Tensor Dictionary Learning for Positive Definite Matrices

Ravishankar Sivalingam, Daniel Boley, *Senior Member, IEEE,* Vassilios Morellas, *Member, IEEE,* and Nikolaos Papanikolopoulos, *Fellow, IEEE*

***Abstract*—Sparse models have proven to be extremely successful in image processing and computer vision. However, a majority of the effort has been focused on sparse representation of vectors and low-rank models for general matrices. The success of sparse modeling, along with popularity of region covariances, has inspired the development of sparse coding approaches for these positive definite descriptors. While in earlier work [1], the dictionary was formed from all, or a random subset of, the training signals, it is clearly advantageous to learn a concise dictionary from the entire training set. In this paper, we propose a novel approach for dictionary learning over positive definite matrices. The dictionary is learned by alternating minimization between sparse coding and dictionary update stages, and different atom update methods are described. A discriminative version of the dictionary learning approach is also proposed, which simultaneously learns dictionaries for different classes in classification or clustering. Experimental results demonstrate the advantage of learning dictionaries from data both from reconstruction and classification viewpoints. Finally, a software library is presented comprising C++ binaries for all the positive definite sparse coding and dictionary learning approaches presented here.**

***Index Terms*—Sparse coding, dictionary learning, positive definite matrices, region covariance descriptors, optimization.**

I. Introduction

Sparse coding methods transform a given signal into a set of sparse coefficients with the help of a dictionary or basis set. In the vector domain, pre-defined dictionaries are available which can be constructed using analytical expressions - for *e.g.*: Fourier, DCT, wavelets, etc. However, for applications involving only specific classes of signals, it is more interesting to use a domain-specific dictionary rather than universal dictionaries.

This paper addresses the issue of learning a *data-driven* dictionary from a training set of positive definite matrices. The dictionary learning problem is formulated, analogous to similar approaches in vector dictionary learning. An alternating minimization approach to learn the dictionary is presented, and iterative gradient and Newton methods for updating the dictionary atoms are derived. When the dimensions of the data become too large, we propose an efficient matrix conjugate gradient approach to compute the Newton direction.

R. Sivalingam is with 3M Corporate Research, St. Paul, MN 55144. Email: ravi@cs.umn.edu

D. Boley, V. Morellas, and N. Papanikolopoulos are with the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, 200 Union Street SE, Minneapolis, MN 55455. Email: {boley,morellas,npapas}@cs.umn.edu.

A. Related Work

This work was primarily motivated by the use of region covariance descriptors are features in computer vision and image processing. Region Covariance Descriptors (RCDs) were first introduced by Tuzel et al. [2] as a novel region descriptor for object detection and texture classification. These descriptors by construction are positive definite[1]. Our earlier work on sparse coding for positive definite matrices [1] naturally led to the development of learning techniques for positive definite dictionaries.

Related work involving similar sparse decompositions of positive definite matrices are given below: In [3], Guo *et al.* take the covariance descriptors to the tangent space, by the logarithm map and perform vector sparse coding in this Euclidean space. The resultant algorithm gives good performance for action recognition in video. Wang and Vemuri [4] also learn sparse representations over positive definite matrices in the tangent space, via the logarithm and exponential maps. In a similar approach, Sra and Cherian [5] learn a generalized dictionary of rank-1 positive semidefinite atoms to sparsely represent covariance descriptors. However, the authors in the above two approaches use the Frobenius norm as the error metric. Pfander *et al.* [6] decompose a general matrix as a sparse linear combination of a dictionary of matrices by multiplying all the involved matrices on a known vector reducing the matrix problem to a known vector problem with well-established guarantees. Wang *et al.* [7] present the *Common Component Analysis* problem, where the authors learn a common low-dimensional subspace for a set of high-dimensional covariance matrices.

More recently, Harandi *et al.* [8] use the symmetric Stein divergence [9] to embed the Riemannian manifold into a Reproducing Kernel Hilbert Space (RKHS). They proceed to derive new sparse coding and dictionary learning techniques under this divergence. This is the most closely related work to ours, but with the use of a different divergence to measure the reconstruction error.

The rest of this paper is organized as follows: In Section II, we give a brief overview of our positive definite sparse coding approach from [1]. In Section III, we present the dictionary learning formulation, and proceed with optimization algorithms to the learn the dictionary from data. Section IV presents a variation of the dictionary learning technique incorporating similarity between atoms of dictionaries from

[1] singular descriptors are regularized by adding a small multiple of the identity matrix

different classes, so as to learn a discriminative dictionary for classification and clustering applications. In Section V we present experimental results with synthetic and real texture data. Section VI describes the software library released as part of this work, and we present our conclusions and future directions in Section VII.

II. Positive Definite Sparse Coding

Given a known *dictionary* consisting of K $n \times n$ positive definite matrices $\mathcal{A} = \{A_i\}_{i=1}^K$, where each $A_i \in \mathbb{S}_{++}^n$ is referred to as a dictionary atom, and a signal $S \in \mathbb{S}_{++}^n$, positive definite sparse coding [1] aims to represent S as a linear combination of the dictionary atoms, *i.e.*,

$$S = x_1 A_1 + x_2 A_2 + \ldots + x_K A_K = \sum_{i=1}^{K} x_i A_i, \qquad (1)$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_K)^T$ is the coefficient vector. With a slight abuse of notation, we will henceforth represent the sum $\sum_{i=1}^K x_i A_i$ as $\mathcal{A}\mathbf{x}$ for the sake of convenience[2].

This reconstruction is achieved by minimizing the following sparse coding objective:

$$\min_{\mathbf{x} \geq 0} \quad D_{\mathrm{ld}}\left(\mathcal{A}\mathbf{x}, S\right) + \lambda \left\|\mathbf{x}\right\|_1 \qquad (2a)$$

$$\text{s.t.} \quad \mathcal{A}\mathbf{x} \succeq 0, \qquad (2b)$$

where $D_{\mathrm{ld}}(\cdot,\cdot)$ is the Logdet divergence, given by:

$$D_{\mathrm{ld}}(X, Y) = \mathrm{tr}\left(XY^{-1}\right) - \log\det\left(XY^{-1}\right) - n, \qquad (3)$$

and λ is the regularization parameter influencing the sparsity of $\mathbf{x}$.

In our earlier work on positive definite sparse coding [1], the dictionaries were constructed by sampling from the data. However, when sufficient data is available, learning a dictionary tailored to this class of samples yields a much better sparse reconstruction of the data. Hence, in this work, we propose new dictionary learning techniques to learn dictionaries of positive definite atoms

III. Positive Definite Dictionary Learning

A. Dictionary Learning Formulation

Given a training set $\mathcal{S} = \{S_j\}_{j=1}^N$, $S_j \in \mathbb{S}_{++}^n$, the problem of learning the dictionary $\mathcal{A} = \{A_i\}_{i=1}^K$, $A_i \in \mathbb{S}_{++}^n$ can be formulated as:

Dictionary Learning:

$$\min_{\mathcal{A}, X} \quad \sum_{j=1}^{N} D_{\mathrm{ld}}\left(\mathcal{A}\mathbf{x}_j, S_j\right) + \lambda \left\|\mathbf{x}_j\right\|_1 \qquad (4a)$$

$$\text{s.t.} \quad \mathbf{x}_j \geq 0 \qquad \text{for } j = 1, \ldots, N \qquad (4b)$$

$$A_i \succeq 0 \qquad \text{for } i = 1, \ldots, K \qquad (4c)$$

$$\left\|A_i\right\|_F^2 \leq 1 \qquad \text{for } i = 1, \ldots, K \qquad (4d)$$

[2]This can be distinguished from the regular $A\mathbf{x}$ matrix-vector multiplication through the calligraphic notation of $\mathcal{A}$.

Here $\mathbf{x}_j$ denotes the j-th column of coefficient matrix X. As mentioned in Section II, the atoms should be normalized by their Frobenius norm. However, the constraint $\left\|A_i\right\|_F^2 = 1$ is non-convex, and therefore we have relaxed the constraint to be convex $\left\|A_i\right\|_F^2 \leq 1$.

The dictionary learning problem (4) is non-convex in $(\mathcal{A}, X)$, and therefore there is no unique minimizer $(\mathcal{A}^*, X^*)$. However, the problem is convex in one argument given the other fixed, as is also the case in the vector dictionary learning problem. This naturally leads to an alternating minimization approach to arrive at a stationary point of the optimization problem.

B. Approach: Alternating Minimization

Similar to other dictionary learning algorithms [10], we approach this problem through *alternating minimization*, repeating the following steps:

(a) Given $\mathcal{S}$ and $\mathcal{A}$ fixed, solve for X.
(b) Given $\mathcal{S}$ and X fixed, solve for $\mathcal{A}$.

Although this approach does not guarantee reaching a universal minimizer, we are guaranteed to reach a local minimum of the objective function in (4) [11]. The first step mentioned above is simply the sparse coding of the training set $\mathcal{S}$, which we will refer to as the *sparse coding step* of the dictionary learning procedure. The second step involves updating the dictionary atoms while keeping the sparse coefficients fixed, which we denote as the *dictionary update step*. The training data is sampled to initialize the dictionary $\mathcal{A}^0$.

Motivated by the K-SVD algorithm by [10], the dictionary update is performed sequentially, updating one atom $A_i \in \mathcal{A}$ at a time, keeping the sparsity structure of X fixed, but allowing the corresponding non-zero coefficients of A_i to change in value. At iteration k of the dictionary learning procedure (denoted in the superscript), the atom A_i^{k-1} is updated to A_i^k, given $\left\{A_1^k, A_2^k, \ldots, A_{i-1}^k, A_{i+1}^{k-1}, \ldots A_K^{k-1}\right\}$ and X^k. The dictionary atoms are updated using a gradient projection method, where the constraint set for A_i is defined by $A_i \succ 0$ and $normA_iF^2 \leq 1$. Convergence of block coordinate descent type methods where each iteration comprises a gradient projection step is discussed in [12].

Algorithm 1 Dictionary Learning

Input: Data $\mathcal{S} = \{S_j\}_{j=1}^N$, dictionary size K, sparsity parameter λ
Output: $\mathcal{A} = \{A_i\}_{i=1}^K$

```
k = 0
Initialize A^0 sampled from S
repeat
    k ← k + 1
    Given S and A^{k-1}, compute the sparse coefficients X^k
    for i = 1 to K do
        Update atom A_i^{k-1} to A_i^k, along with the corresponding coefficients in X^k (Algorithm 2)
    end for
until convergence
```

C. Atom Update

In this section, we present the optimization subroutine to update atom A_i in the dictionary update step. Let ω_i be the *active set*, $\omega_i = \{j | j \subseteq \{1, \ldots, N\}, x_{ij} \neq 0\}$, *i.e.*, the subset of signals which use atom A_i.

The reconstruction $\hat{S}_j$ of each S_j, $j \in \omega_i$ can be decomposed into the constant and variable components under the optimization of A_i:

$$\hat{S}_j = \sum_{i' \neq i} x_{i'j} A_{i'} + x_{ij} A_i = \hat{S}_j^{(i)} + x_{ij} A_i. \tag{5}$$

$\hat{S}_j^{(i)}$ is the reconstruction of S_j without the contribution of A_i.

The sub-problem of (4) to optimize atom A_i keeping all other atoms fixed, is given by:

$$\min_{A_i \succeq 0} \quad \sum_{j \in \omega_i} D_{\text{ld}} \left(\sum_{i' \neq i} x_{i'j} A_{i'} + x_{ij} A_i, S_j \right) \tag{6}$$

Expanding Equation (6) and retaining only the terms relevant to A_i, we get:

$$\min_{A_i \succeq 0} \sum_{j \in \omega_i} \left[x_{ij} \text{tr} \left(A_i S_j^{-1} \right) - \log \det \left(\sum_{i' \neq i} x_{i'j} A_{i'} + x_{ij} A_i \right) \right]$$

Denoting the above objective function as $f(A_i)$ and taking the gradient w.r.t. A_i,

$$\nabla f(A_i) = \sum_{j \in \omega_i} x_{ij} S_j^{-1} - x_{ij} \left(\hat{S}_j^{(i)} + x_{ij} A_i \right)^{-1}. \tag{7}$$

We propose iterative descent methods such as gradient descent and Newton descent below.

Algorithm 2 Atom Update

Input: A_i, $\left\{ x_{ij}, S_j, \hat{S}_j \mid j \in \omega_i \right\}$

Output: A_i, $\left\{ x_{ij}, \hat{S}_j \mid j \in \omega_i \right\}$

repeat

Compute descent direction ΔA_i using (9) or (14)

Choose stepsize α by line search s.t. $A_i + \alpha \Delta A_i \succeq 0$

$A_i^{\text{new}} \leftarrow A_i + \alpha \Delta A_i$

$\hat{S}_j \leftarrow \hat{S}_j + x_{ij} \left(A_i^{\text{new}} - A_i \right) \quad \forall j \in \omega_i$

$t = \max \left\{ \| A_i^{\text{new}} \|_F, 1 \right\}$

$A_i \leftarrow A_i^{\text{new}} / t$

$x_{ij} \leftarrow t \; x_{ij} \quad \forall j \in \omega_i$

until convergence

1) Gradient Descent: The gradient of the objective $f(A_i)$ is given by:

$$\nabla f(A_i) = \sum_{j \in \omega_i} x_{ij} \left(S_j^{-1} - \hat{S}_j^{-1} \right). \tag{8}$$

The gradient descent direction ΔA_i^g is given by the negative of the gradient:

$$\Delta A_i^g = \sum_{j \in \omega_i} x_{ij} \left(\hat{S}_j^{-1} - S_j^{-1} \right). \tag{9}$$

The gradient descent update algorithm is, therefore,

$$A_i^k \leftarrow A_i^{k-1} + \alpha \Delta A_i^g \quad \text{s.t. } A_i^k \succeq 0, \tag{10}$$

with stepsize $\alpha \geq 0$ determined using line search techniques. The stepsize should also satisfy the constraint that the updated atom A_i^k is positive semi-definite.

Two possibilities are to use the the *exact* line search or the *backtracking* (*Armijo*) line search. In practice, we see that these two methods do not provide much improvement in the objective function in each atom update iteration. Instead, we use the *Barzilai-Borwein* (BB) step sizes [13]:

$$\alpha_{BB1}^k = \frac{\left\langle A_i^k - A_i^{k-1}, \nabla f\left(A_i^k\right) - \nabla f\left(A_i^{k-1}\right) \right\rangle}{\left\| \nabla f\left(A_i^k\right) - \nabla f\left(A_i^{k-1}\right) \right\|_F^2}, \tag{11}$$

$$\alpha_{BB2}^k = \frac{\left\| A_i^k - A_i^{k-1} \right\|_F^2}{\left\langle A_i^k - A_i^{k-1}, \nabla f\left(A_i^k\right) - \nabla f\left(A_i^{k-1}\right) \right\rangle}, \tag{12}$$

for iteration k. The BB stepsize choice yields a much stronger net decrease in the objective function value compared to exact or backtracking line searches.

2) Newton Descent: Taking the second derivative of the gradient (8), we get the expression for the Hessian:

$$\nabla^2 f(A_i) = \sum_{j \in \omega_i} \left(x_{ij} \hat{S}_j^{-1} \right) \otimes \left(x_{ij} \hat{S}_j^{-1} \right) \tag{13}$$

The Newton descent direction ΔA_i^N is obtained by solving:

$$\begin{aligned} \nabla^2 f(A_i) \; \Delta A_i^N &= -\nabla f(A_i) \\ \sum_{j \in \omega_i} x_{ij}^2 \hat{S}_j^{-1} \Delta A_i^N \hat{S}_j^{-1} &= \sum_{j \in \omega_i} x_{ij} \left(\hat{S}_j^{-1} - S_j^{-1} \right) \end{aligned} \tag{14}$$

The Newton descent update algorithm is, therefore,

$$A_i^k \leftarrow A_i^{k-1} + \alpha \Delta A_i^N \quad \text{s.t. } A_i^k \succeq 0, \tag{15}$$

with stepsize $\alpha \geq 0$.

The Newton direction computation involves solving an $n^2 \times n^2$ system of linear equations, given by:

$$\begin{aligned} \underbrace{\sum_{j \in \omega_i} \left(x_{ij} \hat{S}_j^{-1} \right) \otimes \left(x_{ij} \hat{S}_j^{-1} \right)}_{n^2 \times n^2} \text{vec} \left(\Delta A_i^N \right) \\ = \text{vec} \left(\sum_{j \in \omega_i} x_{ij} \left(\hat{S}_j^{-1} - S_j^{-1} \right) \right). \end{aligned} \tag{16}$$

Let us denote this positive definite system as $A\mathbf{x} = \mathbf{b}$, with $A = \sum_{j \in \omega_i} \left(x_{ij} \hat{S}_j^{-1} \right) \otimes \left(x_{ij} \hat{S}_j^{-1} \right)$, $\mathbf{x} = \text{vec} \left(\Delta A_i^N \right)$, and $\mathbf{b} = \text{vec} \left(\sum_{j \in \omega_i} x_{ij} \left(\hat{S}_j^{-1} - S_j^{-1} \right) \right)$.

Explicitly forming A and solving the system is an expensive operation, even with decomposition methods. The cost of directly solving for the Newton direction has a cost of $\mathcal{O}\left(n^6\right)$, where n denotes the dimension of the dictionary atoms. In most of our applications pertaining to region covariance descriptors, n is very small ($\sim 5 - 10$), and therefore this is still acceptable in practice.

When n is much larger, we can take advantage of the fact that although solving for $A\mathbf{x} = \mathbf{b}$ with an explicit A is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

expensive, it is relatively inexpensive to apply the operator A on a given $\mathbf{x}$. This is due to the fact that A is composed of a sum of Kronecker products. This enables us to use iterative methods like conjugate gradient to directly solve Equation (14).

D. Matrix Conjugate Gradient

In this section, we present a conjugate gradient method to directly solve (14) for the Newton descent direction. Writing the general form[3] of Equation (14),

$$\sum_{i=1}^{M} A_i X A_i^T = B. \tag{17}$$

The matrix conjugate gradient algorithm to iteratively solve Equation (17) for X is given in Algorithm 3.

Algorithm 3 Matrix Conjugate Gradient

Input: $\{A_i\}_{i=1}^M$, B
Output: X^*
$X_0 = 0_{n\times n}$
$R_0 = B - \sum_{i=1}^M A_i X_0 A_i^T$
$P_0 = R_0$
$k = 0$
repeat
$\alpha_k = \frac{\langle\ R_k\ ,\ R_k\ \rangle}{\langle\ P_k\ ,\ \sum_{i=1}^M A_i P_k A_i^T\ \rangle}$
$X_{k+1} = X_k + \alpha_k P_k$
$R_{k+1} = R_k - \alpha_k \sum_{i=1}^M A_i P_k A_i^T$
$\beta_k = \frac{\langle\ R_{k+1}\ ,\ R_{k+1}\ \rangle}{\langle\ R_k\ ,\ R_k\ \rangle}$
$P_{k+1} = R_{k+1} + \beta_k P_k$
$k \leftarrow k + 1$
until convergence
$X^* = X_{k+1}$

We compare the direct inversion approach and the matrix conjugate gradient approach for computing the Newton direction during actual dictionary update iterations for synthetic datasets of varying dimensions n. The matrix conjugate gradient is implemented in MATLAB without any further code optimization. The time taken to explicitly construct A is also included in the computation time of the direct approach. The dimensions n is varied from 5 to 50 in steps of 5. The computation times for dimension n are averaged over $25n$ trials. In all comparisons, the returned solution from the conjugate gradient method X^{cg} is within 10^{-5} relative error of the direct solution X^*.

The average speedup obtained by using the matrix conjugate gradient algorithm over the direct inversion method is presented in Figure 1, along with 1σ standard deviation bars. The horizontal line at speedup of 1 shows the cross-over point when the conjugate gradient method overtakes the direct inversion approach in computation time. For $n \leq 15$, it is faster to directly solve for $\mathbf{x}$ than using iterative methods. For $n \geq 20$, the matrix conjugate gradient method gives significant speedups in solving systems of the presented structure.

[3]The notations A_i, X, B in this section are different from the variables of the dictionary learning problem.

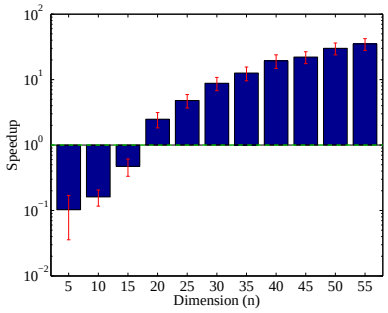


Fig. 1. Average speedup of matrix conjugate gradient vs. direct $A\mathbf{x} = \mathbf{b}$ linear system solution for computation of the Newton descent direction. 1σ bars are also shown.

E. Comparison of Atom Update Techniques

We compare the gradient and Newton atom update techniques in terms of their effectiveness in optimizing the dictionary learning objective function. A set of K 5×5 positive definite atoms $\mathcal{A}_0$ were synthesized. N k-sparse vectors $\{\mathbf{x}_j\}_{j=1}^N$ were sampled, where $N = 100$ and $k = 2$, and signals $\mathcal{S} = \{S_1, \ldots, S_N\}$ were constructed. The dictionary learning was run for a maximum of 15 iterations, and the net reduction in the objective function was compared.

We used dictionary sizes of $K = \{10, 15, 20, 25, 30\}$, and the results were averaged over 25 random trials. The choice of K covers three different scenarios - $K < M$, $K = M$, and $K > M$, where $M = n(n+1)/2$ - *undercomplete*, *complete*, and *overcomplete* cases.

Four atom update techniques were compared:

1) gradient descent with backtracking line search
2) gradient descent with BB stepsize (11)
3) gradient descent with BB stepsize (12)
4) Newton descent

The different techniques were initialized with the same random dictionary. The objective function values $f(\hat{\mathcal{A}})$ at the end of the learning procedure relative to the initial objective $f(\mathcal{A}^0)$ are estimated as an indicator of the quality of the local minimum attained in each learning procedure. This is shown in Figure 2(a). The Newton update method performs the best, as is expected, but the BB stepsize methods greatly improve upon the gradient descent with backtracking line search.

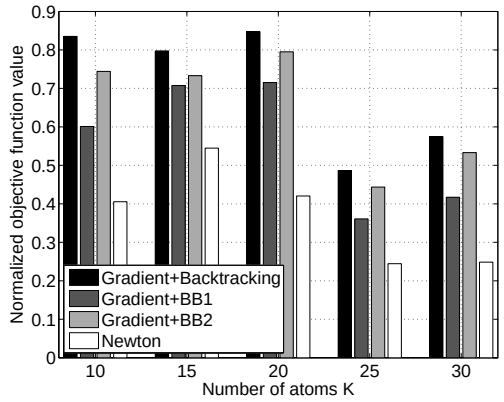
We also test the number of atoms correctly recovered in the learned dictionary in each update technique. The learned atoms $\hat{A}_i$ are matched with the ground truth atoms A_j^* using a coherence threshold of $\mu = \text{tr}\left(\hat{A}_i, A_j^*\right) \geq 0.95$. The Newton dictionary update approach performs the best at recovering the ground truth dictionary atoms, followed by the gradient approaches using Barzilai-Borwein step sizes.

We recommend using the Newton updates for smaller matrices, *i.e.*, $n \leq 10$, and the gradient approach with BB step size selection for larger dimensions. The matrix conjugate gradient can be used to speed up the Newton updates in cases of larger n as well.

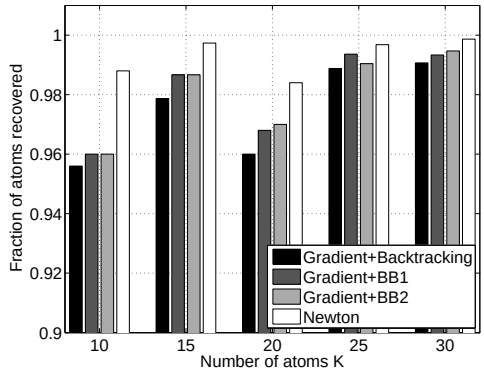
F. Time Complexity

1) Sparse Coding: The MAXDET problem that forms the fundamental part of the sparse coding step [1] has a time

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing



(a)



(b)

Fig. 2. Comparison of atom update methods: (a) Objective function values $f(\hat{\mathcal{A}})$ at the end of the learning procedure relative to the initial objective $f(\mathcal{A}^0)$ and (b) Fraction of recovered atoms with a coherence threshold of $\mu_{\min} = 0.95$. Size of the dictionary K is varied in $\{10, 15, 20, 25, 30\}$ and the results are averaged over 25 different random trials.

complexity of $\mathcal{O}(K^2n^2)$ per Newton iteration [14], with a worst-case complexity of $\mathcal{O}(\sqrt{n})$ Newton iterations at each step in the interior point algorithm. In this work, we use a coordinate descent approach which updates one coordinate of $\mathbf{x}$ at a time, and repeats this over all the coordinates until convergence. This has a time complexity of $\mathcal{O}(Kn^3)$ per iteration over all the coordinates. The n^3 term comes from the computation of matrix inverses, generalized eigenvalues and their sums. Since n is pretty small in most of our applications, and the size of the dictionary is large, our specialized approach and implementation yields faster run times. The coordinate descent approach is used in our software implementation presented in Section VI, where the sparse coding times for practical problem sizes of (n, K) are shown.

2) Dictionary Learning: The dictionary learning approach has a time complexity of $\mathcal{O}(n^3 L_{\max})$ per atom update for the gradient descent methods, and the Newton descent has a time complexity of $\mathcal{O}(n^6 L_{\max})$ since it involves solving an $n^2 \times n^2$ system of equations. $L_{\max}$ denotes the maximum number of inner iterations within each atom update step (Usually this is small in practice: ≤ 5 for initial iterations and just 1 or 2 for later iterations). The rest of the computation in each atom update step is subsumed by complexity of computing the descent direction.

IV. Discriminative Dictionary Learning

Sparse models have been used extensively to classify or cluster data. Learning dictionaries for each class independently without information from the other classes can be compared to *generative modeling*, which may not be able to classify or cluster data with sufficient accuracy when different classes share features. Such a scenario calls for the use of *discriminative modeling*, where the learning should promote discrimination between the sparse models of each class. In other words, the dictionary learned for a certain class should provide good reconstruction for the signals from that class, and poor reconstruction for signals that do not belong to that class. Conversely, a signal from a certain class should be reconstructed best by a dictionary of the same class, compared to all other class dictionaries.

In the vector sparse modeling literature, [15], [16] have used different formulations to solve the dictionary learning problem while increasing the discriminative power of the learned dictionaries. [15] use a logistic loss term in their objective function that penalizes for misclassification of signals. In [16], however, the discrimination is learned in terms of the incoherence between atoms of different class dictionaries. We follow the latter approach in learning discriminative positive definite dictionaries.

Sparse coding has been applied to classification problems in many domains. Here we present applications where we use the tensor sparse coding for classification. Let us denote the number of classes by C. The typical approach to classifying signals with dictionaries is to maintain separate dictionaries for each class $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_C$. The test signal S is sparse-coded independently over each dictionary to get the coefficients $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_C$ respectively. The different class reconstructions are computed as $\hat{S}_c = \mathcal{A}_c \mathbf{x}_c$, $c = 1, \ldots, C$, and the test signal is assigned the label c^* of the class which gives the closest approximation:

$$\text{label} \quad c^* = \arg\min_c \quad D_{\text{ld}}\left(\hat{S}_c, S\right). \tag{18}$$

Throughout this work, this will be the classification approach used.

A. Atom Coherence

Before we proceed any further, we define a fundamental property of dictionaries of positive definite atoms, *coherence*, by extension from vector dictionaries.

The inner product in a positive definite matrix space is given by $\langle A_i, A_j \rangle = \operatorname{tr}(A_i A_j)$.

Definition 1. *The **coherence** between two symmetric positive (semi-)definite dictionary atoms A_i and A_j is given by*

$$\mu(A_i, A_j) = \langle A_i, A_j \rangle = \operatorname{tr}(A_i A_j), \tag{19}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in matrix space. Therefore if the atoms are normalized to unit Frobenius norm as mentioned in the previous section, we have the following bounds on the coherence measure (by Cauchy-Schwarz inequality):

$$\begin{aligned} 0 \quad &\leq \quad \mu(A_i, A_j) \quad \leq \quad 1, \\ A_i, A_j \in \mathbb{S}^n_+, \quad &\|A_i\|_F = \|A_j\|_F = 1. \end{aligned} \tag{20}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

For non-trivial A_i and A_j:

- $\mu(A_i, A_j) = 0$ if and only if they are low-rank (semi-definite) and their eigenspaces are disjoint.
- $\mu(A_i, A_j) = 1$ if and only if $A_i = A_j$.

This can be further extended to define the average coherence between two dictionaries $\mathcal{A}$ (of size K_A) and $\mathcal{B}$ (of size K_B) by

$$\mathcal{Q}(\mathcal{A}, \mathcal{B}) = \frac{1}{K_A K_B} \sum_{i=1}^{K_A} \sum_{j=1}^{K_B} \langle A_i, B_j \rangle . \tag{21}$$

B. Formulation & Approach

Given training data from C different classes, we will attempt to learn the dictionary for each class $c = 1, \ldots, C$. The sizes of the training data from each class c is given by N_c. The training data from class c is specified as $\mathcal{S}^{(c)} = \left\{ S_j^{(c)} \right\}$, $j = 1, \ldots, N_c$, and the dictionary learned to model this data is denoted by $\mathcal{A}^{(c)} = \left\{ A_i^{(c)} \right\}$, $i = 1, \ldots, K_c$, K_c being the dictionary size for class c.

The discriminative power of the dictionaries is induced by including a term which promotes incoherence between the dictionaries of different classes - *i.e.*, between each class c dictionary $\mathcal{A}^{(c)}$ and all other dictionaries $\mathcal{A}^{(c')}$, $c' \neq c$. This is motivated by the work of [16] in learning discriminative dictionaries for classification and clustering. Similar to their work, we will use our definition of atom coherence from Section IV-A and penalize for the coherence between atoms from dictionaries of different classes.

The discriminative dictionary learning problem is given by:

$$\min_{\mathcal{A}^{(1)}, \ldots, \mathcal{A}^{(C)}} \sum_{c=1}^{C} \left[\frac{1}{N_c} \sum_{j=1}^{N_c} \min_{\substack{\mathbf{x} \geq 0 \\ \mathcal{A}^{(c)}\mathbf{x} \succeq 0}} D_{\text{ld}}\left(\mathcal{A}^{(c)}\mathbf{x}, S_j^{(c)} \right) + \lambda \|\mathbf{x}\|_1 + \eta \sum_{c' \neq c} \mathcal{Q}\left(\mathcal{A}^{(c)}, \mathcal{A}^{(c')} \right) \right] \tag{22a}$$

$$A_i^{(c)} \succeq 0, \ \left\| A_i^{(c)} \right\|_F^2 \leq 1, \quad i = 1, \ldots, K_c, \ c = 1, \ldots, C \tag{22b}$$

This coherence term is convex (in fact, linear) in one argument, given the other fixed. Therefore, while updating the class c dictionary $\mathcal{A}^{(c)}$, all other class dictionaries are fixed. The alternating minimization between the sparse coding and dictionary update stages is the same as in the usual dictionary learning approach.

Writing out the coherence term $\mathcal{Q}$ in (22),

$$\begin{aligned} \sum_{c' \neq c} \mathcal{Q}\left(\mathcal{A}^{(c)}, \mathcal{A}^{(c')} \right) &= \sum_{c' \neq c} \frac{1}{K_c K_{c'}} \sum_{i=1}^{K_c} \sum_{i'=1}^{K_{c'}} \operatorname{tr}\left(A_i^{(c)} A_{i'}^{(c')} \right) \\ &= \sum_{i=1}^{K_c} \operatorname{tr}\left(A_i^{(c)} M^{(c)} \right) \\ \text{where } M^{(c)} &= \frac{1}{K_c} \sum_{c' \neq c} \left[\frac{1}{K_{c'}} \sum_{i'=1}^{K_{c'}} A_{i'}^{(c')} \right] . \end{aligned}$$

While updating the dictionary from class c, the factor $M^{(c)}$ encompasses the influence of all the other class dictionary atoms. This is independent of the atom number i in dictionary $\mathcal{A}^{(c)}$. The linear penalty $\operatorname{tr}\left(A_i^{(c)} M^{(c)} \right)$ merely adds an $\eta M^{(c)}$ term to the gradient expression for the dictionary learning problem in Equation (8).

The Hessian from the dictionary learning problem in Equation (13) does not change since the coherence term $\mathcal{Q}$ is linear.

The discriminative atom update in Algorithm 4 can be performed using either the gradient descent or Newton descent methods in Section III-C.

Algorithm 4 Discriminative Dictionary Learning

Input: Data $\mathcal{S}^{(c)} = \left\{ S_j^{(c)} \right\}_{j=1}^{N_c}$, $c = 1, \ldots, C$, dictionary size K, sparsity parameter λ, incoherence parameter η

Output: $\mathcal{A}^{(c)} = \left\{ A_i^{(c)} \right\}_{i=1}^{K}$, $c = 1, \ldots, C$

```
k = 0
for c = 1 to C do
    Initialize A_0^(c) sampled from S^(c)
end for
repeat
    k ← k + 1
    for c = 1 to C do
        Given S^(c) and A_{k-1}^(c), compute the sparse coefficients X_k^(c)
    end for
    for c = 1 to C do
        Given S^(c), X_k^(c), and other class dictionaries {A_k^(1), ..., A_k^(c-1), A_{k-1}^(c+1), ..., A_{k-1}^(C)}, compute the updated dictionary A_k^(c)
    end for
until convergence
```

V. Experiments

A. Dictionary Representation Error

In this experiment, we show a comparison of sparse reconstruction performance with randomly sampled dictionaries, dictionaries learned using K-means, and dictionaries learned using our proposed learning approach. We use the Newton method for the atom update in our dictionary learning approach. 10 different texture images from mosaics #1 and #2 of the Brodatz texture dataset [17] are chosen, and from each texture image, $N = 225$ blocks of size 32×32 are extracted. Covariance descriptors of size 5×5 are computed using the features $I, |I_x|, |I_y|, |I_{xx}|$, and $|I_{yy}|$.

For varying dictionary sizes $K = 5, 10, 15, 20, 25$, and 30, dictionaries are constructed by randomly sampling from the covariance descriptors in each class. These random dictionaries are used as the initialization for the K-means procedure and the dictionary learning approach, both of which are run for 25 iterations. The average reconstruction errors for the covariance descriptors for the initial dictionaries, the K-means dictionaries and the dictionaries learned using our proposed approach are shown in Figure 3, for varying values of dictionary size K. The results are averaged across the 10 different texture images. As is expected, learning a dictionary does better at reconstruction performance than random sampling. Further,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

our proposed approach produces a dictionary that yields a better reconstruction than K-means clustering. This is similar to the comparison to K-means for reconstruction performance shown in [8].

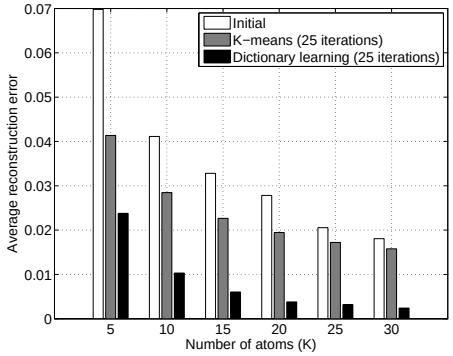


Fig. 3. Comparison of average reconstruction error with dictionaries learned using K-means and our proposed dictionary learning approach, along with the initial random dictionaries, for various dictionary sizes ($n = 5, \lambda = 0$).

B. Classification with Positive Definite Dictionaries

1) Texture Classification: We apply the classification approach described by Equation (18) to classify the 12 different Brodatz texture mosaics [17]. This dataset comprises five 5-class, two 16-class, two 10-class, and three 2-class classification problems. We show classification accuracies for both a randomly initialized dictionary sampled from the data, and dictionaries learned from the data. A dictionary of size $K = 5$ was used for each class, with $\lambda = 0.01$. The results averaged over 10-fold cross-validation are shown in Table I. As can be seen from the results, the learned dictionaries perform better at classifying the textures compared to randomly initialized dictionaries. Average accuracies for KNN classification with $K = 5$ is also shown for a baseline comparison. Note that KNN needs to retain all the training data, whereas in our approach, a compact trained dictionary represents all the training knowledge.

Mosaic	Random Dictionary	Learned Dictionary	KNN ($K = 5$)
1(5)	99.11 %	**100.00** %	100.00 %
2(5)	91.73 %	**93.51** %	98.04 %
3(5)	85.41 %	**96.00** %	97.87 %
4(5)	82.68 %	**94.75** %	98.40 %
5(5)	87.11 %	**89.86** %	98.58 %
6(16)	83.67 %	**90.17** %	94.72 %
7(16)	73.81 %	**86.14** %	94.86 %
8(10)	85.69 %	**93.87** %	97.29 %
9(10)	75.82 %	**87.11** %	94.85 %
10(2)	99.78 %	**100.00** %	100.00 %
11(2)	99.11 %	**100.00** %	100.00 %
12(2)	97.33 %	**98.89** %	100.00 %

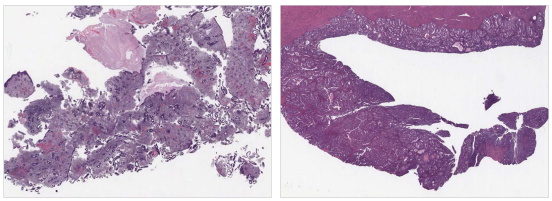
TABLE I
AVERAGE CLASSIFICATION ACCURACY OF BRODATZ TEXTURES WITH 10-FOLD CROSS-VALIDATION.

2) Cancer Tissue Classification: Early diagnosis of any disease is quintessential for effective treatment. This is no less true in the diagnosis and treatment of cancer. For a surgical pathologist, the most time-consuming aspect of the diagnostic process involves arduously scrutinizing tissue slides under a microscope for the evidence of disease. As a result, even a skilled pathologist is able to diagnose only a few patients every day. However, it is possible to expedite this process through computer-assisted diagnosis. Towards this end, we apply the positive definite dictionary learning algorithms towards classification of tissue image regions as cancerous or benign. We use region covariance descriptors to characterize the image blocks extracted from the tissue, since the distinction between the two classes of healthy vs. cancerous tissue is based on the architecture or texture. Our work on using region covariances for this classification, along with vector sparse dictionary learning, has been published earlier in [18] and [19] where we deal with endometrial and prostate cancer tissue images respectively.

We show results with positive definite dictionary learning with the endometrial tissue images from [18]. Sample images from the healthy and endometrioid carcinoma tissue classes, the description of the covariance features used are shown in Figure 4. A combination of spatial and intensity features, with a block size of 200x200 pixels at 5x resolution was seen to give the best performance in [18]. The features used were

$$\phi(x, y) = \left[I, I_x, I_y, \sqrt{I_x^2 + I_y^2}, x, y, \sqrt{x^2 + y^2}, \tan^{-1}(y/x)\right].$$

We choose 4 images each from the healthy and carcinoma classes, and sample 200 blocks from each image. We use this set of 1600 covariance descriptors and perform 4-fold cross-validation - using 1 image from each class for testing and keeping the remaining 3 for training.



(a) (b)

Fig. 4. Samples from the healthy (left) and cancerous (right) images.

In each fold, we use the 8×8 training covariances to learn tensor dictionaries of varying sizes, and classify the test features by the usual least reconstruction error approach. The parameter λ was set to 0.001. We compare with learned dictionaries constructed by randomly sampling the training data. We also compare the performance with a baseline K-NN classifier (with $K = 5$ chosen by cross-validation). The results are shown in Table II for different values of the dictionary size K.

The dictionary learning procedure helps in improving the accuracy of dictionary-based classification, compared to randomly choosing data points for the model. We beat the baseline K-nearest-neighbor classification, while just maintaining only

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

Algorithm		Accuracy	
5-NN		94.31 %	
Tensor Dictionaries	K	Random	Learned
	4	90.75 %	92.75 %
	8	92.25 %	93.25 %
	16	92.44 %	93.31 %
	20	93.13 %	93.88 %
	28	93.88 %	94.63 %
	32	94.50 %	**95.38** %

TABLE II
AVERAGE CLASSIFICATION ACCURACY WITH 4-FOLD CROSS-VALIDATION BETWEEN HEALTHY AND ENDOMETRIOID CANCER TISSUE IMAGE PATCHES.

a few atoms derived from the data - a 32-atom dictionary stores only about 5% of the number of matrices as the K-NN classifier.

C. Discriminative Dictionary Learning and Mutual Coherence

1) Synthetic Data: To demonstrate that discriminative dictionary learning reduces the mutual coherence between class dictionaries, we run this approach on synthetic data generated from known dictionaries. A dictionary of size K atoms is constructed for each of M classes (with $n = 5$, $K = 5$). $N = 100$ samples per class are generated by constructing N T-sparse vectors and multiplying with the dictionary. The support and coefficients of the sparse vectors are chosen uniformly at random.

Figure 5 shows the average coherence between atoms of different class dictionaries for $M = 2, 3$, and 4 classes, for different values of the regularization parameter η. As η increases, the between-class coherence decreases. This decrease is more significant when more classes are present.

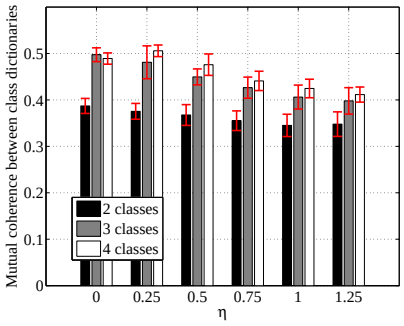
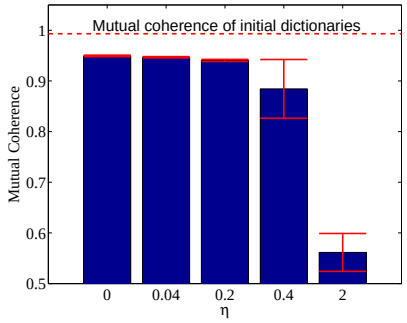
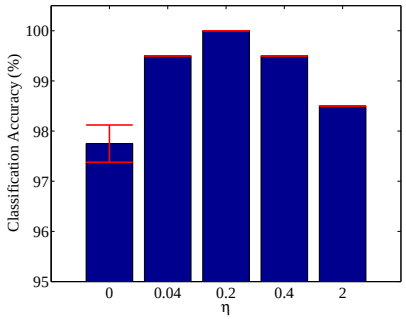


Fig. 5. Mutual coherence between class dictionaries for different number of classes and varying values of η, averaged over 10 iterations (1σ bars are shown). Each dictionary is of size $K = 5$ atoms, with parameter $\lambda = 0$.

2) Texture Data: Another example demonstrating the effect of η on mutual coherence as well as on the classification accuracy is shown next. We take the two texture images from Brodatz mosaic #12 [17] and sample 100 blocks of size 32×32 from each image. Covariance descriptors of size 5×5 are computed using the features $I, |I_x|, |I_y|, |I_{xx}|$, and $|I_{yy}|$. We then learn discriminative dictionaries of size $K = 5$ atoms each for the two classes, with varying values for the incoherence regularizer η.



(a)



(b)

Fig. 6. (a) Mutual coherence between class dictionaries and (b) classification accuracy for 2 Brodatz texture images. $\lambda = 0.1, K = 5, N_1 = N_2 = 100$, 20 training iterations, averaged over 10 trials (1σ bars shown). Coherence drops as η increases, but beyond a certain point accuracy starts to deteriorate.

Figure 6(a) shows the mutual coherence between the dictionaries after 20 training iterations. All trials were initialized using the same set of initial dictionaries, the mutual coherence between which is shown in the dotted line. As η increases, the net effect is to reduce the coherence between the class dictionaries. However, as can be seen in the accuracy plot of Figure 6(b), the accuracy increases when η increases from 0, but only up to a certain point. Beyond that, the effect of over-emphasizing the mutual incoherence between the dictionaries affects the reconstruction performance and results in a drop in classification accuracy. Similar trends can be observed in other experiments as well. Empirically, suitable values for η were found to be close to the average of reconstruction error terms in Equation (22).

D. Discriminative Dictionary Learning for Classification

1) Discriminative Texture Classification: We apply the discriminative dictionary learning algorithm from Algorithm 4 to classify two example textures from the Brodatz texture mosaics dataset [17]. Each of these examples have 5 different texture classes. The different types of dictionaries used were:

1) Randomly sampled from the data
2) Learned from the data independently in each class (denoted as *DL*)
3) Learned from the data discriminatively with the coherence penalty (denoted as *DDL*)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

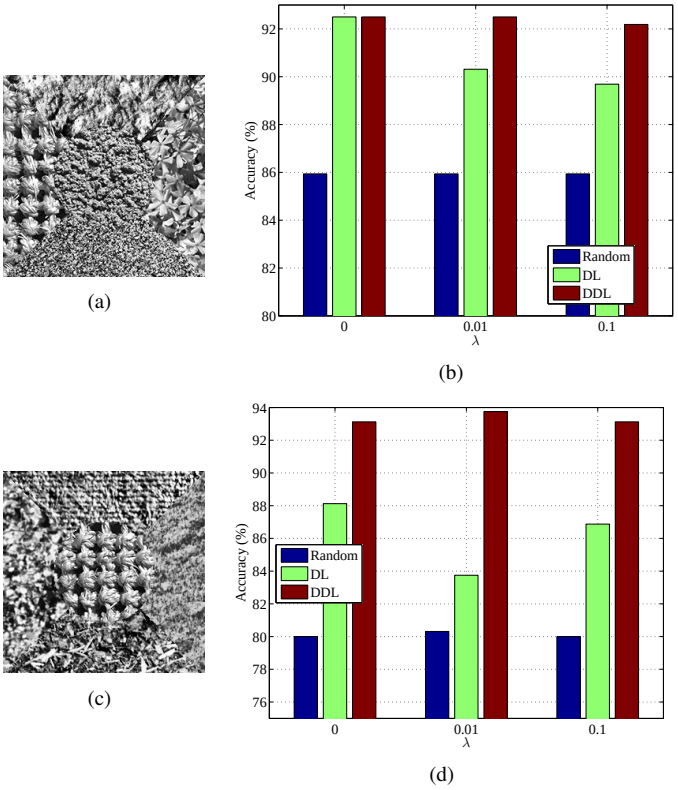


Fig. 7. Comparison of accuracy between randomly initialized dictionaries (*random*) and dictionaries learned with (*DDL*) and without (*DL*) the discriminative penalty. The corresponding textures are shown on the left.

We chose a dictionary size of $K = 4$, and varied the sparsity regularizer λ. The value of η was set to be 0.1. The improvement of accuracy in the texture classification is shown in Figure 7. The learned dictionary improves the classification performance, and the discriminative training proves a further boost to the accuracy, sometimes substantially.

Classification Approach	Accuracy
K-NN ($K = 6$)	82.50 %
Vectorized Log-covariance [3]	83.89 %
DL ($K = 15$)	83.43 %
DDL ($K = 15$)	83.78 %
DL ($K = 30$)	84.01 %
DDL ($K = 30$)	84.59 %
DL ($K = 60$)	85.75 %
DDL ($\mathbf{K = 60}$)	**86.37** %

TABLE III
CLASSIFICATION ACCURACY ON THE KTH DATASET.

2) Action Recognition with the KTH Dataset: We apply the discriminative dictionary learning approach to classify actions from the KTH dataset from [20]. There are 6 different actions performed about 4 times each by 25 subjects (for a total of 598 sequences). We use the 12×12 covariance feature representation from [3], using the optical flow of the video frames.

We use the 8 training and 9 test subjects indicated in the dataset, and test our discriminative dictionary learning approach. We compare this with the baseline K-nearest-neighbors classification (best $K = 6$), as well as the vectorized-log-covariance sparse coding approach from [3] (best sparsity $k = 2$ with our implementation). We only compare with these two methods using optical flow-based region covariance descriptors, and the classification accuracy is shown in Table III. *DL* implies dictionary learning without discrimination ($\eta = 0$), and DDL denotes discriminative dictionary learning ($\eta = 0.1$). λ was set to 0.1. Note that in the first two approaches, the entire training data is available to the classifier during test time, which is not the case in our approach. The learned dictionary models the features from the different classes, and the discriminative term improves the overall classification accuracy.

VI. SOFTWARE

As part of this work, we present a software suite entitled the *Tensor Sparse Library*[4], comprising C++ binaries for the algorithms presented here, namely:

- Sparse coding
- Sparse classification
- Dictionary learning
- Discriminative dictionary learning

The sparse coding algorithms are implemented using a coordinate descent approach which works much faster than interior point methods using generic solvers for our problem sizes. We use the Eigen library [21] with OpenMP in our sparse coding and dictionary learning implementation. For a typical dimension of $n = 5$ with region covariances, and for reasonable dictionary sizes $K \leq 50$ the sparse coding takes *under 1 millisecond* using our implementation.

VII. CONCLUSIONS AND FUTURE WORK

We have proposed a new formulation for dictionary learning over positive definite matrices, and different approaches to learn these dictionaries given training data. A discriminative variant of dictionary learning for learning dictionaries of multiple classes simultaneously is also presented, for classification and clustering applications. Experimental results demonstrate the performance of the dictionary learning algorithms as well as the applicability to real-world texture data. Finally, a software library has been release comprising C++ binaries for all the positive definite sparse coding and dictionary learning approaches presented here.

ACKNOWLEDGMENT

This material is based upon work supported in part by the National Science Foundation through grants #IIP-0443945, #IIP-0934327, #CNS-1039741, #IIS-1017344, #IIP-1032018, #SMA-1028076, #CNS-1338042, #IIS-1427014, #IIP-1439728, and #CNS-1514626.

[4]Available at http://www.ece.umn.edu/users/sival001/research.html

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2015.2440766, IEEE Transactions on Image Processing

REFERENCES

[1] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Tensor sparse coding for positive definite matrices," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 99, no. PrePrints, p. 1, 2013.

[2] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *ECCV 2006*, 2006, pp. 589–600.

[3] K. Guo, P. Ishwar, and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow," in *Seventh IEEE Intl. Conf. on Advanced Video and Signal Based Surveillance 2010*, Sep. 2010, pp. 188–195.

[4] Z. Wang and B. Vemuri, "Dti segmentation using an information theoretic tensor dissimilarity measure," *IEEE Transactions on Medical Imaging*, vol. 24, no. 10, pp. 1267–1277, Oct. 2005.

[5] S. Sra and A. Cherian, "Generalized dictionary learning for symmetric positive definite matrices with application to nearest neighbor retrieval," in *Proc. 2011 European Conf. on Machine learning and knowledge discovery in databases - Volume Part III*. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 318–332.

[6] G. Pfander, H. Rauhut, and J. Tanner, "Identification of matrices having a sparse representation," *IEEE Trans. Signal Process.*, vol. 56, no. 11, pp. 5376–5388, Nov. 2008.

[7] H. Wang, A. Banerjee, and D. Boley, "Modeling time varying covariance matrices in low dimensions," Dept. of Computer Science and Engineering, University of Minnesota, Technical Report TR-10-017, Aug. 2010.

[8] M. Harandi, C. Sanderson, R. Hartley, and B. Lovell, "Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach," in *Computer Vision ECCV 2012*, ser. Lecture Notes in Computer Science, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds. Springer Berlin Heidelberg, 2012, pp. 216–229.

[9] S. Sra, "Positive definite matrices and the Symmetric Stein Divergence," *ArXiv e-prints*, Oct. 2011.

[10] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[11] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Applications*, vol. 109, no. 3, pp. 475–494, 2001.

[12] A. Beck and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.

[13] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988. [Online]. Available: http://imajna.oxfordjournals.org/content/8/1/141.abstract

[14] L. Vandenberghe, S. Boyd, and S.-P. Wu, "Determinant maximization with linear matrix inequality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 19, pp. 499–533, Apr. 1998.

[15] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis," in *IEEE Conference on Computer Vision and Pattern Recognition, 2008*, Jun. 2008, pp. 1–8.

[16] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," in *IEEE Conference on Computer Vision and Pattern Recognition 2010*, Jun. 2010, pp. 3501–3508.

[17] T. Randen and J. H. Husøy, "Filtering for texture classification: A comparative study," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 21, pp. 291–310, Apr. 1999.

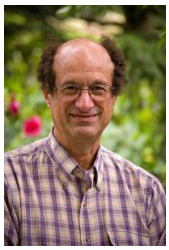
[18] R. Sivalingam, G. Somasundaram, A. Ragipindi, A. Banerjee, V. Morellas, N. Papanikolopoulos, and A. Truskinovsky, "Diagnosing endometrial carcinoma via computer-assisted image analysis," in *Annual Meeting of the United States and Canadian Academy of Pathology (USCAP)*, Mar. 2011.

[19] R. Sivalingam, G. Somasundaram, X. Li, A. Kaplan, J. Henriksen, A. Banerjee, V. Morellas, N. Papanikolopoulos, , and A. Truskinovsky, "Diagnosing adenocarcinoma of the prostate by computer vision methods," in *Annual Meeting of the United States and Canadian Academy of Pathology (USCAP)*, Mar. 2012.

[20] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *17th International Conference on Pattern Recognition*, ser. ICPR '04, 2004, pp. 32–36.

[21] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.

Ravishankar Sivalingam received his Bachelors in Electronics and Communication Engineering from Anna University, India in 2006. He received his M.S in Computer Science, M.S. in Electrical Engineering, and Ph.D. in Electrical Engineering from the University of Minnesota in 2009, 2010, and 2015, respectively. He has been a full-time researcher at 3M Corporate Research Labs since 2012. His primary interests lie in the domains of computer vision, pattern recognition, and machine learning. His current and past projects include 3D object recognition and analysis, aerial image processing (image registration, mosaicing and region annotation), people detection, tracking and crowd counting, generic object detection and application of sparsity and dictionary learning techniques.



Daniel Boley received his A.B. degree Summa Cum Laude in Mathematics and with Distinction in All Subjects from Cornell University in 1974, and his M.S. and Ph.D. degrees in Computer Science from Stanford University in 1976 and 1981, respectively. Since 1981, he has been on the faculty of the Department of Computer Science and Engineering at the University of Minnesota, where he is now a full professor. He has had extended visiting positions at the Los Alamos Scientific Laboratory, the IBM Research Center in Zurich (Switzerland), the Australian National University in Canberra, Stanford University, and the University of Salerno (Italy). Dr. Boley is known for his past work on numerical linear algebra methods for control problems, parallel algorithms, iterative methods for matrix eigenproblems, error correction for floating point computations, inverse problems in linear algebra, as well as his more recent work on computational methods in statistical machine learning and unsupervised document categorization in data mining and bioinformatics. He is an associate editor for the SIAM Journal of Matrix Analysis and has chaired several technical symposia at major conferences. His current interests involve scalable methods for data mining with applications in bioinformatics, computational biology, large collections of text documents (most recently e-mail for the study of social networks), etc.

Vassilios Morellas received his Diploma of Engineering in Mechanical Engineering, from the National Technical University of Athens in 1983. He received his M.S. in Mechanical Engineering from Columbia University in 1988, and Ph.D. in Mechanical Engineering from the University of Minnesota in 1995. Vassilios Morellas' research interests are in the area of geometric image processing, machine learning, robotics and sensor integration to enhance automation of electromechanical systems. He is the Program Director in the department of Computer Science and Engineering and Executive Director of the NSF Center for Safety Security and Rescue. Prior to his current position he was a Senior Principal Research Scientist at Honeywell Laboratories where he developed technologies in the general areas of access control, security and surveillance and biometrics with emphasis on the problem of tracking of people and vehicles across non overlapping cameras. Past research experience also includes work on Intelligent Transportation Systems where he developed innovative technologies to reduce run-off-the-road accidents.

Nikolaos Papanikolopoulos received his Diploma of Engineering in Electrical and Computer Engineering, from the National Technical University of Athens in 1987. He received his M.S. in 1988 and Ph.D. in 1992 in Electrical and Computer Engineering from Carnegie Mellon University. Professor Papanikolopoulos specializes in robotics, computer vision and sensors for transportation uses. His research interests include robotics, sensors for transportation applications, computer vision, and control systems. As the director of the Center for Distributed Robotics and a faculty member of the Artificial Intelligence and Robotic Vision Laboratory, his transportation research has included projects involving vision-based sensing and classification of vehicles, and the recognition of human activity patterns in public areas and while driving.