Complex Image Data Basis Classification

Maxim Berman

March 5, 2014

Abstract

In order to improve the speed and scalability of detection systems, new feature coding methods have emerged from ideas related to manifold learning. Locality-constrained Linear Coding (LLC) achieves state-of-the art classifying performance on competitive datasets using only linear SVM, where other methods require the computationally costly use of non-linear classifiers to achieve optimal performance. The aim of this work is to assess the recognition performance of LLC on the Caltech 101 dataset, to experiment with the parameters of this method, and to compare LLC with other coding methods like Vector Quantization or Sparse Coding.

Contents

1 Introduction

Image classification is a field of constant research in Computer Vision. State-of-the art methods rely either on representations learned from deep belief networks, specifically designed features such as HOG [1, 2], or aggregation of multiple descriptors. Once a feature has been chosen, a popular approach to classification consist in using bag-of-features (BoF) and spatial pyramid matching. BoF represents an image as an orderless collection of local features, which is an efficient representation of all the objects contained in the image and performs well in global image categorization, but disregards the spatial layout of the features about the objects, which might however be valuable for classification. Spatial Pyramid Matching (SPM) [3] addresses this issue by aggregating the statistics of local features over increasingly finer subregions. SPM adds spatial ordering

to the representation, is computationally efficient, and gives good performance on image classification tasks.

The coding procedure of SPM is illustrated by figure 1: first, the features are locally or densely extracted from the image; in the case of HOG, these features consist in histograms of the directions of the gradients in the neighborhood of each point. Then, each feature point $\mathbf{x}_i$ is coded using a codebook $\mathbf{B}$: one obtains a code $\mathbf{c}_i$ such that $\mathbf{B}\mathbf{c}_i$ is close to $\mathbf{x}_i$, and if hard Vector Quantization (VQ) is used, each feature is mapped to a single visual word, i.e. $\|\mathbf{c}_i\|_0 = 1$. As will be discussed later, this quantization step is crucial for the representation efficiency, and is the main difference between the different classifying methods studied in this work. After the coding, the different layers of the pyramid are computed: at each level, the image is subdivided into increasingly finer subregions, and within the different subregions the codes are aggregated by taking their maximum or average. Finally, the feature vector is obtained by concatenation of all subregions histograms.

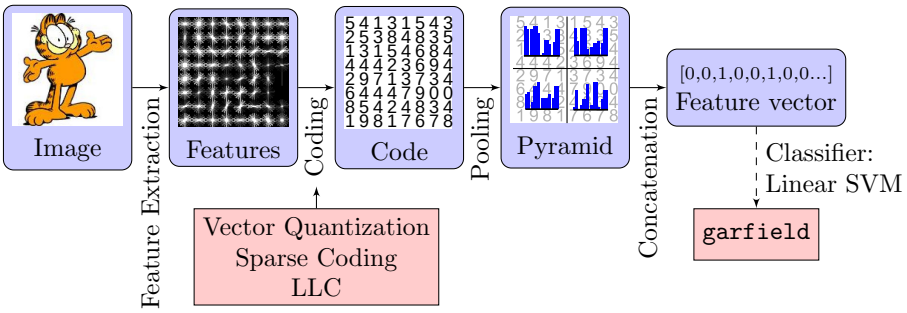


Figure 1: Feature vector coding and classification in the SPM approach.

While this simple coding scheme allows one to have a good understanding of the detection system (as opposed to the use of convolutional networks, for instance), one finds out experimentally that linear SVM is not sufficient to achieve optimal classification performance on the induced representation when a simple a VQ coding step is used. Although this absence of linear separability in feature space can be addressed by using nonlinear kernels such as Chi-Square kernels, the associated cost of $\mathcal{O}(n^2 \sim n^3)$ in training and $\mathcal{O}(n)$ in testing, where n is the training size, is often intractable in real-world applications. Linear SVM on the other hand has an associated cost of $\mathcal{O}(n)$ in training and constant in testing. Obtaining nonlinear features that work better with linear classifiers is therefore an important research direction in order to improve the scalability of detection systems.

Several approaches have been proposed to address this problem, among which the use of Sparse Coding (SC), as in ScSPM [6]. The idea is to enforce sparsity on the coefficients $\mathbf{c}_i$ of $\mathbf{x}_i$ the over-complete basis $\mathbf{B}$ during the coding step – by enforcing a constraint on the ℓ^1 norm of $\mathbf{c}_i$ – in order to generate compact representations of the data, while using a softer quantization method than Vector Quantization, therefore reducing the quantization error.

This work investigates yet another approach by Wang et al. [6]. Their hypothesize that the efficiency of Sparse Coding stems from the locality of the encoding that it generates, and accordingly propose a new coding method, Lo-

cal Coordinate Coding (LCC), that explicitly enforces a locality constraint on the learned basis and feature vectors. Yang et al. [5] proposes variant of LCC, named Locality-constrained Linear Coding or LLC, with the benefit of a fast implementation.

In section 2, various ideas related to locality enforcement in coding will be formalized, and some connections to manifold learning will be made. In section 3, the performance of LCC on the classification of the Caltech 101 image dataset will be assessed, and the influence of the parameters of the method will be studied.

2 Locality constraints in coding

Using feature extraction, an input image I is mapped to a set of D-dimensional local descriptors $X = [\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_N] \in \mathbf{R}^{D \times N}$, which might be sampled densely or at selected interest points; $D = 128$ in the case of SIFT descriptors. Coding aims at extracting a more compact representation of these features. In this section, it will be shown that local coding emerges naturally from the ideas of manifold learning, which are based on the intuition that the features lie on a lower-dimensional manifold.

2.1 Nonlinear learning in high dimensions

Before specifically discussing the coding problem, lets take a step back and consider the classification problem as a whole, predicting the label Y from a single feature $\mathbf{x}_i$. We want to learn a function f from the training data $(\mathbf{x}_m, Y_m)_{m=1\dots n}$ such that $f(\mathbf{x}) \approx Y$. The function f is a real-valued function that includes both the coding and classification part in figure 1, and would represent how likely the feature $\mathbf{x}$ belong to an image from the `garfield` category in that particular example (although only one feature $\mathbf{x}$ is considered here in the classification problem, the SPM pooling takes care of aggregating the features in the whole problem).

Kernel smoothing is one approach to be considered in order to learn the high dimensional nonlinear function f. For a given kernel K, e.g. a Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/2\sigma^2)$, kernel smoothing predicts the label Y of $\mathbf{x}$ as the weighted average of close-by observations

$$f(\mathbf{x}) = \frac{\sum_{m=1}^{n} k(\mathbf{x}_m, \mathbf{x}) Y_m}{\sum_{m=1}^{n} k(\mathbf{x}_m, \mathbf{x})}. \tag{1}$$

This learning method is said to be *zero-order* because the prediction is equivalent to

$$\hat{Y} = \arg\min_{y} \sum_{m=1}^{n} k(\mathbf{x}_m, \mathbf{x})[y - Y_m]^2 \tag{2}$$

which shows that it will be locally constant in a neighborhood of $\mathbf{x}$. Locally first-order Kernel smoothing can also be derived by considering the parametrization $\hat{Y} = \hat{\mathbf{w}}_{\mathbf{x}}^{\mathbf{T}}\mathbf{x} + \hat{b}_{\mathbf{x}}$, the parameters being learned according to

$$[\hat{\mathbf{w}}_{\mathbf{x}}, \hat{b}_{\mathbf{x}}] = \arg\min_{\mathbf{w},\mathbf{b}} \sum_{m=1}^{n} k(\mathbf{x}_m, \mathbf{x})[y - Y_m]^2; \tag{3}$$

the learned function will change linearly in the neighborhood of $\mathbf{x}$. One drawback of this method is its lack of adaptivity, the bandwidth of the kernel being fixed; a natural extension that addresses this issue is to consider the k-Nearest Neighbors (k-NN) of $\mathbf{x}$. Still, this local learning method is not used in high dimension learning. The reason is that in high dimension, each point $\mathbf{x}$ is "far away" from its neighbors; this leads to overfitting and sensitivity to noise. Kernel smoothing is moreover computationally expensive, particularly for the first-order problem (eq. (3)).

2.2 Basis learning

The failure of Kernel Smoothing or k-NN in high dimension can be addressed using basis learning. Instead of selecting all the neighbors of $\mathbf{x}$ in order to classify this feature, one considers a small number of precomputed anchor points $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2 \dots \mathbf{b}_M] \in \mathbf{R}^{D \times M}$; typically $M \sim 1024$. These anchor points can be thought of as representative of the feature space; they are to be less noisy than individual observations of the training sample. The use of anchor points instead of nearest neighbors also solves the computation issues of the k-NN method. A feature point will have an approximate decomposition

$$\mathbf{x}_i \approx \mathbf{B}\mathbf{c}_i \tag{4}$$

with $\mathbf{c}_i \in \mathbf{R}^M$ the coefficients of $\mathbf{x}_i$ in the over-complete basis $\mathbf{B}$. It is here that the coding problem appears specifically: what constraints should be enforce on the codebook $\mathbf{B}$ and on the code $\mathbf{c}_i$ in order to give a formal meaning to the decomposition in equation (4) and learn anchor points ($\mathbf{b}_i$) and code ($\mathbf{c}_i$) best suited for the classification task? Next, we will describe various such coding schemes: VQ, ScSPM, LCC and LLC.

VQ: Vector Quantization

Vector Quantization coding solves the constrained least-square problem

$$\begin{aligned} &\arg\min_{\mathbf{C}} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}\mathbf{c}_i\|^2 \\ &\text{s.t. } \|\mathbf{c}_i\|_0 = \|\mathbf{c}_i\|_1 = 1, \mathbf{c}_i \succeq 0, \forall i. \end{aligned} \tag{5}$$

Each feature $\mathbf{x}_i$ is thus mapped to exactly one element of the codebook, in practice found by searching the nearest neighbor. This simple 0-order scheme was used in SPM original implementation [3]; the hard assignment of x_i to one codebook element introduces however a high quantization loss, i.e. loss of information on the original features due to the coding.

ScSPM: Sparse Coding

ScSPM uses sparse coding to lower the quantization loss of VQ; a feature $\mathbf{x}_i$ is soft assigned to a small number of codebook elements by solving the ℓ^1 norm regularized problem

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}\mathbf{c}_i\|^2 + \lambda \|\mathbf{c}_i\|_1. \tag{6}$$

The generated codes still have only a small number of nonzero values, while much less quantization error than VQ are introduced. Accordingly, this method easily outperforms VQ-SPM on standard image benchmarks [6].

LCC: Local Coordinate Coding

Yu et al. [7] explain the success of ScSPM not only by the soft-assignment, but also by the fact that Sparse Coding produces local codes: for a reconstruction $\mathbf{x} \approx \mathbf{B}\mathbf{c}_i$, $\mathbf{c}_i \approx \mathbf{0}$ when $\mathbf{b}_i$ is far from $\mathbf{x}$. They introduce LCC in order to explicitly enforce such a locality constraint. In this method, the code $\mathbf{C}$ is obtained by solving

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}\mathbf{c}_i\|^2 + \lambda \|\mathbf{\Delta}_i^{1+p} \mathbf{c}_i\|_1 \tag{7}$$

where $\mathbf{\Delta}_i = \mathrm{diag}(\|\mathbf{b}_1 - \mathbf{x}_i\|, \ldots, \|\mathbf{b}_M - \mathbf{x}_i\|)$.

Here the sparsity term in problem (6) has been replaced by a localization term, which favors basis vectors that are close to x_i in the coding. Interestingly, sparse coding appears as a limiting case with no locality constraints with $p = -1$. A codebook $\mathbf{B}$ can be learned by optimizing problem (7) jointly on the code and on $\mathbf{B}$ on the training data, using EM-like alternate minimization on $\mathbf{B}$ (lasso problem) and $\mathbf{C}$ (least-square problem when $p = 1$).

The authors show [7] that LCC has strong theoretical grounds in manifold learning, on the contrary of sparse coding which is based on heuristic arguments. Using the approximate decomposition

$$\mathbf{x}_i \approx \sum_{m=1}^{M} c_{im} \mathbf{b}_m, \tag{8}$$

the nonlinear classifying function we want to learn $f(\mathbf{x}_i) \approx Y$ is locally approximated as a linear function

$$f(\mathbf{x}_i) \approx \sum_{m=1}^{M} c_{im} f(\mathbf{b}_m), \tag{9}$$

in the local bases around $\mathbf{x}_i$. Assuming that the features lie in on a compact manifold $\mathcal{M} \subset \mathbf{R}^D$, the objective function in problem (7) can be shown to be a first-order approximation to the approximation error in equation (9), under some Lipschitz-smoothness assumptions on f. Moreover, for a typical manifold with well-defined curvature, we may choose $p = 1$ in the localization term. An important result is the result of consistency: as the number of training samples $n \to \infty$, LCC can learn any nonlinear function on the manifold $\mathcal{M}$, and the rate of convergence depends on the intrinsic dimension of the manifold $m(\mathcal{M})$ and not on the dimension of the feature space D.

LCC can be seen as a locality constrained and theoretically sound version of sparse coding. The high dimensional nonlinear function becomes linear with respect to the anchor point. Therefore, LCC is a first-order coding method, a high-dimensional generalization of the first-order smoothing kernel method (eq. (3)) where the use of a codebook addresses the computation and dimensionality issues.

LLC: Locality-constrained Linear Coding

Building upon LCC, Yang et al. [6] propose a faster alternative to localize the coding. Under LLC, the code learning problem becomes

$$\begin{aligned} \arg\min_{\mathbf{C}} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}\mathbf{c}_i\|^2 + \lambda \|\tilde{\mathbf{\Delta}}_i \mathbf{c}_i\|^2 \\ \text{s.t. } \mathbf{1}^\intercal \mathbf{c}_i = 1, \forall i \end{aligned} \tag{10}$$

where

$$\tilde{\mathbf{\Delta}}_i = \exp\left(\mathbf{\Delta}_i / \sigma\right) / Z_i \tag{11}$$

Z_i being chosen such that $\tilde{\mathbf{\Delta}}_i \preceq \mathbf{1}$. Compared to LCC coding (problem (7)), the localization term in equation (10) is an ℓ^2 constraint, and the localization penalty $\tilde{\mathbf{\Delta}}$ grows exponentially around x instead of linearly. The added constraints $\mathbf{1}^\intercal \mathbf{c}_i = 1$ ensures that the learned codes are shift-invariant.

Here the localization term departs more significantly from sparse coding than in LCC, by the use of the ℓ^2 norm. Still, because of the fast increase of the localization penalty (equation (11)), only a few values of $\mathbf{c}_i$ will be nonzero. The added benefit of this variant is that its solution can be derived analytically as

$$\begin{aligned} \tilde{\mathbf{c}}_i &= (\mathbf{C}_i + \lambda \tilde{\mathbf{\Delta}}_i^2) \setminus \mathbf{1} \\ \mathbf{c}_i &= \tilde{\mathbf{c}}_i / \mathbf{1}^\intercal \tilde{\mathbf{c}}_i \end{aligned} \tag{12}$$

where $\mathbf{C}_i = (\mathbf{B} - \mathbf{1}\mathbf{x}_i^\intercal)(\mathbf{B} - \mathbf{1}\mathbf{x}_i^\intercal)^\intercal$ is the data covariance matrix.

2.3 Fast implementation of LLC

Coding

In practice, the few basis vectors selected by the significant values of $\mathbf{c}_i$ after solving equation 3 will be roughly the nearest neighbors of $\mathbf{x}_i$. Therefore, LLC coding can be approximated by using the K nearest neighbors of $\mathbf{x}_i$ as the local basis $\mathbf{B}_i$, and solving equation (10) using only this local basis $\mathbf{B}_i$ as the codebook, finding

$$\begin{aligned} \arg\min_{\mathbf{C}} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}_i\mathbf{c}_i\|^2 + \lambda \|\tilde{\mathbf{\Delta}}_i \mathbf{c}_i\|^2 \\ \text{s.t. } \mathbf{1}^\intercal \mathbf{c}_i = 1, \forall i. \end{aligned} \tag{13}$$

This approximation reduces the cost of computing the code using equation (12) from $\mathcal{O}(M^2)$ to $\mathcal{O}(M + K^2)$, where the number of nearest neighbors we have to consider $K \ll M$ and is linked to the chosen bandwidth σ in the localization penalty. In practice, the code computation in further simplified by setting the localization penalty $\tilde{\mathbf{\Delta}}$ proportional to the identity inside the K-NN.

The only overhead is the need to find the nearest neighbors of the $\mathbf{x}_i$. This K-NN search can be made efficient by hierarchical representation of the feature space, allowing the use of much larger codebooks.

Codebook learning

A simple way to generate the codebook $\mathbf{B}$ is to use a clustering method such as K-Means clusterings to form M clusters from the training feature samples. As suggested by the experiments in next section, this approach already gives satisfactory results.

One may however improve classification results by optimizing the codebook for LLC coding specifically. The idea is to optimize the objective function in equation (10) jointly on the code and the codebook, adding some ℓ^2 constraint on the code in order to define the problem and allow for efficient optimization as in

$$\begin{aligned} \arg\min_{\mathbf{C},\mathbf{B}} & \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{B}\mathbf{c}_i\|^2 + \lambda \|\tilde{\mathbf{\Delta}}_i \mathbf{c}_i\|^2 \\ \text{s.t. } & \mathbf{1}^\intercal \mathbf{c}_i = 1, \forall i \\ & \|\mathbf{b}_j\|^2 \leq \mathbf{1}, \forall j. \end{aligned} \tag{14}$$

3 Experiments and results

Classification: speed and performance on Caltech 101

Caltech 101 is an image dataset for classification. It consists of 9144 images grouped into 102 categories, including one background category.

With the code given by the authors of LLC, using a dictionary of size $M = 1024$ trained by k-means, one obtains the results indicated in the first row of table 1. This is to be compared with the second row, where a dictionary of size $M = 2048$ is trained according to the partial gradient descent method described subsection 2.3 (the values are taken from article [5]). The measured coding time averages to 0.48 second per image, including the save of the figures in a separate file. The results of a 7-layers convolutional network trained on ImageNet [8] is listed along the results (7L CN), as well as the result of a 2-layer scattering network (2L SN) [4]. The results of Linear SVM classification with pyramid pooling but no LLC coding is shown as the entry "SIFT+SPM".

type, `dict`, pool.	5	10	15	20	25	30
LLC, `k-means`, max	48.2 ± 0.6	58.6 ± 0.4	63.75 ± 0.35	66.9 ± 0.9	69.4 ± 0.6	71.1 ± 0.4
LLC, `trained`, max	51.15	59.77	65.43	67.74	70.16	73.44
7L CN, -, max	-	-	83.8 ± 0.5	-	-	86.5 ± 0.5
2L SN, -, mean	-	-	-	-	-	67.3 ± 0.5
SIFT+SPM, -, max	-	-	43.73 ± 0.8	-	-	50 ± 1
LLC, `k-means`, mean	27 ± 1	36.7 ± 0.9	40 ± 1	46.9 ± 0.8	50.0 ± 0.7	52 ± 1
LLC, `sparse`, max	46.3 ± 0.9	-	66.35 ± 0.51	65.8 ± 0.7	-	69 ± 1
LLC, `triangles`, max	46 ± 1	-	55 ± 19	57 ± 20	60 ± 21	62 ± 21
ScSPM, `sparse`, max	52.4 ± 0.4	61.9 ± 0.8	66.35 ± 0.51	69.3 ± 0.6	70.6 ± 0.5	72.9 ± 0.7

Table 1: Classification results in various methods, type of pooling, dictionary, and number of training samples (indicated by the first row numbers).

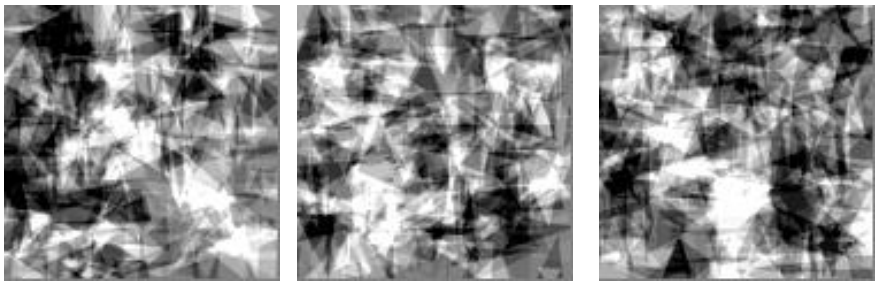


Figure 2: Examples of images whose features constitute the `triangles` dictionary

3.1 Role of the dictionary and pooling

Various tests were made to clarify the role played by the choice and training of the dictionary, and the type of pooling. The results are listed together in 1. The different dictionaries that were used are:

- `k-means`, where 1024 vectors are computed from k-means in feature spaces over more than 200000 examples extracted from PascalVOC dataset (in order to avoid dataset bias);
- `sparse`, a dictionary trained by sparse coding constraint by the code given with ScSPM
- `triangles`, a dictionary trained by picking samples of the SIFT features associated with a database of 100×100 images containing 500 triangles drawn at random (examples in figure 2)

The SIFT features in the dictionary have $4 \times 4 \times 8 = 128$ dimensions (16 cells and 8 bins for the gradient). The visualization of a few of these histograms is presented in figure 3. One notices a surprising fact in the second row: the features contained in the trained sparse dictionary appear localized in feature space.

Notice that LLC optimization of the dictionary doesn't add much to the performance of the algorithm ($0.3 \sim 1.4\%$). More generally, we notice that the use of other dictionaries, including one generated at random, does not have great impact on the classification performance. Notice however the fact that the accuracy has a huge variance using the `triangles` dictionary. This can be explained by the redundancy in the images: some features will be very close to each other, and produce more than one local basis for a vector to be encoded in which breaks down classification. Therefore the algorithm is dependent on the random choice of training images, since some good choices will have less redundancy than others.

Results 1 also show that max pooling is more efficient than average pooling. This is surprising, because the whole point of LLC was to construct a linear separable space, and by applying a non-linear operation on this space, one should in principle loose this property. In fact, the maximum step already selects features. One explanation would be that this "subselection" acts as a detector and remove unnecessary redundancies from the codebook, thereby improving the global richness of the representation.

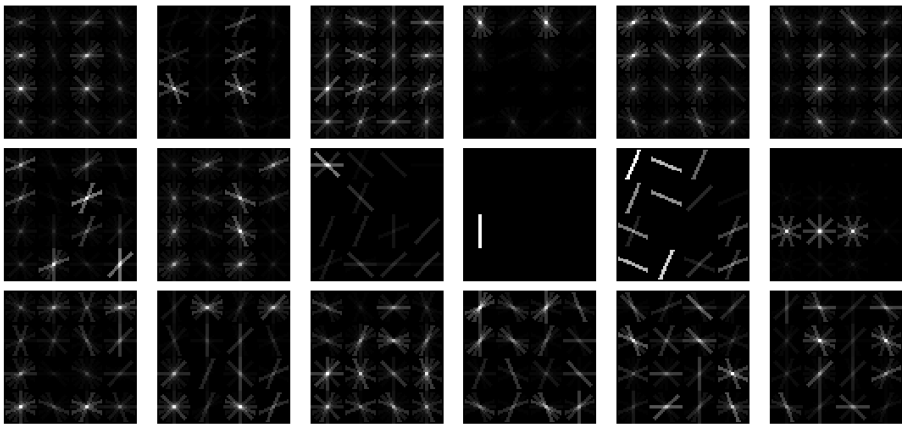
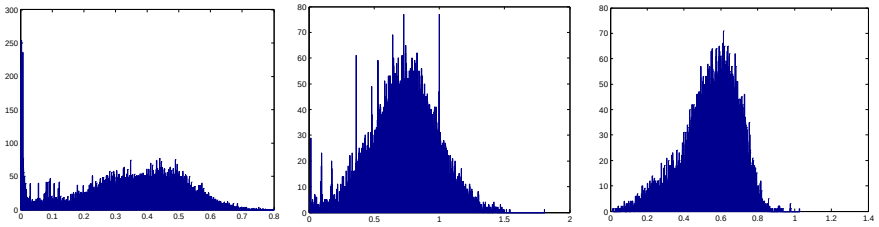


Figure 3: Visualization of some elements of the `k-means` (top), `sparse` (middle) and `triangles` (bottom) dictionaries.



(a) LLC + `kmeans` (b) Sparse Code + `sparse` (c) LLC + `sparse`

Figure 4: Measured distances between a query feature and the vectors of its associated local basis, using different methods and dictionaries.

Locality of the coding

Figure 4 presents the distances measured between feature points and their closes local basis. One sees that the LLC code on a k-means dictionary is indeed more local than the sparse code in ScSPM or than the LLC code on a `sparse` dictionary. The fact that ScSPM performs as well as LLC despite less apparent locality suggests that locality is not the end of the picture, and that the choice of the dictionary still has its importance when it comes to having the best possible results.

4 Conclusion

In terms of generating image descriptors suited for large-scale classification, LLC codes are promising. They are fast to compute, and induce the same level of classification accuracy as sparse coding. The success of LLC can moreover be explained theoretically using ideas related to manifold learning; they pave the way to a better mathematical understanding of features, coding and classification.

However, some design choices in LLC are still subject to experimentation; in particular, the role played by dictionary learning still raises questions, as does the use of max-pooling.

The experiments done in this work suggest that while optimizing the dictionary for a locality criterion does improve the classification accuracy, the gain compared to a simple k-means trained dictionary is very modest. Moreover, the use of random dictionaries, or exemplar dictionaries by simple aggregation of features, still produces reasonable results. Hence the key to explain the success of sparse and locality-constrained coding is the coding itself and not the dictionary.

It remains to show whether there is better to do for generating dictionaries without learning them. The appearance of structures in feature spaces in trained dictionaries suggests that this is the case.

References

[1] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. of CVPR'05*, volume 1, pages 886–893. IEEE, 2005.

[2] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[3] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. of CVPR'06*, volume 2, pages 2169–2178. IEEE, 2006.

[4] Edouard Oyallon, Stéphane Mallat, and Laurent Sifre. Generic deep networks with wavelet scattering. *arXiv preprint arXiv:1312.5940*, 2013.

[5] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Proc. of CVPR'10*, pages 3360–3367. IEEE, 2010.

[6] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proc. of CVPR'09*, pages 1794–1801. IEEE, 2009.

[7] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Proc. of NIPS'09*, volume 9, page 1, 2009.

[8] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.