



This is an electronic reprint of the original article. This reprint may differ from the original in pagination and typographic detail.

Ghorbani Veshki, Farshad; Vorobyov, Sergiy

Efficient approximate online convolutional dictionary learning

Published in: IEEE Transactions on Computational Imaging

DOI: 10.1109/TCI.2023.3340612

Published: 15/12/2023

Document Version Publisher's PDF, also known as Version of record

Published under the following license: CC BY

Please cite the original version:

Ghorbani Veshki, F., & Vorobyov, S. (2023). Efficient approximate online convolutional dictionary learning. *IEEE Transactions on Computational Imaging*, *9*, 1165-1175. https://doi.org/10.1109/TCI.2023.3340612

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Efficient Approximate Online Convolutional Dictionary Learning

Farshad G. Veshki¹⁰, Member, IEEE, and Sergiy A. Vorobyov¹⁰, Fellow, IEEE

Abstract—Most existing convolutional dictionary learning (CDL) algorithms are based on batch learning, where the dictionary filters and the convolutional sparse representations are optimized in an alternating manner using a training dataset. When large training datasets are used, batch CDL algorithms become prohibitively memory-intensive. An online-learning technique is used to reduce the memory requirements of CDL by optimizing the dictionary incrementally after finding the sparse representations of each training sample. Nevertheless, learning large dictionaries using the existing online CDL (OCDL) algorithms remains highly computationally expensive. In this paper, we present a novel approximate OCDL method that incorporates sparse decomposition of the training samples. The resulting optimization problems are addressed using the alternating direction method of multipliers. Extensive experimental evaluations using several image datasets and based on an image fusion task show that the proposed method substantially reduces computational costs while preserving the effectiveness of the state-of-the-art OCDL algorithms.

Index Terms—Convolutional sparse coding, online convolutional dictionary learning.

I. INTRODUCTION

S PARSE representations have achieved significant success and widespread adoption as models for solving inverse problems in signal processing, image processing, and computational imaging [1], [2], [3], [4], [5], [6]. The sparse representation model approximates a signal using a product of a matrix called a dictionary and a vector that only has a few non-zero entries (sparse representation). There are numerous applications where the use of the sparse representation model coupled with a learned dictionary results in remarkably improved performance. A learned dictionary aims to produce sparser representations and more accurate approximations of its domain signals [7], [8], [9]. Dictionary learning has been utilized in diverse computational imaging tasks such as image reconstruction [10], [11], image super-resolution [5], and image fusion [12].

Manuscript received 12 May 2023; revised 29 September 2023; accepted 24 November 2023. Date of publication 7 December 2023; date of current version 20 December 2023. This work was supported by the Academy of Finland under Grant 357715. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. M. Salman Asif. (*Corresponding author: Sergiy A. Vorobyov.*)

Farshad G. Veshki is with the Department of Information and Communications Engineering, Aalto University, 02150 Espoo, Finland, and also with Nokia, 11351 Espoo, Finland (e-mail: farshad.ghorbaniveshki@aalto.fi).

Sergiy A. Vorobyov is with the Department of Information and Communications Engineering, Aalto University, 02150 Espoo, Finland (e-mail: sergiy.vorobyov@aalto.fi).

Digital Object Identifier 10.1109/TCI.2023.3340612

Typically, dictionary learning and sparse approximation are used to extract local patterns and features from high-dimensional signals (such as images). Therefore, a prior decomposition of the original signals into vectorized overlapping blocks is usually required (e.g., patch extraction in image processing). However, relations between neighboring blocks are ignored, which results in multi-valued sparse representations and dictionaries composed of similar (shifted) atoms.

Convolutional sparse coding (CSC) provides a single-valued and shift-invariant model for describing high-dimensional signals [13], [14], [15], [16]. The CSC model has also recently seen numerous applications in deep learning models employed for various image processing tasks [17], [18], [19].

The CSC model replaces the matrix-vector product used in the standard sparse approximation by a sum of convolutions of dictionary filters $\{d_k \in \mathbb{R}^m\}_{k=1}^K$ and convolutional sparse representations (CSRs) $\{x_k \in \mathbb{R}^P\}_{k=1}^K$ (also called sparse feature maps). The convolutional sparse approximation problem can be formulated as follows

$$\underset{\{\boldsymbol{x}_k\}_{k=1}^K}{\text{minimize}} \frac{1}{2} \left\| \sum_{k=1}^K \boldsymbol{d}_k * \boldsymbol{x}_k - \boldsymbol{s} \right\|_2^2 + \lambda \sum_{k=1}^K \|\boldsymbol{x}_k\|_1, \quad (1)$$

where $s \in \mathbb{R}^{P}$ is the signal, $\lambda > 0$ is the regularization parameter that controls the sparsity of the representations, * denotes the convolution operator (here, with "same" padding), and $\|\cdot\|_{1}$ and $\|\cdot\|_{2}$ represent the ℓ_{1} -norm and the Euclidean norm of a vector, respectively.

The convolutional dictionary learning (CDL) problem is typically addressed using a batch approach in which the sparse representations and the dictionary filters are optimized alternately (batch CDL) [16], [20], [21], [22], [23], [24], [25]. The following is the formulation of the dictionary optimization problem over a batch of N training signals $\{s^n \in \mathbb{R}^P\}_{n=1}^N$,

$$\underset{\left\{\boldsymbol{d}_{k}\right\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \sum_{n=1}^{N} \left\| \sum_{k=1}^{K} \boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n} \right\|_{2}^{2} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{d}_{k}\right), \quad (2)$$

where $\Omega(\cdot)$ represents the indicator function of the constraint set for the dictionary filters, that is,

$$\mathbf{\Omega}\left(\boldsymbol{d}\right) = \begin{cases} 0, & \text{if } \|\boldsymbol{d}\|_2 \leq 1\\ \infty, & \text{otherwise.} \end{cases}$$

The existing batch CDL methods require access to all training signals and their CSRs at once. As a result, a memory of the order of NPK is required [26], which can be extremely expensive when using large training datasets, i.e., when $N \gg K$. It is

© 2023 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/ reminded that K is the number of dictionary filters, N is the number of training signals (the batch size), and P is the dimension of the training signals, for example, the number of pixels in an image (usually $P \gg K$ and $P \gg N$). The memory requirement of CDL can be reduced using an online-learning approach, where the dictionary is optimized incrementally after observing each training signal and finding its sparse representations [9]. The online CDL (OCDL) methods are also useful when the training signals are not available all at once, but they are observed gradually over time, such as in streaming data settings. The stateof-the-art OCDL methods have achieved memory requirements of the order of K^2P [27], [28], which is independent of the number of training signals. Nevertheless, when learning large dictionaries or using high-dimensional signals, these methods can still incur excessive computational costs.

This paper presents a novel approximate OCDL method that significantly improves the computational efficiency of the stateof-the-art algorithms while providing competitive performance compared to the existing methods.¹ As a result, we propose a method that requires a memory of the order of *KP* only. More specifically, our method approximates the OCDL problem by minimizing an upper bound of the objective function, where the dictionary optimization problem is decentralized with respect to the convolutional filters. We then solve the resulting optimization problem using the *alternating direction method of multipliers* (ADMM). MATLAB implementations of the proposed algorithms are available at *https://github.com/FarshadGVeshki/ Approximate-Online-Convolutional-Dictionary-Learning*.

The rest of the paper is organized as follows. Section II briefly reviews CDL in the Fourier domain. The proposed CDL method and derivation of the algorithms are presented in detail in Section III. Thorough experimental evaluation results in terms of convergence properties and reconstruction accuracy using multiple image datasets of varying sizes and in the context of a multimodal image fusion task are presented in Section IV. The conclusions are provided in Section V.

II. OCDL IN THE FOURIER DOMAIN

Most efficient CDL methods are based on the Fourier transform [16], [25], [27], [28]. In the frequency (Fourier) domain, problem (2) is equivalent to

$$\underset{\left\{\boldsymbol{d}_{k}\right\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2NP} \sum_{n=1}^{N} \left\| \sum_{k=1}^{K} \hat{\boldsymbol{d}}_{k} \odot \hat{\boldsymbol{x}}_{k}^{n} - \hat{\boldsymbol{s}}^{n} \right\|_{2}^{2} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{d}_{k}\right),$$
(3)

where $(\hat{\cdot})$ and \odot denote the discrete Fourier transform (DFT) and the elementwise multiplication operator, respectively. The filters $\{d_k\}_{k=1}^K$ are zero-padded prior to DFT, so that $\{\hat{d}_k\}_{k=1}^K$ are of the same size as the CSRs.

Defining
$$\boldsymbol{\delta}_p \triangleq [\boldsymbol{d}_1(p), \dots, \boldsymbol{d}_K(p)]^T$$
 and $\boldsymbol{\chi}_p^n \triangleq [\boldsymbol{\hat{x}}_1^n(p), \dots, \boldsymbol{\hat{x}}_K^n(p)]^T$, problem (3) can be rewritten as

$$\underset{\left\{\boldsymbol{d}_{k}\right\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2NP} \sum_{p=1}^{P} \sum_{n=1}^{N} \left\| (\boldsymbol{\chi}_{p}^{n})^{T} \boldsymbol{\delta}_{p} - \hat{\boldsymbol{s}}^{n}(p) \right\|_{2}^{2} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{d}_{k}\right),$$

$$(4)$$

where $(\cdot)^T$ is the transpose operator. The most efficient solutions to problem (4) (the batch CDL problem) have been proposed based on ADMM, and the *fast iterative shrinkage-thresholding algorithm* (FISTA) [25], [26]. The complexities of these algorithms are of $\mathcal{O}(KNP)$ and they require memory of the order of KNP. As a result, when the training dataset is large, batch CDL becomes excessively computationally demanding in practice.

OCDL alleviates the problem of large required memory by storing sufficient statistics of the training signals and their CSRs in compact history arrays. An online reformulation of problem (4) can be written as

$$\underset{\left\{\boldsymbol{d}_{k}\right\}_{k=1}^{K}}{\text{minimize}} \ \frac{1}{2} \sum_{p=1}^{P} \boldsymbol{\delta}_{p}^{H} \boldsymbol{A}_{p}^{N} \boldsymbol{\delta}_{p} - \sum_{p=1}^{P} \boldsymbol{\delta}_{p}^{T} \boldsymbol{b}_{p}^{N} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{d}_{k}\right), \quad (5)$$

where $(\cdot)^H$ is the Hermitian transpose operator, and the history arrays $\mathbf{A}_p^N \in \mathbb{R}^{K \times K}$ and $\mathbf{b}_p^N \in \mathbb{R}^K$, $p = 1, \ldots, P$, are defined as

$$\boldsymbol{A}_{p}^{N} \triangleq \frac{1}{NP} \sum_{n=1}^{N} (\boldsymbol{\chi}_{p}^{n})^{*} (\boldsymbol{\chi}_{p}^{n})^{T}, \quad \boldsymbol{b}_{p}^{N} \triangleq \frac{1}{NP} \sum_{n=1}^{N} \hat{\boldsymbol{s}}^{n} (p)^{*} \boldsymbol{\chi}_{p}^{n},$$
(6)

with $(\cdot)^*$ standing for the element-wise complex conjugate of an array vector. After observing each training signal and finding its sparse representations, the history arrays are recalculated incrementally using the following formulas

$$\boldsymbol{A}_{p}^{N} = \frac{1}{NP} (\boldsymbol{\chi}_{p}^{N})^{*} (\boldsymbol{\chi}_{p}^{N})^{T} + \frac{N-1}{N} \boldsymbol{A}_{p}^{N-1}, \ p = 1, \dots, P,$$
$$\boldsymbol{b}_{p}^{N} = \frac{1}{NP} \hat{\boldsymbol{s}}^{N} (p)^{*} \boldsymbol{\chi}_{p}^{N} + \frac{N-1}{N} \boldsymbol{b}_{p}^{N-1}, \ p = 1, \dots, P.$$
(7)

The history arrays are initialized using zero arrays. In OCDL, the dictionary is optimized by solving problem (5) only after the updated history arrays are available. As a result, a memory requirement of K^2P and a complexity of $\mathcal{O}(K^2NP)$ are achieved [27], [28].

III. THE PROPOSED METHOD

In the proposed method, the training signals are approximated in a distributed manner using N distinct dictionaries $\{c_k^n \in \mathbb{R}^m\}_{k=1}^K$. A fusion of the separately optimized dictionaries based on the respective CSRs is used to calculate the dictionary $\{d_k\}_{k=1}^K$. Specifically, the quadratic term in CDL problem (2) is approximated using the following upper-bound estimate

$$\begin{split} &\sum_{n=1}^{N} \left\| \sum_{k=1}^{K} \boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n} \right\|_{2}^{2} \\ &= \sum_{n=1}^{N} \left\| \sum_{k=1}^{K} \boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \sum_{k=1}^{K} \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} + \sum_{k=1}^{K} \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n} \right\|_{2}^{2} \end{split}$$

¹The preliminary findings and results of the proposed method have been reported in [29]. This paper provides a comprehensive presentation, encompassing in-depth derivations, additional algorithmic developments, and a more extensive array of experimental results including the application demonstrating the utility of OCDL in image fusion.

$$\leq \sum_{n=1}^{N} \sum_{k=1}^{K} \|\boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n}\|_{2}^{2} + \sum_{n=1}^{N} \left\|\sum_{k=1}^{K} \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n}\right\|_{2}^{2},$$
(8)

where the inequality is due to the triangle inequality. Accordingly, the proposed approximate CDL problem is formulated as

$$\underset{\{\boldsymbol{d}_{k}\}_{k=1}^{K}, \\ \{\boldsymbol{c}_{k}^{n}\}_{k=1}^{K}\}_{n=1}^{N}}{\underset{1}{\sum}} \sum_{n=1}^{N} \sum_{k=1}^{K} \|\boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n}\|_{2}^{2} + \sum_{k=1}^{K} \Omega\left(\boldsymbol{d}_{k}\right)$$
$$+ \frac{1}{2N} \sum_{n=1}^{N} \left\| \sum_{k=1}^{K} \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n} \right\|_{2}^{2} + \sum_{n=1}^{N} \sum_{k=1}^{K} \Omega\left(\boldsymbol{c}_{k}^{n}\right).$$
(9)

In the following, two ADMM-based online methods for addressing (9) are presented. The first algorithm uses a standard approach for optimization of $\{d_k\}_{k=1}^K$ and $\{c_k^N\}_{k=1}^K$, while the second algorithm incorporates pragmatic modifications to the first algorithm to improve the effectiveness of the proposed approximation method and lower computational costs.

A. Algorithm 1

Optimization problem (9) is jointly convex with respect to $\{d_k\}_{k=1}^K$ and $\{\{c_k^n\}_{k=1}^K\}_{n=1}^N$. Thus, using the OCDL framework, problem (9) can be addressed for the joint optimization variables $\{c_k^N, d_k\}_{k=1}^K$ after observing the *N*th training signal s^N and obtaining its CSRs $\{x_k^N\}_{k=1}^K$. Compact history arrays are used to store sufficient statistics of $\{\{c_k^n\}_{k=1}^K\}_{n=1}^{N-1}$ and $\{\{x_k^n\}_{k=1}^K\}_{n=1}^{N-1}$. The following ADMM formulation is used to solve (9) for

 $\{oldsymbol{c}_k^N,oldsymbol{d}_k\}_{k=1}^K$

where $\{f_k^N, g_k\}_{k=1}^K$ are the (joint) ADMM auxiliary variables. The ADMM iterations consist of the following three steps.

The $\{f, g\}$ -update step: In this step the auxiliary variables $\{\boldsymbol{f}_{k}^{N},\boldsymbol{g}_{k}\}_{k=1}^{K}$ are updated as

$$\left(\{\boldsymbol{f}_{k}^{N}\}_{k=1}^{K} \right)^{t+1} = \operatorname{argmin}_{\{\boldsymbol{f}_{k}^{N}\}_{k=1}^{K}} \frac{1}{2N} \sum_{k=1}^{K} \left\| \boldsymbol{f}_{k}^{N} * \boldsymbol{x}_{k}^{N} - \boldsymbol{z}_{k}^{N} \right\|_{2}^{2}$$

$$+ \frac{1}{2N} \left\| \sum_{k=1}^{K} \boldsymbol{f}_{k}^{N} * \boldsymbol{x}_{k}^{N} - \boldsymbol{s}^{N} \right\|_{2}^{2}$$

$$+ \frac{\rho}{2} \sum_{k=1}^{K} \left\| \boldsymbol{f}_{k}^{N} - (\boldsymbol{c}_{k}^{N})^{t} + (\boldsymbol{u}_{k})^{t} \right\|_{2}^{2}, \quad (11)$$

$$\left(\{ \boldsymbol{g}_k \}_{k=1}^K \right)^{t+1} = \underset{\{ \boldsymbol{g}_k \}_{k=1}^K}{\operatorname{argmin}} \frac{1}{2N} \sum_{n=1}^N \sum_{k=1}^K \| \boldsymbol{g}_k * \boldsymbol{x}_k^n - \boldsymbol{t}_k^n \|_2^2 + \frac{\rho}{2} \sum_{k=1}^K \| \boldsymbol{g}_k - (\boldsymbol{d}_k)^t + (\boldsymbol{v}_k)^t \|_2^2,$$
 (12)

where $\{\boldsymbol{u}_k, \boldsymbol{v}_k\}_{k=1}^K$ are the scaled Lagrangian variables, $\rho > 0$ is the ADMM penalty parameter, $\boldsymbol{z}_k^N \triangleq (\boldsymbol{g}_k)^t * \boldsymbol{x}_k^N$ and $\boldsymbol{t}_k^n \triangleq$ $(\boldsymbol{f}_k^n)^{t+1}*\boldsymbol{x}_k^n.$

The $\{c, d\}$ -update step In this step $\{c_k^N, d_k\}_{k=1}^K$ is updated

$$\left(\{\boldsymbol{c}_{k}^{N}\}_{k=1}^{K}\right)^{t+1} = \underset{\{\boldsymbol{c}_{k}^{N}\}_{k=1}^{K}}{\operatorname{argmin}} \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{c}_{k}^{N}\right)$$

$$+ \frac{\rho}{2} \sum_{k=1}^{K} \left\| (\boldsymbol{f}_{k}^{N})^{t+1} - \boldsymbol{c}_{k}^{N} + (\boldsymbol{u}_{k})^{t} \right\|_{2}^{2}, (13)$$

$$\left(\{ \boldsymbol{d}_k \}_{k=1}^K \right)^{t+1} = \operatorname*{argmin}_{\{ \boldsymbol{d}_k \}_{k=1}^K} \sum_{k=1}^K \boldsymbol{\Omega} \left(\boldsymbol{d}_k \right)$$

+ $\frac{\rho}{2} \sum_{k=1}^K \left\| (\boldsymbol{g}_k)^{t+1} - \boldsymbol{d}_k + (\boldsymbol{v}_k)^t \right\|_2^2.$ (14)

Updating the scaled Lagrangian parameters: Finally, the scaled Lagrangian variables are updates as

$$(\boldsymbol{u}_{k})^{t+1} = (\boldsymbol{f}_{k}^{N})^{t+1} - (\boldsymbol{c}_{k}^{N})^{t+1} + (\boldsymbol{u}_{k})^{t}, \quad k = 1, \dots, K,$$
$$(\boldsymbol{v}_{k})^{t+1} = (\boldsymbol{g}_{k})^{t+1} - (\boldsymbol{d}_{k})^{t+1} + (\boldsymbol{v}_{k})^{t}, \quad k = 1, \dots, K.$$
(15)

The {c, d}-update step involves projecting $(f_k^N)^{t+1} + (u_k)^t$ (in (13)) and $(g_k)^{t+1} + (v_k)^t$ (in (14)) onto the constraint set. First, the entries outside the support (\mathbb{R}^m) are mapped to zero (recall that the filters are zero-padded), followed by projection onto the unit ℓ_2 -norm ball.

In the $\{f, g\}$ -update step, solving problem (11) is equivalent to solving the following optimization problem

$$\begin{array}{l} \underset{\{\boldsymbol{f}_{k}^{N}\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \sum_{k=1}^{K} \left\| \hat{\boldsymbol{f}}_{k}^{N} \odot \hat{\boldsymbol{x}}_{k}^{N} - \hat{\boldsymbol{z}}_{k}^{N} \right\|_{2}^{2} \\ + \frac{1}{2N} \left\| \sum_{k=1}^{K} \hat{\boldsymbol{f}}_{k}^{N} \odot \hat{\boldsymbol{x}}_{k}^{N} - \hat{\boldsymbol{s}}^{N} \right\|_{2}^{2} + \frac{\rho}{2} \sum_{k=1}^{K} \left\| \hat{\boldsymbol{f}}_{k}^{N} - \hat{\boldsymbol{q}}_{k} \right\|_{2}^{2}, \ (16) \end{array}$$

where $\boldsymbol{q}_{k} \triangleq (\boldsymbol{c}_{k}^{N})^{t} - (\boldsymbol{u}_{k})^{t}$. By equating the derivative of the objective in (16) to zero and using the Sherman-Morrison (SM) formula, the solution to the f-update step is found as

$$\begin{pmatrix} \hat{\boldsymbol{f}}_{k}^{N}(p) \end{pmatrix}^{t+1} = \left(a_{p}^{k} + \frac{(a_{p}^{k})^{2} |\hat{\boldsymbol{x}}_{k}^{N}(p)|^{2}}{1 + \sum_{k=1}^{K} a_{p}^{k} |\hat{\boldsymbol{x}}_{k}^{N}(p)|^{2}} \right) \\ \times \left((\hat{\boldsymbol{x}}_{k}^{N}(p))^{*} \left(\hat{\boldsymbol{z}}_{k}^{N}(p) + \hat{\boldsymbol{s}}^{N}(p) \right) + N \rho \hat{\boldsymbol{q}}_{k}(p) \right),$$
(17)

Algorithm 1: OCDL Method Proposed in Section III-A.

Input: Training signals $\{s^n \in \mathbb{R}^P\}_{n=1}^N$, initial dictionary $\{d_k^0 \in \mathbb{R}^m\}_{k=1}^K$, sparsity regularization parameter λ ; *Initialisation* : History arrays $\alpha_k^0 \in \mathbb{R}^P$ and $\beta_k^0 \in \mathbb{R}^P$, $k = 1, \ldots, K$ as zero arrays, $\{d_k\}_{k=1}^K = \{d_k^0\}_{k=1}^K$; 1: **for** n = 1 to N **do**

- 2: Find $\{\boldsymbol{x}_k^n\}_{k=1}^K$ for \boldsymbol{s}^n using $\{\boldsymbol{d}_k\}_{k=1}^K$ and λ by solving (1);
- 3: Calculate $\{\alpha_k^n\}_{k=1}^K$ using (21);
- 4: Optimize $\{c_k^n, d_k\}_{k=1}^K$ using the ADMM-based method in Section III-A (recalculate $\{\beta_k^n\}_{k=1}^K$ using (22) in every iteration);

5: end for

6: return Learned convolutional dictionary $\{d_k\}_{k=1}^K$.

where $a_p^k \triangleq (|\hat{x}_k^N(p)|^2 + N\rho)^{-1}$. Using precalculated values of $\sum_{k=1}^{K} a_p^k |\hat{x}_k^N(p)|^2$, the **f**-update step can be carried out with the complexity of $\mathcal{O}(KP)$ using (17).

Problem (12) can be addressed via solving the following optimization problem

$$\underset{\{\boldsymbol{g}_{k}\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \sum_{n=1}^{N} \sum_{k=1}^{K} \left\| \hat{\boldsymbol{g}}_{k} \odot \hat{\boldsymbol{x}}_{k}^{n} - \hat{\boldsymbol{t}}_{k}^{n} \right\|_{2}^{2} + \frac{\rho}{2} \sum_{k=1}^{K} \left\| \hat{\boldsymbol{g}}_{k} - \hat{\boldsymbol{w}}_{k} \right\|_{2}^{2},$$
(18)

where $\boldsymbol{w}_k \triangleq (\boldsymbol{d}_k)^t - (\boldsymbol{v}_k)^t$.

The solution to (18) can be found as

$$(\hat{\boldsymbol{g}}_{k}(p))^{t+1} = \frac{\boldsymbol{\beta}_{k}^{N}(p) + \hat{\boldsymbol{w}}_{k}(p)}{\boldsymbol{\alpha}_{k}^{N} + \boldsymbol{\rho}}, \ p = 1, \dots, P, \ k = 1, \dots, K,$$
(19)

where history arrays $\alpha_k^N \in \mathbb{R}^P$ and $\beta_k^N \in \mathbb{R}^P$, k = 1, ..., K, are defined as

$$\boldsymbol{\alpha}_{k}^{N} \triangleq \frac{1}{N} \sum_{n=1}^{N} (\hat{\boldsymbol{x}}_{k}^{n})^{*} \odot \hat{\boldsymbol{x}}_{k}^{n}, \ \boldsymbol{\beta}_{k}^{N} \triangleq \frac{1}{N} \sum_{n=1}^{N} (\hat{\boldsymbol{x}}_{k}^{n})^{*} \odot \hat{\boldsymbol{t}}_{k}^{n}.$$
(20)

The history arrays are incrementally updated using

$$\boldsymbol{\alpha}_{k}^{N} = \frac{N-1}{N} \boldsymbol{\alpha}_{k}^{N-1} + \frac{1}{N} (\hat{\boldsymbol{x}}_{k}^{N})^{*} \odot \hat{\boldsymbol{x}}_{k}^{N}, \qquad (21)$$

$$\beta_{k}^{N} = \frac{N-1}{N} \beta_{k}^{N-1} + \frac{1}{N} (\hat{x}_{k}^{N})^{*} \odot \hat{t}_{k}^{N}.$$
(22)

Algorithm 1 summarizes the main steps of the proposed approximate OCDL algorithm detailed in this section. Unitnorm Gaussian distributed random arrays can be used as initial dictionary $\{d_k^0\}_{k=1}^K$. At the first iteration, dictionary $\{d_k\}_{k=1}^K$ can be used to initialize $\{c_k^n\}_{k=1}^K$ and $\{g_k\}_{k=1}^K$. Note that, before each iteration of the ADMM algorithm, $\{\beta_k^n\}_{k=1}^K$ needs to be recalculated using (22) based on the latest values of $\{f_k^n\}_{k=1}^K$.

B. Algorithm 2

To improve the performance of the proposed OCDL algorithm, dictionary optimization can be performed *exactly* for the latest observed signal s^N , while the proposed approximation method is used for $\{s^n\}_{n=1}^{N-1}$. Thus, the modified approximate

CDL problem is now formulated as

$$\begin{array}{l} \underset{\{\boldsymbol{d}_{k}\}_{k=1}^{K}, \\ \{\boldsymbol{c}_{k}^{n}\}_{k=1}^{K}\}_{n=1}^{N}}{\underset{\boldsymbol{c}_{k}^{n}}{\sum_{k=1}^{N}}} \frac{1}{2N} \left\| \sum_{k=1}^{K} \boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{N} - \boldsymbol{s}^{N} \right\|_{2}^{2} \\ + \frac{1}{2N} \sum_{n=1}^{N-1} \sum_{k=1}^{K} \left\| \boldsymbol{d}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} \right\|_{2}^{2} + \sum_{k=1}^{K} \Omega\left(\boldsymbol{d}_{k}\right) \\ + \frac{1}{2N} \sum_{n=1}^{N-1} \left\| \sum_{k=1}^{K} \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{s}^{n} \right\|_{2}^{2} + \sum_{n=1}^{N} \sum_{k=1}^{K} \Omega\left(\boldsymbol{c}_{k}^{n}\right). \end{aligned}$$

$$(23)$$

The alternating procedure for addressing (23) consists of the following steps.

1) Optimization of $\{d_k\}_{k=1}^K$: Solving (23) with respect to $\{d_k\}_{k=1}^K$ can be addressed using the following ADMM formulation

$$\begin{array}{l} \underset{\{\boldsymbol{d}_{k}\}_{k=1}^{K},\{\boldsymbol{g}_{k}\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \left\| \sum_{k=1}^{K} \boldsymbol{g}_{k} \ast \boldsymbol{x}_{k}^{N} - \boldsymbol{s}^{N} \right\|_{2}^{2} \\ + \frac{1}{2N} \sum_{n=1}^{N-1} \sum_{k=1}^{K} \left\| \boldsymbol{g}_{k} \ast \boldsymbol{x}_{k}^{n} - \boldsymbol{r}_{k}^{n} \right\|_{2}^{2} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{d}_{k}\right) \\ \text{s.t.} \quad \boldsymbol{g}_{k} = \boldsymbol{d}_{k}, \ k = 1, \dots, K \end{array}$$
(24)

where $\boldsymbol{r}_{k}^{n} \triangleq \boldsymbol{c}_{k}^{n} \ast \boldsymbol{x}_{k}^{n}$.

The ADMM iterations consist of the following steps:

- the *g*-update step: a convolutional least-squares fitting problem;
- the *d*-update step: projection on the constraint set (similar to (14));
- 3) updating the Lagrangian multipliers (similar to (15)).

The g-update step requires solving the optimization problem in the form of

$$\begin{array}{l} \underset{\{\boldsymbol{g}_{k}\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \left\| \sum_{k=1}^{K} \hat{\boldsymbol{g}}_{k} \odot \hat{\boldsymbol{x}}_{k}^{N} - \hat{\boldsymbol{s}}^{N} \right\|_{2}^{2} \\ + \frac{1}{2N} \sum_{n=1}^{N-1} \sum_{k=1}^{K} \| \hat{\boldsymbol{g}}_{k} \odot \hat{\boldsymbol{x}}_{k}^{n} - \hat{\boldsymbol{r}}_{k}^{n} \|_{2}^{2} + \frac{\rho}{2} \sum_{k=1}^{K} \| \hat{\boldsymbol{g}}_{k} - \hat{\boldsymbol{e}}_{k} \|_{2}^{2} . \end{array}$$

$$(25)$$

Equating the derivative to zero and using the SM formula, optimization problem (25) can be solved as

$$(\hat{\boldsymbol{g}}_{k}(p))^{t+1} = \left(b_{p}^{k} + \frac{(b_{p}^{k})^{2} |\hat{\boldsymbol{x}}_{k}^{N}(p)|^{2}}{N + \sum_{k=1}^{K} b_{p}^{k} |\hat{\boldsymbol{x}}_{k}^{N}(p)|^{2}} \right) \\ \times \left(\frac{1}{N} (\hat{\boldsymbol{x}}_{k}^{N}(p))^{*} \hat{\boldsymbol{s}}^{N}(p) + \tilde{\boldsymbol{\beta}}_{k}^{N-1}(p) + \rho \hat{\boldsymbol{e}}_{k}(p) \right),$$
(26)

Algorithm 2: OCDL Method Proposed in Section III-B.

Input: Training signals $\{s^n \in \mathbb{R}^P\}_{n=1}^N$, initial dictionary $\{d_k^0 \in \mathbb{R}^m\}_{k=1}^K$, sparsity regularization parameter λ ; *Initialisation*: History arrays $\tilde{\alpha}_k^0 \in \mathbb{R}^P$ and $\tilde{\beta}_k^0 \in \mathbb{R}^P$, $k = 1, \ldots, K$ as zero arrays, $\{d_k\}_{k=1}^K = \{d_k^0\}_{k=1}^K$; 1: **for** n = 1 to N **do** 2: Find $\{x_k^n\}_{k=1}^K$ for s^n and $\{d_k\}_{k=1}^K$ by solving (1); 3: Optimize $\{d_k\}_{k=1}^K$ as in Section III-B1; 4: Optimize $\{c_k^n\}_{k=1}^K$ and $\{\tilde{\beta}_k^n\}_{k=1}^K$ using (28), (29); 5: Calculate $\{\tilde{\alpha}_k^n\}_{k=1}^K$ and $\{\tilde{\beta}_k^n\}_{k=1}^K$ using (28), (29); 6: **end for** 7: **return** learned convolutional dictionary $\{d_k\}_{k=1}^K$.

with $b_p^k \triangleq (\tilde{\alpha}_k^{N-1}(p) + \rho)^{-1}$, where history arrays $\tilde{\alpha}_k^N \in \mathbb{R}^P$ and $\tilde{\beta}_k^N \in \mathbb{R}^P$, $k = 1, \dots, K$, are defined as

$$\tilde{\boldsymbol{\alpha}}_{k}^{N} \triangleq \frac{1}{N+1} \sum_{n=1}^{N} (\hat{\boldsymbol{x}}_{k}^{n})^{*} \odot \hat{\boldsymbol{x}}_{k}^{n}, \ \tilde{\boldsymbol{\beta}}_{k}^{N} \triangleq \frac{1}{N+1} \sum_{n=1}^{N} (\hat{\boldsymbol{x}}_{k}^{n})^{*} \odot \hat{\boldsymbol{r}}_{k}^{n}$$
(27)

The incremental update rules for $\tilde{\alpha}_k^N$ and $\tilde{\beta}_k^N$ can be found as

$$\tilde{\boldsymbol{\alpha}}_{k}^{N} = \frac{N}{N+1} \tilde{\boldsymbol{\alpha}}_{k}^{N-1} + \frac{1}{N+1} (\hat{\boldsymbol{x}}_{k}^{N})^{*} \odot \hat{\boldsymbol{x}}_{k}^{N}, \qquad (28)$$

$$\tilde{\beta}_{k}^{N} = \frac{N}{N+1}\tilde{\beta}_{k}^{N-1} + \frac{1}{N+1}(\hat{x}_{k}^{N})^{*} \odot \hat{r}_{k}^{N}.$$
 (29)

The *g*-update (26) can be performed with the complexity of $\mathcal{O}(KP)$ using precalculated values of $\sum_{k=1}^{K} b_p^k |\hat{x}_k^N(p)|^2$.

2) Optimization of $\{c_k^N\}_{k=1}^K$: In the modified algorithm, dictionary $\{c_k^N\}_{k=1}^K$ is optimized only to provide a more accurate approximation of s^N (in comparison with the approximation provided using $\{d_k\}_{k=1}^K$). It means that the second quadratic term in (23) is ignored in the step of $\{c_k^N\}_{k=1}^K$ optimization. Here we rely on the fact that CSRs $\{x_k^N\}_{k=1}^K$ are direct products of $\{d_k\}_{k=1}^K$. As a result, considering that the approximation is based on $\{x_k^N\}_{k=1}^K$, the resulting $\{c_k^N\}_{k=1}^K$ cannot unfavorably deviate from $\{d_k\}_{k=1}^K$. Problem (23), which needs to be solved now for $\{c_k^N\}_{k=1}^K$ only, is then reduced to the following optimization problem

$$\underset{\left\{\boldsymbol{c}_{k}^{N}\right\}_{k=1}^{K}}{\text{minimize}} \frac{1}{2N} \left\| \sum_{k=1}^{K} \boldsymbol{c}_{k}^{N} \ast \boldsymbol{x}_{k}^{N} - \boldsymbol{s}^{N} \right\|_{2}^{2} + \sum_{k=1}^{K} \boldsymbol{\Omega}\left(\boldsymbol{c}_{k}^{N}\right), \quad (30)$$

which is a CDL problem involving a single training signal, and can be addressed using the existing CDL methods (e.g., [25]).

The main steps of the presented approximate OCDL algorithm are summarized in Algorithm 2. Optimization of dictionaries $\{d_k\}_{k=1}^K$ and $\{c_k^n\}_{k=1}^K$ (lines 3 and 4) can be initialized using the existing $\{d_k\}_{k=1}^K$.

C. Memory Requirements and Computational Complexity

The largest arrays used in the proposed algorithms are of size KP. The most computationally expensive steps of performing updates (17) and (26) both have a complexity of $\mathcal{O}(KP)$, which is slightly dominated by the complexity of DFT that is of

 $\mathcal{O}(KP\log(P))$ when performed using *Fast Fourier Transform*. Thus, the computational complexity of the proposed algorithm is of the order of KP sequentially performed N times (once for each signal in the training dataset).

IV. EXPERIMENTAL RESULTS

A. Compared Methods

The performance of the proposed algorithms is benchmarked against the following state-of-the-art OCDL methods:

OCSC The ADMM-based OCDL method of [27], which uses the iterative Sherman-Morrison formula for updating the history arrays;

FISTA The FISTA-based OCDL method of [28] that uses gradient calculated in the Fourier domain.

In addition, we compare the OCDL methods to the following batch-CDL algorithm,

ADMM-cns The batch-CDL method of [25] that is based on consensus-ADMM.

Algorithms 1 and 2 are referred to as "Proposed-1" and "Proposed-2", respectively.

B. Datasets

The experiments are conducted using the following 6 image datasets:

Fruit and **City** Two small datasets, each composed of 10 images of size 100×100 . These datasets are typically used as benchmarks for CSC and CDL [20], [21], [27];

SIPI A dataset composed of 20 training images and 5 test images collected from the UCS-SIPI image database (*http:* //sipi.usc.edu/database/). The original images are resized to dimensions of 256×256 pixels.

Flicker A dataset composed of 40 training images and 5 test images collected from the MIRFLICKR-1 M image dataset (*https://press.liacs.nl/mirflickr/mirdownload.html*). The original images are cropped and resized to dimensions of 256×256 pixels.

Flicker-large A dataset composed of 1000 training images and 50 test images collected from the MIRFLICKR-1 M image dataset. The original images are cropped and resized to dimensions of 256×256 pixels.

RGB-NIR A dataset composed of 10 pairs of multimodal visible-light (VL) and near-infrared (NIR) images size 382×586 taken from the EPFL RGB-NIR scene dataset (*https://www.epfl.ch/labs/ivrl/research/downloads/rgb-nir-scene-dataset/*). The RGB-NIR dataset is used for the VL-NIR imaging experiment presented in Section IV-I.

The initial images are transformed into greyscale and the 8-bit pixel values are normalized to a range of 0-1 by dividing by 255. As the CSC model is not capable of effectively handling low-frequency signals, it is a common practice to use high-pass filtered images for CDL [16], [26], [28]. In the experiments, the low-frequency components of all images are eliminated using the *lowpass* function of the SPORCO toolbox [30] with a regularization parameter of 5.



Fig. 1. Datsets Fruit (first row) and City (second row).

TABLE I $\lambda_{max} \text{ Values Used for Different Datasets}$



Fig. 2. Comparison of training objective values and training times obtained using all methods compared for datasets *Fruit* (top) and *City* (bottom).

C. Implementation Details

The proposed algorithms employ the unconstrained convolutional sparse approximation method of [25]. We use publicly available online implementations of the compared methods, as provided by their respective authors. In all ADMM-based algorithms (both sparse approximation and dictionary learning) the maximum number of iterations² is set to 300, and stopping criteria discussed in [31, Subsection 3.3] with absolute and relative tolerance values of 10^{-4} are used. We use dictionary filters of size 8×8 in all experiments.

All ADMM-based algorithms except OCSC use ADMM extensions *over-relaxation* [31, Subsection 3.4.3] and *varying penalty parameter* [31, Subsection 3.4.1] with initial penalty parameter $\rho = 10$ (the same parameters are used in all methods).

 $^2\mathrm{A}$ single ADMM iteration corresponds to the execution of the three ADMM steps.



Fig. 3. Comparison of the average number of (ADMM and FISTA) iterations of the OCDL algorithms compared for datasets *Fruit* (left) and *City* (right).

The OCSC method incorporates the ADMM penalty parameter ρ in the history arrays. Thus, this method cannot use *varying penalty parameter* extension. For methods OCSC and FISTA, we use the default parameters set by the authors of the paper (the stopping criteria of the OCSC method are modified to be uniform with other ADMM-based algorithms compared).

In all experiments, we use $\lambda = 0.1\lambda_{\max}$, where λ_{\max} is the smallest value that results in all-zero sparse representations and can be obtained using ℓ_{∞} -norm of the gradient of the objective of convolutional sparse approximation problem (1) at $\{x_k\}_{k=1}^K = 0$. Here, the value of λ_{\max} is calculated only once using the first image in the training datasets. Table I reports the values of λ_{\max} calculated for different training datasets used in the experiments.

All algorithms are implemented using MATLAB. All experiments are performed using a PC equipped with an Intel(R) Core(TM) i5-8365 U 1.60 GHz CPU and 16 GB memory.

D. Comparison Criteria

The effectiveness of the CDL algorithms is typically evaluated based on the objective values of the convolutional sparse approximation problem (1) averaged over the entire test datasets [27], [28], [32]. A lower objective value indicates a better performance. For the small datasets *Fruit* and *City*, since there is no test data, the average training objective values are reported to compare the effectiveness of the optimization algorithms [20]. Using visualized dictionary filters, the OCDL algorithms are evaluated for their ability to extract (learn) visual features. The efficiency of the algorithms is measured using the training times.

E. Small Datasets Fruit and City

Fig. 1 shows the images in the small datasets *Fruit* and *City*. Tables II and III report the average training objective values and



Fig. 4. Ten randomly selected images from datasets SIPI (top) and Flickr (bottom).

TABLE II Average Training Objective Values and Training Times Obtained Using the Methods Compared for Dataset *Fruit*

	Objective	Training Time (s)
Initial dictionary	19.5422	-
FISTA [28]	16.0159	167
OCSC [27]	14.5529	530
Proposed-1	16.6867	39
Proposed-2	14.3059	33
ADMM-cns (batch) [25]	11.8088	122

TABLE III Average Training Objective Values and Training Times Obtained Using the Methods Compared for Dataset *City*

	Objective	Training Time (s)
Initial dictionary	33.9411	-
FISTA [28]	28.6235	190
OCSC [27]	24.5472	462
Proposed-1	30.1463	42
Proposed-2	25.2740	32
ADMM-cns (batch) [25]	18.9411	153

the training times obtained using the methods tested for these two datasets. To facilitate comparison, the results are presented as bar plots in Fig. 2. The experiments based on datasets *Fruit* and *City* are performed using dictionary size K = 64.

As can be observed, the ADMM-cns batch CDL algorithm yields the lowest objective function values. However, this method is not suitable for large datasets as mentioned earlier. The proposed methods produce objective values that are comparable to other OCDL algorithms tested. In particular, Algorithm 2 (Proposed-2) results in the smallest objective for the *Fruit* dataset among all OCDL algorithms. For the *City* dataset, the OCSC method has the lowest objective compared to other OCDL methods (slightly better than that of Proposed-2), but shows a longer training time. As shown in Tables II and III, the proposed algorithms result in substantially shorter training times, especially Algorithm 2, which is noticeably faster than Algorithm 1.

Fig. 3 presents a comparison of the average number of (ADMM and FISTA) iterations utilized for performing dictionary learning on datasets *Fruit* and *City* using the OCDL methods compared. Specifically, for Proposed-2 method, we provide the cumulative count of ADMM iterations used for optimizing the dictionaries $\{d_k\}_{k=1}^K$ and $\{c_k^n\}_{k=1}^K$ (lines 3 and 4 in Algorithm 2). As previously explained in Section IV-C, in the OCSC method, the ADMM penalty parameter ρ is incorporated into the history arrays. Consequently, this method is unable



Fig. 5. Comparison of test objective values and training times obtained using all methods compared for datasets *SIPI* (top) and *Flickr* (bottom).

to take advantage of the varying penalty parameter extension of ADMM, which may account for its comparatively slower convergence when compared to the other OCDL methods. The average number of iterations used by the proposed methods and the FISTA method falls within a similar range. However, since the proposed methods are significantly computationally more efficient, they result in substantially shorter training times.

F. Datasets SIPI and Flickr

Fig. 4 depicts 10 images randomly selected from the *SIPI* and *Flickr* datasets. The experiments for *SIPI* dataset are carried out using a dictionary size of K = 80. A dictionary size of K = 100 is used for the experiments based on *Flickr* dataset. The average test objective values and the training times obtained using all methods tested for these two datasets are reported in Tables IV and V, and displayed in bar charts in Fig. 5.

As can be seen in Tables IV and V, the ADMM-cns method achieves the lowest test objective values. However, its advantage over the OCDL methods is not as noticeable as in the case of experiments on small datasets *Fruit* and *City*. Specifically,



Fig. 6. Images from datasets SIPI (top) and Flickr (bottom) and their reconstructions obtained using different CDL methods.

TABLE IV Average Test Objective Values and Training Times Obtained Using the Methods Compared for Dataset SIPI

	Objective	Training Time (s)
Initial dictionary	103.0952	-
FISTA [28]	63.6088	4904
OCSC [27]	67.2540	5598
Proposed-1	65.0985	685
Proposed-2	63.4867	513
ADMM-cns (batch) [25]	61.1713	2248

TABLE V Average Test Objective Values and Training Times Obtained Using the Methods Compared for Dataset *Flickr*

	Objective	Training Time (s)
Initial dictionary	51.6432	-
FISTA [28]	31.3904	16032
OCSC [27]	35.4325	12689
Proposed-1	32.4064	1362
Proposed-2	31.6799	1102
ADMM-cns (batch) [25]	30.6657	16049

in the experiments on the larger dataset *Flickr*, ADMM-cns performs only slightly better than FISTA and Proposed-2, while requiring the longest training time. Among the OCDL methods, FISTA results in the smallest test objective in the experiments on *Flickr*, although it takes the longest training time. The proposed methods result in comparable test objective values to other OCDL methods while substantially shortening the training time. In particular, Algorithm 2 has the smallest objective among all OCDL algorithms for the *SIPI* dataset.

In Fig. 6, we visually compare images reconstructed using different methods. As can be observed, the proposed approximate methods yield results of comparable accuracy to those obtained using the state-of-the-art algorithms.

G. Learning Large Dictionaries

In this experiment, we use the proposed algorithms to learn large dictionaries of sizes K = 200, K = 300, and K = 400based on the *Flickr* dataset. Learning such large dictionaries over the images of the size of those in *Flickr* is not practically feasible using the OCDL methods, OCSC and FISTA. Indeed,

TABLE VI TRAINING TIMES (SECONDS) USED BY THE PROPOSED METHODS FOR DATASET FLICKR

	K = 200	K = 300	K = 400
Proposed-1	2691	3695	4574
Proposed-2	2466	3487	4258

in single precision, for K = 200, only the larger history array of these methods, that is of size K^2P , would require more than 10 Gigabytes memory, and thus, would require a specialized computer, which is not desirable. The learned large dictionaries are visualized in Fig. 7. It can be seen that all dictionaries learned are mostly composed of visually valid features. The obtained training times are reported in Table VI and Fig. 8. As can be seen, the longest training times obtained using the proposed methods are still significantly shorter than those resulting from using other methods tested for learning smaller dictionaries (see Table V, for example).

H. CDL Over a Large Dataset

In this section, we demonstrate the scalability of the proposed algorithms using the *Flickr-large* dataset (with 1000 training images). Dictionaries composed of K = 100 filters are used in this experiment. Fig. 9 shows the average test objective values obtained using the learned dictionaries after processing 1, 10, 100, and 1000 images. The results show that both proposed algorithms are applicable to large training datasets. However, Algorithm 2 leads to considerably lower objective values.

I. VL-NIR Imaging

This section assesses the efficacy of the OCDL algorithms through an image processing task involving the fusion of VL-NIR images. NIR images are known for their high contrast resolutions, particularly when capturing vegetation scenes or imaging in low-visibility atmospheric conditions like fog or haze. Therefore, they are used to enhance outdoor VL images based on these properties.

In recent work, we proposed a VL-NIR fusion method based on OCDL [33]. This method captures correlated features in NIR and VL images as pairs of convolutional filters in two



Fig. 7. Large dictionaries learned using the proposed algorithms (top: Proposed-1, bottom: Proposed-2) for dataset *Flickr* with K = 200 (left), K = 300 (middle), and K = 400 (right).



Fig. 8. Comparison of training times obtained using the proposed algorithms and dataset *Flickr* for learning dictionaries of different sizes.



Fig. 9. Results for CDL on *Flickr-large* dataset using the proposed algorithms: average test objective values over the number of processed training images (left) and training time (right).

dictionaries, which are learned in a coupled manner via CSRs with identical supports, also known as simultaneous CSC.

In this experiment, we utilize the proposed OCDL method and the FISTA method to perform *convolutional coupled feature learning* (CCFL) using the method of [33] and compare the performances.



Fig. 10. Coupled learned VL-NIR dictionaries.

Specifically, to perform simultaneous CSC in the method of [33], the ℓ_1 -norm and $\ell_{1,2}$ -norm regularization parameters are set as 0.001 and 0.005, respectively. The parameter setting of the proposed OCDL method and the FISTA method are as described in Section IV-C.

Our results demonstrate a significant improvement in terms of training time, as CCFL over the RGB-NIR dataset using the proposed method (Algorithm 2) took only 640 seconds compared to 2547 seconds for the FISTA method.

The learned coupled VL-NIR dictionaries are presented in Fig. 10, where correlations between the corresponding filters (coupled VL-NIR features) are clearly visible. An example of VL-NIR image fusion using the proposed OCDL algorithm



NIR image



VL image



Fused VL-NIR image

Fig. 11. VL-NIR image fusion results.

is depicted in Fig. 11, revealing a noticeable improvement in contrast resolution and visibility range.

V. CONCLUSION AND DISCUSSION

An efficient approximate method for CDL has been presented. The proposed method is based on a novel formulation of the CDL problem that incorporates approximate sparse decomposition of training data samples. We have developed two computationally efficient OCDL algorithms based on ADMM to address the proposed approximate CDL problem. The proposed OCDL algorithms substantially reduce the required memory and improve the computational complexities of the state-of-the-art CDL algorithms. Extensive experimental evaluations using multiple image datasets for OCDL-based VL-NIR image fusion task have demonstrated the effectiveness of the proposed OCDL algorithms. It is worth mentioning that one advantageous feature of using ADMM (as it is in the proposed computationally efficient OCDL algorithms) is its ability to leverage distributed optimization. As an illustrative scenario, within the proposed Algorithm 2, the convolutional sparse approximation step (line 2 in Algorithm 2) can be executed in parallel for multiple input data samples. Subsequently, the *d*-update step (line 3 in Algorithm 2) can be efficiently carried out using consensus ADMM, which is a distributed implementation of ADMM. This approach can optimize the dictionary based on input from multiple sources.

With respect to strategies for reducing a possible performance gap between the exact (but computationally and memory hungry) OCDL and the proposed efficient approximate OCDL methods, one approach is to repeat the inner steps of the proposed Algorithms (lines 2-4 of Algorithm 1, for example) for a certain number of iterations. In our extensive simulations, we however did not observe a noticeable performance gap between the exact and approximate OCDL methods.

REFERENCES

- [1] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.
- [2] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.
- [3] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.
- [4] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral image classification using dictionary-based sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 10, pp. 3973–3985, Oct. 2011.
- [5] P. Song, X. Deng, J. F. C. Mota, N. Deligiannis, P. L. Dragotti, and M. R. D. Rodrigues, "Multimodal image super-resolution via joint sparse representations induced by coupled dictionaries," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 57–72, 2020.
- [6] G. Xu, L. Yang, G. Bi, and M. Xing, "Enhanced ISAR imaging and motion estimation with parametric and dynamic sparse Bayesian learning," *IEEE Trans. Comput. Imag.*, vol. 3, no. 4, pp. 940–952, Dec. 2017.
- [7] K. Engan, S. O. Aase, and J. H. Husøy, "Method of optimal directions for frame design," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1999, pp. 2443–2446.
- [8] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
- [9] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 689–696.
- [10] M. Xu, D. Hu, F. Luo, F. Liu, S. Wang, and W. Wu, "Limited-angle x-ray ct reconstruction using image gradientlo-norm with dictionary learning," *IEEE Trans. Radiat. Plasma Med. Sci.*, vol. 5, no. 1, pp. 78–87, Jan. 2021.
- [11] T. Liu, A. Chaman, D. Belius, and I. Dokmanić, "Learning multiscale convolutional dictionaries for image reconstruction," *IEEE Trans. Comput. Imag.*, vol. 8, pp. 425–437, 2022.
- [12] F. G. Veshki, N. Ouzir, S. A. Vorobyov, and E. Ollila, "Multimodal image fusion via coupled feature learning," *Signal Process.*, vol. 200, 2022, Art. no. 108637.
- [13] M. Lewicki and T. J. Sejnowski, "Coding time-varying signals using sparse, shift-invariant representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998, pp. 730–736.
- [14] M. Mørup, M. N. Schmidt, and L. K. Hansen, "Shift invariant sparse coding of image and music data," DTU Inform., Tech. Univ. Denmark, Kongens Lyngby, Denmark, Tech. Rep. IMM2008-04659, 2008.
- [15] V. Papyan, Y. Romano, M. Elad, and J. Sulam, "Convolutional dictionary learning via local processing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5306–5314.
- [16] B. Wohlberg, "Efficient algorithms for convolutional sparse representations," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 301–315, Jan. 2016.

- [17] A. Chen, Z. Xu, X. Wei, S. Tang, H. Su, and A. Geiger, "Dictionary fields: Learning a neural basis decomposition," ACM Trans. Graph., vol. 42, no. 4, pp. 1–12, 2023.
- [18] N. Janjušević, A. Khalilian-Gourtani, and Y. Wang, "CDLNet: Noiseadaptive convolutional dictionary learning network for blind denoising and demosaicing," *IEEE Open J. Signal Process*, vol. 3, pp. 196–211, 2022.
- [19] H. Zheng, H. Yong, and L. Zhang, "Deep convolutional dictionary learning for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 630–641.
- [20] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5135–5143.
- [21] H. Bristow, A. Eriksson, and S. Lucey, "Fast convolutional sparse coding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 391–398.
- [22] B. Choudhury, R. Swanson, F. Heide, G. Wetzstein, and W. Heidrich, "Consensus convolutional sparse coding," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 4290–4298.
- [23] R. Chalasani, J. C. Principe, and N. Ramakrishnan, "A fast proximal method for convolutional sparse coding," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2013, pp. 1–5.
- [24] G. Peng, "Adaptive ADMM for dictionary learning in convolutional sparse representation," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3408–3422, Jul. 2019.
- [25] F. G. Veshki and S. A. Vorobyov, "Efficient ADMM-based algorithms for convolutional sparse coding," *IEEE Signal Process. Lett.*, vol. 29, pp. 389–393, 2022.
- [26] C. Garcia-Cardona and B. Wohlberg, "Convolutional dictionary learning: A comparative review and new algorithms," *IEEE Trans. Comput. Imag.*, vol. 4, no. 3, pp. 366–381, Sep. 2018.
- [27] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Scalable online convolutional sparse coding," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 4850–4859, Oct. 2018.
- [28] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin, "First-and secondorder methods for online convolutional dictionary learning," *SIAM J. Imag. Sci.*, vol. 11, no. 2, pp. 1589–1628, 2018.
- [29] F. G. Veshki and S. A. Vorobyov, "Efficient online convolutional dictionary learning using approximate sparse components," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2023, pp. 1–5.
- [30] B. Wohlberg, "SParse optimization research COde (SPORCO)," Software library, 2016. [Online]. Available: http://purl.org/brendt/software/sporco
- [31] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
- [32] E. Zisselman, J. Sulam, and M. Elad, "A local block coordinate descent algorithm for the CSC model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8200–8209.
- [33] F. G. Veshki and S. A. Vorobyov, "Convolutional simultaneous sparse approximation with applications to RGB-NIR image fusion," in *Proc. IEEE 56th Asilomar Conf. Signals Syst. Comput.*, 2022, pp. 872–876.



Farshad G. Veshki (Student Member, IEEE) received the M.Sc. degree in communications and systems engineering from the University of Vaasa, Vaasa, Finland, in 2018, and the Ph.D. degree in signal processing technology from Aalto University, Espoo, Finland, in June 2023. He is currently an R&D Engineer with Nokia, Espoo. He is also a visiting Researcher with the Department of Information and Communications Engineering, Aalto University. His research interests include statistical signal processing, optimization, and machine learning.



Sergiy A. Vorobyov (Fellow, IEEE) received the M.Sc. and Ph.D. degrees in systems and control from the Kharkiv National University of Radio Electronics, Kharkiv, Ukraine, in 1994 and 1997, respectively. He is currently a Professor with the Department of Information and Communications Engineering, Aalto University, Espoo, Finland. Since his graduation, he held various faculty and research positions with the University of Alberta, Edmonton, AB, Canada, the Joint Research Institute between Heriot-Watt University and Edinburgh University, Edinburgh, U.K.,

Duisburg-Essen University, Duisburg, Germany, Darmstadt University of Technology, Darmstadt, Germany, McMaster University, Hamilton, ON, Canada, the Institute of Physical and Chemical Research, Wako, Japan, and the Kharkiv National University of Radio Electronics, Kharkiv, Ukraine. His research interests include optimization and multi-liner algebra methods in signal processing and data analysis, statistical and array signal processing, sparse signal processing, computational imaging, estimation, detection and learning theory and methods, and multi-antenna, very large, cooperative, and cognitive systems. Dr. Vorobyov was the recipient of the 2004 IEEE Signal Processing Society Best Paper Award, 2007 Alberta Ingenuity New Faculty Award, 2011 Carl Zeiss Award (Germany), 2012 NSERC Discovery Accelerator Award, and IEEE ICASSP 2023 Top 3% paper recognition, and other awards. Since 2016, he has been the Senior Area Editor of the IEEE SIGNAL PROCESSING LETTERS. He was an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING during 2006-2010 and IEEE SIGNAL PROCESSING LETTERS during 2007-2009. He was a Member of the Sensor Array and Multi-Channel Signal Processing and Signal Processing for Communications and Networking Technical Committees of the IEEE Signal Processing Society during 2007–2012 and 2010–2016, respectively. He was the Track Chair for Asilomar 2011, Pacific Grove, CA, USA, Technical Co-Chair for IEEE CAMSAP 2011, Puerto Rico, Tutorial Chair for ISWCS 2013, Ilmenau, Germany, the Technical Co-Chair for IEEE SAM 2018, Sheffield, U.K., Technical Co-Chair for IEEE CAMSAP 2023, Costa Rica, and General Co-Chair for EUSIPCO 2023, Helsinki, Finland.