Published in Image Processing On Line on YYYY–MM–DD.
ISSN 2105–1232
This article is available online with supplementary materials, software, datasets and online demo at
http://dx.doi.org/10.5201/ipol.YYYY.XXXXXXXX

PREPRINT August 7, 2013

LSSC

Yohann Salaun[1] &Marc Lebrun[2]

[1] Polytechnique, France (yohann.salaun@polytechnique.org)
[2] CMLA, ENS Cachan, France (marc.lebrun@cmla.ens-cachan.fr)

Abstract

1 Overview

2 Theoritical Description

2.1 Notations

In order to keep coherence with [3], the notations used are the same. A picture of n pixels is seen as a column vector in $\mathbb{R}^n$. The noisy picture is noted $\mathbf{y}$ and the denoised one $\mathbf{x}$. The i-th pixel of x is noted $\mathbf{x}[i]$ and the patch centered in $\mathbf{x}[i]$ and of size m is noted $\mathbf{x}_i$.

2.2 Learned Sparse Coding

The idea behind this method is to assume that the denoised picture is a signal that can be approximated by a sparse linear combinations of elements from a basis set. The basis set is called a dictionary $\mathbf{D} \in \mathbb{R}^{m\times k}$ and is composed of k elements. The denoised patches are then computed from $\mathbf{D}$ with:

$$min_{\boldsymbol{\alpha}\in\mathbb{R}^k}||\boldsymbol{\alpha}||_p \;\; s.t. \;\; ||\mathbf{y}_i - \mathbf{D}\boldsymbol{\alpha}||_2^2 \leqslant \epsilon \tag{1}$$

$\mathbf{D}\boldsymbol{\alpha}$ is the estimate of the denoised patch and $||\boldsymbol{\alpha}||_p$ is a regularization term that impose sparsity for $\boldsymbol{\alpha}$.
p is usually 0 or 1. Eq. 1 becomes NP-hard to solve when $p = 0$ and a greedy algorithm such as Orthogonal Matching Pursuit [5] can give an approximation. With $p = 1$, the problem is convex and solved efficiently with the LARS algorithm [6]. Experimental observations [7] have shown that the learning part is better with $p = 1$ and the recomposition part with $p = 0$.
ϵ can be chosen according to the value of the estimated standard deviation of the noise.

2.3 Simultaneous Sparse Coding

3 Algorithm Description

3.1 Algorithm Overview

The first part consists in initializing a dictionary that will denoise roughly the picture.
Once the picture is denoised a first time, a clustering is made in order to regroup similar patches for further treatment.
Then, iteratively for each clusters, the dictionary is updated using simultaneous sparse coding and the cluster is denoised.

3.2 Dictionnary Initialization

The initial dictionnary is first learned offline on the 10 000 images of the PASCAL VOC'07 database using the online dictionary learning procedure of [2]. This procedure is then used on the noisy picture in order to improve the dictionary efficiency.
In fact, only a fixed number T of patches in the picture are used to update the dictionary. However they are chosen so that they are independently and identically distributed in the picture.
The algorithm corresponds then to the minimization of eq.1 on the T patches with the l_1 norm using the LARS [6] algorithm.

Input : number of iterations $\mathbf{T}$, i.i.d. sampling of $\mathbf{T}$ patches of the noisy picture $\mathbf{Y}_T$, initial dictionary $\mathbf{D}^0 \in \mathbb{R}^{m\times k}$, regularization parameter λ
Output : learned dictionary $\mathbf{D}$
Initialization : $\mathbf{A}^0 \in \mathbb{R}^{k\times k} \leftarrow 0$, $\mathbf{B}^0 \in \mathbb{R}^{m\times k} \leftarrow 0$
for *t = 1..**T*** **do**

$\mathbf{y}_t = \mathbf{Y}_T[\mathrm{t}]$
Sparse coding: compute with LARS algorithm:

$$\boldsymbol{\alpha}^t = \mathrm{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^k} ||\boldsymbol{\alpha}||_1 \text{ s.t. } ||\mathbf{y} - \mathbf{D}^{t-1}\boldsymbol{\alpha}||_2^2 \leq \lambda$$

$\mathbf{A}^t \leftarrow \mathbf{A}^{t-1} + \boldsymbol{\alpha}^t\boldsymbol{\alpha}^{tT}$
$\mathbf{B}^t \leftarrow \mathbf{B}^{t-1} + \mathbf{y}^t\boldsymbol{\alpha}^{tT}$
Update dictionary from $\mathbf{D}^{t-1}$ to $\mathbf{D}^t$ so that:

$$\mathbf{D}^t = \mathrm{argmin}_{\mathbf{D}\in\mathbb{R}^{m\times k}} \frac{1}{t}\sum_{i=1}^{t}(\frac{1}{2}|\mathbf{y}^i - \mathbf{D}\boldsymbol{\alpha}^i||_2^2 + \lambda||\boldsymbol{\alpha}^i||_1)$$

end
return $\mathbf{D}^T$

Algorithm 1: Online Dictionary Learning

Input :input dictionary $\mathbf{D}= [\mathbf{d}^1, \dots , \mathbf{d}^k] \in \mathbb{R}^{m\times k}$,
$\mathbf{A} = [\mathbf{a}^1, \dots , \mathbf{a}^k] \in \mathbb{R}^{k\times k}$, $\mathbf{B} = [\mathbf{b}^1, \dots , \mathbf{b}^k] \in \mathbb{R}^{m\times k}$
Output : updated dictionary $\mathbf{D}$
repeat

for *j = 1..k* **do**

update the j^{th} column:
if ***A****(j,j) = 0* **then**

$$\mathbf{d}^j \leftarrow 0$$

end
else

$$\mathbf{u}^j \leftarrow \frac{1}{\mathbf{A}(j,j)}(\mathbf{b}^j - \mathbf{D}\mathbf{a}^j) + \mathbf{d}^j$$

$$\mathbf{d}^j \leftarrow \frac{1}{\max(||\mathbf{u}^j||_2, 1)}\mathbf{u}^j$$

end

end

until *convergence* ;
return $\mathbf{D}$

Algorithm 2: Dictionary Update

Input : Gram matrix of the dictionary G = $\mathbf{D}^T\mathbf{D} \in \mathbb{R}^{k\times k}$ where $\mathbf{D} = [\mathbf{d}^1, \dots , \mathbf{d}^k] \in \mathbb{R}^{m\times k}$, noisy patch $\mathbf{y}$, constraint λ

Output : sparse vector $\boldsymbol{\alpha} \in \mathbb{R}^k$

—INITIALIZATION—

normX $\in \mathbb{R} \leftarrow ||\mathbf{y}||_2^2$

coeffs $\in \mathbb{R}^k \leftarrow \mathbf{0}$

Most correlated element

$\hat{\mathbf{c}} \in \mathbb{R}^k \leftarrow \mathbf{D}^T\mathbf{y}$

$C \in \mathbb{R}^+ \leftarrow \max_{j=1..m}(|\hat{\mathbf{c}}_j|)$

currentInd $\leftarrow j$ s.t. $\hat{\mathbf{c}}_j = C$

```
if normX > λ then
    return 0
end
newAtom ← True
for i = 1..k do
    —NEW ATOM—
    if newAtom then
        A[i] ← currentInd
        G_A[i^th line] ← G[A[i]^th line]
        G_S ← D_A^T D_A
        UPDATE G_S^{-1}
    end
    —VARIABLES UPDATES—
    u ← G_S^{-1}(sgn(ĉ_j))_{j∈A}
    ratio ← (−coeffs[j]/u_j)_{j∈[1;i]}
    stepMAX ← min⁺(ratio) (min⁺ means its the minimum between positive values)
    criticalInd ← j s.t. ratio[j] = stepMAX
    C ← ĉ[0]
    γ ← min⁺((C±ĉ_j)/(1±(G_A u)[j]))_{j∉A}
    —POLYNOMIAL RESOLUTION—
    a ← Σ_{j∈A} sgn(ĉ[A[j]])u[j]
    b ← Σ_{j∈A} ĉ[A[j]]u[j]
    c ← normX - λ
    Δ ← b² − ac
    stepMAX2 ← min((b−√Δ)/a, C)
    —FINAL STEP & BREAK—
    γ ← min(γ, stepMAX, stepMAX2)
    coeffs ← coeffs + γu
    ĉ ← ĉ − γ G_A u
    normX ← normX + aγ² − 2bγ
    if |γ| < 1e^{−15} || γ = stepMAX2 || normX < 1e^{−15} || normX − λ < 1e^{−15} then
        break
    end
    if γ = stepMAX then
        DOWNDATE G_S^{-1}
        newAtom ← False
        i ← i-2
    end
    else
        newAtom ← True
    end
end
```

return $\boldsymbol{\alpha} = sort(\text{coeffs}, \mathcal{A})$ (sorted coefficient with respect to the increasing order of $\mathcal{A}$ indexes and filled with 0 if necessary)

Algorithm 3: LARS algorithm - Mairal Version

Input : Gram matrix $G_S \in \mathbb{R}^{i \times i}$, and its former inverse to update $G_S^{-1} \in \mathbb{R}^{i-1 \times i-1}$
Output : updated $G_S^{-1} \in \mathbb{R}^{i \times i}$
if $i = 1$ **then**
    return $\frac{1}{G_S}$
end
$u \leftarrow G_S^{-1} G_S[i^{th} \text{ line}]$
$\sigma \leftarrow \frac{1}{G_s(i,i) - u.G_S[i^{th} \text{ line}]}$
$G_s^{-1}(i,i) \leftarrow \sigma$
$G_s^{-1}[i^{th} \text{ line}] \leftarrow -\sigma u$
return $G_s^{-1} \leftarrow G_s^{-1} + \sigma u u^T$

Algorithm 4: Update invert algorithm

Input : pseudo-Gram matrix $G_A \in \mathbb{R}^{i \times i}$ Gram matrix $G_S \in \mathbb{R}^{i \times i}$, and its inverse $G_S^{-1} \in \mathbb{R}^{i \times i}$, active indexes list $\mathcal{A} \in \mathbb{R}^i$, sparse coefficient list coeffs $\in \mathbb{R}^k$, criticalInd $\in [1; k]$, current iteration i
Output : downdated matrices $G_S, G_S^{-1}, G_A \in \mathbb{R}^{i-1 \times i-1}$, downdated lists $\mathcal{A} \in \mathbb{R}^{i-1}$, coeffs $\in \mathbb{R}^k$
$\sigma \leftarrow \frac{1}{G_S^{-1}(\text{criticalInd}, \text{criticalInd})}$
$u \leftarrow G_S^{-1}[\text{criticalInd}^{th} \text{ line}]$ without its criticalIndth coefficient
for *j= criticalInd:i* **do**
    $G_A[j^{th} \text{ line}] \leftarrow G_A[(j+1)^{th} \text{ line}]$
    for *k= 1:criticalInd-1* **do**
        $G_S(j,k) \leftarrow G_S(j+1,k)$
        $G_S^{-1}(j,k) \leftarrow G_S^{-1}(j+1,k)$
    end
    for *k= criticalInd:i* **do**
        $G_S(j,k) \leftarrow G_S(j+1,k+1)$
        $G_S^{-1}(j,k) \leftarrow G_S^{-1}(j+1,k+1)$
    end
    $\mathcal{A}[j] \leftarrow \mathcal{A}[j+1]$
    coeffs[j] ← coeffs[j+1]
end
coeffs[i] ← 0
$G_s^{-1} \leftarrow G_s^{-1} - \sigma u u^T$

Algorithm 5: Downdate invert algorithm

Glossary

Image Credits

© IPOL (there's no need to credit this image, here is used as an example.)

4 References

References

[1] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani *Least angle regression*. Ann. Statist., 32(2):407499, 2004.

[2] J.Mairal, F. Bach, J. Ponce, and G. Sapiro *Online dictionary learning for sparse coding.* ICML, 2009.

[3] J. Mairal, F. Bach, J. Ponce, G. Sapiro and A. Zisserman *Non-local Sparse Models for Image Restoration.* International Conference on Computer Vision, 2009.

[4] J. Mairal *Reprsentations parcimonieuses en apprentissage statistique, traitement dimage et vision par ordinateur.* PhD thesis, 2010.

[5] S. Mallat and Z. Zhang *Matching pursuit in a timefrequency dictionary.* IEEE T. SP, 41(12):33973415, 1993.

[6] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani *Least angle regression.* Ann. Statist., 32(2):407499, 2004.

[7] M. Elad and M. Aharon *Image denoising via sparse and redundant representations over learned dictionaries.* IEEE T. IP, 54(12):37363745, 2006.