Product Quantization (PQ)

Sánchez and Perronnin, “High-dimensional signature compression for large-scale image classification”, CVPR’11.

+

Simple machine learning: one-vs-all linear SVMs

Linear classifiers learned in primal using Stochastic Gradient Descent (SGD)

Our system in a nutshell

High-dimensional image signatures: Fisher Vectors (FV)

Perronnin, Sánchez and Mensink, “Improving the Fisher kernel for large-scale image classification”, ECCV’10.

+

Compression: Product Quantization (PQ)

Sánchez and Perronnin, “High-dimensional signature compression for large-scale image classification”, CVPR’11.

+

Simple machine learning: one-vs-all linear SVMs

Linear classifiers learned in primal using Stochastic Gradient Descent (SGD)

The need for fine-grained information

As the number of classes increases we need finer-grained information.

⇒ BOV answer to the problem: increase visual dictionary size.

See e.g. Li, Crandall and Huttenlocher, “Landmark classification in large-scale image collections”, ICCV’09.

☹ High computational cost

How to increase the image signature dimensionality without increasing the visual dictionary size?

Embedding view of BOV: given a single local descriptor

$$\varphi_{BOV}(x_t) = [0, \dots, 0, 1, 0, \dots, 0]$$

A possible solution: include **higher-order statistics** in embedding

⇒ **Fisher vector**

Fisher vector

Patch embedding

Notations:

- $X = \{x_t, t = 1 \dots T\}$: set of D-dim local descriptors (e.g. SIFT)
- $u_\lambda(x) = \sum_{i=1}^K w_i u_i(x)$: GMM with parameters $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1 \dots N\}$

Fisher vector: $G_\lambda^X = \frac{1}{T} \sum_{t=1}^T \varphi_{FV}(x_t)$

$$\text{with: } \varphi_{FV}(x_t) = \left[0, \dots, 0, \overbrace{\frac{1}{\sqrt{w_i}} \left(\frac{x_t - \mu_i}{\sigma_i} \right), \frac{1}{\sqrt{2w_i}} \left(\frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right)}^{2D \text{ non-zero dim}}, 0, \dots, 0 \right]$$

A linear classifier on φ induces in the descriptor space

- BOV case: a piecewise constant decision function
 - FV case: a piecewise quadratic decision function
- 😊 FV leads to more complex decision functions than BOV

Perronnin and Dance, “Fisher kernels on visual categories for image categorization”, CVPR’07.

See also: Csurka and Perronnin, “An efficient approach to semantic segmentation”, IJCV’10.

Fisher vector

Improvements

FV normalization is crucial to get state-of-the-art results:

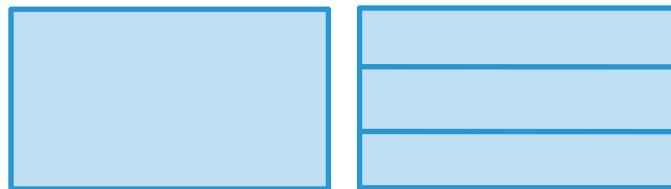
Perronnin, Sánchez and Mensink, “Improving the Fisher kernel for large-scale image classification”, ECCV’10.

- Power normalization: variance stabilization of a compound Poisson

See also: Jégou, Perronnin, Douze, Sánchez, Pérez and Schmid, “Aggregating local descriptors into compact codes”, upcoming TPAMI.

- L2 normalization: invariance to amount of background

Spatial pyramids:



⇒ Final representation = concatenation of per-region Fisher vectors

Let us denote: D = feature dim, N = # Gaussians and R = # regions

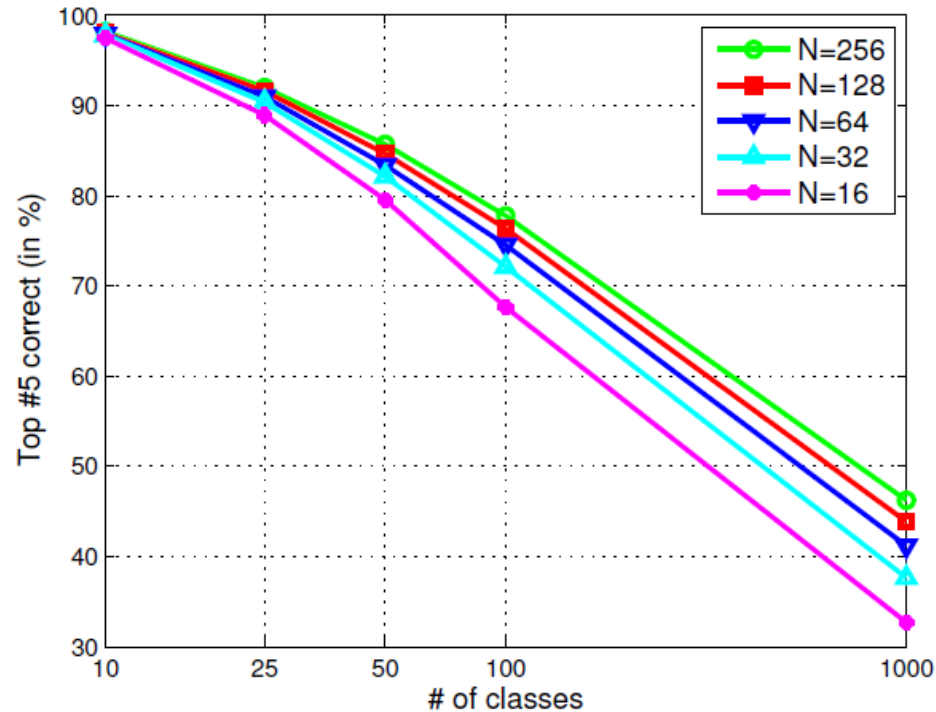
⇒ FV dim = $2DNR$, e.g. $2 \times 64 \times 1,024 \times 4 = 0.5M$ dimensions

⇒ **large (informative) signatures even for small vocabularies**

See also: Chatfield, Lempitsky, Vedaldi, Zisserman, “The devil is in the details: an evaluation of recent feature encoding methods”, BMVC’11.

The need for HD signatures

ILSVRC2010: use subset of classes and vary # Gaussians N



⇒ HD signatures make a difference for large number of classes

See also: Tsuda, Kawanabe and Müller, “Clustering with the Fisher score”, NIPS’02.

The need for compression

The FV is high dimensional (e.g. 0.5M dim) and weakly sparse (only half of the dimensions are non-zero)

☹ **Very high storage / memory / IO cost**

Example using 4-byte floating point arithmetic:

- ILSVRC 2010 \approx 2.8TBs
- ImageNet \approx 23TBs

... per low-level descriptor type

We explored two forms of compression:

- **dimensionality reduction**: dense and sparse
 - ⇒ reduces storage and learning cost
- **coding**: product quantization
 - ⇒ reduces storage only

Dimensionality reduction

Dense projections: e.g. PCA, Gaussian Random Projections, PLS, etc.

- cost linear in the # of input and output dim
- useful only if we can keep a very small # of output dimensions

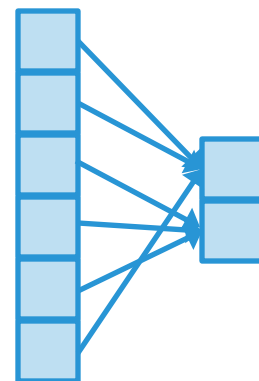
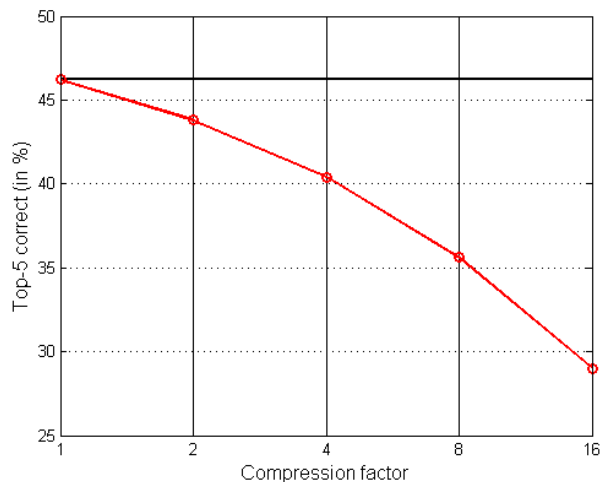
⇒ Huge accuracy drop for reasonable computational cost

Sparse projections: random aggregation with the Hash Kernel (HK)

Shi, Petterson, Dror, Langford, Smola, Strehl, Vishwanathan, “Hash kernels”, AISTATS’09.

Weinberger, Dasgupta, Langford, Smola and Attenberg, “Feature hashing for large scale multitask learning”, ICML’09.

Results on ILSVRC 2010:

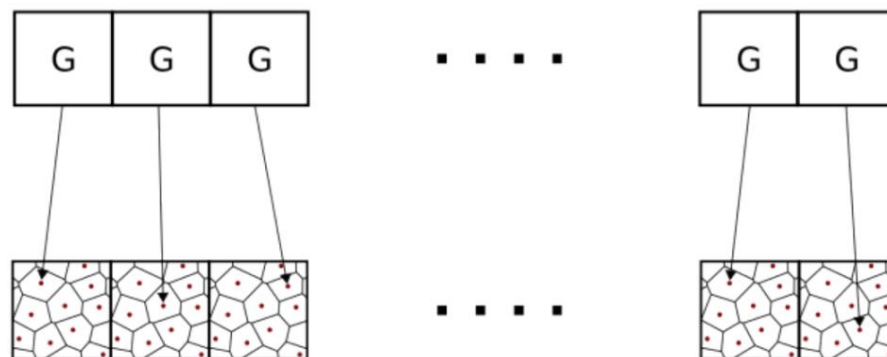


Coding with Product Quantization (PQ)

Principle

Split vector into (small) sub-vectors of G dimensions

- Training step: k-means clustering for each sub-vector (2^K centroids)
- ⇒ $b = K/G$ bits per dimension
- Compression step: encode each sub-vector by its closest centroid index
- ⇒ FV encoded as a vector of indices



Jégou, Douze and Schmid, “Product quantization for nearest neighbor search”, TPAMI’11.

The larger b and G , the higher the accuracy...

... and the higher the computational/storage cost which is in $O(2^{bG})$

⇒ fix b according to storage budget and G according to computational budget

Coding with PQ

Learning

Learning is done in the space of FVs.

Using **Stochastic Gradient Descent** (SGD) to learn linear SVMs:

- one sample is decompressed (look-up table access)
- it is fed to the SGD learning algorithm
- the decompressed sample is discarded

⇒ only one uncompressed sample is “alive” at any time in RAM

No compression needed at test time.

For reference SGD implementation: see Bottou’s webpage

<http://leon.bottou.org/projects/sgd>

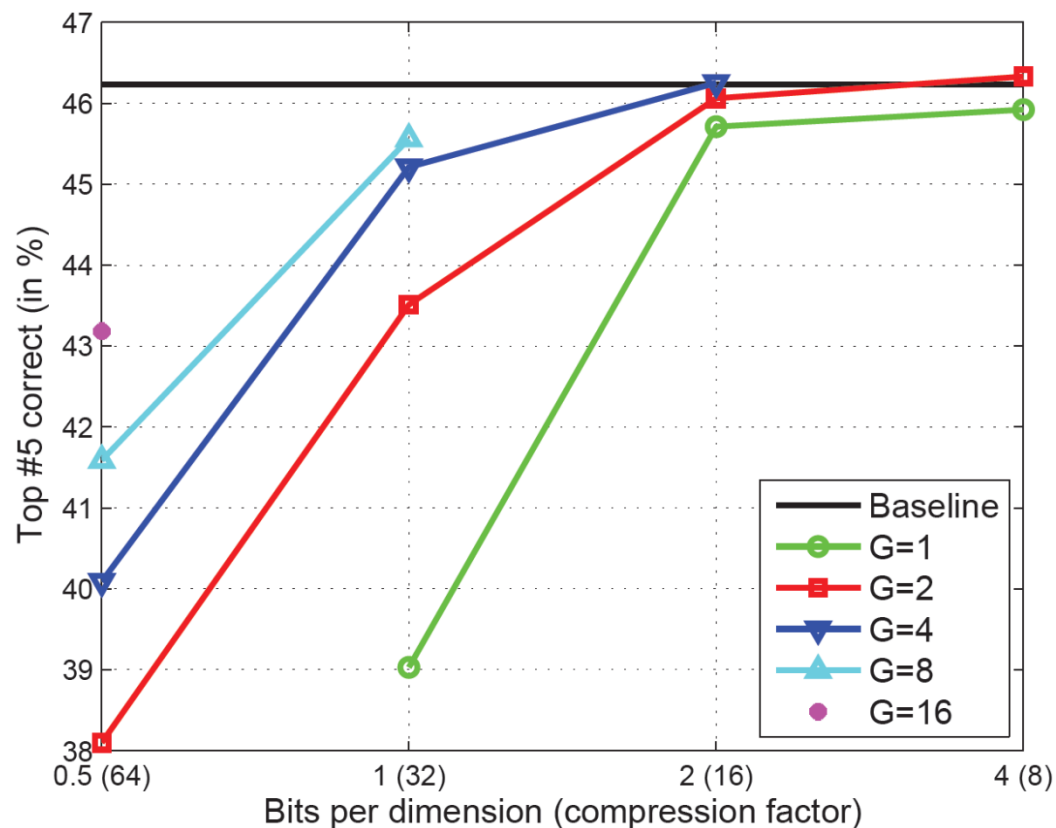
Coding with PQ

Results on ILSVRC 2010

If one has a large storage budget (large b), scalar quantization ($G=1$) is sufficient

G has a large impact for a small number of bits b

PQ performs significantly better than dim-reduction for any compression rate



Summary

Low-level feature extraction \approx 10k patches per image

- SIFT: 128-dim
 - color: 96-dim
- } reduced to 64-dim with PCA

FV extraction and compression:

- $N=1,024$ Gaussians, $R=4$ regions \Rightarrow 520K dim x 2
- compression: $G=8$, $b=1$ bit per dimension

One-vs-all SVM learning with SGD

Late fusion of SIFT and color systems

For details, see: Sánchez and Perronnin, “High-dimensional signature compression for large-scale image classification”, CVPR’11.

Results on ILSVRC 2010

Challenge winners: combination of 6 systems \Rightarrow 71.8 %

- sparse coding and super-vector coding (very close to FVs)
- HOG and LBP features
- various codebooks and spatial pyramids

Lin, Lv, Zhu, Yang, Cour, Yu, Cao and Huang, “Large-scale image classification: fast feature extraction and SVM training”, CVPR’11.

Our results: **74.3%**

\Rightarrow 1 week on a single 16 CPUs machine

Can we get reasonable results with fewer dimensions? Yes!

Concatenation of SIFT FV + color FV (N=256, no SP) \Rightarrow 65K dimensions

+ PQ compression + linear SVMs: 71.4 %

\Rightarrow training set fits in 3.4GB!!!

Can we handle larger datasets?

ImageNet10K:

- 10,184 classes (leaves and internal nodes)
- 9M images: $\frac{1}{2}$ training and $\frac{1}{2}$ test
- accuracy measured as % top-1 correct

SIFT + BOV + SP + fast IKSVM: 6.4 %

Deng, Berg, Li, Fei-Fei, What does classifying more than 10,000 image categories tell us?", ECCV'10.

SIFT + FV (N=256) + SP (R=4) \Rightarrow 130K-dim image signatures
+ PQ compression + linear SVM: **16.7%**

For details, see: Sánchez and Perronnin, "High-dimensional signature compression for large-scale image classification", CVPR'11.