ADMM-Net for Robust Compressive Sensing Image Reconstruction in the Presence of Symmetric α-Stable Noise

Yunyi Li, Liya Huang, Yue Yin, Yu Wang and Guan Gui
Nanjing University of Posts and Telecommunications, Nanjing 210003, China.
E-mails: huangly@njupt.edu.cn, guiguan@njupt.edu.cn

***Abstract*—This work aims at reconstructing image from compressed-measured data in the presence of symmetric α-stable (SαS) noise. We first employ the ℓ_1-norm as the estimator to depress the influence of SαS noise, and then the ADMM framework is employed to address the resulting optimization problem. Moreover, a smoothing strategy is adopted to address the ℓ_1-norm resulting in a nonsmooth optimization problem. To exploit more prior knowledge and image features, a robust composite regularization model is proposed for training by deep neural network (DNN). In the training phase, the DNN can be utilized to train the samples for the optimal parameters, the optimal shrinkage function and the optimal transform domain. Experiments show that our proposed algorithm can obtain higher reconstruction Peak Signal to Noise Ratio (PSNRs) than some existing state-of-the-art robust CS methods.**

I. Introduction

Compressive sensing (CS) [1] is an emerging promising approach that aims for accurate acquisition and reconstruction of the sparse signal (e.g., image) from a small amount of sub-Nyquist sampling data. Typical applications include magnetic resonance imaging (MRI) [2], radar imaging [3], and hyper-spectral imaging [4]. The basic linear observation system can be formulated as follows:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n} \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^{M\times 1}$ denotes the observation data, $\mathbf{A} \in \mathbb{R}^{M\times N}, (M \ll N)$ represents the linear operator or random sampling matrix, $\mathbf{x} \in \mathbb{R}^{N\times 1}$ is the desired signal vector that is sparse in some transform domain, and $\mathbf{n} \in \mathbb{R}^{M\times 1}$ is often considered Gaussian with the bounded norm $\|\mathbf{n}\|_2 \leqslant \xi$. The CS theory states that, if the desired unknown signal is inherently, then the ill-posed problem of recovering $\mathbf{x}$ from $\mathbf{y}$ can be accurately addressed by

$$\hat{\mathbf{x}} \leftarrow \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda g(\mathbf{x}) \tag{2}$$

where the ℓ_2-norm term $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$ is the observation fidelity term that ensures the concurrence of $\mathbf{y}$ and $\mathbf{x}$, and λ is the regularization parameter. The penalty function $g(\mathbf{x})$ usually provides prior knowledge for the optimization problem via a norm function

$$g(\mathbf{x}) = \|\mathbf{D}\mathbf{x}\|_p^p = \sum_{i=1}^{n} |(\mathbf{D}\mathbf{x})_i|^p,\ 0 \leqslant p \leqslant 1 \tag{3}$$

Here, $\mathbf{D}$ denotes the sparsifying transform operator.

Symmetric α-stable (SαS) noise is a typical impulsive noise often generated in signal/information measurement and transmission systems that will cause the traditional CS reconstruction algorithms in (2) to degrade severely. To suppress the outliers caused by SαS noise, one popular effective optimization model employs the ℓ_1-norm as the metric for the residual error by

$$\hat{\mathbf{x}} \leftarrow \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_1 + \lambda g(\mathbf{x}) \tag{4}$$

The CS formulation in (4) is known as robust CS, compared with the quadratic estimator of ℓ_2-norm in (2), is more suited for using the ℓ_1-norm to model large outliers; hence, it has been widely used in designing robust CS algorithms. However, optimization of the objective function is intractable because of the resulting nonsmooth cost function term of the ℓ_1-norm. Among all the existing effective robust recovery algorithms, the alternating direction method of multipliers (ADMM) framework is regarded as one of the more effective and efficient approaches [5]–[7]. Using an operator splitting [8], this framework suggests to separate the regularization term by an additional variable. However, it is intractable to tune these parameters, e.g., penalty parameter and regularization parameter. What's more, it is also challenging in CS to choose optimal sparsifying transform domain and the corresponding regularization function $g(\mathbf{x})$. Inspired by recently work of ADMM-net for CS reconstruction under Gaussian-assumption environment [9], in this paper, we propose a robust-ADMM-Net to address the reconstruction problem in the presence of SαS noise. The proposed robust-ADMM-net algorithm uses the deep neural network method to train the corrupted image data for optimal parameters. In addition, the ADMM-net can also be utilized for training the shrinkage function as well as the sparsifying transform dictionary $\mathbf{D}$. Experiments results demonstrate that the proposed robust CS reconstruction approach can outperform the state-of-the-art CS robust methods.

II. Proposed Robust-ADMM-net framework

A. Robust ADMM Framework

ADMM is a simple and powerful framework for the high-dimension optimization problem in machine learning and signal processing that adopts a variable-splitting strategy to separate coupled components via auxiliary variables [10]. In

this paper, we design the following robust optimization model by

$$\hat{\mathbf{x}} \leftarrow \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1 + \sum_{l=1}^{L} \lambda_l g(\mathbf{D}_l\mathbf{x}) \tag{5}$$

where $g(\cdot)$ denotes the regularization function, such as ℓ_p -norm ($0 < p \leq 1$), $\mathbf{D}_l$ denotes the sparsifying transform dictionary (e.g., sines, wavelet bases), with examples of a typical sparsifying transform including the Discrete Cosine Transform (DCT) and the Discrete Wavelet Transform (DWT); and λ_l denotes the regularization parameter. Unlike the optimization problem (4), the advantage of this model can obtain more physical mechanism and prior knowledge for optimization [11].

To solve the above optimization problem (5) based on the ADMM framework, we typically use an auxiliary variable $\mathbf{z}_l \in \mathbb{R}^{N\times 1}$, then the optimization problem (5) can be rewritten as

$$\begin{aligned} &\arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-y\|_1 + \sum_{l=1}^{L} \lambda_l g(\mathbf{z}_l) \\ &s.t.\ \mathbf{z_l}{=}\mathbf{D_l}\mathbf{x},\ \forall l \in [1,2,\cdots,L]. \end{aligned} \tag{6}$$

The augmented Lagrangian of problem (6) is

$$\begin{aligned} L(\mathbf{x},\mathbf{z},\alpha) &= \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1 + \sum_{l=1}^{L} \lambda_l g(\mathbf{z}_l) \\ &- \sum_{l=1}^{L} \langle \alpha_l, \mathbf{z}_l - \mathbf{D}_l\mathbf{x}\rangle + \sum_{l=1}^{L} \frac{\rho_l}{2} \|\mathbf{z}_l - \mathbf{D}_l\mathbf{x}\|_2^2 \end{aligned} \tag{7}$$

where $\{\alpha_l\} \in \mathbb{R}^{N\times 1}$ are Lagrangian multipliers, $\{\rho_l\} > 0$ are penalty parameters, and $\{\lambda_l\}$ denotes the regularization parameter; these parameters will play critical roles in the optimization process. According to the ADMM framework, we have the following three steps:

$$\begin{aligned} \mathbf{x}^{(n+1)} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1 \\ &- \sum_{l=1}^{L} \left\langle \alpha_l^{(n)}, \mathbf{z}_l^{(n)} - \mathbf{D}_l\mathbf{x} \right\rangle + \sum_{l=1}^{L} \frac{\rho_l}{2} \left\| \mathbf{z}_l^{(n)} - \mathbf{D}_l\mathbf{x} \right\|_2^2 \end{aligned} \tag{8}$$

$$\begin{aligned} \mathbf{z}^{(n+1)} &= \arg\min_{\mathbf{x}} \sum_{l=1}^{L} \lambda_l g(\mathbf{z}_l) \\ &- \sum_{l=1}^{L} \left\langle \alpha_l^{(n)}, \mathbf{z}_l^{(n)} - \mathbf{D}_l\mathbf{x}^{(n+1)} \right\rangle + \sum_{l=1}^{L} \frac{\rho_l}{2} \left\| \mathbf{z}_l - \mathbf{D}_l\mathbf{x}^{(n+1)} \right\|_2^2 \end{aligned} \tag{9}$$

$$\alpha^{(n+1)} = \arg\min_{\mathbf{x}} \sum_{l=1}^{L} \left\langle \alpha_l, \mathbf{D}_l\mathbf{x}^{(n+1)} - \mathbf{z}_l^{(n+1)} \right\rangle \tag{10}$$

The $\mathbf{x}$-step in (8) in fact is a penalized least square (LS) problem. To solve this optimization problem, we first smooth the L_1-norm term $\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1$, specifically, we linearize the term $\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1$ at the given $\tilde{\mathbf{x}}$ as

$$\begin{aligned} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_1 &= \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_{1,\varepsilon} \\ &= \frac{1}{2}\|\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y}\|_{1,\varepsilon} + \frac{1}{2}\langle \mathbf{A}\mathbf{x}-\mathbf{A}\tilde{\mathbf{x}}, d(\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y})\rangle \\ &+ \frac{1}{2\tau}\|\mathbf{A}\mathbf{x}-\mathbf{A}\tilde{\mathbf{x}}\|_2^2 \end{aligned} \tag{11}$$

where $\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_{1,\varepsilon} = \sum_i [(\mathbf{A}\mathbf{x}-\mathbf{y})_i^2 + \varepsilon^2]^{\frac{1}{2}}$, $\varepsilon = 10^{-3}$, $\tau > 0$ is a proximal parameter. Thus, we have

$$\begin{aligned} \mathbf{x}^{(n+1)} &= \arg\min_{\mathbf{x}} + \frac{1}{2}\langle \mathbf{A}\mathbf{x}, d(\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y})\rangle + \frac{1}{2\tau}\|\mathbf{A}\mathbf{x}-\mathbf{A}\tilde{\mathbf{x}}\|_2^2 \\ &- \sum_{l=1}^{L} \left\langle \alpha_l^{(n)}, \mathbf{z}_l^{(n)} - \mathbf{D}_l\mathbf{x} \right\rangle + \sum_{l=1}^{L} \frac{\rho_l}{2} \left\| z_l^{(n)} - \mathbf{D}_l\mathbf{x} \right\|_2^2 \end{aligned} \tag{12}$$

We can obtain the closed-form solution by derivation

$$\begin{aligned} \mathbf{x}^{(n+1)} = \left[\frac{1}{\tau}\mathbf{A}^T\mathbf{A} + \sum_{l=1}^{L} \rho_l \mathbf{D}_l^T\mathbf{D}_l\right]^{-1} \Bigg[&\frac{1}{\tau}\mathbf{A}^T\mathbf{A}\tilde{\mathbf{x}} \\ &- \frac{1}{2}\mathbf{A}^T d(\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y}) + \sum_{l=1}^{L} \rho_l \mathbf{D}_l^T(\mathbf{z}_l - \beta_l)\Bigg] \end{aligned} \tag{13}$$

where $d(\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y})_i = ((\mathbf{A}\tilde{\mathbf{x}}-\mathbf{y})_i^2 + \varepsilon^2)^{-1/2}$. Specifically, in this paper, we set , where $\mathbf{P} \in \mathbb{R}^{N\times N_1}$ denotes the under-sampling matrix, and $\mathbf{F} \in \mathbb{R}^{N_1\times M}$ is a Fourier transform; thus, we have the solution of the $\mathbf{x}$-step,

$$\begin{aligned} \mathbf{x}^{(n+1)} = \mathbf{F}^T\left[\frac{1}{\tau}\mathbf{P}^T\mathbf{P} + \sum_{l=1}^{L} \rho_l \mathbf{F}\mathbf{D}_l^T\mathbf{D}_l\mathbf{F}^T\right]^{-1} \Bigg[&\frac{1}{\tau}\mathbf{P}^T\mathbf{P}\mathbf{F}\tilde{\mathbf{x}} \\ &- \frac{1}{2}\mathbf{P}^T d(\mathbf{P}\mathbf{F}\tilde{\mathbf{x}}-\mathbf{y}) + \sum_{l=1}^{L} \rho_l \mathbf{F}\mathbf{D}_l^T(\mathbf{z}_l - \beta_l)\Bigg] \end{aligned} \tag{14}$$

where $d(\mathbf{P}\mathbf{F}\tilde{\mathbf{x}}-\mathbf{y})_i = ((\mathbf{P}\mathbf{F}\tilde{\mathbf{x}}-\mathbf{y})_i^2 + \varepsilon^2)^{-1/2}$

To simplify the above solution, we specially set $\tilde{\mathbf{x}} = 0$ to simplify our algorithm. In the $n+1$ -th iteration,

$$\begin{aligned} \mathbf{x}^{(n+1)} = \mathbf{F}^T\left[\frac{1}{\tau_1}\mathbf{P}^T\mathbf{P} + \sum_{l=1}^{L} \rho_l \mathbf{F}\mathbf{D}_l^T\mathbf{D}_l\mathbf{F}^T\right]^{-1} \Bigg[&\frac{1}{2}d(\mathbf{P}^T\mathbf{y}) \\ &+ \sum_{l=1}^{L} \rho_l \mathbf{F}\mathbf{D}_l^T(\mathbf{z}_l - \beta_l)\Bigg] \end{aligned} \tag{15}$$

In the first iteration for initialization, we set $\mathbf{z}_l = 0$, $\beta_l = 0$ then we have

$$\mathbf{x}^{(1)} = \mathbf{F}^T\left[\frac{1}{\tau_1}\mathbf{F}^T\mathbf{P}^T\mathbf{P}\mathbf{F} + \sum_{l=1}^{L} \rho_l \mathbf{F}H_l^T H_l\mathbf{F}^T\right]^{-1} \left[\frac{1}{2}d(\mathbf{P}^T\mathbf{y})\right] \tag{16}$$

where $d(\mathbf{p}^T\mathbf{y})_i = ((\mathbf{p}^T\mathbf{y})_i^2 + \varepsilon^2)^{-1/2}$.

The solution of the $\mathbf{z}$-step (9) can be obtained by the shrinkage function determined by $g(\cdot)$, e.g., the L_1-norm

function corresponding with the soft-shrinkage operator [12]. Then, the output of **z**-step can be described as

$$\mathbf{z}_l^{(n+1)} = S(\mathbf{D}_l\mathbf{x}^{(n)} + \beta_l^{(n)}; {}^{\lambda_l}/_{\rho_l}) \tag{17}$$

where $S(\cdot)$ represents the corresponding nonlinear shrinkage function. For initialization, in the first iteration, we set the initial $S(\cdot)$ as the soft-shrinkage operator and set $\mathbf{D}_l$ as the DCT basis. For simplicity, let $\beta_l = {}^{\alpha_l}/_{\rho_l}$, then the α-step is converted into the β-step, and the subproblem have the following solution:

$$\beta_l^{(n+1)} = \beta_l^{(n)} + \eta_l \cdot (\mathbf{D}_l\mathbf{x}^{(n+1)} - \mathbf{z}_l^{(n+1)}) \tag{18}$$

where the parameter η_l denotes the update rate.

B. Deep Neural Network for ADMM framework

Deep learning has be regarded as the representative advance of artificial intelligence, deep learning approaches are capable of extracting features from images for recognition and restoration [13]. To connect the ADMM framework and the deep neural network, we first map the iterations in ADMM to the layers of the deep neural network. If every iteration in ADMM is considered as one layer of the deep neural network, in this case, the above three steps can be regarded as three layers: Reconstruction layer $\mathbf{x}^{(n+1)}$ in (16); Nonlinear transform layer $\mathbf{z}_l^{(n+1)}$ in (17); and the Multiplier update layer $\beta_l^{(n+1)}$ in (18). As mentioned before, finding an optimal transform domain is an active research area because a sparser representation often leads to higher reconstruction accuracy. Some popular sparsifying transforms, such as DCT, Fourier and Haar, are often not optimal. In this paper, we also employ a conventional layer $\mathbf{c}_l^{(n+1)}$ to obtain the optimal sparsifying transform domain [9]; thus, we have the following four layers: (1) Reconstruction layer $\mathbf{x}^{(n+1)}$:

$$x^{(n+1)} = \mathbf{F}^T\left[\frac{1}{\tau_1}\mathbf{P}^T\mathbf{P} + \sum_{l=1}^{L}\rho_l\mathbf{F}\mathbf{D}_l^T\mathbf{D}_l\mathbf{F}^T\right]^{-1}\left[\frac{1}{2}d(\mathbf{P}^T\mathbf{y}) + \sum_{l=1}^{L}\rho_l\mathbf{F}\mathbf{D}_l^T(\mathbf{z}_l - \beta_l)\right] \tag{19}$$

(2) Nonlinear transform layer $\mathbf{z}_l^{(n+1)}$:

$$\mathbf{z}_l^{(n+1)} = S(\mathbf{D}_l\mathbf{x}^{(n)} + \beta_l^{(n)}; \frac{\lambda_l}{\rho_l}) \tag{20}$$

(3) Multiplier update layer $\beta_l^{(n+1)}$:

$$\beta_l^{(n+1)} = \beta_l^{(n)} + \eta_l \cdot (\mathbf{D}_l\mathbf{x}^{(n+1)} - \mathbf{z}_l^{(n+1)}) \tag{21}$$

(4) Convolution layer $\mathbf{c}_l^{(n+1)}$:

$$\mathbf{c}_l^{(n+1)} = \mathbf{D}_l^{(n+1)}\mathbf{x}^{(n+1)} \tag{22}$$

Then the general ADMM framework can be represented using deep neural network, after considering the link between each layers, we adopt a more effective network structure [9] to design our ADMM-net shown in Fig.1.

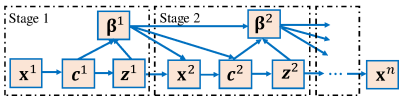

Fig. 1. An improved robust ADMM network structure.

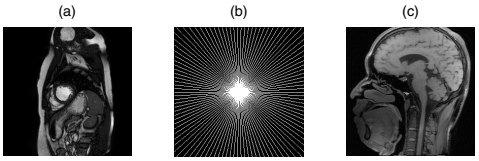

Fig. 2. (a) A 'Chest' magnetic resonance image for training; (b) A magnetic resonance image and the k-space data with 0.3 sampling rate; (c) A 'Brain' magnetic resonance image for reconstruction.

C. Updating by Backpropagation over Robust-ADMM-Network

In this paper, we choose the popular normalized mean square error (NMSE) as the loss-function for training. The loss-function between the output and the ground-truth is described as

$$Loss = \frac{1}{\Gamma}\sum_{i=1}\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_2}{\|\mathbf{x}\|_2} \tag{23}$$

where $\hat{\mathbf{x}}$ is the network output (or reconstruction result), $\mathbf{x}$ is the ground-truth and the set Γ represents the number of pairs of under-sampling data and ground-truth images. To obtain the optimal parameters, transform domain and shrinkage function, we employ the backpropagation strategy to compute the gradient w.r.t. the parameters. The procedure of $\mathbf{x}^{(n)} \rightarrow \mathbf{c}^{(n)} \rightarrow \mathbf{z}^{(n)} \rightarrow \beta^{(n)} \rightarrow \mathbf{x}^{(n+1)}$ is the forward pass. In the forward pass, we calculate the NMSE for using the updated parameters, the learned shrinkage function, and the learned transform domain. The procedure of $\mathbf{x}^{(n)} \leftarrow \mathbf{c}^{(n)} \leftarrow \mathbf{z}^{(n)} \leftarrow \beta^{(n)}$ is the backward pass. In the backward pass, we calculate the gradient of NMSE w.r.t. each parameter in every layer, see [10] for more details.

III. Experiments

Considering the fact that the desired MR images are unknown in applications, we first randomly choose real-world 'Chest' MR images (see Fig.2 (a) and (b)) corrupted by eight levels of α noise with 0.3 sampling rate for training, and then reconstruct the different 'Brain' MR image (see Fig.2 (c)) using the learned Robust ADMM-net in the presence of SαS noise. All experiments are performed for the real-world magnetic resonance (MR) images (https://masi.vuse.vanderbilt.edu/workshop2013 /index.php/Segmentation-Challenge Details).

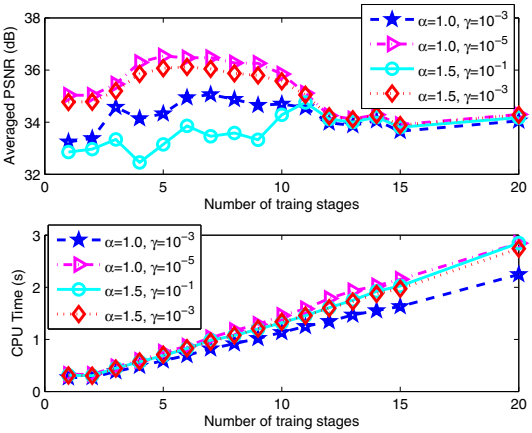

Fig. 3. Impact of different stage numbers on reconstruction. Top: The reconstruction averaged PNSRs versus different number of stage; Bottom: The reconstruction averaged CPU time versus different number of stage.

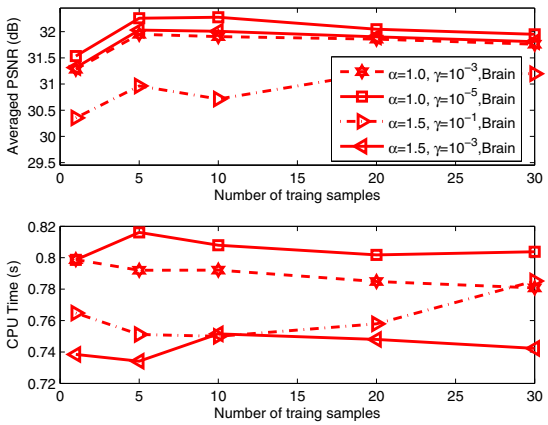

Fig. 4. Reconstruction Performance versus the training number; Top: The reconstruction averaged PNSRs versus different number of training samples; Bottom: The reconstruction averaged CPU time versus different number of training samples.

A. Reconstruction Performance versus the number of network stages

We evaluate the performance with different number of network stages. In general, a deeper network can often obtain higher reconstruction accuracy, but also costs more computational overhead. To make a tradeoff between the reconstruction accuracy and speed, in the following, we will investigate the impact of stage number on reconstruction results. We first empirically choose one 'Chest' MR image as experiment for training, and then reconstruct the 'Brain' MR image (See Fig.2 (c)) using the trained robust-ADMM network. Figure 3 presents the reconstruction results versus the number of stages. From the results, we can observe that the reconstruction PSNR tend to a higher value with the stage number increase, but then decrease, as shown in the top of figure 3, the reconstruction algorithm can obtain the highest PSNR value at 5, 6 and 7 stages. The bottom of Figure 3 describes the CPU time versus the stage number, the result shows that the computational overhead increase linearly with the number of stage number.

B. Reconstruction Performance versus the number of training image

To evaluate the reconstruction results of proposed Robust-ADMM-net using different number of training samples, in this experiment, we employ 1, 5, 10, 20 and 30 different 'Chest' magnetic resonance (MR) images for training respectively, and then reconstruct the 'Brain' images using the learned network under four different levels of SαS noise. Figure 4 detail the reconstruction PSNRs and the corresponding CPU time under different cases. From the results we can explicitly observe that the more training samples can achieve higher reconstruction accuracy, and then tend to stable gradually. Meanwhile, the computational overhead have no significant increase. However, the increasing number of training samples will causes overwhelmingly computational overhead in the training process, hence it is not unnecessary to increase the sampling number in application.

C. Reconstruction Performance compared with State-of-the art Algorithms

To further demonstrate the superiority of the proposed Robust-ADMM-net algorithm, in this section, we compare the proposed algorithm with some existing corresponding work. The YALL1 algorithm [10] is a well-known robust reconstruction algorithm, and the LqLA-ADMM algorithm [5] is a more recent approach for robust compressive sensing reconstruction. Both of them employ the ADMM framework to address the robust reconstruction problem and have achieved good performances. The ADMM-net algorithm [9] is a similar work based on a denoise model called BM3D-MRI [14], although the ADMM-net is proposed in Gaussian environment, we find taht the algorithm can suppress the outliers effectively, and the algorithm of Robust-ADMM is the proposed algorithm before training. For a fair comparison, the robust reconstruction algorithms of YALL1, LqLA-ADMM and Robust-ADMM employ the discrete cosine transformation (DCT) as the sparsifying transform; the algorithms of ADMM-net and Robust-ADMM-net employ DCT as the initial transform. Following the above analysis, we employ the proposed network with 5 stages for training using five different 'Chest' MR images. In this experiment, we train the samplings only under a very impulsive noise environment ($\alpha = 1.0, \gamma = 10^{-4}$), and then reconstruct the different 'Brain' MR image using the learned parameters under four different levels of noise. As shown in Fig.2 (c), this 'Brain' MR image is widely used to evaluate the CS reconstruction performance. Table I detail the reconstruction results of five algorithms for different levels of SαS noise-corrupted 'Brain' data with four sampling rates of 0.2, 0.3, 0.4 and 0.5. In the tables, '–' indicates that the performance of the

TABLE I
RECONSTRUCTION PSNR (DB) FOR 'BRAIN' DATA WITH DIFFERENT SAMPLING RATES AND NOISES.

Method	$\alpha = 1.0, \gamma = 10^{-3}$				$\alpha = 1.0, \gamma = 10^{-5}$				$\alpha = 1.5, \gamma = 10^{-1}$			
	20%	30%	40%	50%	20%	30%	40%	50%	20%	30%	40%	50%
YALL1	–	–	–	–	21.06	22.83	25.39	26.31	–	–	–	–
LqLA-ADMM	–	–	–	–	22.18	24.53	26.74	28.37	–	–	–	–
ADMM-net	27.53	30.38	31.48	32.04	28.97	31.45	**33.51**	35.21	27.21	29.11	28.10	28.74
Robust-ADMM	25.53	26.96	28.12	28.83	25.58	27.07	28.20	28.96	25.55	27.05	28.18	28.93
Robust-ADMM-net	**28.27**	**31.17**	**32.17**	**33.43**	**29.05**	**31.68**	33.42	**35.50**	**28.55**	**30.67**	**30.68**	**31.91**

corresponding algorithm is very poor. We can observe that the algorithms of YALL1 and LqLA-ADMM fail to reconstruct the images when the strength of $S\alpha S$ noise is too high (e.g., $\gamma = 10^{-1}$ and 10^{-3}). The algorithms of ADMM-net and Robust-ADMM can obtain considerable reconstruction results, with the proposed Robust-ADMM-net achieving the best performance. Moreover, the proposed Robust-ADMM-net can improve the reconstruction results significantly after training via the network.

IV. CONCLUSIONS

This paper caters a robust sparse CS reconstruction algorithm called robust-ADMM-net jointing the ADMM framework and deep neural network for robust sparse composite Regularization combined with the composite regularization model. A robust composite model is employed to further exploit more physicalism and prior knowledge of trained image. The strategy of deep neural network can effectively train the parameters through the backpropagation methods. Compared with the no training algorithm of Robust-ADMM, the proposed Robust-ADMM-net algorithm can improve the reconstruction accuracy significantly. Simulation results tell us that the strategy of jointing the traditional CS reconstruction algorithms with deep neural network can improve the recovery performance, and should be our future work.

REFERENCES

[1] E. J. Candes and M. B. Wakin, "An Introduction To Compressive Sampling," IEEE Signal Process. Mag., vol. 25, no. 2, pp. 21-30, 2008.
[2] Y. Li, et. al., "Sparse Adaptive Iteratively-Weighted Thresholding Algorithm (SAITA) for Lp-Regularization Using the Multiple Sub-Dictionary Representation," Sensors, vol. 17, no. 12, pp. 2920-2936, 2017.
[3] X. Cong et al., "A novel adaptive wide-angle SAR imaging algorithm based on Boltzmann machine model," 2016.
[4] W. L. Chan, M. L. Moravec, R. G. Baraniuk, and D. M. Mittleman, "Terahertz imaging with compressed sensing and phase retrieval," Opt. Lett., vol. 33, no. 9, pp. 974-976, 2008.
[5] F. Wen, Y. Yang, L. Pei, W. Yu, and P. Liu, "Efficient and Robust Recovery of Sparse Signal and Image Using Generalized Nonconvex Regularization," IEEE Trans. Comput. Imaging, no. 99, pp. 1-13, 2017.
[6] Y. H. Xiao, H. Zhu, and S. Y. Wu, "Primal and dual alternating direction algorithms for L1-L1-norm minimization problems in compressive sensing," Comput. Optim. Appl., vol. 54, no. 2, pp. 441-459, 2013.
[7] D.-S. Pham and S. Venkatesh, "Efficient algorithms for robust recovery of images from compressed data," IEEE Trans. Image Process., vol. 22, no. 12, pp. 4724-4737, 2013.
[8] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," IEEE Trans. Image Process., vol. 19, no. 9, pp. 2345-2356, 2010.
[9] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep ADMM-Net for Compressive Sensing MRI," in 30th Conference on Neural Information Processing Systems (NIPS 2016), 2016, no. NIPS, pp. 10-18.
[10] J. Yang and Y. Zhang, "Alternating Direction Algorithms for L1 Problems in Compressive Sensing," SIAM J. Sci. Comput., vol. 33, no. 1, pp. 250-278, 2011.
[11] Y. Li, et. al., "MUSAI-L1/2: MUltiple Sub-wavelet-dictionaries-based Adaptively-weighted Iterative Half Thresholding Algorithm for Compressive Imaging," IEEE Access, vol. 6, pp. 16795-16805, 2018.
[12] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," SIAM J. Imaging Sci., vol. 2, no. 1, pp. 183-202, 2009.
[13] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.
[14] E. M. Eksioglu, "Decoupled Algorithm for MRI Reconstruction Using Nonlocal Block Matching Model: BM3D-MRI," J. Math. Imaging Vis., vol. 56, no. 3, pp. 430-440, 2016.