Stochastic Convolutional Sparse Coding

Anonymous CVPR submission

Paper ID 3722

Abstract

State-of-the-art methods for Convolutional Sparse Coding usually employ Fourier-domain solvers in order to speed up the convolution operators. However, this approach is not without shortcomings. For example, Fourier-domain representations implicitly assume circular boundary conditions and make it hard to fully exploit the sparsity of the problem as well as the small spatial support of the filters.

In this work, we propose a novel stochastic spatial-domain solver, in which a randomized subsampling strategy is introduced during the learning sparse codes. Afterwards, we extend the proposed strategy in conjunction with online learning, scaling the CSC model up to very large sample sizes. In both cases, we show experimentally that the proposed subsampling strategy, with a reasonable selection of the subsampling rate, outperforms the state-of-the-art frequency-domain solvers in terms of execution time without losing the learning quality. Finally, we evaluate the effectiveness of the over-complete dictionary learned from large-scale datasets that has not been studied by precedent CSC work, which demonstrates an improved sparse representation of the natural images on account of more abundant learned image features.

1. Introduction

Convolutional Sparse Coding (CSC) is a method for learning *generative* models in the form of translationally invariant dictionaries for a large variety of different training signals. These generative models have been shown effective for solving problems in neural and brain information processing [11, 22], as well as in a variety of image processing tasks, for instance, image inpainting [10], super-resolution [9], high dynamic range imaging [25], and high-dimensional signal reconstructions [6, 1]. CSC differs from conventional sparse coding by formulating the signals as the sum of a set of convolutions on dictionary filters and sparse codes instead of patch-wise linear combinations of filters. In traditional sparse dictionary learning, the patch structure significantly degrades the expressiveness of the dictionaries by introducing a strong dependency on the position of a feature, which the convolutional nature of CSC avoids.

Of course this convolutional approach is also at the heart of many deep learning-based methods in the form of CNNs [16, 13, 15], which have in recent years been extraordinarily successful for a broad range of high-level image understanding applications. However, while CNNs generally are used in a *supervised* setting and produce *discriminative*, task-specific models, CSC is *unsupervised* and produces *generative* models that easily transfer between tasks.

To solve the optimization problems inherent to CSC, Zeiler et al. [30] iteratively solve two subproblems (updating sparse codes and updating filters) using gradient decent in the form of convolutional operations in the spatial domain, which is computationally expensive. Recent algorithms tackle the problem by exploiting Parseval's theorem to express the spatial convolution by multiplication in the frequency domain and using proximal solver such as Alternating Direction Method of Multipliers (ADMM) [4] to separate the linear least squares parts from the non-smooth terms in the optimization problem. These approaches show tremendous improvements over prior spatial-domain solvers with respect to running time [5, 10, 29, 6]. Most of the prior work learns the dictionary filters in a batch mode, which indicates that all training signals are involved in every training iteration, and this restricts it from applying to large datasets or streaming data.

In contrast to batch mode learning, online learning [26] is a well established strategy which processes a single image or a small portion (mini-batch) of the whole data at each training step, and incrementally updates model variables. Herein, the required memory and computing sources are only dependent on the sample size in every observation, independent of the training data size. It alleviates the scalability issue that arises in batch approaches, and the convergence of the algorithm was firstly analyzed using stochastic approximation tools [2]. Bottou et al. [3] further showed better generalization performance of the stochastic algorithms than standard gradient descent on large scale learning systems. Later on, online learning strategies were synergetic with sparse coding, which was then scaled

up for learning dictionary from millions of training samples [19, 20], and for large-scale matrix factorization with an additionally introduced subsampling strategy [21]. More recently, Liu et al. [18] and Wang et al. [28] separately proposed similar online learning frameworks for the CSC model, alleviating the memory issues arise in batch-based CSC model on large datasets.

Contributions. We mainly make three contributions in this work. First, we introduce a randomization strategy for the CSC model and solve the entire problems in spatial domain. We demonstrate that the proposed stochastic spatial-domain solver, with a reasonably selected subsampling rate, outperforms the state-of-the-art frequency-domain solvers with regard to computing efficiency. We then formulate an online-learning version of the proposed algorithm, and show dramatic runtime improvement over current online CSC methods, while producing comparable outcomes. Finally, we demonstrate the capability to learn the over-complete dictionary from thousands of images, and analyze the effectiveness of the learned over-complete dictionary for a number of reconstruction tasks.

2. Convolutional Sparse Coding (CSC)

The dictionary learning problem for CSC problem has the form

$$\begin{aligned} &\min_{\mathbf{d},\mathbf{z}} \ \frac{1}{2}\|\mathbf{x} - \sum_{\mathrm{k}=1}^{K} \mathbf{d}_\mathrm{k} * \mathbf{z}_\mathrm{k}\|_2^2 + \lambda \sum_{\mathrm{k}=1}^{K} \|\mathbf{z}_\mathrm{k}\|_1 \\ &\text{subject to } \|\mathbf{d}_\mathrm{k}\|_2^2 \leq 1 \ \ \forall k \in \{1, \dots, K\}, \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^D$ is a D-dimensional signal or a vectorized image[1], $\mathbf{d}_\mathrm{k} \in \mathbb{R}^M$ is the k-th dictionary, $\mathbf{z}_\mathrm{k} \in \mathbb{R}^D$ is the sparse code associated with that dictionary, $\lambda > 0$ is a sparsity inducing penalty parameter, K is the number of dictionary filters, and $*$ is the convolution operator. The above model will be applied to all the training images $\mathbf{x} \in \mathbb{X}$.

Most recent CSC algorithms exploit Parseval's theorem and introduce two slack variables to separate the non-smooth L_1 penalty term and the L_2 constraints, making it feasible to efficiently compute the latter in the frequency domain. Furthermore, the whole Problem (1) can be split into alternating subproblems for updating $\mathbf{z}$ and $\mathbf{d}$, which are jointly solved by coordinate descent [5, 10, 29]. This approach suffers from several issues:

- While CSC overcomes the independence assumption held in patch-based learning algorithms, far more variables (K times more) are introduced to represent a single image to compensate for this. This creates more severe memory and computational burdens.
- We observe through experiments that the vast majority of the entries of the reconstructed sparse codes do not provide useful information about the represented image. For $K = 100$, 99.5% entries are not informative. This indicates that the subproblem for updating $\mathbf{z}$ solves a highly sparse LASSO problem. Transforming the problem into frequency domain imposes restrictions on exploiting this sparsity.
- While prior work shows its efficiency in solving the CSC problem in the frequency domain, this is only applicable for updating $\mathbf{z}$, and does not hold for updating $\mathbf{d}$. The dictionary filters usually have much smaller spatial support than the dimension size of the sparse codes ($M \ll D$). However, in order to tackle the problem in the frequency domain, it is necessary to process the $\mathbf{d}$-subproblem over the full support of the sparse codes, and then project the results onto the much smaller spatial support of the filters.

[1] In this manuscript, we work on 2D images.

3. Stochastic Convolutional Sparse Coding

3.1. The Model

Based on the strong sparsity of the codes $\mathbf{z}$, we propose to implement the CSC learning problem (1) iteratively, where at t-th iteration we only consider a random subset $\mathcal{M}^t(\mathbf{z})$ of the codes:

$$\begin{aligned} &(\mathbf{d}^t, \mathbf{z}^t) = \\ &\operatorname*{argmin}_{\mathbf{d},\mathbf{z}} \ \frac{1}{2}\|\mathbf{x} - \sum_{\mathrm{k}=1}^{K} \mathbf{d}_\mathrm{k} * \mathcal{M}_\mathrm{k}^t(\mathbf{z}_\mathrm{k})\|_2^2 + \lambda \sum_{\mathrm{k}=1}^{K} \|\mathcal{M}_\mathrm{k}^t(\mathbf{z}_\mathrm{k})\|_1 \\ &\text{subject to } \|\mathbf{d}_\mathrm{k}\|_2^2 \leq 1 \ \ \forall k \in \{1, \dots, K\}. \end{aligned} \tag{2}$$

Here $\mathcal{M}_\mathrm{k}^t(\mathbf{z}_\mathrm{k})$ is a $\mathbb{R}^D \to \mathbb{R}^D$ operator subsampling the corresponding code $\mathbf{z}_\mathrm{k}$ following a Bernoulli distribution with probability p, drawn at current iteration. Concretely, it retains the sampled codes and zero out the others. In the case of $p = 1$, the proposed model is identical with the classical CSC model. When $p < 1$, the algorithm only solves a subset of the sparse codes at chosen positions in each iteration, and accordingly, the update of dictionary $\mathbf{d}$ is based on the selected portion of the sparse codes. Similar to solving the classical CSC problem, we can apply a coordinate descent algorithm, alternating on subproblems of $\mathcal{M}^t(\mathbf{z})$ and $\mathbf{d}$, to solve the above bi-convex optimization problem. Specifically, the modified minimization problem for $\mathbf{z}$ can be formulated as:

$$\mathbf{z}^t = \operatorname*{argmin}_{\mathbf{z}} \ \frac{1}{2}\|\mathbf{x} - \mathbf{D}^{t-1}\mathbf{M}^t\mathbf{z}\|_2^2 + \lambda\|\mathbf{M}^t\mathbf{z}\|_1, \tag{3}$$

where $\mathbf{D}^{t-1} = [\mathbf{D}_1^{t-1}, \dots, \mathbf{D}_\mathrm{K}^{t-1}] \in \mathbb{R}^{D \times DK}$, which is constructed from the dictionary learned by $(t-1)$-th it-

eration, $\mathbf{z} = [\mathbf{z}_1, \ldots, \mathbf{z}_K] \in \mathbb{R}^{DK}$. The convolution operators are expressed as a matrix multiplication so that $\mathbf{Dz} = \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k$. Therefore, each part of $\mathbf{D}$ is a Toeplitz matrix. Matrix $\mathbf{M}^t$ is a $\mathbb{R}^{Dk \times DK}$ random binary diagonal matrix with 1s on the diagonal entries corresponding to the sampled codes and 0 elsewhere, performing the function of $\mathcal{M}^t(\mathbf{z})$, which satisfies $\mathrm{Tr}(\mathbf{M}^t) = pDK$. When $p = 1$, $\mathbf{M}^t$ is just an identity matrix. Due to the introduced subsampling matrix, the convolution operator cannot be implemented in the Fourier domain. However, owing to the subsampling strategy, the number of variables that need to be computed for this subproblem is pDK instead of DK, which leads to a reduction of the spatial-domain computation time by a factor of $\frac{1}{p}$.

After computing the subsampled sparse codes $\mathcal{M}^t(\mathbf{z})$, we can then project them onto the original spatial support by substituting 0 for the masked-out values, obtaining $\mathbf{z}^t$. Afterwards, the dictionary can be updated by:

$$\begin{aligned} \mathbf{d}^t = \underset{\mathbf{d}}{\mathrm{argmin}} \ & \frac{1}{2}\|\mathbf{x} - \mathbf{Z}^t\mathbf{d}\|_2^2 \\ \text{subject to } & \|\mathbf{d}_k\|_2^2 \leq 1 \ \forall k \in \{1, \ldots, K\}, \end{aligned} \tag{4}$$

where $\mathbf{Z}^t = [\mathbf{Z}_1^t, \ldots, \mathbf{Z}_K^t] \in \mathbb{R}^{D \times MK}$ is a concatenation of Toeplitz matrices, and $\mathbf{Z}_k^t$ is constructed from the associated $\mathbf{z}_k^t$, $\mathbf{d} = [\mathbf{d}_1, \ldots, \mathbf{d}_K] \in \mathbb{R}^{MK}$ such that $\mathbf{Zd} = \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k$. In typical CSC settings, $M \ll D$, hence there is no need to perform subsampling on the dictionary.

The work of [21] also applies a so-called stochastic subsampling strategy for dictionary learning, where the subsampling is randomly performed on the observed signals, differing from that of our proposed. In addition, it is basically applied for factorizing matrix that is large in both dimensions. Although CSC can be expressed in a similar mathematical formula as matrix factorization, the two problems are in practice quite different due to the convolutional property of the CSC model. Our proposed randomization approach shares a similar idea as utilized in the stochastic Frank-Wolfe algorithm [23, 14], in which the optimization is performed on a subset of the variables which are randomly extracted based on a certain probability distribution at each iteration. For the proposed algorithm, each iteration extracts $(pDK + MK)$ variables, where pDK variables are randomly picked from a total DK variables, and the rest remain unchanged. Owing to the fact that the codes are highly sparse, the original signals can still be represented by a portion of the codes under a reasonable subsampling rate. Therefore, the convergence of the proposed algorithm will not be significantly affected by the subsampling manipulation, unlike the general case of the stochastic Frank-Wolfe algorithm, which usually requires more iterations to reach convergence. This insight is experimentally verified in Section 4.

In the following we introduce two different outer loop structures to utilize the proposed subsampling strategy. First, we introduce a *batch mode* method (stochastic batch CSC) that learns from all images simultaneously, and second we introduce an *online* variant (stochastic online CSC).

3.2. Stochastic Batch CSC (SBCSC)

We first introduce a batch-mode version of the proposed method as shown in Algorithm 1, where N is the number of total input images, $(\mathbf{z}^i)^t$ is the sparse code associated with i-th image at t-th iteration, and p is the uniform probability for one code been selected. We choose $p = \{1, 0.5, 0.2, 0.1, 0.05\}$ for testing in this work, where $p = 1$ indicates no subsampling, and $p = 0.05$ indicates a subsampling rate of 5%.

Algorithm 1 SBCSC

```
Initialize t = 0, d^t, p
while not converge do
    t ← t + 1
    Randomly sample z^t with rate p
    for i=1 to N do
        Compute (z^i)^t by solving problem (3)
    end for
    Compute d^t by solving problem (4)
end while
```

Problem (3) is the standard LASSO, which can be solved by plenty of optimization frameworks. We found that solving it with ADMM delivers a good balance between computation time and convergence within a moderate number of iterations. Specifically, the data fitting term and the L_1 penalty term are split, forming two separate substeps. The first substep is a quadratic programming (QP) problem, and we can either cache the matrix factorization by Cholesky decomposition (when N is relatively large), or solve it iteratively by Conjugate Gradient (when N is relatively small). The second substep can be solved by a point-wise shrinkage operation. Problem (4) is a quadratic constrained quadratic programming (QCQP) problem, and it can be efficiently solved by projected block coordinate decent. Empirically, a single iteration is enough with $\mathbf{d}$ computed in previous iteration as a warm start. We set the hyperparameters $\lambda = 1$, the ADMM iteration fixed to 10, the augmented Lagrangian penalty ρ to 10λ, and the over-relaxation strategy within ADMM is applied with $\alpha = 1.8$. For a detailed description of the above two solvers, please refer to the supplement.

Every outer loop of SBCSC involves all of the training images, which makes it computationally expensive to process them all simultaneously. Furthermore, memory consumption quickly becomes an issue with increasing numbers of images. Thus, the batch-based learning algorithm

lacks the ability to scale up to very large datasets or to handle dynamically changing training data.

3.3. Stochastic Online CSC (SOCSC)

Algorithm 2 SOCSC

Initialize $t = 0,\ \mathbf{d}^t,\ p,\ \mathbf{C}^t = 0,\ \mathbf{B}^t = 0$
while not converge **do**
  $t \leftarrow t + 1$
  draw $\mathbf{x}^t$ from training images
  Randomly sample $\mathbf{z}^t$ with rate p
  Compute $\mathbf{z}^t$ by solving problem (5) using $\mathbf{d}^{t-1}$
  Compute $\mathbf{C}^t$ and $\mathbf{B}^t$ by Eq. 7
  Compute $\mathbf{d}^t$ by solving problem (8)
end while

In order to address this scalability issue, we can tackle the Stochastic CSC problem in an online fashion. In the online learning setting as shown in Algorithm 2, each iteration only draws one or a subset (mini-batch) of the total training images, hence the complexity per loop is independent of the training sample size. Then, given the sampled image $\mathbf{x}^t$ at t-th iteration, we can compute the corresponding subsampled sparse codes $\mathbf{z}^t$ by

$$\mathbf{z}^t = \underset{\mathbf{z}}{\operatorname{argmin}}\ \frac{1}{2}\|\mathbf{x}^t - \mathbf{D}^{t-1}\mathbf{M}^t\mathbf{z}\|_2^2 + \lambda\|\mathbf{M}^t\mathbf{z}\|_1. \quad (5)$$

The only difference between Eq. 3 is that $\mathbf{x}^t$ only contains a portion of the total images. After obtaining the sparse codes, the dictionary is updated by:

$$\begin{aligned}\mathbf{d}^t = \underset{\mathbf{d}}{\operatorname{argmin}}\ & \frac{1}{2t}\sum_{\mathrm{i}=1}^{t}\|\mathbf{x}^i - \mathbf{Z}^i\mathbf{d}\|_2^2 \\ \text{subject to}\quad & \|\mathbf{d}_\mathrm{k}\|_2^2 \le 1\ \forall k \in \{1,\dots,K\}.\end{aligned} \quad (6)$$

Note that updating the dictionary in this fashion involves all of the past training images and sparse codes. Using techniques developed for regular (non-convolutional) dictionary learning [19, 20], we can get rid of explicitly storing this data by introducing two surrogate matrices $\mathbf{C} \in \mathbb{R}^{KM\times KM}$ and $\mathbf{B} \in \mathbb{R}^{KM\times 1}$. These carry all of the required information for updating $\mathbf{d}$, and can be iteratively updated by:

$$\begin{aligned}\mathbf{C}^t &= \frac{t-1}{t}\mathbf{C}^{t-1} + \frac{1}{t}(\mathbf{Z}^t)^\top\mathbf{Z}^t \\ \mathbf{B}^t &= \frac{t-1}{t}\mathbf{B}^{t-1} + \frac{1}{t}(\mathbf{Z}^t)^\top\mathbf{x}^t\end{aligned} \quad (7)$$

With these surrogate matrices, the updated dictionary can be obtained by solving

$$\begin{aligned}\mathbf{d}^t = \underset{\mathbf{d}}{\operatorname{argmin}}\ & \frac{1}{2}\mathbf{d}^\top\mathbf{C}^t\mathbf{d} - \mathbf{d}^\top\mathbf{B}^t \\ \text{subject to}\quad & \|\mathbf{d}_\mathrm{k}\|_2^2 \le 1\ \forall k \in \{1,\dots,K\}.\end{aligned} \quad (8)$$

Problem (5) and problem (8) are solved in the same way as that for SBCSC.

3.4. Complexity Analysis

Recall that D is the number of pixels for a single image, K is the number of filters, and M is the size of the filter support. Commonly, we can assume $K \approx M$. State-of-the-art frequency-domain solvers then have the time complexity $\mathcal{O}(K^2D + KDlog(D))$ for a single data pass.

The time complexity of updating $\mathbf{z}$ in our SBCSC algorithm using Conjugate Gradient is $\mathcal{O}(pKMD\sqrt{\tau})$, where $pKMD$ is the number of non-zero elements in $(\mathbf{D}\mathbf{M}^\top)$ and τ is the condition number of $(\mathbf{A}^\top\mathbf{A} + \rho I)$ where $\mathbf{A} = \mathbf{D}\mathbf{M}^\top$. With a reasonable selection of the subsampling rate, this time complexity is comparable to that of frequency-domain solvers.

Updating the filters $\mathbf{d}$ takes $\mathcal{O}(K^2M^2)$ time. This is comparable to $\mathcal{O}(K^2D)$ in the common CSC setting ($M \ll D$). However, multiple ADMM iterations are required in the frequency domain to compute $\mathbf{d}$ while only a single pass is required by the proposed method, which greatly reduces the computation time. Overall, the proposed method has the time complexity of $\mathcal{O}(pKMD\sqrt{\tau} + K^2M^2)$.

The time complexity of SOCSC for one data pass is similar to that of SBCSC, apart from two additional steps to update the surrogate matrices. Updating $\mathbf{C}$ and $\mathbf{B}$ involves computing $\mathbf{Z}^\top\mathbf{Z}$ and $\mathbf{Z}^\top x$. Although $\mathbf{Z}$ has dimensions of $D \times KM$, it is a highly sparse matrix with only $\mathcal{O}(D)$ non-zero elements. Therefore, the total performance is not affected significantly.

4. Results

We first validate the proposed algorithms on the fruit and city datasets [30], which each consist of 10 training images of size 100×100. The online-mode algorithms are then adapted to one thousand 100×100 image patches randomly picked up from ImageNet [7]. Note that batch-based CSC commonly can only handle less than 100 images simultaneously. The dictionary size is set to 100 filters of size 11×11 pixels in all experiments except for over-complete dictionary. All training and evaluation processes in this manuscript are performed on contrast normalized images [30, 10].

4.1. Subsampling Strategy

Convergence. Comparisons of the convergence between the proposed method (SBCSC) and the state-of-the-art batch-mode algorithm [10] are shown in Fig. 1. A different selection of the subsampling rate reveals that the proposed strategy will slightly influence the convergence and the training objective of the minimization problem. Specifically, the more subsampled, the relatively slower convergence and the higher objective will be obtained. On the

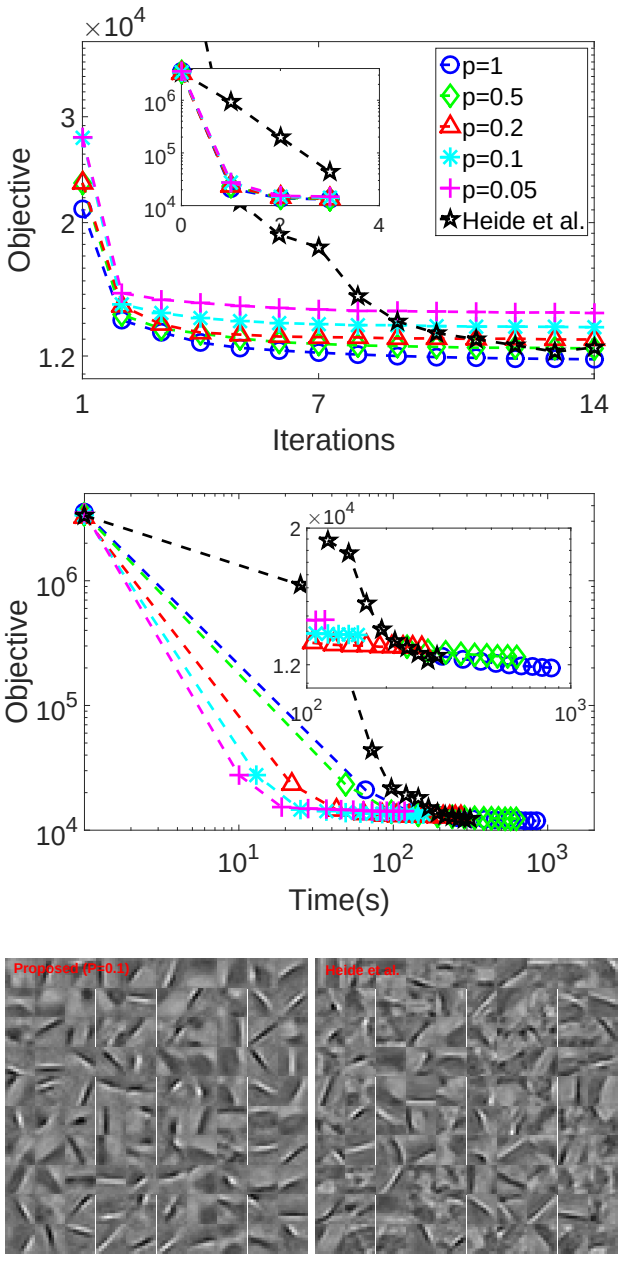

Figure 1: Top: Convergence comparison between the-state-of-art method [10] and the proposed method with different subsampling rate, all of which on performed on fruit dataset. Convergence is evaluated by monitoring the objective value of Eq. 1 on training images versus iterations and time, respectively. Bottom: Learned filters by the proposed method with $p = 0.1$ and the comparable method. In these represented learned filters, our method learns more smooth Gabor-like filters and less number of noise-like filters with the same λ. Beyond this, it runs faster with regard to each iteration and shows better convergence behavior.

other hand, small subsampling rate will significantly accelerate the computation process, where 10% subsampling achieves about $6\times$ speedup over the not subsampled spatial domain solver and a $2\times$ speedup over state-of-the-art Fourier-domain solver for one iteration. We observe that a subsampling rate between $p = 0.1$ and $p = 0.2$ delivers empirically good enough convergence in our settings, as well as achieving at least $3\times$ speedup. In general, the proposed method with various subsampling rates converges at around 10-12 iterations in all testing cases, acting similar to the competing methods.

We have observed that the comparison method has a relatively high objective value during the first $6 - 8$ iterations. The reason is that it introduces an additional mask term handling the boundary issues, therefore different splitting strategies are utilized and the subproblems are interleaved on their primary variables (in some sense, the primary variables can be regarded as intermediate results in the ADMM iteration). See [29] for details. Despite of this, it converges to an optimum with comparable objective after 10 iterations. In summary, the convergence behaviors of the proposed algorithm is only slightly influenced by the subsampling strategy within the testing subsampling rates. Comparing to the state-of-the-art frequency solver, the proposed stochastic spatial-domain solver with a subsampling rate of 0.1 reduces the computation time by a factor of two for the tested example. The robustness of the proposed algorithm is evaluated by additional experiments. Please see supplementary materials for reference.

Reconstruction. The SBCSC-learned filters with a subsampling rate of $p = 0.1$ are shown in the bottom of Fig. 1. For a visual comparison, we also show the filters learned from the competing method (bottom left of Figure 3 in [10]). As can be observed, both of them learn some seemingly similar Gabor-like filters. However, zooming into the details (please see supplemental materials for high resolution versions) reveals that our filters appear less noisy than the ones from [10]. We then demonstrate the effectiveness of the reconstructed filters in the application of image inpainting, which refers to reconstructing a full image from partial measurements. A numerical comparison of the reconstruction quality is shown in Fig. 2. The filters learned by the proposed method not only appear less noisy by themselves, but also achieve better reconstruction quality on partially observed images with respect to the PSNR value. Furthermore, the learning process only takes about 50% of the time for our method. Specifically, SBCSC takes 170 seconds and the comparable method takes 350 seconds for 14 iterations on a Core i7 PC.

4.2. Online Learning

Convergence. Unlike the batch-based learning approaches which evaluate its convergence by monitoring the

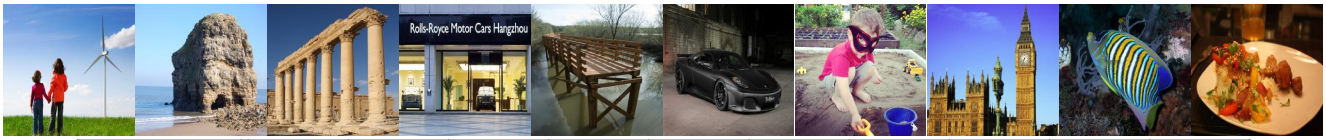

Image	1	2	3	4	5	6	7	8	9	10
PSNR [10]	29.54	28.16	29.40	29.22	28.89	29.29	28.10	29.57	27.46	30.72
PSNR ours	**29.65**	**28.18**	**29.52**	**29.43**	**28.99**	**29.38**	**28.19**	**29.63**	**27.62**	**30.98**

Figure 2: Numerical comparisons of the reconstruction quality obtained from the presented filters and its comparison. The reconstructions are performed on 50% randomly observed images, with $\lambda = 0.4$ and 50 ADMM iterations for both cases. Obtained PSNR values are averaged on 5 trials. Note that none of the testing images are in the training sets.

objective value on training datasets, a common way to evaluate the learning process of online learning model is to monitor its objective value on test datasets. In Fig. 3 we plot the objective values against the iteration number for the proposed method (SOCSC) and a recent online frequency-domain CSC method [18]. In the same figure, we also keep track of the capability of the updated filters during the learning process to sparsely represent the test images, which is demonstrated by the time evolution of PSNR. These two approaches stop at optimum positions with similar objective values. The final PSNR values for both methods also reveal a similar reconstruction performance of the learned filters. In terms of runtime comparison, however, the proposed method runs at least 6 times faster than the comparison method. Specifically, SOCSC takes 70 seconds and the comparable method takes 440 seconds on a Core i7 PC to process all 10 training images. The supplement shows additional comparisons for the other datasets.

Over-complete dictionary. To our knowledge, none of the existing CSC work reported or analyzed the over-complete dictionary (number of the dictionary is more than its degrees of freedom). One of the reasons could be that most of the prior work is batch-based, thus learning over-complete dictionary from small datasets would cause over-fitting issues, which may contain quite a few data-specific filters, and therefore limit the ability to generalize the filters to other data (we verify this explanation in the supplement). The proposed online-based learning strategy (SOCSC) can overcome this issue by scaling the model up to arbitrary sample sizes.

We demonstrate this ability on 1000 image patches with the size of 100×100, and learn an $11 \times 11 \times 400$ over-complete dictionary, which is shown in Fig. 4. For a visual comparison, we also show 100 learned filters generated by the same algorithm and another 100 filters generated by [18]. As can be observed, both of the approaches learn visually similar under-complete dictionary, while the proposed method takes $6\times$ less training time than the comparison method. The learned over-complete dictionary is composed of the Gabor-like image features as represented in

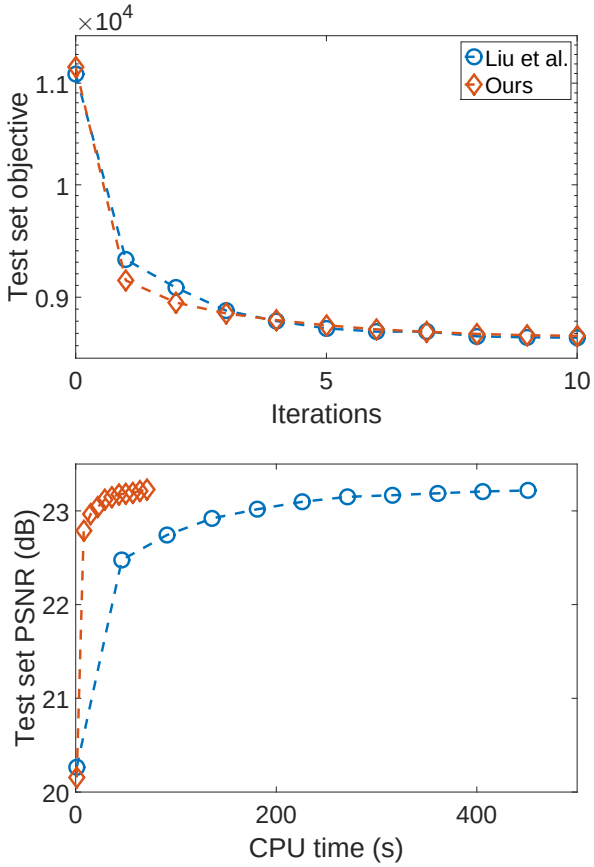

Figure 3: The experiments are performed on fruit dataset, and each iteration randomly choose one samples from the training datasets. Top: Convergence of the test set objectives (objective value of Eq. 1 on testing datasets) for our method (SOCSC) and the state-of-the-art online approach [18]. Bottom: Testing PSNR with respect to execution time. While the quality of the output is comparable, our method achieves $6\times$ speedup.

the under-complete dictionaries, as well as a number of low contrast features which are not typical for under-complete

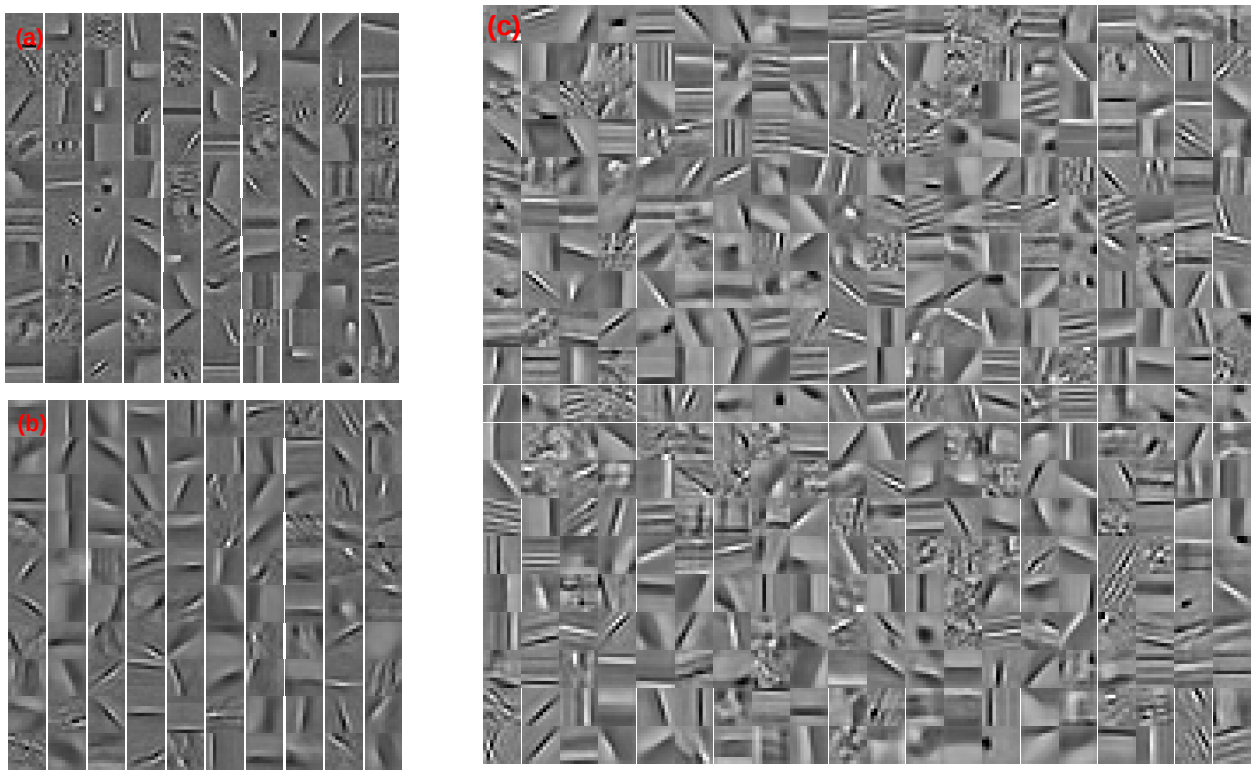

Image	1	2	3	4	5	6	7	8	9	10
PSNR by (a)	29.58	28.19	29.44	29.63	28.89	29.33	28.13	30.14	27.42	30.89
PSNR by (b)	29.63	28.22	29.57	29.90	29.12	29.59	28.05	30.17	27.53	31.08
PSNR by (c)	**30.24**	**28.34**	**29.95**	**30.30**	**29.43**	**29.96**	**28.24**	**30.57**	**27.72**	**31.67**

Figure 4: Filters learned from 1000 images by our method (SOCSC) and the state-of-the-art online method [18], and their corresponding reconstruction quality using the same hyperparameter settings as in Fig. 2. (a) The under-complete dictionary ($11 \times 11 \times 100$) learned by [18]; (b) The under-complete dictionary ($11 \times 11 \times 100$) learned by SOCSC. (c) The over-complete dictionary ($11 \times 11 \times 400$) learned by SOCSC. These under-complete dictionaries, mainly composed of Gabor-like filters, can be seen as a subset of the represented over-complete dictionary, which contains a number of extra low contrast image features.

dictionaries. This additional feature information would play an essential role to better reconstruct the natural images, which can be demonstrated by the results of reconstruction quality shown at the bottom of the same figure.

We further show a bottleneck revealed by the under-complete dictionary in the online-based CSC model. The top of Fig. 5 demonstrates that no more apparent progress could be observed when the number of training images is higher than a specific value for both of the online approaches ($K = 100$). However, owing to more abundant filters, learning over-complete dictionary overcomes this bottleneck, and it shows a considerable improvement in the PSNR of image representations. This is also verified by applying these learned dictionaries to the image inpainting application as discussed above.

Mini-batching. In practice, a mini-batching strategy would be preferred in order to gain advantages from modern parallel computing architectures. This is also a standard extension to stochastic optimization algorithms [27, 24, 17]. We denote the mini-batch size as η. In the proposed online algorithm (SOCSC), the time complexity for one step dictionary update will not increase linearly with the increase of η. Concretely speaking, updating $\mathbf{z}$ can be implemented by caching the Cholesky decomposition, and one computation of the matrix factorization can be applied to all of the currently selected batches. Herein, the complexity for doing updating $\mathbf{z}$ η times all together is cheaper than η times the complexity of updating one $\mathbf{z}$. In addition, the time complexity for updating $\mathbf{d}$ is not linearly affected by the value of η, which will be executed only once in each training step regardless of η. One exception is the update of surrogate matrices which has a complexity that is linear in η. How-

ever, this step is not dominant in the runtime.

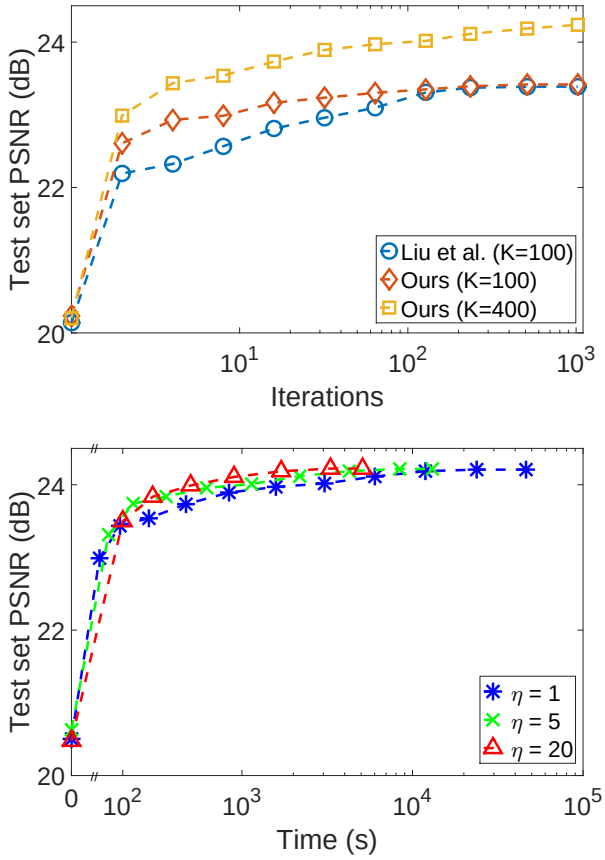

Figure 5: Top: Testing PSNR for the comparable method [18] with $K = 100$, and our method (SOCSC) with $K = 100$ and $K = 400$, respectively. Every iteration draws a single image from those 1000 image patches. Bottom: Testing PSNR for SOCSC ($K = 400$) with varying values of η. The learned filters are examined on the test sets every 2^i iterations and also at the last iteration, where $i = 0, 1, \ldots$. Note that all the results are generated by a single-core program.

The runtime comparisons for various mini-batch sizes are shown in the bottom of Fig. 5. Note that larger η will result in a smaller number of iterations to process all 1000 samples. The plots show that a larger mini-batch size will generally lead to a greater progress in first few training steps, though it takes additional running time and memory. Overall, mini-batched updates provides a more runtime efficient learning process in the online-based CSC model, and according to the obtained experimental results, $\eta = 20$ achieves a one order of magnitude speedup over $\eta = 1$ to reach a a comparable level of convergence. Since computing sparse codes is a data-independent process, this mini-batched approach can be further accelerated in a distributed-computing system.

5. Conclusions

In this work, we present a novel stochastic subsampling strategy for solving the CSC problem in the spatial domain. This method significantly improves the runtime performance over the prior frequency-domain solvers, which applies to both batch mode and online-learning mode. The proposed algorithm, for the first time, demonstrates the feasibility that tackling the CSC problem in spatial domain while still holding, or even improving the runtime efficiency. Since the subproblem of updating the code is a highly sparse LASSO, other specific optimization strategies can be applied to further accelerate the computation, for instance the idea proposed in [12], which solves the LASSO problem by coordinate descent and skips unnecessary updates using the method of safe screening [8]. It is worth emphasizing that Fourier-domain methods cannot benefit from these kinds of speedup strategies.

In the future, alternative subsampling strategies can be studied to improve learning efficiency. Instead of employing uniform subsampling strategy, for instance one could employ an importance or correlation based sampling strategy that considers the inter-dependency between the signals and the sparse codes.

We have also shown the capability of the developed online algorithm to learn representative and meaningful overcomplete dictionary from arbitrary large datasets, and the availability of the dictionary is further verified by the application of image inpainting. It can be foreseen that this capability has widespread applications in audio and image related tasks, and higher dimensional signal processing. The source code for SBCSC and SOCSC is attached in the supplement for reference.

References

[1] A. Bibi and B. Ghanem. High order tensor formulation for convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1772–1780, 2017. 1

[2] L. Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9):142, 1998. 1

[3] O. Bousquet and L. Bottou. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008. 1

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011. 1

[5] H. Bristow, A. Eriksson, and S. Lucey. Fast convolutional sparse coding. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 391–398. IEEE, 2013. 1, 2

[6] B. Choudhury, R. Swanson, F. Heide, G. Wetzstein, and W. Heidrich. Consensus convolutional sparse coding. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4290–4298. IEEE, 2017. 1

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 4

[8] L. E. Ghaoui, V. Viallon, and T. Rabbani. Safe feature elimination for the lasso and sparse supervised learning problems. In *Pacific Journal of Optimization*, page 8(4):667698, 2012. 8

[9] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang. Convolutional sparse coding for image super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1823–1831, 2015. 1

[10] F. Heide, W. Heidrich, and G. Wetzstein. Fast and flexible convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5135–5143, 2015. 1, 2, 4, 5, 6, 11, 12, 13

[11] M. Jas, T. D. La Tour, U. Simsekli, and A. Gramfort. Learning the morphology of brain signals using alpha-stable convolutional sparse coding. In *Advances in Neural Information Processing Systems*, pages 1099–1108, 2017. 1

[12] T. B. Johnson and C. Guestrin. Stingycd: Safely avoiding wasteful updates in coordinate descent. In *International Conference on Machine Learning*, pages 1752–1760, 2017. 8

[13] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. L. Cun. Learning convolutional feature hierarchies for visual recognition. In *Advances in neural information processing systems*, pages 1090–1098, 2010. 1

[14] T. Kerdreux, F. Pedregosa, and A. d'Aspremont. Frank-Wolfe with subsampling oracle. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2591–2600, 2018. 3

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[17] L. Lei and M. I. Jordan. Less than a single pass: Stochastically controlled stochastic gradient. In *PMLR: Proceedings of Machine Learning Research (AISTATS 2017)*, volume 54, 2017. 7

[18] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin. First-and second-order methods for online convolutional dictionary learning. *SIAM Journal on Imaging Sciences*, 11(2):1589–1628, 2018. 2, 6, 7, 8, 11

[19] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009. 2, 4

[20] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(Jan):19–60, 2010. 2, 4

[21] A. Mensch, J. Mairal, B. Thirion, and G. Varoquaux. Dictionary learning for massive matrix factorization. In *International Conference on Machine Learning*, pages 1737–1746, 2016. 2, 3

[22] S. Peter, E. Kirschbaum, M. Both, L. Campbell, B. Harvey, C. Heins, D. Durstewitz, F. Diego, and F. A. Hamprecht. Sparse convolutional coding for neuronal assembly detection. In *Advances in Neural Information Processing Systems*, pages 3678–3688, 2017. 1

[23] S. J. Reddi, S. Sra, B. Póczos, and A. Smola. Stochastic frank-wolfe methods for nonconvex optimization. In *Communication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference on*, pages 1244–1251. IEEE, 2016. 3

[24] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1-2):433–484, 2016. 7

[25] A. Serrano, F. Heide, D. Gutierrez, G. Wetzstein, and B. Masia. Convolutional sparse coding for high dynamic range imaging. In *Computer Graphics Forum*, volume 35, pages 153–163. Wiley Online Library, 2016. 1

[26] S. Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012. 1

[27] M. Takáč, A. Bijral, P. Richtárik, and N. Srebro. Mini-batch primal and dual methods for svms. *In 30th International Conference on Machine Learning*, 28:537–552, 2013. 7

[28] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Scalable online convolutional sparse coding. *IEEE Transactions on Image Processing*, 2018. 2

[29] B. Wohlberg. Efficient algorithms for convolutional sparse representations. *IEEE Transactions on Image Processing*, 25(1):301–315, 2016. 1, 2, 5

[30] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010. 1, 4