CONVOLUTIONAL FACTOR ANALYSIS INSPIRED COMPRESSIVE SENSING

Xin Yuan[1] *and Yunchen Pu*[2]

[1]Nokia Bell Labs, 600 Montain Avenue, Murray Hill, NJ, 07974, USA
[2]Department of ECE, Duke University, Durham, NC, 27708, USA

ABSTRACT

We solve the compressive sensing problem via convolutional factor analysis, where the convolutional dictionaries are learned *in situ* from the compressed measurements. An alternating direction method of multipliers (ADMM) paradigm for compressive sensing inversion based on convolutional factor analysis is developed. The proposed algorithm provides reconstructed images as well as features, which can be directly used for recognition ($e.g.$, classification) tasks. We demonstrate that using $\sim 30\%$ (relative to pixel numbers) compressed measurements, the proposed model achieves the classification accuracy comparable to the original data on MNIST.

Index Terms— Deconvolutional network, convolutional factor analysis, compressive sensing, compressive imaging, image processing

1. INTRODUCTION

The compressive sensing (CS) problem [1–3] can be formulated as:

$$\min \frac{1}{2}\|\boldsymbol{y}-\mathbf{A}\boldsymbol{x}\|_2^2+\lambda\|\boldsymbol{c}\|_\star, \qquad \text{s.t. } \boldsymbol{x}=\mathbf{B}\boldsymbol{c}, \tag{1}$$

where $\mathbf{A}\in\mathbb{R}^{M\times N}$ is the sensing matrix and usually $M\ll N$. $\boldsymbol{x}$ is the desired signal, $\boldsymbol{c}$ denotes the coefficients which are sparse ($\|\cdot\|_\star=\|\cdot\|_1$) [4] or low rank ($\|\cdot\|_\star$ symbolizes the nuclear norm) [5,6], and given $\boldsymbol{c}$, we can recover $\boldsymbol{x}$ via $\mathbf{B}$. This $\mathbf{B}$ can be *known a priori*, (*e.g.*, a wavelet or DCT basis) or learned from the measurement $\boldsymbol{y}$ during reconstruction. λ is a parameter to balance the two terms in (1).

There has been over a decade research on CS, both on theory and applications. Various algorithms have been proposed [7–9] for CS inversion. On the other hand, deep learning methods, especially the convolutional [10] and deconvolutional networks [11], have achieved excellent recognition results on benchmark datasets [12–14]. The convolutional networks, used in a supervised manner, usually aim to extract features to achieve high classification performance. By contrast, the deconvolutional networks [11, 15, 16], used in an *unsupervised* manner, aim to reconstruct the input signals (*e.g.*, minimize the reconstruction error to the input images), as well as extracting features. Therefore, it is applicable to use this deconvolutional network to solve the CS problem in (1).

Most existing dictionary learning algorithms learn dictionaries on small patches [17, 18]. Compressive sensing, however, usually imposes compression on the entire image [19–21]. Similarly, the convolutional factor analysis (CFA) [15, 16] models learn dictionaries on entire images, too. Thereby, it is feasible and appropriate to leverage this CFA technique to reconstruct desired signals $\boldsymbol{x}$ from compressed measurements $\boldsymbol{y}$. Regarding the regularizer, in CFA, the coefficients (a.k.a., features) are usually imposed to be sparse and therefore the ℓ_1-norm is utilized in (1).

This paper makes the following contributions: i) A new convolutional factor analysis algorithm based on *compressed measurements* is developed using the alternating direction method of multipliers (ADMM) paradigm [22]. ii) As the features are obtained during reconstruction, our algorithm provides features simultaneously with reconstruction results. Therefore, joint classification and reconstruction is straightforward. We demonstrate that using $\sim 30\%$ (relative to pixel numbers) compressed measurements, we can achieve the classification performance comparable to the original data on MNIST.

2. CONVOLUTIONAL FACTOR ANALYSIS VIA ADMM

Considering the image case investigated in CS, let $\mathbf{X}_n\in\mathbb{R}^{N_x\times N_y\times N_c}$ denote the three-dimensional (3D) image, which can be a gray-scale image ($N_c=1$), an RGB image ($N_c=3$), or a hyperspectral image [23] (N_c denoting the spectral channels). Under the convolutional factor model, we jointly consider N images, and for n^{th} image

$$\mathbf{X}_n=\textstyle\sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}+\mathbf{E}_n, \tag{2}$$

where $\mathbf{D}_k\in\mathbb{R}^{n_x\times n_y\times N_c}$ is the convolutional dictionary (kernels or filters), $\mathbf{S}_{k,n}\in\mathbb{R}^{(N_x+n_x-1)\times(N_y+n_y-1)\times N_c}$ denotes the coefficients (features) and $\mathbf{E}_n$ signifies the residual or noise. The two-dimensional convolution '$*$' is performed on each slice ($n_c=1,\ldots,N_c$) of $\mathbf{D}_k$ and $\mathbf{S}_{k,n}$. Note the spatial size of $\mathbf{S}_{k,n}$ is $(N_x+n_x-1)\times(N_y+n_y-1)$ such that the image $\mathbf{X}_n$ will be of 'valid' size after convolution. As mentioned before, this is different from the convolutional neural

networks, which impose the features to be of valid size. We develop the CFA algorithm based on ADMM below, which is different from [11]. Note that the dictionaries $\{\mathbf{D}_k\}_{k=1}^K$ are shared across N images, while the features $\{\mathbf{S}_{n,k}\}_{k,n=1}^{N,K}$ vary for each image.

Sparsity is imposed on $\mathbf{S}$ to solve the problem. Without considering compressive sensing, the problem during training can be modeled as

$$\min \ \frac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \lambda\|\mathbf{S}\|_1, \quad (3)$$

where $\|\mathbf{S}\|_1 = \sum_{n,k}\|\mathbf{S}_{k,n}\|_1$. Equation (3) results in the following objective function

$$\mathcal{L}(\mathbf{D},\mathbf{S},\lambda) = \frac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \lambda\sum_{n,k}\|\mathbf{S}_{k,n}\|_1. \quad (4)$$

In order to simplify the problem, we introduce an auxiliary variable $\mathbf{Z}$, and the problem in (3) can be formulated as

$$\min \ \tfrac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \lambda\|\mathbf{Z}\|_1, \quad (5)$$
$$\text{s.t. } \mathbf{Z} = \mathbf{S}. \quad (6)$$

Consider the Lagrange multiplier $\{\mathbf{V}, \eta\}$ and denote $\boldsymbol{s} = \text{vec}(\mathbf{S}), \boldsymbol{z} = \text{vec}(\mathbf{Z}), \boldsymbol{v} = \text{vec}(\mathbf{V})$. This leads to another objective function

$$\mathcal{L}(\mathbf{D},\mathbf{S},\mathbf{Z},\mathbf{V},\lambda,\eta) = \tfrac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \lambda\|\mathbf{Z}\|_1 + \tfrac{\eta}{2}\|\mathbf{S}-\mathbf{Z}\|_2^2 + \boldsymbol{v}^\top(\boldsymbol{s}-\boldsymbol{z}). \quad (7)$$

Define $\mathbf{U} = (1/\eta)\mathbf{V}$, and we have

$$\mathcal{L}(\mathbf{D},\mathbf{S},\mathbf{Z},\mathbf{U},\lambda,\eta) = \tfrac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \lambda\|\mathbf{Z}\|_1 + \tfrac{\eta}{2}\|\mathbf{S}-\mathbf{Z}+\mathbf{U}\|_2^2 - \tfrac{\eta}{2}\|\mathbf{U}\|_2^2. \quad (8)$$

ADMM cyclically solves (8) via the following sub problems:

$$\mathbf{D}^{t+1} := \arg\min_{\mathbf{D}}\Big(\frac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2\Big) \quad (9)$$

$$\mathbf{S}^{t+1} := \arg\min_{\mathbf{S}}\big(\tfrac{1}{2}\sum_{n=1}^N \|\mathbf{X}_n - \sum_{k=1}^K \mathbf{D}_k * \mathbf{S}_{k,n}\|_2^2 + \tfrac{\eta}{2}\|\mathbf{S}-\mathbf{Z}+\mathbf{U}\|_2^2\big) \quad (10)$$

$$\mathbf{Z}^{t+1} := \arg\min_{\mathbf{Z}}\ (\lambda\|\mathbf{Z}\|_1 + \tfrac{\eta}{2}\|\mathbf{S}-\mathbf{Z}+\mathbf{U}\|_2^2) \quad (11)$$

$$\mathbf{U}^{t+1} := \mathbf{U}^t + \eta(\mathbf{S}-\mathbf{Z}) \quad (12)$$

where t denotes the iteration. Below we solve these subproblems one by one.

1) Eq. (9) can be solved by gradient descent. Via the following definitions: a) $\boldsymbol{x}_n = \text{vec}(\mathbf{X}_n)$, b) $\boldsymbol{d}_k = \text{vec}(\mathbf{D}_k)$, c) $\mathbf{F}_{k,n}$ being the 2D sparse convolution matrix that implements $\mathbf{D}_k * \mathbf{S}_{k,n} = \mathbf{F}_{k,n}\boldsymbol{d}_k$, and d)

$$\boldsymbol{x}_n^{-k} = \boldsymbol{x}_n - \sum_{k'=1,k'\neq k}^K \mathbf{F}_{k',n}\boldsymbol{d}_{k'}, \quad (13)$$

taking derivative to $\boldsymbol{d}_k$ in (9), we have

$$\frac{\partial(\frac{1}{2}\sum_{n=1}^N \|\boldsymbol{x}_n - \sum_{k=1}^K \mathbf{F}_{k,n}\boldsymbol{d}_k\|_2^2)}{\partial \boldsymbol{d}_k} = -\sum_{n=1}^N \mathbf{F}_{k,n}^\top(\boldsymbol{x}_n^{-k} - \mathbf{F}_{k,n}\boldsymbol{d}_k). \quad (14)$$

Therefore,

$$\boldsymbol{d}_k^{t+1} = \boldsymbol{d}_k^t + \beta\sum_{n=1}^N \mathbf{F}_{k,n}^\top(\boldsymbol{x}_n^{-k} - \mathbf{F}_{k,n}\boldsymbol{d}_k), \quad (15)$$

where β is the learning rate.

2) Eq. (10) is a quadratic optimization problem and can be simplified via the following definitions: a) $\boldsymbol{s}_{k,n} = \text{vec}(\mathbf{S}_{k,n})$ and b) $\mathbf{T}_{k,n}$ being the 2D sparse convolution matrix that implements $\mathbf{D}_k * \mathbf{S}_{k,n} = \mathbf{T}_k\boldsymbol{s}_{k,n}$. Since the feature is unique for each image, given $\{\mathbf{X}_n, \mathbf{D}, \mathbf{Z}_n, \mathbf{U}_n\}$, we define

$$\mathcal{C}_n \overset{\text{def}}{=} \tfrac{1}{2}\|\boldsymbol{x}_n - \sum_{k=1}^K \mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \tfrac{\eta}{2}\sum_k \|\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}\|_2^2.$$

Taking derivative to $\boldsymbol{s}_{k,n}$, we have

$$\frac{\partial \mathcal{C}_n}{\partial \boldsymbol{s}_{k,n}} = \eta(\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}) - \mathbf{T}_k^\top(\boldsymbol{x}_n - \sum_{k=1}^K \mathbf{T}_k\boldsymbol{s}_{k,n}). \quad (16)$$

Denote $\boldsymbol{x}_n^{-k} = \boldsymbol{x}_n - \sum_{k'=1,k'\neq k}^K \mathbf{T}_{k'}\boldsymbol{s}_{k',n}$, setting $\frac{\partial \mathcal{C}_n}{\partial \boldsymbol{s}_{k,n}} = (\mathbf{T}_k^\top\mathbf{T}_k + \eta\mathbf{I})\boldsymbol{s}_{k,n} - (\mathbf{T}_k^\top\boldsymbol{x}_n^{-k} + \eta\boldsymbol{z}_{k,n} - \eta\boldsymbol{u}_{k,n}) = 0$. Given $\{\mathbf{T}, \boldsymbol{z}, \boldsymbol{u}\}$, the optimal $\boldsymbol{s}_{k,n}$ is the solution of the following linear system

$$(\mathbf{T}_k^\top\mathbf{T}_k + \eta\mathbf{I})\boldsymbol{s}_{k,n} = \mathbf{T}_k^\top\boldsymbol{x}_n^{-k} + \eta(\boldsymbol{z}_{k,n} - \boldsymbol{u}_{k,n}), \quad (17)$$

which can be solved effectively using conjugate gradient (CG) algorithms [24].

We list the implementation details in MATLAB:

$$\mathbf{F}_{k,n}\boldsymbol{d}_k = \mathbf{D}_k * \mathbf{S}_{k,n} = \text{conv2}(\mathbf{S}_{k,n}, \mathbf{D}_k, \text{'valid'}) \quad (18)$$

$$\mathbf{F}_{k,n}^\top\boldsymbol{x}_n = \text{conv2}(\text{rot90}(\mathbf{S}_{k,n}, 2), \mathbf{X}_n, \text{'valid'}) \quad (19)$$

$$\mathbf{T}_k\boldsymbol{s}_{k,n} = \mathbf{D}_k * \mathbf{S}_{k,n} = \text{conv2}(\mathbf{S}_{k,n}, \mathbf{D}_k, \text{'valid'}) \quad (20)$$

$$\mathbf{T}_k^\top\boldsymbol{x}_n = \text{conv2}(\mathbf{X}_n, \text{rot90}(\mathbf{D}_k, 2), \text{'full'}) \quad (21)$$

We also found that using "fft2( )" in MATLAB is at least $4\times$ faster than "conv2( )" by providing almost the same results.

3) Eq. (11) can be solved via the shrinkage operator, *i.e.*, soft thresholding

$$\mathbf{Z}^{t+1} = \text{soft}(\mathbf{S}^t + \tfrac{\mathbf{U}^t}{\eta}, \boldsymbol{\gamma}^t), \quad (22)$$

which can be performed element-wise, *i.e.*, for i^{th} element,

$$z_i^{t+1} = \text{sign}(s_i^t + \tfrac{u_i^t}{\eta})\ \max(|s_i^t + \tfrac{u_i^t}{\eta}| - \gamma_i^t, 0), \quad (23)$$

and this threshold $\boldsymbol{\gamma}^t$ can be updated in each iteration, thus *adaptively* soft thresholding. Note that λ in (11) can be normalized and thus observed in η. In our experiments, we set the number of the non-zero elements (thus sparsity) in $\mathbf{Z}$ to be the same [21] in each iteration and update this $\boldsymbol{\gamma}^t$ based on $(\mathbf{S}^t + \mathbf{U}^t/\eta)$.

3. CFA BASED COMPRESSIVE SENSING

We now extend the CFA model to the CS scenario. Consider N images jointly with the same sensing matrix $\mathbf{A}$, (which can also be different for each image).

$$\boldsymbol{y}_n = \mathbf{A}\boldsymbol{x}_n, \tag{24}$$

where $\boldsymbol{x}_n$ denotes the n^{th} vectorized image. Similar to (1), using the CFA model, the problem can be formulated as

$$\min \ \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\boldsymbol{x}_n\|_2^2 + \lambda\sum_{n=1}^{N}\sum_{k=1}^{K}\|\boldsymbol{s}_{k,n}\|_1, \tag{25}$$

$$\text{s.t. } \mathbf{X}_n = \textstyle\sum_{k=1}^{K}\mathbf{D}_k * \mathbf{S}_{k,n}. \tag{26}$$

Employing the definition in (20), and using the vectorization forms, we have

$$\min \ \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\boldsymbol{x}_n\|_2^2 + \lambda\sum_{n=1}^{N}\sum_{k=1}^{K}\|\boldsymbol{s}_{k,n}\|_1, \tag{27}$$

$$\text{s.t. } \boldsymbol{x}_n = \textstyle\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}. \tag{28}$$

This leads to the following objective function

$$\mathcal{L}(\mathbf{T},\mathbf{S},\lambda) = \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{n,k}\|\boldsymbol{s}_{k,n}\|_1. \tag{29}$$

We consider to solve (29) in two ways:

a) The convolutional dictionary ($\{\mathbf{D}_k\}_{k=1}^K$ or $\{\mathbf{T}_k\}_{k=1}^K$) is pre-learned by training data. In this case, (29) aims to solve

$$\arg\min_{\{\boldsymbol{s}\}} \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{n,k}\|\boldsymbol{s}_{k,n}\|_1. \tag{30}$$

b) The convolutional dictionary ($\{\mathbf{D}_k\}_{k=1}^K$ or $\{\mathbf{T}_k\}_{k=1}^K$) is unknown *a priori* and will be learned *in situ* from the raw measurements $\{\boldsymbol{y}_n\}_{n=1}^N$. In this case, (29) aims to solve

$$\arg\min_{\{\mathbf{T},\boldsymbol{s}\}} \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{n,k}\|\boldsymbol{s}_{k,n}\|_1. \tag{31}$$

3.1. CS Inversion with Pre-Learned Dictionary

In this case, we aim to solve $\{\boldsymbol{s}_{k,n}\}_{n,k=1}^{N,K}$ given $\{\boldsymbol{y}_n\}_{n=1}^N$ and $\{\mathbf{T}_k\}_{k=1}^K$. We again employ the ADMM framework to solve this problem. Recall (5) and introducing the auxiliary variable $\{\boldsymbol{z}_{n,k}\}_{n,k=1}^{N,K}$, we have the following problem

$$\min \ \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{k,n}\|\boldsymbol{z}_{k,n}\|_1,$$

$$\text{s.t. } \boldsymbol{s}_{k,n} = \boldsymbol{z}_{k,n}, \ \ \forall k,n. \tag{32}$$

This results in the objective function

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{z},\boldsymbol{u},\lambda,\eta) = \tfrac{1}{2}\textstyle\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{k,n}\|\boldsymbol{z}_{k,n}\|_1 + \frac{\eta}{2}\sum_{k,n}\|\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}\|_2^2 + \text{Const.} \tag{33}$$

ADMM cyclically solves (33) via the following sub-problems:

$$\boldsymbol{s}^{t+1} := \arg\min_{\boldsymbol{s}} (\tfrac{1}{2}\textstyle\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \tfrac{\eta}{2}\sum_{n,k}\|\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}\|_2^2) \tag{34}$$

$$\boldsymbol{z}^{t+1} := \arg\min_{\boldsymbol{z}} (\lambda\sum_{k,n}\|\boldsymbol{z}_{k,n}\|_1 + \tfrac{\eta}{2}\textstyle\sum_{k,n}\|\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}\|_2^2) \tag{35}$$

$$\boldsymbol{u}^{t+1} := \boldsymbol{u}^t + \eta(\boldsymbol{s} - \boldsymbol{z}) \tag{36}$$

Note that the updates of $\boldsymbol{z},\boldsymbol{u}$ in (35-36) are the same as in (11)-(12). For (34), it is different from (9). Following similar derivations, via defining

$$\boldsymbol{y}_n^{-k} \stackrel{\text{def}}{=} \boldsymbol{y}_n - \mathbf{A}\textstyle\sum_{k'=1,k'\neq k}^{K}\mathbf{T}_{k'}\boldsymbol{s}_{k',n},$$

given $\{\mathbf{T},\boldsymbol{z},\boldsymbol{u},\mathbf{A}\}$, the optimal $\boldsymbol{s}_{k,n}$ is the solution of the following linear system

$$(\mathbf{T}_k^\top\mathbf{A}^\top\mathbf{A}\mathbf{T}_k + \eta\mathbf{I})\boldsymbol{s}_{k,n} = \mathbf{T}_k^\top\mathbf{A}^\top\boldsymbol{y}_n^{-k} + \eta(\boldsymbol{z}_{k,n} - \boldsymbol{u}_{k,n}). \tag{37}$$

Again, this can be solved effectively via CG algorithms.

Algorithm 1 CS-CFA

Require: Input measurements $\{\boldsymbol{y}_n\}_{n=1}^N$, sensing matrix $\mathbf{A}$, parameters $\{\beta,\eta\}$

1: Initialize $\mathbf{D},\mathbf{S},\mathbf{Z},\mathbf{U}$.
2: **for** $t = 1$ **to** MaxIter **do**
3: Update $\mathbf{D}$ by Eq. (39).
4: Update $\mathbf{S}$ by Eq. (37).
5: Update $\mathbf{Z}$ by shrinkage operator, Eq. (23).
6: Update $\mathbf{U}$ by Eq. (36).
7: **end for**

3.2. Learning Dictionary from Measurements

When the pre-learned dictionary is not available, we need to learn the dictionary from the raw measurements $\{\boldsymbol{y}_n\}_{n=1}^N$. The problem in Eq. (31) results in the following objective function

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{z},\boldsymbol{u},\mathbf{T},\lambda,\eta) = \tfrac{1}{2}\textstyle\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K}\mathbf{T}_k\boldsymbol{s}_{k,n}\|_2^2 + \lambda\sum_{k,n}\|\boldsymbol{z}_{k,n}\|_1 + \frac{\eta}{2}\sum_{k,n}\|\boldsymbol{s}_{k,n} - \boldsymbol{z}_{k,n} + \boldsymbol{u}_{k,n}\|_2^2 + \text{Const.}$$

In addition to the subproblems described in (34)-(36), we also need to update $\{\mathbf{D}_k\}_{k=1}^K$,

$$\boldsymbol{d}^{t+1} := \arg\min_{\boldsymbol{d}} (\frac{1}{2}\sum_{n=1}^{N} \|\boldsymbol{y}_n - \mathbf{A}\sum_{k=1}^{K} \mathbf{F}_{k,n}\boldsymbol{d}_k\|_2^2). \quad (38)$$

Similar to (9), this can be solved by the gradient descent:

$$\boldsymbol{d}_k^{t+1} = \boldsymbol{d}_k^t + \beta \textstyle\sum_{n=1}^{N} \mathbf{F}_{k,n}^\top \mathbf{A}^\top (\boldsymbol{y}_n^{-k} - \mathbf{A}\mathbf{F}_{k,n}\boldsymbol{d}_k), \quad (39)$$

where $\boldsymbol{y}_n^{-k} = \boldsymbol{y}_n - \mathbf{A}\sum_{k'=1,k'\neq k}^{K} \mathbf{F}_{k',n}\boldsymbol{d}_{k'}$. The complete algorithm of compressive sensing convolutional factor analysis (CS-CFA) is summarized in Algorithm 1.

The CS-CFA model can be extended to a deep (multi-layer) deconvolutional networks using the recently proposed stochastic "unpooling" approach in [15], which imposes that inside each pooling block, there is at most one nonzero element. Therefore, this "unpooling" process does not lose any information from the bottom layer (touching the data) to the top layer. Details and results will be reported elsewhere [25].

4. JOINT RECONSTRUCTION AND CLASSIFICATION

The deep convolutional and deconvolutional networks are used to extract features $\{\mathbf{S}^{(\ell)}\}_{\ell=1}^L$ and then these features are used for classification. In our work, we have developed a deep CS-CFA model and these features are already obtained during reconstruction. Indeed, after we get these features, we reconstruct images as

$$\hat{\mathbf{X}}_n = \textstyle\sum_{k_1=1}^{K} \mathbf{D}_{k_1}^{(1)} * \mathbf{S}_{k_1,n}^{(1)}. \quad (40)$$

When performing classification, we can either superimpose a layer on the top of the CS-CFA model or use a separate classifier, *e.g.*, employing a support vector machine (SVM) on features trained by our model during reconstruction.

For joint classification and reconstruction task, we have labeled (compressed) data $\{\boldsymbol{y}_n, c_n\}_{n=1}^N$, where $c_n = \{1, \ldots, C\}$ considering C classes in total. Introducing the classifier weight matrix $\mathbf{H} \in \mathbb{R}^{C\times N_s}$ and the bias vector $\boldsymbol{\alpha} \in \mathbb{R}^{N_s}$, we have $p(c_n = i|\boldsymbol{s}_n, \mathbf{H}, \boldsymbol{\alpha}) = \text{softmax}_i(\boldsymbol{h}_i\boldsymbol{s}_n^{(L)} + \boldsymbol{\alpha}) = \frac{\exp(\boldsymbol{h}_i\boldsymbol{s}_n^{(L)}+\alpha_i)}{\sum_j \exp(\boldsymbol{h}_j\boldsymbol{s}_n^{(L)}+\alpha_j)}$, where $\boldsymbol{h}_i$ denotes the i^{th} row of the weight matrix $\mathbf{H}$ and α_i symbolizes the i^{th} element of the vector $\boldsymbol{\alpha}$. These weights $\{\mathbf{H}, \boldsymbol{\alpha}\}$ can be learned jointly with the CFA network, thus constituting a supervised CS-CFA model. Similarly, a C-class SVM can also be used [26].

5. RESULTS

The MNIST [27] data has 60,000 training and 10,000 testing images, each 28×28, for digits 0 through 9. We randomly select 100 digits (10 for each) for CS reconstruction. We conduct the experiments by both training the convolutional dictionary $\mathbf{D}$ from another set of digits, and learning $\mathbf{D}$ from the measurements directly, thus *in situ*. The dictionary size is set to $7 \times 7 \times 16$ and a single layer model is used.

The average PSNR of the reconstructed digits versus CSr is plotted in Figure 1. TwIST [7] , and GAP [28, 29] are used as baselines. It can be observed that our proposed algorithm, both learned dictionary *in situ* and with pre-learned dictionary, performs better than other algorithms. If training data are available, the algorithm performs best, $i.e.$, a 2dB increase compared to *in situ* learned dictionary.

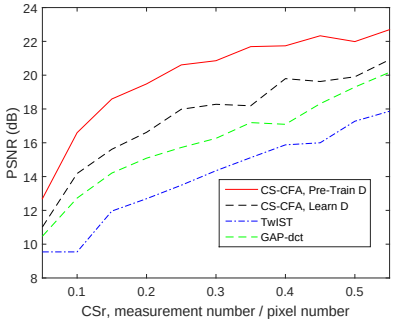

Fig. 1. Average PSNR of reconstructed results vs. CSr with different algorithms on MNIST.

Table 1. Reconstruction PSNR (dB) and classification accuracy (%) at various CSr on MNIST.

CSr	0.1	0.2	0.3	0.4	0.5	no CS
Reconstruction PSNR	15.65	19.73	21.55	22.51	23.53	-
Softmax	75.88	90.88	92.89	93.78	94.47	93.84
Linear SVM	70.30	88.57	91.64	93.45	93.79	92.13
Nonlinear SVM	77.62	92.38	94.93	95.29	96.37	96.32

Next we conduct our model on the complete MNIST dataset for joint reconstruction and classification. We perform our CFA model on the 60000 training digits to extract features and these features are sent to classifiers for training. During testing, the features are extracted directly from the *compressed measurements* of 10000 digits. The reconstruction and classification results are summarized in Table 1. It can be observed that our model can simultaneously reconstruct the images and classify the digits using *compressed measurements*. According to [30], the classification accuracy of the original data using softmax is 92.6%, while we have achieved 92.89% when CSr = 0.3. For comparison, we also present the classification results of both linear and nonlinear SVM [31, 32]. Without comparing these different classifiers, we observe that for every classier, we can achieve comparable classification result to the original data with $30 \sim 40\%$ compressed measurements, which reduced more than half of the data volume. We further notice that due to the randomness introduced by the compressive sensing matrix, the classification results are improved compared to the original data. Specifically, the softmax classification accuracy is 93.84% using CFA features extracted from the original data, but we get $> 94\%$ accuracies using CFA features extracted from the *compressed* data when CSr> 0.4. This is in agreement with the theory recently developed in [33].

6. REFERENCES

[1] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, 2006.

[2] E.J. Candes, M.B. Wakin, and S.P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 877–905, 2008.

[3] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, 2006.

[4] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution," *CPAM*, 2006.

[5] W. Dong, G. Shi, X. Li, Y. Ma, and F. Huang, "Compressive sensing via nonlocal low-rank regularization," *IEEE Transactions on Image Processing*, vol. 23, no. 8, pp. 3618–3632, 2014.

[6] X. Yuan, H. Jiang, G. Huang, and P. Wilford, "Compressive sensing via low-rank Gaussian mixture models," *arXiv:1508.06901*, 2015.

[7] J.M. Bioucas-Dias and M.A.T. Figueiredo, "A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2992–3004, December 2007.

[8] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," pp. 586–597, Dec. 2007.

[9] X. Yuan, V. Rao, S. Han, and L. Carin, "Hierarchical infinite divisibility for multiscale shrinkage," *IEEE Transactions on Signal Processing*, vol. 62, no. 17, pp. 4363–4374, Sep. 1 2014.

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, 1989.

[11] M. D. Zeiler, D. Kirshnan, G. Taylor, and R. Fergus, "Deconvolutional networks," *CVPR*, 2010.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[14] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," in *Advances in Neural Information Processing Systems 28*, 2016.

[15] Y. Pu, X. Yuan, A. Stevens, C. Li, and L. Carin, "A deep generative deconvolutional image model," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[16] B. Chen, G. Polatkan, G. Sapiro, D. M. Blei, D. B. Dunson, and L. Carin, "Deep learning with hierarchical convolutional factor analysis," *IEEE T-PAMI*, 2013.

[17] 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 (ICML)*, 2009, pp. 689–696.

[18] G. Yu and G. Sapiro, "Statistical compressed sensing of Gaussian mixture models," *IEEE Transactions on Signal Processing*, vol. 59, no. 12, pp. 5842–5858, 2011.

[19] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[20] G. Huang, H. Jiang, K. Matthews, and P. Wilford, "Lensless imaging by compressive sensing," *IEEE International Conference on Image Processing*, 2013.

[21] X. Yuan, H. Jiang, G. Huang, and P. Wilford, "SLOPE: Shrinkage of local overlapping patches estimator for lensless compressive imaging," *IEEE Sensors Journal*, vol. 16, no. 22, pp. 8091–8102, November 2016.

[22] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, January 2011.

[23] X. Yuan, T.-H. Tsai, R. Zhu, P. Llull, D. J. Brady, and L. Carin, "Compressive hyperspectral imaging with side information," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 6, pp. 964–976, September 2015.

[24] J. R Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Tech. Rep., Pittsburgh, PA, USA, 1994.

[25] X. Yuan, Y. Pu, and L. Carin, "Compressive sensing via convolutional factor analysis," *arXiv:1701.03006*, 2017.

[26] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," *CVPR*, 2009.

[27] MNIST, "{http://yann.lecun.com/exdb/mnist/}," .

[28] X. Liao, H. Li, and L. Carin, "Generalized alternating projection for weighted-$\ell_{2,1}$ minimization with applications to model-based compressive sensing," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, pp. 797–823, 2014.

[29] X. Yuan, "Generalized alternating projection based total variation minimization for compressive sensing," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2539–2543.

[30] UFLDL, "http://ufldl.stanford.edu/wiki/index.php/Exercise:Softmax_Regression," .

[31] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, 2011.

[32] R. Henao, X. Yuan, and L. Carin, "Bayesian nonlinear SVMs and factor modeling," *NIPS*, 2014.

[33] J. Huang, Q. Qiu, R. Calderbank, M. Rodrigues, and G. Sapiro, "Alignment with intra-class structure can improve classification," in *ICASSP*, April 2015, pp. 1921–1925.