GPX-ADMM-Net: ADMM-based Neural Network with Generalized Proximal Operator

Shih-Wei Hu
Institute of Information Science
Academia Sinica,
Taipei, Taiwan, ROC
vacuityhu@iis.sinica.edu.tw

Gang-Xuan Lin
Institute of Information Science
Academia Sinica,
Taipei, Taiwan, ROC
spybeiman@gmail.com

Chun-Shien Lu
Institute of Information Science
Academia Sinica,
Taipei, Taiwan, ROC
lcs@iis.sinica.edu.tw

***Abstract*—In this paper, we propose a highly efficient and well interpretable deep learning solver, called Generalized ProXimal ADMM-Net (GPX-ADMM-Net), for the linear inverse problem, which is conventionally solved with intensive computations.**

GPX-ADMM-Net is characterized by the generalized proximal operator, convolutional dictionary, and modified loss function. Without loss of interpretability, GPX-ADMM-Net only needs a small number of parameters in a learning model to retain elegant reconstruction quality.

Different from traditional optimization methods, all the parameters of GPX-ADMM-Net need no more hand-crafted but determined by learning strategies. Furthermore, unlike other deep learning-based methods, GPX-ADMM-Net is able to adapt to various measurement rates with only one single set of training parameters. Extensive experimental results further demonstrate the advantages of our proposed method.

***Index Terms*—ADMM, Convolution neural network, Deep learning, Linear inverse problem, Proximal operator**

I. Introduction

A. Background

The linear inverse problem (LIP) has been extensively studied and can be written as $y = \Phi\bar{x}$, where $\Phi \in \mathbb{R}^{m\times n}$ is the linear mapping, and $\bar{x}$, depending on different applications, represents different roles like monitoring-data or signal (e.g., audio, image, video), and y is the vector of samples. Note that the ratio $\frac{m}{n}$ is defined as the measurement rate (MR) in the context of compressive sensing (CS) [1]–[3]. Usually, we solve LIPs with the ℓ_1-regularized optimization problem as:

$$\min_x \frac{\alpha}{2}\|y-\Phi x\|_2^2 + \|\Psi(x)\|_1, \tag{1}$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm, $\Psi(x)$ is the dictionary of x, and $\alpha > 0$ is a constant.

Nevertheless, solving problem (1) is time-consuming. Recently, deep learning has been introduced as an extremely low complexity solver for LIPs [4]–[12]. In this paper, we propose a highly efficient solver, called Generalized ProXimal ADMM-Net (GPX-ADMM-Net), for LIPs by incorporating deep learning and optimization methods.

B. Related Work

For learning-based models of solving LIPs, interpretability, in our opinion, means that when the network architecture design is based on a certain iterative algorithm, the theoretical insights can interpret the network architecture. For example, by unrolling an algorithm for ℓ_1-norm minimization, the network output can be interpreted by the theory of convex optimization. Interpretability plays a fundamental role of understanding the potential limitation of designed networks. We classify the learning-based solvers to LIPs into two categories.

The first category [4]–[9] aims at learning an inverse mapping to solve LIPs by convolutional neural network (CNN). Nevertheless, CNN-based methods perform well only under low MRs and lack interpretability due to the inherent architecture of stacking convolution layers.

The second category unrolls traditional optimization algorithms into a deep learning architecture for the purpose of maintaining the interpretability. [10]–[12] unroll the iterative shrinkage-thresholding algorithm (ISTA), and [13]–[15] unroll the alternating direction method of multipliers algorithm (ADMM). This category performs well in a full range (between 0 and 1) of MRs.

In ADMM-Net [13], the authors used a piecewise linear function determined by learning strategies instead of a shrinkage function to avoid hand-craft regularization term. Afterward, [11] proposed ISTA-Net$^+$ that outperforms ADMM-Net and achieves state-of-the-art recovery performances.

Both ADMM-Net and ISTA-Net$^+$ add the convolution layers and skip connections heuristically to seek better performances, but become hard to interpret the architecture. Note that while [14] obtains inferior reconstruction quality than ISTA-Net$^+$, it indeed provides a strict theoretical guarantee that the algorithm will reach a stationary point.

C. Motivation and Contribution

Following [10]–[15], we learn that an insight into traditional optimization algorithms could be a great hint for designing a learning model-based solver for LIPs. In [16], the authors show that the global convergence rates of ADMM and ISTA are both $O(\frac{1}{k})$ and that ADMM always achieves the minimum with fewer iterations than ISTA and FISTA [16]. Such an observation inspires us to develop a new learning-based ADMM algorithm with the contributions summarized as follows:

1) Different from ISTA-Net$^+$ [11], we unroll the ADMM algorithm, which is more efficient than ISTA, as our network basis. After unrolling the ADMM algorithm,

we employ a generalized proximal operator due to the theory of proximal gradient methods. Moreover, a loss function is modified to be interpreted by the behavior observation of the ADMM algorithm. Meanwhile, we keep the soft-thresholding function as the shrinkage function rather than a piecewise linear function, which is different from ADMM-Net [13] as well. As such, GPX-ADMM-Net finds a new way to enjoy the advantages of fast and accurate reconstruction (See Table II) without loss of interpretability.

2) Inspired by [14], GPX-ADMM-Net trains only the dictionary, threshold parameter, and step size to maintain the adaptability to different MRs. This implies that, despite a slight performance drop, GPX-ADMM-Net can reconstruct signals under various MRs with only a single set of trained parameters (See Table III).
3) As claimed in [11], ISTA-Net$^+$ obtains state-of-the-art performances when the phase number equals 9 (*i.e.*, the number of parameters equals $336,978$) but achieves flat and saturated performance when the phase number is greater than 9. In comparison with ISTA-Net$^+$, GPX-ADMM-Net achieves state-of-the-art performance with only half the amount of parameters. Moreover, GPX-ADMM-Net is capable of obtaining higher reconstruction performance if more parameters are allowed for training (See Fig. 2).

II. Proposed Method: GPX-ADMM-Net

GPX-ADMM-Net contains three main components: convolutional dictionary, generalized proximal operator, and modified loss function, as shown in Fig. 1, where Ψ and $\tilde{\Psi}$ represent the convolutional dictionaries, and the dotted box indicates the generalized proximal operator. To enjoy the interpretability coming from the traditional optimization algorithms, we design the network architecture by unrolling the ADMM algorithm. Thus, it is necessary to introduce ADMM to make this paper self-contained.

A. ADMM algorithm

In problem (1), by splitting the decision variable x in the error term, the objective function becomes $\frac{\alpha}{2}\|y-\Phi z\|_2^2 + \|\Psi(x)\|_1$ with constraint $x=z$, whereas the corresponding augmented Lagrange function is in the form:

$$\begin{aligned}\mathcal{L}(x,z;M) = \ & \tfrac{\alpha}{2}\|y-\Phi z\|_2^2 + \|\Psi(x)\|_1 - \langle M, x-z\rangle \\ & + \tfrac{\beta}{2}\|x-z\|_2^2,\end{aligned} \tag{2}$$

where M is Lagrange multiplier associated with the constraint $x-z=0$, and $\beta>0$ is a constant. At each iteration k, the ADMM algorithm [17] minimizes $\mathcal{L}$ associated with z and x, individually, followed by updating the Lagrange multiplier M. Note that the minimization subproblem $\mathcal{L}(x,z;M)$ associated with z is convex quadratic programming provided x and M are fixed. Their closed-form solutions are summarized as:

$$z^{k+1} = G\left(\alpha\Phi^T y - M^k + \beta x^k\right), \tag{3}$$

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\ \tfrac{1}{\beta}\|\Psi(x)\|_1 + \tfrac{1}{2}\left\|x - (\tfrac{1}{\beta}M^k + z^{k+1})\right\|_2^2, \tag{4}$$

$$M^{k+1} = M^k - \beta(x^{k+1} - z^{k+1}), \tag{5}$$

where $G = \left(\alpha\Phi^T\Phi + \beta I_n\right)^{-1}$. By letting $\mathcal{Z}^{k+1} = \frac{1}{\beta}M^k + z^{k+1}$, Eqs. (3)-(5) can be rewritten as:

$$\mathcal{Z}^{k+1} = W_1 y + W_2 M^k + W_3 x^k + \tfrac{1}{\beta}M^k, \tag{6}$$

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\ \tfrac{1}{\beta}\|\Psi(x)\|_1 + \tfrac{1}{2}\|x - \mathcal{Z}^{k+1}\|_2^2, \tag{7}$$

$$M^{k+1} = \beta(\mathcal{Z}^{k+1} - x^{k+1}), \tag{8}$$

where $W_1 = \alpha G\Phi^T$, $W_2 = -G$, and $W_3 = \beta G$.

The solution to the sub-problem (7), called proximal operator, has been well studied [18]. Let $f(x) = \|\Psi(\cdot)\|_1$, the proximal operator of $\frac{1}{\beta}f$ at $\mathcal{Z}^{k+1}$ is defined as:

$$\operatorname{prox}_{\frac{1}{\beta}f}(\mathcal{Z}^{k+1}) = \underset{x}{\operatorname{argmin}}\ \tfrac{1}{\beta}\|\Psi(x)\|_1 + \tfrac{1}{2}\|x - \mathcal{Z}^{k+1}\|_2^2. \tag{9}$$

B. Generalized proximal operator and convolutional dictionary

It is observed that subproblem (7) is in the form of the proximal operator of composite function $\frac{1}{\beta}\|\Psi(\cdot)\|_1$ at the point $\mathcal{Z}^{k+1}$. But, to solve subproblem (7), there is no useful calculus rule for computing the proximal operator of a composite function. Nevertheless, if the dictionary Ψ is linear and satisfies a certain orthogonality condition, we can employ the following theorem to maintain the interpretability.

Theorem II.1. *[18] Let $g:\mathbb{R}^N \to (-\infty,\infty]$ be a proper close convex function, and let $f(x) = g(\mathcal{A}(x)+b)$, where $b\in\mathbb{R}^N$ and $\mathcal{A}:\mathbb{R}^n\to\mathbb{R}^N$ is a linear transformation satisfying $\mathcal{A}\circ\mathcal{A}^T = \gamma\cdot I_N$ for some constant $\gamma>0$. Then for any $x\in\mathbb{R}^n$,*

$$\operatorname{prox}_f(x) = x + \frac{1}{\gamma}\mathcal{A}^T\left(\operatorname{prox}_{\gamma g}(\mathcal{A}(x)+b) - (\mathcal{A}(x)+b)\right).$$

If we replace $g(\cdot)$ and $\mathcal{A}(\cdot)$ in theorem II.1 by $\|\cdot\|_1$ and $\Psi(\cdot)$ respectively, together with the fact that the solution to the proximal operator of $\rho\|x\|_1$ is $\theta(\ \cdot\ ;\rho)$ (the soft-thresholding function with threshold parameter ρ), we have the closed-form solution to the sub-problem (7) as:

$$\operatorname{prox}_{\frac{1}{\beta}f}(\mathcal{Z}^{k+1}) = \mathcal{Z}^{k+1} + \frac{1}{\gamma}\Psi^T(\theta(\Psi(\mathcal{Z}^{k+1});\frac{\gamma}{\beta}) - \Psi(\mathcal{Z}^{k+1})), \tag{10}$$

which we call a generalized proximal operator.

On the other hand, Ψ in problem (10) is generally treated to be an over-complete dictionary (*i.e.*, $\Psi\in\mathbb{R}^{N\times n}$ with $N>n$). We can see that, as $N>n$, the assumption $\Psi\circ\Psi^T = \gamma I_N$ in Theorem II.1 is not satisfied at all. Thus, it is required to seek the approximation $\Psi\circ\Psi^T \approx \gamma I_N$. Meanwhile, w.l.o.g., we may assume the left inverse of Ψ, *i.e.*, $\Psi^\dagger$, exists due to $N>n$ such that $\Psi^\dagger\circ\Psi = I_n$. Therefore, the assumption in

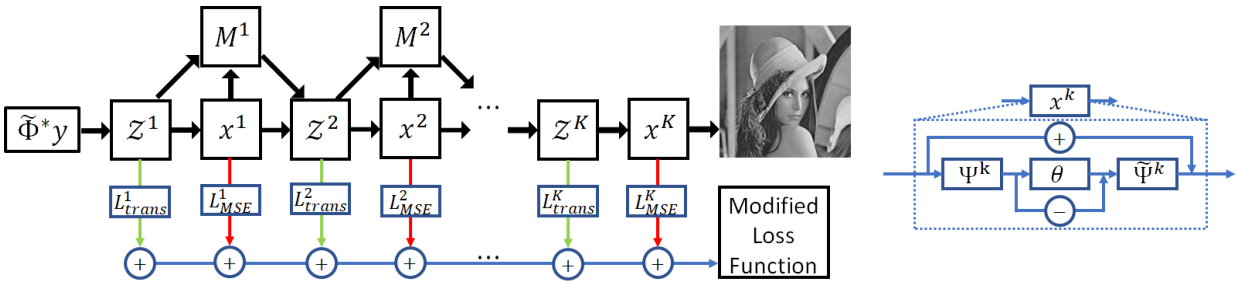

Fig. 1. The data flow of GPX-ADMM-Net.

TABLE I
LIST OF INPUT FEATURE AND OUTPUT FEATURE.

	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_4$
Input feature	1	32	32	32
Output feature	32	32	32	1

Theorem II.1 becomes $\Psi^T = (\Psi^\dagger \cdot \Psi) \cdot \Psi^T = \Psi^\dagger \cdot (\Psi \cdot \Psi^T) \approx \Psi^\dagger \cdot \gamma I_N = \gamma \Psi^\dagger$, and can be relaxed as $\Psi^T = \gamma' \gamma \Psi^\dagger = \bar{\gamma} \Psi^\dagger$, where γ' is a constant. Now, Eq. (10) can be rewritten as:

$$x^{k+1} = \mathcal{Z}^{k+1} + \gamma' \Psi^\dagger \left(\theta(\Psi(\mathcal{Z}^{k+1}); \frac{\gamma}{\beta}) - \Psi(\mathcal{Z}^{k+1}) \right). \quad (11)$$

Inspired by the representation power of CNN [19] and its universal approximation property [20], the dictionary Ψ is adopted in the form of convolutional dictionary as $\Psi = \mathcal{C}_2(\text{ReLU}(\mathcal{C}_1(x)))$, where both $\mathcal{C}_1$ and $\mathcal{C}_2$ are convolution operators, and ReLU is a rectified linear unit.

Moreover, to exhibit a "left-inverse" structure of Ψ, $\Psi^\dagger$ is also adopted in the form of the convolutional dictionary as $\Psi^\dagger = \tilde{\Psi} = \mathcal{C}_4(\text{ReLU}(\mathcal{C}_3(x)))$, where both $\mathcal{C}_3$ and $\mathcal{C}_4$ are convolution operators. Based on the above, it is also required to present a suitable loss function appropriately to ensure the left-inverse relation between Ψ and $\tilde{\Psi}$. The new loss function will be described in Sec. II-C.

In our experiments, all the kernels of convolution operators were set to 3×3 and the numbers of input features and output features of $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ are listed in Table I.

After determining the dictionaries and proximal operator, the forwarding in a single layer of GPX-ADMM-Net can be summarized as:

$$\mathcal{Z}^{k+1} = \hat{W}_1 y + \hat{W}_2 M^k + \hat{W}_3 x^k + \beta^k_{\mathcal{Z}} M^k, \quad (12)$$
$$x^{k+1} = \mathcal{Z}^{k+1} + \gamma^k \tilde{\Psi}^k (\theta(\Psi^k(\mathcal{Z}^{k+1}); \lambda^k) - \Psi(\mathcal{Z}^{k+1})), \quad (13)$$
$$M^{k+1} = \beta^k_M (\mathcal{Z}^{k+1} - x^{k+1}), \quad (14)$$

where $\left\{ \beta^k_{\mathcal{Z}}, \gamma^k, \lambda^k, \beta^k_M, \Psi^k, \tilde{\Psi}^k \right\}$ are training parameters, $\hat{W}_1 = \hat{G}\Phi^T$, $\hat{W}_2 = -\hat{G}$, $\hat{W}_3 = 0.1\hat{G}$, and $\hat{G} = \left(\Phi^T \Phi + 0.1 \cdot I_n\right)^{-1}$.

Inspired by [14], we do not train $\hat{W}_1$, $\hat{W}_2$, and $\hat{W}_3$ since all of them are determined whenever Φ is given. Due to this reason, GPX-ADMM-Net can use unrestricted $\Phi \in \mathbb{R}^{m \times n}$ in terms of m and adapt to different measurement rates.

Moreover, according to [21], skip connection plays a key role in the architecture design of deep learning in that it can remarkably improve the performance. Comparing the architectures of ISTA-Net and ISTA-Net$^+$ [11], ISTA-Net$^+$ outperforms ISTA-Net with the help of skip connection. We find, however, such a heuristical design of skip connection may not transmit information completely.

Benefit from Theorem II.1, the computation of x^{k+1} provides two skip connections without loss of interpretability, *i.e.* $\mathcal{Z}^{k+1}$ plus the mapping of itself is the first one, and $-\Psi(\mathcal{Z}^{k+1})$ plus the mapping of itself is the second one. That is the main difference between GPX-ADMM-Net and ISTA-Net$^+$, and we believe it is also the main reason why GPX-ADMM-Net can achieve state-of-the-art performance.

C. Loss function

Mean-square-error (MSE)-based loss function has been widely adopted in the literature [4]–[7], [10], [13]–[15]. In general, the loss function is defined as: $L_{\text{MSE}} = \frac{1}{b \cdot n} \sum_{i=1}^{b} \left\| x_i^K - \bar{x}_i \right\|_2^2$, where b is the batch size, and $x^K \in \mathbb{R}^n$ and $\bar{x} \in \mathbb{R}^n$ are the output of the network (of K layer) and the ground-truth, respectively.

Recall the ADMM algorithm in Eqs. (3)-(5), the subproblem (4) is to minimize the function (2) associated with x. This means that the optimal solution to subproblem (4) in each iteration tries to achieve the global minimum of problem (1). With this interpretation, each x^k of GPX-ADMM-Net should be constrained by the loss function $L^k_{\text{MSE}} = \frac{1}{b \cdot n} \sum_{i=1}^{b} \left\| x_i^k - \bar{x}_i \right\|_2^2$, as shown in Fig. 1 (red line), for all k. In this manner, we modify the MSE-based loss function as:

$$\mathfrak{L}_{\text{tMSE}} = \sum_{k=1}^{K} L^k_{\text{MSE}} = \frac{1}{b \cdot n} \sum_{k=1}^{K} \sum_{i=1}^{b} \| x_i^k - \bar{x}_i \|_2^2,$$

called total MSE (tMSE) loss function.

Furthermore, considering backpropagated gradients of the network in deep learning is multiplied by both the small multipliers and small learning rates at each layer, the gradients cumulate more small multipliers as further away from the output layer [22]. When the network is deeper enough (with a larger total number of layers K), the gradients will finally vanish, preventing the parameters from being updated.

To address this issue, our study finds that the tMSE loss function can mitigate the gradient vanish problem since it

provides the backpropagated gradients to every layers directly (red line in Fig. 1). To our knowledge, we are the first one in using tMSE loss function to mitigate the gradient vanish problem.

On the other hand, we also consider the MSE-loss function in specifying the left-inverse relation between Ψ^k and $\tilde{\Psi}^k$, as mentioned in Sec. II-B. Specifically, we enforce the relationship between the Ψ^k and $\tilde{\Psi}^k$ by $L^k_{\text{trans}} = \frac{1}{b\cdot n}\sum_{i=1}^{b}\|\tilde{\Psi}^k(\Psi^k(\mathcal{Z}_i^k)) - \mathcal{Z}_i^k\|_2^2$ in each layer k, as shown in Fig. 1 (green line). Thus, the summation of L^k_{trans} in each layer k is defined as:

$$\mathfrak{L}_{\text{trans}} = \sum_{k=1}^{K} L^k_{\text{trans}} = \frac{1}{b\cdot n}\sum_{k=1}^{K}\sum_{i=1}^{b}\|\tilde{\Psi}^k(\Psi^k(\mathcal{Z}_i^k)) - \mathcal{Z}_i^k\|_2^2.$$

Overall, given the training data $\{y_i, \bar{x}_i\}_{i=1}^b$, the loss function for GPX-ADMM-Net is defined as:

$$\mathfrak{L}(\mathcal{P}; y_i) = \mathfrak{L}_{\text{tMSE}} + \varepsilon \cdot \mathfrak{L}_{\text{trans}}, \tag{15}$$

where ε denotes the regularization parameter, $\mathcal{P} = \left\{\beta_{\mathcal{Z}}^k, \beta_M^k, \gamma^k, \lambda^k, \mathcal{C}_1^k, \mathcal{C}_2^k, \mathcal{C}_3^k, \mathcal{C}_4^k\right\}_{k=1}^K$ denotes the training parameter set, both $\beta_{\mathcal{Z}}^k$ and β_M^k are the step sizes, γ^k is the penalty parameter of skip connection, λ^k is the threshold parameter, and $\Psi^k = \mathcal{C}_2^k\left(\text{ReLU}\left(\mathcal{C}_1^k\right)\right)$ and $\tilde{\Psi}^k = \mathcal{C}_4^k\left(\text{ReLU}\left(\mathcal{C}_3^k\right)\right)$ are specified in Eqs. (12)-(14).

D. Initialization

Considering the initialization of GPX-ADMM-Net, instead of adopting a random matrix as Φ, a reliable estimation could enhance the performance significantly. Let $\bar{X} \in \mathbb{R}^{n\times B}$, where B is the total number of training data, and let $Y \in \mathbb{R}^{m\times B}$ be the measurement matrix, which is collected from measurement vectors $y_i = \Phi\bar{x}_i$. By using all training data, the estimated initialization $\tilde{\Phi}^*$ can be computed by solving a least squares problem $\tilde{\Phi}^* = \text{argmin}_{\tilde{\Phi}}\|\tilde{\Phi}Y - \bar{X}\|_F^2 = \bar{X}Y^T(YY^T)^{-1}$, where $\|\cdot\|_F$ denotes the Frobenius norm. Hence, we set $x^0 = \tilde{\Phi}^* y$ as the initialization of GPX-ADMM-Net and M^0 is simply set as $\mathbf{0}$.

III. Experimental Results

We verify the performance of GPX-ADMM-Net in terms of sparse signal recovery and super-resolution.

A. Setting

For fair comparison, we used the same set of 91 images as in [11] for training. Each training image was cropped into 88912 blocks of size 33×33 and vectorized, *i.e.* $B = 88912$ and $n = 1089$. We also drew a standard normal matrix $\Phi \in \mathbb{R}^{m\times n}$ as the sensing matrix, and constructed the measurement vector by $y_i = \Phi\bar{x}_i$, where $\bar{x}_i$ are vectorized image blocks. Thus, the training data pairs $\{y_i, \bar{x}_i\}_{i=1}^B$ were generated. To train the networks, we used Adam optimization [23] with a learning rate 0.0001 and a batch size 64. The regularization parameter ε in Eq. (15) was set to 0.01. The benchmark dataset Set11 [4] was used for testing and the reconstruction results were measured by Peak Signal-to-Noise Ratio (PSNR) and/or structural similarity (SSIM) index [24].

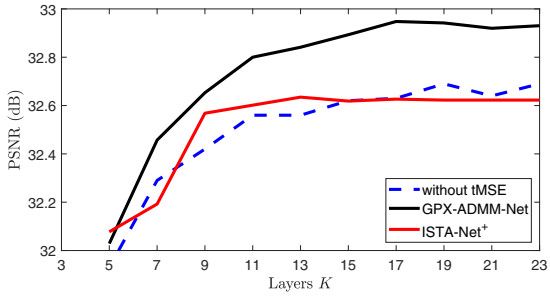

Fig. 2. PSNR comparison among GPX-ADMM-Net, ISTA-Net, and ISTA-Net$^+$ under various number of total layers. At $K = 9$, GPX-ADMM-Net used around 51% of training parameters of ISTA-Net$^+$ but attained slightly better performance.

B. Sparse Signal Recovery in Compressive Sensing

First, we show the recovery performance of GPX-ADMM-Net and compare it with ISTA-Net and ISTA-Net$^+$ [11].

As claimed in [11], the ISTA-Net$^+$ achieves the best performances when the phase* number equals 9, while allowing more parameters could not provide better performance. The total number of training parameters equals $336,978$ (37442 for each phase) whenever the phase number is $N_p = 9$.

In this paper, as shown in Table I, the number of features is 1×32 for $\mathcal{C}_1$ and $\mathcal{C}_4$, and 32×32 for $\mathcal{C}_2$ and $\mathcal{C}_3$. Moreover, the kernal size for $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ is 3×3. Therefore, the total number of training parameters used in GPX-ADMM-Net in each layer is 19012 $(= (1 \times 32 + 32 \times 32) \times (3 \times 3) \times 2 + 4)$ ($+4$ corresponds to $\beta_{\mathcal{Z}}^k$, β_M^k, γ^k, and λ^k). In comparison with ISTA-Net$^+$, GPX-ADMM-Net only used 51% of training parameters in each layer. Thus, the total number of training parameters used in GPX-ADMM-Net with $k = 17$ is still not greater than ISTA-Net$^+$ with $K = 9$.

Here, we show the average PSNR curves for the testing set (Set11) concerning different number of total layers. As shown in Fig. 2, GPX-ADMM-Net performs slight better than ISTA-Net$^+$ when $K = 5, 7, 9$, but the former needs only half of the training parameters in ISTA-Net$^+$. Moreover, the blue dashed curve in Fig. 2 illustrates the performance of GPX-ADMM-Net without using "total" MSE loss function, *i.e.*, the loss function is chosen as $L_{\text{MSE}} + \varepsilon \cdot \mathfrak{L}_{\text{trans}}$. We can see that, in GPX-ADMM-Net, tMSE indeed gets better recovery performance than MSE.

Second, the reconstruction performance with respect to different measurement rates in terms of average PSNR and SSIM is summarized in Table II. When $K = 9$, GPX-ADMM-Net used around 51% of training parameters in ISTA-Net$^+$ but obtained slightly better performance, whereas both GPX-ADMM-Net and ISTA-Net$^+$ comsumed almost the same computation time (in GPU). When $K = 17$, GPX-ADMM-Net used around 96% of training parameters in ISTA-Net$^+$ but obtained much better performance than that with $K = 9$.

The above two experiments demonstrate that the recovery performance of GPX-ADMM-Net can be gradually improved

*A phase in ISTA-Net$^+$ corresponds to a layer in this paper.

TABLE II
AVERAGE PSNR (IN DB) /SSIM AND COMPUTATION TIME (IN SECOND (S)) UNDER DIFFERENT MEASUREMENT RATES (MRS).

	ISTA-Net$^+$	GPX-ADMM-Net		
MRs	$K = 9$	$K = 9$	$K = 17$	$K = 41$
25%	32.57 / 0.92	32.65 / 0.92	32.95 / 0.93	33.02 / 0.93
10%	26.64 / 0.81	26.94 / 0.81	27.28 / 0.82	27.46 / 0.83
4%	21.31 / 0.61	21.86 / 0.64	22.15 / 0.65	22.40 / 0.66
1%	17.34 / 0.43	17.54 / 0.44	17.58 / 0.45	17.60 / 0.45
Time	0.047s	0.038s	0.070s	0.145s

TABLE III
AVERAGE PSNR (DB) UNDER DIFFERENT MEASUREMENT RATES WITH THE SAME SET OF TRAINED PARAMETERS.

	MRs	Retrain	$\mathcal{P}_{25}$	$\mathcal{P}_{10}$	$\mathcal{P}_4$	$\mathcal{P}_1$
GPX-ADMM-Net	25%	33.02	**33.02**	32.50	30.82	27.45
	10%	27.46	26.74	**27.46**	26.39	22.66
	4%	22.40	21.13	21.85	**22.40**	19.70
	1%	17.60	17.32	17.39	17.41	**17.60**
ISTA-Net$^+$	25%	32.57	**32.57**	29.33	10.60	9.08
	10%	26.64	26.34	**26.64**	22.62	11.59
	4%	21.31	20.85	21.21	**21.31**	15.70
	1%	17.34	17.33	17.35	17.39	**17.34**

if more layers and training parameters are allowed but ISTA-Net$^+$ does not exhibit the similar phenomenon.

Third, since GPX-ADMM-Net only needs to be trained in terms of step size, threshold, and dictionary, the trained parameters should be able to fit in different sensing matrices. Here, we trained a parameter set with measurement rate $q\%$, called $\mathcal{P}_q$, and solved the signal recovery problem under other measurement rates. In the "Retrain" column of Table III, we show the PSNR results of retraining GPX-ADMM-Net individually with respect to different measurement rates. For the rest of Table III, we show the results by using $\mathcal{P}_q$ to testing all different MRs. Comparing with "ISTA-Net$^+$," the "GPX-ADMM-Net" attains better recovery in almost all cases.

Finally, we also have the same conclusions for the dataset BSD68.

C. Super-Resolution

By using the same training setting, we let $\Phi \in \mathbb{R}^{121\times 1089}$ be the downsample matrix, which makes the image downsampled into $\frac{1}{3}$ size of the original data via box-averaging. The high-resolution results (in average PSNR of all images in Set11) reconstructed by GPX-ADMM-Net and ISTA-Net$^+$ are 28.07 dB and 27.41 dB, respectively.

IV. CONCLUSION

Since constructing an effective architecture of deep learning for solving optimization problems is not trivial, consulting the structure insights of traditional optimization algorithms is helpful and interpretable. Motivated by this concept, GPX-ADMM-Net features a set of characteristics, including the generalized proximal operator and a new loss function. Moreover, despite a slight performance drop, GPX-ADMM-Net can reconstruct signals under various MRs with only a single set of trained parameters. Extensive comparisons validate its state-of-the-art performance.

V. ACKNOWLEDGMENT

This work was supported by Ministry of Science and Technology, Taiwan, ROC, under Grants MOST 107-2221-E-001-015-MY2 and MOST 108-2634-F-007-010.

REFERENCES

[1] R. G. Baraniuk, "Compressive sensing," *IEEE Signal Process. Mag.*, vol. 24, no. 4, pp. 118–121, July 2007.
[2] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.
[3] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
[4] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok, "Reconnet: Non-iterative reconstruction of images from compressively sensed measurements," in *CVPR*, 2016.
[5] A. Mousavi and R. G. Baraniuk, "Learning to invert: Signal recovery via deep convolutional networks," in *ICASSP*, 2017.
[6] A. Mousavi, G. Dasarathy, and R. G. Baraniuk, "Deepcodec: Adaptive sensing and recovery via deep convolutional neural networks," in *Annu. Allerton Conf. on Commun., Control, and Comput. (Allerton)*, 2017.
[7] S. Lohit, K. Kulkarni, R. Kerviche, P. Turaga, and A. Ashok, "Convolutional neural networks for noniterative reconstruction of compressively sensed images," *IEEE Trans. Comput. Imag.*, vol. 4, no. 3, pp. 326–340, 2018.
[8] J. Schlemper, J. Caballero, J. V. Hajnal, A. N. Price, and D. Rueckert, "A deep cascade of convolutional neural networks for dynamic MR image reconstruction," *IEEE Trans. Med. Imag.*, vol. 37, no. 2, pp. 491–503, 2018.
[9] W. Shi, F. Jiang, S. Liu, and D. Zhao, "Image compressed sensing using convolutional neural network," *IEEE Trans. Image Process.*, vol. 29, pp. 375–388, 2020.
[10] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *ICML*, 2010.
[11] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," in *CVPR*, 2018.
[12] J. Liu, X. Chen, Z. Wang, and W. Yin, "ALISTA: Analytic weights are as good as learned weights in LISTA," in *ICLR*, 2019.
[13] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep ADMM-Net for compressive sensing MRI," in *NeurIPS*, 2016.
[14] J. H. Rick Chang, C.-L. Li, B. Poczos, B. V. K. Vijaya Kumar, and A. C. Sankaranarayanan, "One network to solve them all - solving linear inverse problems using deep projection models," in *ICCV*, 2017.
[15] Y. Li, X. Cheng, and G. Gui, "Co-robust-ADMM-Net: Joint ADMM framework and DNN for robust sparse composite regularization," *IEEE Access*, vol. 6, pp. 47 943–47 952, 2018.
[16] S. Tao, D. Boley, and S. Zhang, "Convergence of common proximal methods for ℓ_1-regularized least squares," in *IJCAI*, 2015.
[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[18] A. Beck, *First-Order Methods in Optimization*. MOS-SIAM Ser. Optim., 2017.
[19] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *ECCV*, 2014.
[20] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 5, no. 2, pp. 359–366, 1989.
[21] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," in *NeurIPS*, 2018.
[22] A. Asadi and R. Safabakhsh, "The encoder-decoder framework and its applications," in *Deep Learn.: Concepts and Archit.* Springer, 2020, pp. 133–167.
[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.
[24] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.