Neurocomputing 157 (2015) 61–69

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Online dictionary learning for Local Coordinate Coding with Locality Coding Adaptors

Junbiao Pang [a], Chunjie Zhang [b], Lei Qin [c], Weigang Zhang [d], Laiyun Qing [b], Qingming Huang [a,b,c,*], Baocai Yin [a,**]

[a] Beijing Key Laboratory of Multimedia and Intelligent Software Technology, College of Metropolitan Transportation, Beijing University of Technology, No. 100 Pingleyuan Road, Chaoyang District 100124, China
[b] School of Computer and Control Engineering, University of Chinese Academy of Sciences, No. 19 Yuquan Road, Shijingshan District, Beijing 100049, China
[c] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, No. 6 Kexueyuan South Road, Haidian District, Beijing 100190, China
[d] School of Computer Science and Technology, Harbin Institute of Technology at Weihai, No. 2 West Wenhua Road, Weihai 26209, China

ARTICLE INFO

Article history:
Received 20 January 2014
Received in revised form
31 October 2014
Accepted 17 January 2015
Communicated by Steven Hoi
Available online 2 February 2015

Keywords:
Local Coordinate Coding
Surrogate function
Locality Coding Adaptor
Large scale problem
Online training

ABSTRACT

Dictionary in Local Coordinate Coding (LCC) is important to approximate a non-linear function with linear ones. Optimizing dictionary from predefined coding schemes is a challenge task. This paper focuses on learning dictionary from two Locality Coding Adaptors (LCAs), i.e., locality Gaussian Adaptor (GA) and locality Euclidean Adaptor (EA), for large-scale and high-dimension datasets. Online dictionary learning is formulated as two cycling steps, local coding and dictionary updating. Both stages scale up gracefully to large-scale datasets with millions of data. The experiments on different applications demonstrate that our method leads to a faster dictionary learning than the classical ones or the state-of-the-art methods.

1. Introduction

Local Coordinate Coding (LCC) [1] is a general framework that uses linear functions to approximate any non-linear Lipschitz smooth one. LCC generally consists of two key components: (1) the coding schemes that define the local coordinates [2]; and (2) a dictionary (data points) which consists of the local coordinates. LCC has been successfully applied to many challenging problems, e.g., approximating non-linear kernels [3], feature learning in multi-class classification [4].

The problem to learn dictionary for LCC, especially for high-dimension visual data, is that time complexity grows quadratically with both the dictionary size and the dimension of data. Because sparse coding [5,6] is usually used to lean dictionary [1]. For a large-scale dataset with millions of samples, the time cost of this sparse coding-based approach [1] becomes unacceptable. For instance, on a single-core 2.6 GHz machine, sparse coding takes about a week to lean 1000 items of a dictionary from about one million samples via feature-sign search [7].

To avoid the sparse coding during learning dictionary for LCC, Locality Coding Adaptors (LCAs) [4,2] are proposed to replace the locality error in LCC. The dictionary size in real applications seemingly increases explosively for high-dimension data, as items in a dictionary should be "local enough" to encode a sample. For LCAs, however, one of the recent results [2] discovers that both locality Gaussian Adaptor (GA) [4] and locality Euclidean Adaptor (EA) [2,8] have no relation with the dimension of data. Therefore, the motivation behind this paper is to fast and accurately learn dictionary for LCC with LCAs.

The key notation of our solution is that dictionary can be fast computed with both surrogate function [9] and warm restart technique [10]. Rather than adopting Stochastic Gradient Descent (SGD) (which requires to tune a learning speed), we instead use the surrogate method [9,11] which aggregates the past information computed during the previous steps with warm restart. The advantage

* Corresponding author at: Beijing Key Laboratory of Multimedia and Intelligent Software Technology, College of Metropolitan Transportation, Beijing University of Technology, No. 100 Pingleyuan Road, Chaoyang District 100124, China.
** Corresponding author.
E-mail addresses: junbiao_pang@bjut.edu.cn (J. Pang), cjzhang@jdl.ac.cn (C. Zhang), lqin@jdl.ac.cn (L. Qin), wgzhang@jdl.ac.cn (W. Zhang), lyqing@ucas.ac.cn (L. Qing), qmhuang@jdl.ac.cn (Q. Huang), ybc@bjut.edu.cn (B. Yin).

http://dx.doi.org/10.1016/j.neucom.2015.01.035
0925-2312/

of warm restart is that a good initialization is supplied when dictionary is learned with the block-wise coordinate descent [12]. This learning scheme is not only significantly faster than the batch alternatives, but also hopes to avoid tuning hyper-parameters in SGD, e.g., learning speed.

The core contributions of this paper can be summarized as follows: technically, we introduce an online dictionary learning method for LCC with LCAs. Our learning approach achieves approximate 100 times faster than the batch one [8] on large-scale datasets. Besides, the theoretical justification on the convergence of the proposed algorithm is presented.

In the next section, related work is briefly summarized. Section 3 introduces problem of learning dictionary for LLC with LCAs. Section 4 first outlines the dictionary learning algorithm and details two cycling steps. After that are the experiment and conclusion sections.

2. Related work

The seemingly most similar work to LCC may be dictionary learning in sparse coding [13,14]: adding different constraints into the reconstruction losses. However, the goal of sparse coding is to represent a signal approximately as the globally linear combination of a small number of the overcomplete dictionary. Given data $\mathbf{x}_i$ ($\mathbf{x}_i \in \mathbb{R}^D$) and dictionary $\mathbf{V}=[\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_M]$ ($\mathbf{V} \in \mathbb{R}^{D\times M}$), sparse coding seeks a linear reconstruction of the given data $\mathbf{x}_i$ as $\mathbf{x}_i = \gamma_{i1}\mathbf{v}_1 + \gamma_{i2}\mathbf{v}_2 + \cdots + \gamma_{iM}\mathbf{v}_M$. The reconstruction coefficients $\gamma_i = [\gamma_{i1}, \ldots, \gamma_{iM}]^T$ ($\gamma_i \in \mathbb{R}^M$) are sparsity, requiring only a small fraction of entries in γ_i are nonzeros. Denoting $\|\gamma\|_0$ as the number of nonzero entries of the vector γ, sparse coding can be formulated as follows:

$$\begin{aligned} \min_{\gamma_i} \quad & \|\gamma_i\|_0 \\ \text{s.t.}: \quad & \mathbf{x}_i = \mathbf{V}\gamma_i. \end{aligned} \tag{1}$$

However, the minimization of ℓ_0 norm is an NP-hard problem. Recent research usually formulates the sparse coding problem as the minimization of ℓ_1 norm of the reconstruction coefficients. The objective of sparse coding can be reformulated as follows [7,15]:

$$\min_{\gamma_i, \mathbf{V}} \|\mathbf{x}_i - \mathbf{V}\gamma_i\|^2 + \lambda\|\gamma_i\|_1, \tag{2}$$

The first term in (2) is the reconstruction loss, and the second term is used to control the sparsity. λ is the tradeoff parameter used to balance the sparsity and the reconstruction error.

While the locality of LCC tends to bring sparsity into local coding, as only the items in a dictionary closing to the test input would be given more weights. The objective of LCC is formulated as follows [1]:

$$\left| f(\mathbf{x}_i) - \sum_m \gamma_{im} f(\mathbf{v}_m) \right| \le \alpha \left\|\mathbf{x}_i - \mathbf{V}\gamma_i\right\|^2 + \beta \sum_m |\gamma_{im}| \left\|\mathbf{v}_m - \mathbf{V}\gamma_i\right\|^{1+p}, \tag{3}$$

where a nonlinear function $f(\mathbf{x}_i)$ is approximated by a set of linear ones $f(\mathbf{v}_m)$, $\mathbf{V}=[\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_M]$ ($\mathbf{V} \in \mathbb{R}^{D\times M}$) is the dictionary, γ_{im} are the local coding of data $\mathbf{x}_i$ based on the point $\mathbf{v}_m$, and α and β are the tradeoff factors to balance between the reconstruction error $\left\|\mathbf{x}_i - \mathbf{V}\gamma_i\right\|^2$ and the locality one $|\gamma_i| \left\|\mathbf{v}_m - \mathbf{V}\gamma_i\right\|^{1+p}$. Eq. (3) indicates that LCC locally encodes each sample to obtain the non-linear approximation ability. In contrast, the dictionary in sparse coding (2) does not favor this choice. Therefore, the motivation between sparse coding and LCC is totally different.

Learning dictionary for LCC in (3) has to face the non-smooth optimization $|\gamma_i| \left\|\mathbf{v}_m - \mathbf{V}\gamma_i\right\|^{1+p}$ and the choice of hyper-parameter p. To avoid these difficulties, LCAs are proposed to replace the locality error as follows:

$$\left| f(\mathbf{x}_i) - \sum_m \gamma_{im} f(\mathbf{v}_m) \right| \le \alpha \left\|\mathbf{x}_i - \mathbf{V}\gamma_i\right\|^2 + \beta \|\mathbf{p}_i \odot \gamma_i\|^2, \tag{4}$$

where the operation $\odot$ represents the element-wise multiplication, and $\mathbf{p}_i$ ($\mathbf{p}_i \in \mathbb{R}^M$) are LCAs. The second term in (4) enforces local coding γ_{im} to have a similar locality of LCAs [2]. Therefore, instead of learning dictionary for LCC, learning dictionary for LCC with LCAs has several advantages: (1) the smooth objective function in LCC with LCAs avoids the non-smooth optimization in LCC; (2) the dictionary size of LCC with LCAs has no relation with the dimension of data [2]. Concretely, LCC with LCAs avoids the sparse coding problem in LCC with the complexity $\mathcal{O}(DMs + Ds^2)$, where s is the number of the nonzero coefficients, if a Cholesky-based implementation of LASSO/LARS problem [5] is adopted. Moreover, LCC with LCAs turns LCC (3) into convex objective functions when one of the parameters $\{\gamma_i, \mathbf{V}\}$ is fixed (see Section 4.4 for the detailed analysis).

Dictionary learning in LCC with LCAs is in most cases considered as vector quantization (VQ). However, a large part of the classical approaches in VQ barely handle a predefined locality. For example, [3] uses k-means to participate the data space with Euclidean distance (which can be considered as a special case of EA [2]). Other methods define a special locality according to the adopted VQ, e.g., [16] uses LASSO to solve a coding scheme with inverse Euclidean distance. These, however, lose flexibility to optimize dictionary for different LCAs.

Dictionary learning in LCC with LCAs alternates between two steps: local coding and dictionary updating. The local coding is sequentially learned for every sample, only requiring a limited computational cost. Dictionary updating by a batch training algorithm [8] has to process all samples in each iteration. Recently, [4] relaxes the objective function by ignoring LCAs, and learns dictionary by minimizing the reconstruction loss. However, the relaxed the objective function makes the learned dictionary obtain a suboptimal performance.

Inspired by the success of warm restart [10] in online sparse coding [11], our proposed method also applies this technique to update dictionary. Dictionary updating in online sparse coding minimizes the convex reconstruction loss in (2), while our approach optimizes both the reconstruction loss and the locality loss (4). On the other side, recently there has been a trend of introducing surrogate function into different tasks, and thus the optimization problem is viewed as finding a more approximate yet simple objective function [17,18,11]. To the best of our knowledge, this paper is first to apply the surrogated-based method to dictionary learning in LCC with LCAs.

3. Problem formulation

3.1. Dictionary learning with Locality Coding Adaptors

LCC is formulated as a constrained reconstruction problem, as the quality of the non-linear approximation ability is bounded by both the reconstruction and the locality (3). For a matrix $\mathbf{X}=[\mathbf{x}_1, \ldots, \mathbf{x}_N]^T$ with N data, the dictionary matrix $\mathbf{V}=[\mathbf{v}_1, \ldots, \mathbf{v}_M]$, LCC with LCAs can be formulated as the following problem [4]:

$$L(\mathbf{X}, \mathbf{V}) = \min_{\gamma_i, \mathbf{V}} \frac{1}{2} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{V}\gamma_i\|^2 + \lambda \|\mathbf{p}_i \odot \gamma_i\|^2 \tag{5}$$

$$\text{s.t.} \quad \gamma_i^T \mathbf{1} = 1, i = 1, \ldots, N, \tag{6}$$

where the vector $\mathbf{1}$ denotes the identity vector $[1, \ldots, 1]^T$, and the operation $\odot$ represents the element-wise multiplication, and $\mathbf{p}_i = [p_{i1}, p_{i2}, \ldots, p_{iM}]^T$ ($\mathbf{p}_i \in \mathbb{R}^M$) are LCAs. p_{im} can be either GA [4] or EA [8,2]:

1. *Gaussian adaptor* (*GA*) presumes the relation among samples and dictionary as

$$p_{im} = \exp\left(\frac{\|\mathbf{v}_m - \mathbf{x}_i\|^2}{\sigma^2}\right) \tag{7}$$

where the hyper-parameter σ controls the weight decay ability for the locality adaptor.

2. *Euclidean adaptor* (*EA*) uses an inverse Student *t*-distribution with one degree of freedom

$$p_{im} = \sigma^2 + \|\mathbf{v}_m - \mathbf{x}_i\|^2, \tag{8}$$

where σ is also used for adjusting the weight decay speed for the locality adaptor.

Both GA (7) and EA (8) enforce different locality decay schemes between the sample $\mathbf{x}_i$ and the point $\mathbf{v}_m$ [2]. The theoretical results [2] indicate that GA tends to work well than EA, but requires more number of dictionary than EA; besides, the empirical results also validate that GA works well than EA adaptor, if the dictionary size is sufficiently large [8]. The constraint (6) makes dictionary shift invariant. Eq. (5) is convex over the dictionary $\mathbf{V}$ with the fixed local coding γ_i and vice versa. Thus the coordinate descent method can be used to alternatively optimize the local coding γ_i and the dictionary $\mathbf{V}$.

4. Online dictionary learning

4.1. Algorithm outline

Algorithm 1. Online dictionary learning.

require: $\mathbf{x} \in \mathbb{R}^D$ (random variable and an algorithm to draw i.i.d samples), $\lambda \in \mathbb{R}$ (regularization parameter), $\mathbf{V}^0 \in \mathbb{R}^{D\times M}$(initial dictionary), T (number of iterations);
initialization: $\mathbf{A}^0 \leftarrow 0$, $\mathbf{B}^0 \leftarrow 0$(reset history information);
for $t=1$ to T **do**
  Randomly draw $\mathbf{x}_i$ from $\mathbf{X}$;
  Local coding : Compute the local coding γ_i for sample $\mathbf{x}_i$ by (12) : $\gamma_i = \arg\min_{\gamma} \frac{1}{2}\|\mathbf{x}_i - \mathbf{V}^{t-1}\boldsymbol{\gamma}\|^2 + \lambda\|\mathbf{p}_i \odot \gamma\|^2$.
  $\mathbf{A}^t \leftarrow \mathbf{A}^{t-1} + \gamma_i\gamma_i^T + \frac{4\lambda}{\sigma^2}\Lambda_i$; $\mathbf{B}^t \leftarrow \mathbf{B}^{t-1} + \mathbf{x}_i\gamma_i^T + \frac{4\lambda}{\sigma^2}\mathbf{x}_i\Sigma_i$;
  Dictionary updating :
  Updates dictionary using Algorithm 2, such that

$$\mathbf{V}^t = \min_{\mathbf{V}} \sum_{i=1}^{t} \frac{1}{2}\|\mathbf{x}_i - \mathbf{V}\gamma_i\|^2 + \lambda\|\mathbf{p}_i \odot \gamma_i\|^2. \tag{9}$$

end
Return dictionary $\mathbf{V}^T$ (dictionary learned after T iterations.);

Algorithm 2. Dictionary updating procedure.

input: $\mathbf{V}^t = [\mathbf{v}_1^t, \ldots, \mathbf{v}_M^t] \in \mathbb{R}^{D\times M}$ (dictionary learned at t iteration),
$\mathbf{A}^t = [\mathbf{a}_1^t, \ldots, \mathbf{a}_M^t] \in \mathbb{R}^{M\times M}$ (past information at t iteration),
$\mathbf{B}^t = [\mathbf{b}_1^t, \ldots, \mathbf{b}_M^t] \in \mathbb{R}^{D\times M}$ (past information at t iteration);
while *not convergence* **do**
  for $m=1$ to M **do**
    Update the m−th point to optimize for (9) :

$$\mathbf{v}_m^{t+1} \leftarrow \mathbf{v}_m^t - \frac{1}{a_{mm}}\left(\mathbf{V}^t\mathbf{a}_m^t - \mathbf{b}_m^t\right),$$

$$\mathbf{v}_m^{t+1} \leftarrow \frac{1}{\max(\|\mathbf{v}_m^{t+1}\|^2, 1)}\mathbf{v}_m^{t+1}.$$

  end
end
Return dictionary $\mathbf{V}_t$(dictionary learned at the t-th iteration.);

Our optimization procedure is summarized in Algorithm 1. Assuming the training set composed of i.i.d. samples, our algorithm draws one data $\mathbf{x}_i$ at a time, and alternatively optimizes between the local coding γ_i and the dictionary $\mathbf{V}^{t-1}$ at the $t-1$−th iteration. After the new local coding γ_i is computed, dictionary is learned by minimizing the following surrogate function:

$$\tilde{L}_t(\mathbf{V}^t) = \sum_{i=1}^{t} \frac{1}{2}\|\mathbf{x}_i - \mathbf{V}^t\gamma_i\|^2 + \lambda\|\mathbf{p}_i \odot \gamma_i\|^2. \tag{10}$$

The motivation behind our approach is that the past information computed during the previous steps upperbounds the empirical cost (5).

4.2. Local coding

The local coding is a constrained linear least-square problem when dictionary is fixed. To determine the optimal local coding γ_i, the constrained problem can be solved with the Lagrangian function $L(\gamma_i, \nu)$:

$$\begin{aligned} L(\gamma_i, \nu) &= \tfrac{1}{2}\|\mathbf{x}_i - \mathbf{V}\boldsymbol{\gamma}_i\|^2 + \lambda\|\mathbf{p}_i \odot \gamma_i\|^2 + \nu(1 - \gamma_i^T\mathbf{1}) \\ &= \tfrac{1}{2}\gamma_i^T\Phi\gamma_i + \lambda\gamma_i^T\mathrm{diag}(\mathbf{p}_i^2)\gamma_i + \nu(1 - \gamma_i^T\mathbf{1}) \end{aligned} \tag{11}$$

where the matrix Φ is $(\mathbf{x}_i\mathbf{1}^T - \mathbf{V})^T(\mathbf{x}_i\mathbf{1}^T - \mathbf{V})$, ν is the Lagrangian multiplier, and the operation diag(A) reshapes the vector A into the diagonal matrix.

Let $\partial L(\gamma_i, \nu)/\partial\gamma_i = 0$, the optimal local coding γ_i satisfies that

$$\begin{aligned} \tilde{\gamma}_i &= \left(\Phi + 2\lambda\,\mathrm{diag}(\mathbf{p}_i^2)\right)^{-1}\mathbf{1}, \\ \gamma_i &= \tilde{\gamma}_i/(\tilde{\gamma}_i^T\mathbf{1}). \end{aligned} \tag{12}$$

It should be noted that the matrix Φ is symmetric and semi-positive. If the matrix Φ is singular or nearly singular, the matrix $\Phi + 2\lambda\,\mathrm{diag}(\mathbf{p}_i^2)$ is still conditioned, as $2\lambda\,\mathrm{diag}(\mathbf{p}_i^2)$ penalizes large distance that exploits correlation beyond some level of precision between data points.

4.3. Dictionary updating

Dictionary updating uses the block-coordinate descent with warm restart, and one of its main advantages is that it does not require any learning rate tuning. Warm restart therefore is especially suitable for the block-coordinate descent. If GA is adopted,[1] the gradient of the surrogate function $\tilde{L}_t(\mathbf{V}^t)$ with respect to the m-th point $\mathbf{v}_m^t$:

$$\frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{v}_m^t} = \sum_{i=1}^{t}\left(-\gamma_{im}(\mathbf{x}_i - \mathbf{V}^t\gamma_i) + 2\lambda\sum_{m=1}^{M}\frac{\partial p_{im}}{\partial\mathbf{v}_m^t}\gamma_{im}^2 p_{im}\right) \tag{13}$$

where the $\partial p_{im}/\partial\mathbf{v}_m^t = 2p_{im}(\mathbf{v}_m^t - \mathbf{x}_i)/\sigma^2$, if GA (7) is used in p_{im}. Substituting this partial derivative into (13) gives the partial derivative of $\mathbf{v}_m^t$:

$$\frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{v}_m^t} = \sum_{i=1}^{t}\left(-\gamma_{im}(\mathbf{x}_i - \mathbf{V}^t\gamma_i) + 4\lambda\sum_{m=1}^{M}\frac{p_{im}^2\gamma_{im}^2(\mathbf{v}_m^t - \mathbf{x}_i)}{\sigma^2}\right). \tag{14}$$

According to the derivative of a scalar function with respect to a matrix of independent variable [19], the partial derivative $\partial\tilde{L}(\mathbf{V}^t)/\partial\mathbf{V}^t$ is computed as

$$\begin{aligned} \frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{V}^t} &= \left[\frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{v}_1^t}, \ldots, \frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{v}_m^t}, \ldots, \frac{\partial\tilde{L}_t(\mathbf{V}^t)}{\partial\mathbf{v}_M^t}\right] \\ &= \mathbf{V}^t\sum_{i=1}^{t}\left(\gamma_i\gamma_i^T + \frac{4\lambda}{\sigma^2}\Lambda_i\right) - \sum_{i=1}^{t}\left(\mathbf{x}_i\gamma_i^T + \frac{4\lambda}{\sigma^2}\mathbf{x}_i\Sigma_i\right), \end{aligned} \tag{15}$$

[1] The solution for EA is presented in Appendix.

where the matrix Λ_i is the diagonal matrix with $(p_{im}\gamma_{im})^2$ as the elements, and the Σ_i is the vector $\left[(p_{i1}\gamma_{i1})^2,\ldots,(p_{iM}\gamma_{iM})^2\right]$.

The past information in (15) is further stored into two matrices, $\mathbf{A}^t=\sum_{i=1}^{t}\gamma_i\gamma_i^T+(4\lambda/\sigma^2)\Lambda_i$ ($\mathbf{A}^t\in\mathbb{R}^{M\times M}$), and $\mathbf{B}^t=\sum_{i=1}^{t}\mathbf{x}_i\gamma_i^T+(4\lambda/\sigma^2)\mathbf{x}_i\Sigma_i$ ($\mathbf{B}^t\in\mathbb{R}^{D\times M}$). It should be noted that two $M\times M$ and $D\times M$ size matrices are sufficient to store all the past information during iterations, due to the following equations:

$$\mathbf{A}^{t+1}=\mathbf{A}^t+\gamma_i\gamma_i^T+\frac{4\lambda}{\sigma^2}\Lambda_i, \tag{16}$$

$$\mathbf{B}^{t+1}=\mathbf{B}^t+\mathbf{x}_i\gamma_i^T+\frac{4\lambda}{\sigma^2}\mathbf{x}_i\Sigma_i. \tag{17}$$

The limited storage requirement makes Algorithm 1 efficiently deal with large-scale and high-dimension datasets.

Let $\mathbf{a}_m^t$ and $\mathbf{b}_m^t$ denote the m-th columns of matrices $\mathbf{A}^t$ and $\mathbf{B}^t$ individually, let a_{mm}^t denote the (m,m)-th element of $\mathbf{A}^t$, the m-th $\mathbf{v}_m^t$ is updated as

$$\mathbf{v}_m^{t+1}\leftarrow\mathbf{v}_m^t-\frac{1}{a_{mm}}\left(\mathbf{V}^t\mathbf{a}_m^t-\mathbf{b}_m^t\right), \tag{18}$$

where $\mathbf{V}^t$ is the dictionary at the t-th iteration. To make dictionary bounded, the point $\mathbf{v}_m^{t+1}$ is projected onto unit ball:

$$\mathbf{v}_m^{t+1}\leftarrow\frac{1}{\max(\|\mathbf{v}_m^{t+1}\|^2,1)}\mathbf{v}_m^{t+1} \tag{19}$$

4.4. Convergence analysis

The non-convexity objective function (5) and the stochastic approximation assumption make the proof of the convergence of the proposed algorithm to a stationary point somewhat involved. In the following, the convergence of the surrogated objective function (10) with warm restart is proved.

Proposition 1. *If γ_i are fixed, the surrogate function $\tilde{L}_t(\mathbf{V}^t)$ in (10) are strictly convex with lower-bounded Hessians.*

Proof. The function $\tilde{L}_t(\mathbf{V}^t)$ is twice continuously differentiable, and the second-order derivative of $\tilde{L}_t(\mathbf{V}^t)$ with dictionary $\mathbf{V}^t$ is

$$\frac{\partial^2\tilde{L}_t(\mathbf{V}^t)}{\partial{\mathbf{V}^t}^2}=\frac{1}{t}\mathbf{A}^t.$$

If every point $\mathbf{x}_i$ at least has a nonzero local coding γ_i, the matrix $\mathbf{A}^t=\sum_{i=1}^{t}\gamma_i\gamma_i^T+(4\lambda/\sigma^2)\Lambda_i$ for GA ($\mathbf{A}^t=\sum_{i=1}^{t}\gamma_i\gamma_i^T+4\lambda\Lambda_i$ for EA) is the positive and semi-definite matrix. This condition also equals to require that the smallest eigenvalue of the positive semi-definite matrix $(1/t)\mathbf{A}^t$ is greater than or equal to some value $k_1>0$. Consequently, $\tilde{L}(\mathbf{V}^t)$ are strictly convex with lower-bounded Hessians $(1/t)\mathbf{A}^t$. □

Proposition 2. *The quadratic objective function $L(\gamma_i,v)$ (11) is strictly convex and has lower-bounded Hessians.*

Proof. The objective function $L(\gamma_i,v)$ is twice continuously differentiable, and the second-order derivative of $L(\gamma_i,v)$ with local coding γ_i is

$$\frac{\partial^2L(\gamma_i,v)}{\partial\gamma_i^2}=\frac{1}{2}\Phi+\lambda\,\mathrm{diag}(\mathbf{p}_i^2).$$

If the $\mathbf{p}_i$ is not equal to zero, $2\lambda\,\mathrm{diag}(\mathbf{p}_i^2)$ always makes the $\frac{1}{2}\Phi+\lambda\,\mathrm{diag}(\mathbf{p}_i^2)$ conditioned. Therefore, the Hessian matrices $\frac{1}{2}\Phi+\lambda\,\mathrm{diag}(\mathbf{p}_i^2)$ are always positive semi-definite. This condition is also equal to require that the minimum eigenvalue of Hessian matrix be at least larger than a positive value $k_2>0$. □

Given Proportions 1 and 2, we justify that our algorithm converges to a stationary point of the objective function by proving that $L(\mathbf{V}^t)-\tilde{L}(\mathbf{V}^t)$ converges almost surely to 0, where $L(\mathbf{V})$ is the expected objective function over samples, $L(\mathbf{V})=\mathrm{E}_{\mathbf{x}_i}[L(\mathbf{X},\mathbf{V})]$.

Proposition 3. *Let $\tilde{L}(\mathbf{V}^t)$ denote the surrogate function, then,*

1. *$\tilde{L}(\mathbf{V}^t)$ converges almost surely;*
2. *$L(\mathbf{V}^t)-\tilde{L}(\mathbf{V}^t)$ converges almost surely to 0;*
3. *$L(\mathbf{V}^t)$ converges almost surely.*

Proof. Most part of this proof is very similar to Proposition 3 in [20]. Following Proposition 3 in [20] and applying Propositions 1 and 2, we can prove all claims. Here we only present the proof sketch.

First, the positive sequence $u_t=\tilde{L}_t(\mathbf{V}^t)\geq 0$ is a quasi-matingale by showing that the expectation $E\left[E[u_{t+1}-u_t|P_t]^+\right]$ is upper-bounded, where P_t denotes the past information at iteration t and $[\cdot]^+$ denotes the positive part of a number. During this step, we should apply Proportions 1 and 2 to prove that $\mathbf{V}^{t+1}-\mathbf{V}^t=O(1/t)$ almost surely. Second, we can prove that $\sum_{i=1}^{\infty}((\tilde{L}_t(\mathbf{V}^t)-L_t(\mathbf{V})^t)/(t+1)$ is bounded, and the functions $\tilde{L}_t(\mathbf{V}^t)$ and $L_t(\mathbf{V}^t)$ are also bounded and Lipschitz. After that, we can further obtain that almost surely, $\tilde{L}_t(\mathbf{V}^t)-L_t(\mathbf{V}^t)\overset{t\to\infty}{\to}0$, and $L_t(\mathbf{V}^t)$ converges almost surely. □

4.5. Analysis of time complexity

Table 1 compares the time complexity among locality Sensitive Coding (LSC) (a batch training method) [8], locally linear coding (LLC) (a SGD based method) [4], and our approach. Note that LSC, LLC and our approach all use the same objective function (5).

LLC [4], LSC [8] and our approach all use the block-wise coordinate descent, and divide the optimization procedure into local coding and dictionary updating. LLC uses SGD to optimize the approximated loss, our method adopts the surrogate function with warm restart; LLC and our method thus can scale up to millions of samples. In contrast, LSC in each iteration uses the analytical solution which would make LSC efficient for a small size of datasets.

5. Experiments

In this section, we report results based on several widely used datasets: PASCAL VOC 2007 [21] for image classification, ORL database for face recognition [22] and handwritten digit recognition [23] by locally linear classification [3]. To fairly compare our algorithm to other state-of-the-art methods, we follow the same setting (features, sampling rate, classification methods, etc.) to demonstrate the efficiency of our approach. The dictionary learning algorithm is implemented in C++ and runs on a single-CPU, single-core 2.6 GHz machine.

Table 1
A comparison among optimizing algorithms with our method for dictionary learning.

Method	Local coding	Dictionary updating	Comments
LLC [4]	$\mathcal{O}(K^2+M\log(K))$	$T_{llc}\cdot\mathcal{O}(M\cdot(DM+D))$	In local coding, LLC performs the K-nearest search where $K\ll M$; dictionary updating ignores the locality constraints $\lambda\|\mathbf{p}_i\odot\gamma_i\|^2$ in (5).
LSC [8]	$\mathcal{O}(M^2)$	$T_{lsc}\cdot\mathcal{O}(N(M^2+DM)+M^2+DM)$	The number of iteration T_{lsc} is usually smaller than T_{llc} or T_{our}. Because LSC is a typical batch training method.
Ours	$\mathcal{O}(M^2)$	$T_{our}\cdot\mathcal{O}(M\cdot(M^2+DM+D))$	In dictionary learning, we require that the dictionary size M is smaller than the number of data N, i.e., $M\ll N$.

5.1. Training time comparison among LLC, LSC, SGD and our method

In dictionary updating step, a comparison is done among LLC, LSC, SGD and our method. To generate a large-scale training set, we have densely selected approximately 1 million SIFT features with 16×16, 24×24 and 32×32 size image patches from PASCAL VOC 2007, which consists of 9963 images from 20 classes. We normalize these SIFT descriptors by ℓ_2 norm, and the trade-off parameter $\lambda$is set to 0.5, and the hyper-parameter σ in GA (7) is set to 0.6 in this experiment. The mini-batch scheme with 256 samples has been applied in dictionary updating [11]. To measure and compare the performances with the state-of-the-art methods, the values of the objective functions on the test set as the function of the corresponding training times are plotted.

5.1.1. Online learning and batch training

Fig. 1 compares the batch LSC [8] and our method. As discussed in Table 1, objective function (5) outputted by our method decreases faster than the batch one, and as a result the speed of online learning achieves approximately 100 time faster than the batch method. Also noticed that the training times of the batch algorithm in this experiment are just the results on 10% and 20% samples.

Meanwhile, when different dictionary sizes are learned, both Fig. 1(a) and (b) indicates that the training times of our method increase linearly with respect to the dictionary size. In addition, there are two interesting observations:

1. For large-scale datasets, although the decrease of the objective function at each iteration is larger than that of our method, our online approach costs a few training time than that of the batch approach. It validates the advantage of the expectation loss (10) – one should not spend too much effort on accurately minimizing the empirical loss.
2. Two approaches achieve a similar performance if both algorithms are sufficiently learned. The expectation loss (10) is convex when either local coding or dictionary is fixed, and an unique local minimal solution is guaranteed by Proportions 1 and 2. Therefore, the proposed online algorithm well balances between training speed and accuracy for LCC with LCAs.

5.1.2. Comparison with stochastic gradient descent

Our experiments have shown that SGD should carefully choose the learning speed. To give a fair comparison, we have selected a range of learning speed ρ in the projected SGD,

$$\mathbf{V}^{t+1} \leftarrow P\left[\mathbf{V}^t - \frac{\rho}{t}\frac{\partial L(\mathbf{V})}{\partial \mathbf{V}}\right],$$

where $P[A]$ projects vector A onto an unit ball. Two different sizes of the mini-batch are used in each iteration: 1 example and 256 ones. One example for each iteration is adopted in the classical SGD optimization [24].

Fig. 2 compares our method and SGD with the different learning speed ρ. We observe that the larger the values of speed ρ in SGD are, the smaller the values of the objective functions are after many iterations. If the learning speed in SGD is well tuned, SGD and our method generally have similar speed to reach local minimums. In addition, if different mini-batch sizes are adopted, the optimal learning speed of SGD also changes. In contrast, our method consistently achieves a similar performance without tuning learning speeds.

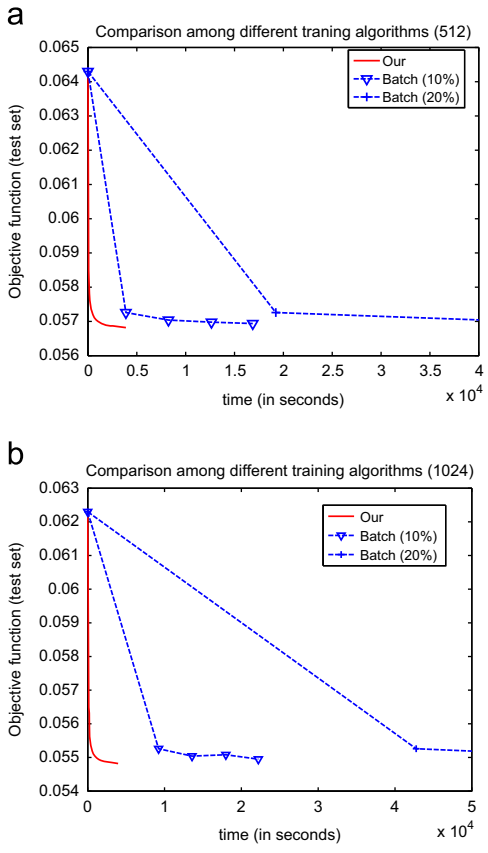


Fig. 1. Comparison between batch training and our method for various dictionary sizes: (a) 512 dictionary and (b) 1024 dictionary.

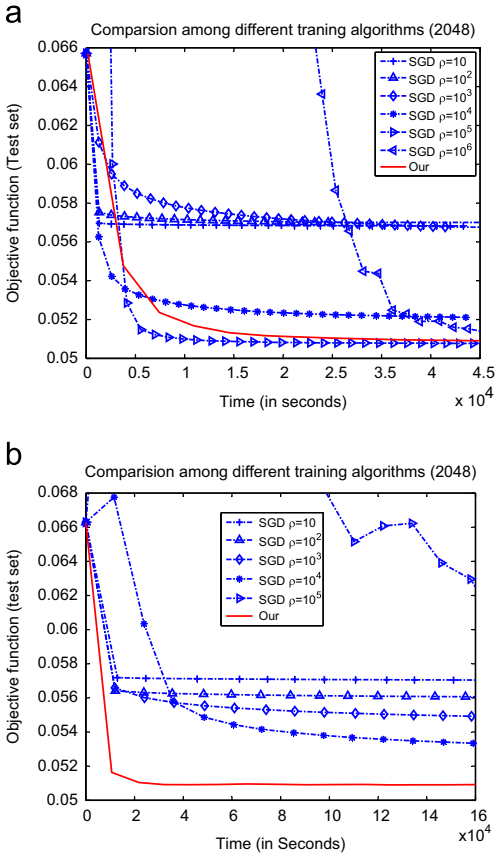


Fig. 2. Comparison between projected SGD and our method with different number of samples in the mini-batch optimization scheme: (a) 1 sample and (b) 256 samples.

5.1.3. Comparison with approximated loss

We compare the proposed dictionary updating in Algorithm 2 with LLC [4] which approximates the loss (5) by ignoring the term $\lambda\|\mathbf{p}_i \odot \gamma_i\|^2$. LLC uses SGD to optimized dictionary. It should be noted that the SGD has to face the selection of the learning speed problem. To give a fair comparison with the approximated loss, the learning rate is selected as $\sqrt{1/t}$ which is also the value assigned in LLC [4].

Fig. 3 shows that although our method has similar training speed to LLC, the objective function outputted by our method is lower than LLC, because two different loss functions are separately used at local coding and dictionary updating steps in LLC. The inconsistent loss functions in LLC make the learned dictionary slightly away from a local minima.

5.2. Application to different tasks

In this subsection, we report results based on three applications of dictionary learning: the reconstruction based classification [8], locality-constrained linear coding for feature learning [4], and locally linear classification [3].

Our intent here is of course *not* to evaluate our learning algorithm in the reconstruction based classification, the feature learning tasks and locally linear classification, which would require thorough comparisons with the state-of-the-art methods on individual task. We here instead wish to demonstrate that it can indeed be comparable to these baselines [8,4] on realistic, non-trivial image classification tasks.

5.2.1. Reconstruction based classification for ORL face recognition

The experiments follow [8] and demonstrate that our online learning technique can be used for a small size datasets. Let $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_J]$, where $\mathbf{V}_j$ is the dictionary learned for class j, a test sample $\mathbf{y} \in \mathbb{R}^D$, the reconstruction-based method classifies $\mathbf{y}$ by its local coding γ which is computed over the entire dictionaries set:

$$\min_{\gamma} \|\mathbf{y} - \mathbf{V}\gamma\|^2 + \lambda\|\mathbf{p} \odot \gamma\|^2.$$

Once the local coding γ is computed, γ is partitioned into $[\gamma^1; \gamma^2; \ldots; \gamma^J]$ where γ^j is the local coding for class j. Then $\mathbf{y}$ is classified as class $\tilde{j}$ by minimizing the class-wise reconstruction error:

$$\tilde{j} = \arg\min_j \|\mathbf{y} - \mathbf{V}_j\gamma^j\|^2.$$

This reconstruction based classification [8] is very similar to the sparse representation-based classification [25].

The ORL database contains 400 face images of size 112×92 pixels from 40 people. The challenges of this dataset include different light conditions, face expression and facial details. To give a fair comparison with other algorithms, we follow the same setting in [8]: randomly and equally split the data into training set and test set, and set the dimension of the eigenface as 100. We perform 10 random trials, and report the average recognition rates of various methods.

It should be noted that our online learning algorithm requires that the size of a dataset should be infinite. To handle the limited size of datasets, the same point is used several times with random replacement. It is very common in online algorithms to simulate an i.i.d. sample by sampling over a randomly permuted datasets: once every sample is already used, we first randomly permute the dataset and stochastically draw the samples again. Fig. 4(a) illustrates the training speed of different methods when the dictionary size per class is 5, and the mini-batch size is 10 samples. Interestingly, the proposed online dictionary learning still outperforms LSC in terms of training speed (see Fig. 4(a)). It clearly indicates that our online approach also works wells on a small size dataset.

The sparse representation based classification (SRC) [25], LLC [4], LSC with GA (LSC-GA) and with ℓ_2 adaptor (LSC-EA) (by assigning σ to 0 in EA (8)) [8] are compared with our method in Fig. 4(b). Following the parameters setting in LSC [8], $\lambda = 0.001$, $\sigma = 0.3$ are used in GA, and $\lambda = 0.1, \sigma = 0$ are used in EA. Our proposed online method with GA achieves the best results for all the dictionary sizes. This may be the random sampling that tends to empirically improve the generalization ability [24]. Although ORL database is relative simple, experiments presented here are used to show the effectiveness of our online learning algorithm on small size datasets.

5.2.2. Locality-constrained linear coding for PASCAL VOC 2007

PASCAL VOC 2007 is an extremely challenging dataset because all the images are daily photos where the size, viewing angle, illumination occlusion, and appearances of objects vary significantly. Spatial Pyramid Matching (SPM) is combined with local

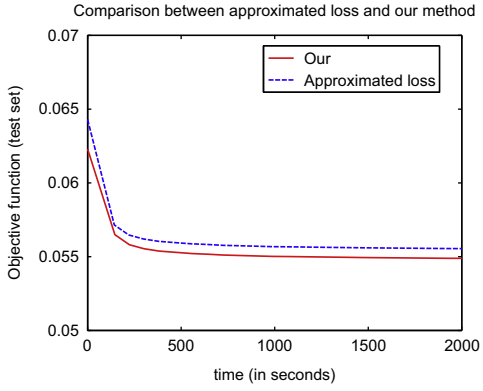


Fig. 3. Comparison between approximated loss and our method.

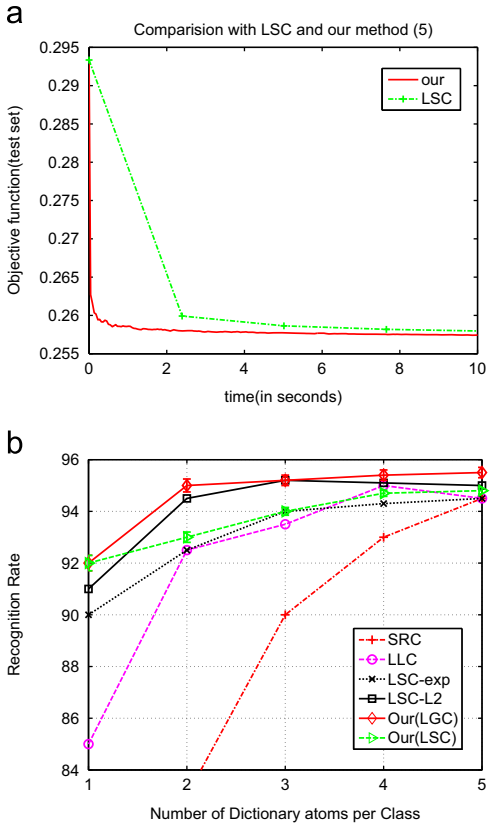


Fig. 4. Recognition performance of different methods and the corresponding training times.

coding to code the discriminative feature [26]. In every SPM layer, for each spatial sub-region, the max pooling is adopted [27,4]:

$$f_i = \max\{|\gamma_{1i}|, |\gamma_{2i}|, \ldots, |\gamma_{Mi}|\},$$

where f_i is the feature coding for the i-th SIFT $\mathbf{x}_i$, γ_{im} is the local coding for the i-th sample. The max pooling is empirically justified by many algorithms in image categorization. The classification performances are evaluated by the Average Precision (AP) measure, and the higher the score is, the better the performance is.

Since a recognition system involves many aspects, such as tuning parameters in classifiers, we have implemented LLC by ourself to repeat the reported results as possibly as we can for a meaningful comparison. Table 2 shows that our method can consistently improve the results of $\text{LLC}_{our.}$, and even obtains the best performance on several object classes (chair, cow and sofa). The improved results indicate that the approximated loss damages the performance of the LLC-based feature.

5.2.3. Locally linear classification for handwritten recognition

Locally linear classification is based on LCC framework. Let $\{\mathbf{x}_i, y_i\}_{i=1}^N$ be a training set, where $\mathbf{x}_i \in \mathbb{R}^D$ denotes the i-th sample, $y_i \in \{+1, -1\}$ denotes the binary label for a given object category, and N is the number of samples. Locally linear support vector machine (LLSVM) [3] combines a set of linear ones $f_m(\mathbf{x})$:

$$F(\mathbf{x}) = \sum_{m=1}^{M} \gamma_m \mathbf{w}_m^T \mathbf{x} + \sum_{m=1}^{M} \gamma_m b_m = \gamma^T \mathbf{W}\mathbf{x} + \gamma^T \mathbf{b},$$

where γ_m is a local coding for the linear classifier $f_m(\mathbf{x}) = \mathbf{w}_m^T \mathbf{x} + b_m$, the transformation $\mathbf{W} \in \mathbb{R}^{M \times D}$ can be considered as a finite kernel transformation which turns a D-dimension problem into a M D-dimension one [3]. If a dictionary produces a lower weighted errors consisting of the reconstruction error and the locality one, LLSVM would have a better non-linear classification ability.

MNIST and USPS, two widely used digit datasets, are evaluated in LLSVM. MNIST contains 40,000 training and 10,000 test 28×28 gray-scale images, which are reshaped directly into the 784 dimension vectors. The label of each image is one of the 10 digits from "0"–"9". USPS [23] consists of 7291 training examples and 2007 gray-scale 16×16 ones for test. Each label corresponds to "0"–"9" digits. During the experiments, the means of raw images are first removed, and then we normalize images with ℓ_2 norm.

Table 3 compares the accuracies among different learning methods with different dictionary size. As analyzed in [2], LCC with GA usually achieves better performances than LLC with EA. Our method with both GA and EA consistently outperforms both the batch one [8] and the approximated one [4]. It means that online learning approach consistently produces the high quality dictionary.

6. Conclusion

In this paper, we present an online dictionary leaning method based on surrogate function with warm restart, leading to results matching or surpassing the state-of-the-art methods in LCC-based applications. There is a significant distinction between the proposed approach and the previous studies: the surrogate function with warm restart for LLC with LCAs enjoys the advantage of simple implementation and a faster training speed than others [8]. We apply online dictionary learning to three tasks: the reconstruction based classification, feature learning for image classification, and locally linear classification. Experimental results show that the dictionary learned by our approach achieves good performances on these LCC-based applications.

The promising results of this paper motive a further examination of our approach with other locality adaptors. Beyond GA and EA, we plan to use the proposed learning framework in Laplacian constraint [28] which is computational cost. Another indication from our work is that the surrogate function with warm restart serves as an efficient method for both supervised dictionary learning [20,29].

Table 2
Image classification results on PASCAL VOC 2007 dataset.

	Aero	Bicyc	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	
Best'07	**77.5**	63.6	**56.1**	**71.9**	**33.1**	60.6	78.0	58.8	53.3	42.6	
$\text{LLC}_{rep.}$	74.8	**65.2**	50.7	70.9	28.7	**68.8**	**78.8**	**61.7**	54.3	48.6	
$\text{LLC}_{our.}$	73.2	63.4	48.7	69.5	28.2	66.3	76.8	59.6	53.7	49.2	
Ours	75.4	64.7	51.2	70.6	29.3	68.4	78.1	61.5	**54.7**	**50.3**	
	Table	Dog	Horse	Mbike	Person	Plant	Sheep	Sofa	Train	Tv	Average
Best'07	**54.9**	**45.8**	**77.5**	64.0	**85.9**	36.3	**44.7**	50.9	**79.2**	53.2	59.4
$\text{LLC}_{rep.}$	51.8	44.1	76.6	**66.9**	83.5	**30.8**	44.6	53.4	78.2	**53.2**	59.3
$\text{LLC}_{our.}$	50.2	42.7	74.2	64.6	82.7	29.5	42.1	51.8	77.5	52.7	57.8
Ours	51.4	44.7	76.3	65.7	83.6	30.4	44.2	**53.7**	78.5	53.2	59.3

$\text{LLC}_{rep.}$ means the reported results in [4], and $\text{LLC}_{our.}$ means the results implemented by ourself.

Table 3
Classification error rate (%) on USPS and MNIST with different number of items in dictionary.

Methods	USPS (number of items)				MNIST (number of items)			
	40	80	120	160	40	80	120	160
LSC-exp [8]	2.42	2.33	2.21	2.17	2.96	2.36	1.98	1.74
LSC-L2 [8]	3.41	3.28	3.15	2.82	4.81	4.13	3.83	3.51
LLC [4]	2.63	2.42	2.37	2.32	3.02	2.52	2.17	1.83
Our (GA)	**2.42**	**2.21**	**2.18**	**2.06**	**2.85**	**2.12**	**1.78**	**1.71**
Our (EA)	3.38	3.28	3.08	2.64	4.62	4.10	3.64	3.22

Acknowledgment

This work was supported in part by National Basic Research Program of China (973 Program): 2012CB316400, in part by Natural Science Foundation of China: 61332016, 61202234, 61202322, 61 303154, 61133003, 61390510 and 61472387, by Beijing Natural Science Foundation: 4132010 and KZ201310005006, and by Funding Project for Academic Human Resources Development in Institutions of Higher Learning Under the Jurisdiction of Beijing Municipality (PHR).

Appendix

If EA (8) is adopted, the partial gradient of (5) with respect to $\mathbf{v}_m$ is

$$\frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{v}_m^t} = \sum_{i=1}^{t} \left(-\gamma_{im}(\mathbf{x}_i - \mathbf{V}^t\gamma_i) + 2\lambda \sum_{m=1}^{M} \frac{\partial p_{im}}{\partial \mathbf{v}_m^t} \gamma_{im}^2 p_{im} \right), \tag{20}$$

where $\partial p_{im}/\partial \mathbf{v}_m^t = 2(\mathbf{v}_m^t - \mathbf{x}_i)$. Substituting this partial derivative into (20) gives the partial derivative of $\mathbf{v}_m^t$:

$$\frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{v}_m^t} = \sum_{i=1}^{t} \left(-\gamma_{im}(\mathbf{x}_i - \mathbf{V}^t\gamma_i) + 4\lambda \sum_{m=1}^{M} p_{im}\gamma_{im}^2(\mathbf{v}_m^t - \mathbf{x}_i) \right). \tag{21}$$

The partial derivative of $\partial L(\mathbf{V}^t)/\partial \mathbf{V}^t$ is computed as

$$\begin{aligned} \frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{V}^t} &= \left[\frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{v}_1^t}, \ldots, \frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{v}_m^t}, \ldots, \frac{\partial \tilde{L}(\mathbf{V}^t)}{\partial \mathbf{v}_M^t} \right] \\ &= \mathbf{V}^t \sum_{i=1}^{t} (\gamma_i\gamma_i^T + 4\lambda\Lambda_i) - \sum_{i=1}^{t} (\mathbf{x}_i\gamma_i^T + 4\lambda\mathbf{x}_i\Sigma_i), \end{aligned} \tag{22}$$

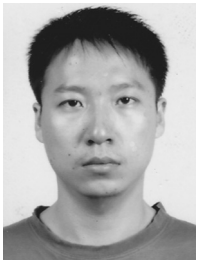
where the matrix Λ_i is the diag matrix with $p_{im}\gamma_{im}^2$ as the elements, and the Σ_i is the vector $[p_{i1}\gamma_{i1}^2, \ldots, p_{im}\gamma_{im}^2]$. Therefore, the past information in (22) is also further stored as

$$\mathbf{A}^t = \sum_{i=1}^{t} \gamma_i\gamma_i^T + 4\lambda\Lambda_i, \tag{23}$$

$$\mathbf{B}^t = \sum_{i=1}^{t} \mathbf{x}_i\gamma_i^T + 4\lambda\mathbf{x}_i\Sigma_i. \tag{24}$$

References

[1] K. Yu, T. Zhang, Y. Gong, Nonlinear learning using local coordinate coding, in: Neural Information Processing System, 2009, pp. 2223–2231.
[2] J. Pang, Q. Huang, B. Yin, L. Qin, D. Wang, Theoretical analysis of learning local anchors for classification, in: International Conference on Pattern Recognition, 2012, pp. 1803–1806.
[3] L. Ladický, P. Torr, Locally linear support vector machines, in: International Conference on Machine Learning, 2011, pp. 985–992.
[4] J. Wang, J. Yang, K. Yu, F. Lv, T.S. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: IEEE Computer Vision and Pattern Recognition, vol. 2, 2010, pp. 3360–3367.
[5] R. Tibshirani, Regression shrinkage and selection via the LASSO, J. R. Stat. Soc. (Ser. B) 58 (1996) 267–288.
[6] S. Zhang, H. Yao, H. Zhou, X. Sun, S. Liu, Robust visual tracking based on online learning sparse representation, Neurocomputing 100 (2013) 31–40.
[7] H. Lee, A. Battle, R. Raina, A. Ng, Efficient sparse coding algorithms, in: Neural Information Processing Systems, 2007, pp. 801–808.
[8] C.-P. Wei, Y.-W. Chao, Y.-R. Yeh, Y.-C.F. Wang, Locality-sensitive dictionary learning for sparse representation based classification, Pattern Recognit. 46 (2013) 1277–1287.
[9] L. Bottou, O. Bousquet, The trade-offs of large scale learning, in: Neural Information Processing Systems, vol. 20, 2008, pp. 161–168.
[10] D. Bertsekas, Nonlinear Programming, Athena Scientific, Belmont, 1999.
[11] J. Mairal, F. Bach, J. Ponce, G. Sapiro, Online dictionary learning for sparse coding, in: International Conference on Machine Learning, 2009, pp. 87–95.
[12] P. Tseng, S. Yun, A coordinate gradient descent method for nonsmooth separable minimization, Math. Prog. 117 (2009) 387–423.
[13] D. Donoho, For Most Large Underdetermined Systems of Linear Equations, The Minimal L1-Norm Solution is also the Sparsest Solution, Technical Report, Stanford University.
[14] D. Donoho, For Most Large Underdetermined Systems of Linear Equations, The Minimal L1-Norm Near-Solution Approximates the Sparsest Near-Solution, Technical Report, Stanford University.
[15] S. Zhang, H. Yao, X. Sun, X. Lu, Sparse coding based visual tracking: review and experimental comparison, Pattern Recognit. 46 (2013) 1772–1788.
[16] B. Xie, M. Song, D. Tao, Large-scale dictionary learning for local coordinate coding, in: British Machine Vision Conference, vol. 2, 2010, pp. 1–9.
[17] S. Nemirovski, D. Yudin, Problem Complexity and Method Efficiency in Optimization, Wiley, New York, 1983.
[18] Y. Nesterov, Introductory Lectures on Convex Optimization, Kluwer Academic Publisher, Dordrecht, The Netherlands, 2004.
[19] C.D. Meyer, Matrix Analysis and Applied Linear Algebra, Society for Industrial and Applied Mathematics, 2000.
[20] J. Mairal, F. Bach, J. Ponce, G. Sapiro, Online learning for matrix factorization and sparse coding, J. Mach. Learn. Res. 11 (1) (2011) 19–60.
[21] M. Everingham, L. Gool, C. Williams, J. Winn, A. Zisserman, The Pascal Visual Object Classes Challenge 2007 (VOC 2007) Results, 2007.
[22] F. Samaria, A. Harter, Parameterisation of a stochastic model for human face identification, in: Proceedings of the IEEE Workshop on Applications of Computer Vision, 1994, pp. 138–142.
[23] J.J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell. 16 (5) (1994) 550–554.
[24] L. Bottou, Online Algorithms and Stochastic Approximations, Cambridge University Press, 1998.
[25] J. Wright, A. Yang, A. Ganesh, S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Learn. 31 (2) (2009) 210–227.
[26] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, in: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, 2006, pp. 2169–2178.
[27] J. Yang, K. Yu, Y. Gong, T.S. Huang, Linear spatial pyramid matching using sparse coding for image classification, in: IEEE Computer Vision and Pattern Recognition, 2009, pp. 1794–1801.
[28] S. Gao, I.W. Tsang, L. Chia, Laplacian sparse coding, hypergraph Laplacian sparse coding, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 92–104.
[29] S. Lazebnik, M. Raginsky, Supervised learning of quantizer codebooks by information loss minimization, IEEE Trans. Pattern Anal. Mach. Intell. 31 (7) (2009) 1294–1309.



Junbiao Pang received the B.S. and the M.S. degrees in computational fluid dynamics and computer science from the Harbin Institute of Technology, Harbin, China, in 2002 and 2004, respectively, and the Ph.D. degree at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2011.

He is currently a Faculty Member with the College of Metropolitan Transportation, Beijing University of Technology, Beijing, China. His research areas include computer vision, multi-media and machine learning, and he has authored or coauthored approximately 10 technical papers.

Chunjie Zhang received the BE from Nanjing University of Posts and Telecommunications, China, in 2006, and the Ph.D. degree at the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, in 2012.

He is currently a faculty member with Graduate University of the Chinese Academy of Sciences (CAS), Beijing, China. His research interests include machine learning, image content analysis, and object categorization.

Lei Qin received the B.S. and M.S. degrees in mathematics from the Dalian University of Technology, Dalian, China, in 1999 and 2002, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2008.

He is currently a faculty member with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His research interests include image/video processing, computer vision, and pattern recognition. He has authored or coauthored over 10 technical papers in the area of computer vision.

Weigang Zhang is an associate professor in the School of Computer Science and Technology, Harbin Institute of Technology at Weihai. He received the B.S. and M.S. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2003 and 2005.

He is currently working toward the Ph.D. degree at the School of Computer Science and Technology, Harbin Institute of Technology. His research interests include multimedia computing, pattern recognition and computer vision. He has authored or coauthored approximately 20 academic papers.

Laiyun Qing received the B.Sc. and M.Sc. degrees in computer science from Northeast University, Shenyang, China, in 1996 and 1999, respectively. She received the Ph.D. degree in computer science from Graduate University of Chinese Academy of Sciences (CAS), in 2005. She is currently an associate professor at the College of Computer and Automatic Controlling, University of Chinese Academy of Sciences. Her research interests include machine learning, pattern recognition and image and video analysis.

Qingming Huang received the B.S. degree in computer science and Ph.D. degree in computer engineering from Harbin Institute of Technology, Harbin, China, in 1988 and 1994, respectively. He is currently a professor with the Graduate University of the Chinese Academy of Sciences (CAS), Beijing, China, and an adjunct research professor with the Institute of Computing Technology, CAS, and with Beijing University of Technology.

He has been granted by China National Funds for Distinguished Young Scientists in 2010. He has authored or coauthored more than 170 academic papers in prestigious international journals and conferences. His research areas include multimedia video analysis, video adaptation, image processing, computer vision, and pattern recognition.

Baocai Yin received the B.S., M.S., and Ph.D. degrees from Dalian University of Technology, Liaoning, China, in 1985, 1988, and 1993, respectively, all in mathematics. He was a postdoctoral researcher in the Computer Science Department of Harbin Institute of Technology (HIT), Harbin, China, during 1993–1995.

He was an associate professor in the Department of Computer Science and Technology, Beijing University of Technology, Beijing, China, from 1995 to 1998. Since 1998, he has been a professor in College of Computer Science and Technology, Beijing University of Technology. He is currently a professor in College of Metropolitan Transportation. His research interests include digital multimedia, data mining, virtual reality and computer graphics.