MCNet: MEASUREMENT-CONSISTENT NETWORKS VIA A DEEP IMPLICIT LAYER FOR SOLVING INVERSE PROBLEMS

Rahul Mourya, João F. C. Mota

Institute of Sensors, Signals, and Systems, Heriot-Watt University, Edinburgh, UK

ABSTRACT

End-to-end deep neural networks (DNNs) have become the state-of-the-art (SOTA) for solving inverse problems. Despite their outstanding performance, during deployment, such networks are sensitive to minor variations in the testing pipeline and often fail to reconstruct small but important details, a feature critical in medical imaging, astronomy, or defence. Such instabilities in DNNs can be explained by the fact that they ignore the forward measurement model during deployment, and thus fail to enforce consistency between their output and the input measurements. To overcome this, we propose a framework that transforms any DNN for inverse problems into a measurement-consistent one. This is done by appending to it an implicit layer (or deep equilibrium network) designed to solve a model-based optimization problem. The implicit layer consists of a shallow learnable network that can be integrated into the end-to-end training while keeping the SOTA DNN fixed. Experiments on single-image super-resolution show that the proposed framework leads to significant improvements in reconstruction quality and robustness over the SOTA DNNs.

Index Terms— Image super-resolution, ADMM, implicit layers, deep equilibrium networks, fixed-point iterations.

1. INTRODUCTION

We consider the problem of reconstructing a vector $\boldsymbol{x}^\star \in \mathbb{R}^n$ of which we have only a few linear, noisy measurements $\boldsymbol{b} = \mathbf{A}\boldsymbol{x}^\star + \boldsymbol{\eta}$, where $\mathbf{A} \in \mathbb{R}^{m\times n}$, with $m \ll n$, and $\boldsymbol{\eta} \in \mathbb{R}^m$ is noise. Such a linear inverse problem is challenging since, even when there is no noise, the system $\boldsymbol{b} = \mathbf{A}\boldsymbol{x}$ contains infinitely many solutions, among which the one we seek, $\boldsymbol{x}^\star$. Classical approaches [1, 2, 3] attempt to reconstruct $\boldsymbol{x}^\star$ by solving an optimization problem that balances (or constrains) measurement fidelity and a regularizer encoding prior knowledge about $\boldsymbol{x}^\star$, e.g., sparsity in a domain. A long-standing challenge is to devise regularizers that, while computational tractable, effectively describe classes of signals.

State-of-the-art (SOTA) methods for image reconstruction tasks such as super-resolution, MRI reconstruction, or inpainting, leverage the power of deep neural networks (DNNs) to directly learn a map from the measurements to the corresponding reconstructed signal [4, 5, 6, 7]. Despite their extraordinary performance, such end-to-end schemes are unreliable, failing, for example, to detect small, unusual signals absent in the training data [8]. Part of the reason is that during deployment they ignore the forward measurement model, that is, $\boldsymbol{b} = \mathbf{A}\boldsymbol{x} + \boldsymbol{\eta}$, even when the model is known [9, 10]. Moreover, end-to-end DNNs for image reconstruction require retraining whenever the measurement operator $\mathbf{A}$ changes. Both shortcomings are absent in classical optimization-based methods, albeit at the cost of poorer performance.

Work supported by EPSRC's New Investigator Award (EP/T026111/1).

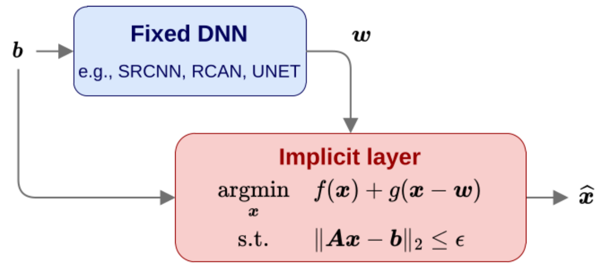
$\boldsymbol{b}$
Fixed DNN
e.g., SRCNN, RCAN, UNET
$\boldsymbol{w}$
Implicit layer
$\underset{\boldsymbol{x}}{\text{argmin}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x} - \boldsymbol{w})$
s.t. $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \leq \epsilon$
$\widehat{\boldsymbol{x}}$

Fig. 1. MCNet framework. The output $\boldsymbol{w}$ of a fixed DNN is fed into an implicit layer designed to solve an inverse problem that ensures measurement consistency (via its constraint), fidelity to $\boldsymbol{w}$ (via g), and prior knowledge of $\boldsymbol{x}$ (via f). f and g can be implemented as DNNs, and the entire pipeline is end-to-end trainable.

Recent work has thus tried to blend classical and DNN approaches to get the benefits of both. We categorize such methodologies as deep unrolling networks (DUNets), plug-and-play methods (PnP), post-processing (PP), and deep equilibrium models (DEQs). We briefly review them in §2.

Our scheme. The scheme we propose combines ideas from PnP, PP, and DEQs, and is represented in Fig. 1. The measurement vector $\boldsymbol{b}$ is passed through an end-to-end DNN, e.g., RCAN [11] for super-resolution or UNET [12] for MRI reconstruction. Its output $\boldsymbol{w}$ will typically be close to $\boldsymbol{x}^\star$ but measurement inconsistent [9]. To leverage the good quality of $\boldsymbol{w}$, the knowledge of the forward measurement model, and any other prior information about $\boldsymbol{x}^\star$, we pass $\boldsymbol{b}$ and $\boldsymbol{w}$ through an implicit layer that solves an optimization problem integrating all these elements. We refer to the resulting scheme as *measurement-consistent network* (MCNet), and describe its architecture and training procedure in §4. While MCNet can be applied to generic image reconstruction problems, this paper focuses on single-image super-resolution. Because MCNet leverages a high-quality DNN network, e.g., RCAN [11], its implicit layer requires only a shallow network, usually with only $4-6$ layers, making it simpler than alternative DEQ approaches [13] or PnP methods [14, 15]. Thus, it also requires less training data than prior approaches.

Contributions. Our contributions are as follows:

- We propose a framework, MCNet, for transforming any DNN for linear inverse problems into a measurement-consistent one.
- By ensuring measurement-consistency, MCNet is more robust to small input perturbations than SOTA DNNs .
- We design an efficient training algorithm based on DEQ techniques. By interpreting an ADMM algorithm to solve the problem in the implicit layer as a fixed-point iteration, we leverage the implicit function theorem to train the network without ever unrolling the iterations. The number of iterations of the fixed-point algorithm can be modified during testing without significantly deteriorating

the quality of the output.

- We establish conditions under which the fixed-point algorithm converges in the forward and backward passes.
- Extensive results on single-image super-resolution show that, with respect to prior methods, MCNet not only makes the DNN outputs measurement-consistent, but also yields significant gains in quality, despite having significantly fewer learnable parameters in the implicit layer.

2. RELATED WORK

We briefly overview work that integrates DNNs into classical optimization-based approaches.

Deep unrolling networks (DUNets). First proposed in [16], DUNets are the current SOTA methods for MRI reconstruction, low-dose CT, and emission tomography reconstruction. They are designed by unrolling a few iterations of an iterative algorithm, e.g., forward-backward splitting [1], in which a proximal step is replaced by a DNN. The number of unrolled iterations is typically small, 5 to 10. And to reduce the memory requirements during training, parameters are shared between layers. A consequence is that outputs from DUNets may not be measurement-consistent [10]. DUNets are also inflexible. For example, the number of unrolled iterations has to be the same during training and testing [13], and modifying the operator $\mathbf{A}$ requires retraining from scratch.

Plug-and-play (PnP) methods. PnP methods [17] modify model-based iterative algorithms, e.g., forward-backward splitting or ADMM, by replacing one or more proximal operators with nonconvex, SOTA denoisers like BM3D [18] or pre-trained DNNs, e.g., DNCNN [19]. The resulting algorithms yield excellent results on a variety of tasks, e.g., image deblurring, MRI, low-dose CT reconstruction, and super-resolution [14]. To converge, however, they require assumptions on the denoiser networks, e.g., contractivity [15]. Unlike DUNets, PnP methods can handle different operators $\mathbf{A}$ during training and testing. However, to achieve reconstruction performance similar to end-to-end DNNs, they need to be trained on very large datasets.

Post-processing (PP). While PnP methods usually output measurement-consistent signals, other strategies include designing data-consistency layers [20, 21, 6] or using cycle consistency losses [22]. A particular effective strategy, however, is post-processing (PP) the output $\boldsymbol{w}$ of an existing DNN with a model-based optimization problem [23]:

$$\widehat{\boldsymbol{x}}(\boldsymbol{w}) \in \begin{array}{ll} \underset{\boldsymbol{x}}{\arg\min} & \|\boldsymbol{x}\|_{\mathrm{TV}} + \beta\|\boldsymbol{x}-\boldsymbol{w}\|_{\mathrm{TV}} \\ \text{s.t.} & \mathbf{A}\boldsymbol{x} = \boldsymbol{b}\,, \end{array} \tag{1}$$

where $\|\cdot\|_{\mathrm{TV}}$ is a total-variation (TV) norm [24], and $\beta > 0$ a tradeoff parameter. Such PP framework achieves better results than existing DNN on several tasks, including image super-resolution [9], MRI reconstruction [10], hyperspectral image super-resolution [25]. Moreover, PP can handle slight variations in $\mathbf{A}$ [9]. A shortcoming, however, is the rigid selection of an explicit function, such as the TV-norm, and the inability to train the network end-to-end. Our work builds on PP by replacing the TV-norms in (1) with shallow neural networks, and by leveraging techniques from deep equilibrium networks to make the entire system end-to-end trainable.

Deep equilibrium networks (DEQs). DEQs [26] achieve SOTA results on several vision [27, 13], language [26], and graph [28] tasks. The output of a DEQ (or implicit) layer, rather than being defined explicitly as in conventional DNNs, is defined implicitly. For example, if $\boldsymbol{w}$ represents its input, the output of an implicit layer is the solution of $h(\boldsymbol{x};\boldsymbol{w}) = 0$, where h is a generic differentiable map. This generic model captures fixed-point equations, generalizing recurrent backpropagation models [29], or optimality conditions of optimization problems, effectively enabling the inclusion of hard-constraints in DNNs [30]. Unlike DUNets, the number of iterations to solve $h(\boldsymbol{x};\boldsymbol{w}) = 0$ in the forward pass can be as large as required, without incurring any additional memory footprint during the backward pass. This is because the gradient can be propagated through the implicit layer via the implicit function theorem. Thus, an implicit layer can be sandwiched between conventional, explicit layers, and the whole network trained end-to-end.

DEQs have been applied to solving inverse problems in [13]. The main difference with respect to our work is that they apply DEQ machinery directly to a PnP method: ADMM solving a LASSO-type problem with the proximal operator replaced by a DNN. Thus, they fail to leverage the power of end-to-end SOTA DNNs. Our method, in contrast, leverages such methodologies and, as a result, requires a significantly shallower DNN, with lower computational cost during both training and testing, while achieving similar results, or better.

3. PROPOSED FRAMEWORK: MCNet

As depicted in Fig. 1, we propose to append an implicit layer to the output of a fixed, pre-trained DNN, whose output is $\boldsymbol{w}$. The implicit layer solves a constrained optimization problem,

$$\widehat{\boldsymbol{x}}(\boldsymbol{w}) \in \underset{\boldsymbol{x}\in\mathcal{S}}{\arg\min}\, f(\boldsymbol{x}) + \beta g(\boldsymbol{x}-\boldsymbol{w})\,, \tag{2}$$

where $\mathcal{S} = \{\boldsymbol{x} : \|\boldsymbol{b} - \mathbf{A}\boldsymbol{x}\|_2 \leq \varepsilon\}$, for $\varepsilon \geq 0$, and $f, g : \mathbb{R}^n \to \mathbb{R}$ are generic, possibly nonconvex, functions. We estimate $\hat{\boldsymbol{x}}$ in (2) by PnP-ADMM iterations similar to [15] and replace the proximal operators of f and/or g by simple CNN denoisers. In this work, we consider simpler case and use $g(\cdot) = \|\cdot\|_2^2$, . The resulting algorithm can be interpreted as a fixed-point iteration.

ADMM as a fixed-point iteration. For simplicity, we consider the case $\varepsilon = 0$. Cloning the variable $\boldsymbol{x}$ into $\boldsymbol{u}$ and setting $g(\cdot) = \|\cdot\|_2^2$, we rewrite (2) as

$$\begin{array}{ll} \underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} & f(\boldsymbol{u}) + \frac{\beta}{2}\|\boldsymbol{x}-\boldsymbol{w}\|_2^2 + \iota_{\mathcal{S}}(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} = \boldsymbol{u}\,, \end{array} \tag{3}$$

where $\iota_{\mathcal{S}}(s) = 0$ if $s \in \mathcal{S}$, and $\iota_{\mathcal{S}}(s) = +\infty$ if $s \notin \mathcal{S}$, is the indicator function of $\mathcal{S}$. Applying ADMM to (3) results in

$$\boldsymbol{u}^{(k+1)} = \underset{\boldsymbol{u}}{\arg\min}\, f(\boldsymbol{u}) + \frac{\rho}{2}\|\boldsymbol{u}-\boldsymbol{p}^{(k)}\|_2^2 = \mathrm{prox}_{\frac{f}{\rho}}\left(\boldsymbol{p}^{(k)}\right) \tag{4a}$$

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x}\in\mathcal{S}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{q}^{(k)}\|_2^2 = \Pi_{\mathcal{S}}\big[\boldsymbol{q}^{(k)}\big] \tag{4b}$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \boldsymbol{x}^{(k+1)} - \boldsymbol{u}^{(k+1)}\,, \tag{4c}$$

where $\rho > 0$ is the augmented Lagrangian parameter, $\mathrm{prox}_f(\boldsymbol{v}) := \arg\min_{\boldsymbol{x}}\ f(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{v}\|_2^2$ the proximal operator of f at $\boldsymbol{v}$, and $\Pi_{\mathcal{S}}[\cdot]$ the projection operator (in this case, it has a closed-form solution). We defined $\boldsymbol{p}^{(k)} := \big(\boldsymbol{x}^{(k)} + \boldsymbol{\lambda}^{(k)}\big)$ and $\boldsymbol{q}^{(k)} := \big(\beta\boldsymbol{w} + \rho(\boldsymbol{u}^{(k+1)} - \boldsymbol{\lambda}^{(k)})\big)/\big(\beta+\rho\big)$.

The key step is replacing $\mathrm{prox}_{f/\rho}$ by a neural network $R_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$. Performing this substitution and plugging $\boldsymbol{u}^{(k+1)}$

directly into $\boldsymbol{q}^{(k)}$ and (4c) yields:

$$\boldsymbol{x}^{(k+1)} = \Pi_{\mathcal{S}}\left[\frac{\beta \boldsymbol{w} + \rho\Big(R_{\boldsymbol{\theta}}\big(\boldsymbol{x}^{(k)} + \boldsymbol{\lambda}^{(k)}\big) - \boldsymbol{\lambda}^{(k)}\Big)}{\beta+\rho}\right] \tag{5a}$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} - R_{\boldsymbol{\theta}}\big(\boldsymbol{x}^{(k)} + \boldsymbol{\lambda}^{(k)}\big) + \Pi_{\mathcal{S}}\left[\frac{\beta \boldsymbol{w} + \rho\Big(R_{\boldsymbol{\theta}}\big(\boldsymbol{x}^{(k)} + \boldsymbol{\lambda}^{(k)}\big) - \boldsymbol{\lambda}^{(k)}\Big)}{\beta+\rho}\right]. \tag{5b}$$

As the right-hand side of (5) depends only on variables indexed by k, i.e., $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\lambda}^{(k)}$, those equations can be interpreted as a fixed-point iteration on the joint variable $\boldsymbol{z} = \big(\boldsymbol{x}, \boldsymbol{\lambda}\big)$. More compactly, they can be expressed as

$$\boldsymbol{z}^{(k+1)} = F_{\boldsymbol{\theta}}\Big(\boldsymbol{z}^{(k)}; \boldsymbol{w}, \mathbf{A}, \boldsymbol{b}\Big), \tag{6}$$

where $F_{\boldsymbol{\theta}} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n \times \mathbb{R}^n$ depends on the fixed parameters $\boldsymbol{w}$, $\mathbf{A}$, and $\boldsymbol{b}$, and on the parameters $\boldsymbol{\theta}$ of the network $R_{\boldsymbol{\theta}}$. If $F_{\boldsymbol{\theta}}$ is contractive, then it has a unique fixed-point $\boldsymbol{z}^{(\infty)} = \big(\boldsymbol{x}^{(\infty)}, \boldsymbol{\lambda}^{(\infty)}\big)$, and the iterations (6) [equivalently, (5)] converge to it. Below, we establish the conditions under which this holds. A similar derivation applies to the case where $\varepsilon > 0$.

Forward and backward passes. In the forward-pass, if we directly use equations (5a)-(5b) to compute the fixed-point of (6), convergence can be very slow. Instead, as in [26], we apply Anderson acceleration [31], which uses a few past iterations, typically 4-6, to identify promising directions to move along in each iteration. In all our experiments, we set $\rho = 1$ and find that 150-200 iterations of Anderson acceleration achieve good accuracy.

In the backward-pass, we need to compute the gradient of the loss ℓ with respect to $\boldsymbol{\theta}$, i.e., $\partial\ell/\partial\boldsymbol{\theta}$. Following [26], we leverage the fact that $\boldsymbol{x}^{(\infty)} := \boldsymbol{x}^{(\infty)}(\boldsymbol{w}, \mathbf{A}, \boldsymbol{b}; \boldsymbol{\theta})$ is a fixed-point of $F_{\boldsymbol{\theta}}$, and calculate the required gradient without unrolling the fixed-point iterations. Using the chain rule and the convention that $\partial/\partial\cdot$ represents a row vector, we write $\partial\ell/\partial\boldsymbol{\theta}$ as a Jacobian-vector product:

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}} = \frac{\partial \boldsymbol{x}^{(\infty)}}{\partial \boldsymbol{\theta}}^{\top} \frac{\partial \ell}{\partial \boldsymbol{x}^{(\infty)}}. \tag{7}$$

To compute the Jacobian $\partial\boldsymbol{x}^{(\infty)}/\partial\boldsymbol{\theta}$, we apply implicit differentiation to (6) at the fixed-point $\boldsymbol{z}^{(\infty)} = (\boldsymbol{x}^{(\infty)}, \boldsymbol{\lambda}^{(\infty)})$. After simplifying,

$$\frac{\partial \boldsymbol{x}^{(\infty)}}{\partial \boldsymbol{\theta}} = \left[\mathbf{I} - \frac{\partial F_{\boldsymbol{\theta}}(\boldsymbol{x};\cdot)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}^{(\infty)}}\right]^{-1} \left[\frac{\partial F_{\boldsymbol{\theta}}(\boldsymbol{x}^{(\infty)};\cdot)}{\partial \boldsymbol{\theta}}\right]. \tag{8}$$

Plugging (8) into (7) yields:

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}} = \left[\frac{\partial F_{\theta}(\boldsymbol{x}^{(\infty)};\cdot)}{\partial \boldsymbol{\theta}}\right]^{\top} \underbrace{\left[\mathbf{I} - \frac{\partial F_{\boldsymbol{\theta}}(\boldsymbol{x};\cdot)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}^{(\infty)}}\right]^{-\top} \frac{\partial \ell}{\partial \boldsymbol{x}^{(\infty)}}}_{=:\boldsymbol{s}^{(\infty)}}, \tag{9}$$

where $\mathbf{I}$ is the identity matrix. The quantity $\boldsymbol{s}^{(\infty)}$ in (9) can be computed as the fixed-point of

$$\boldsymbol{s}^{(k+1)} = \left[\frac{\partial F_{\boldsymbol{\theta}}(\boldsymbol{x};\cdot)}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}^{(\infty)}}\right]^{\top} \boldsymbol{s}^{(k)} + \frac{\partial \ell}{\partial \boldsymbol{x}^{(\infty)}}. \tag{10}$$

We again use Anderson acceleration to compute this fixed-point. This can be done efficiently with recent auto-differentiation tools that compute the Jacobian-vector product in (10). Once $\boldsymbol{s}^{(\infty)}$ is computed, we replace it in (9) and obtain $\partial\ell/\partial\boldsymbol{\theta}$. In all our training, we find that 50-80 iterations of Anderson acceleration are enough to achieve good accuracy in both the forward and backward passes.

Convergence of fixed-point iterations. Fixed-point theory guarantees that the iterations (6) and (10) converge to a unique fixed-point if the associated iteration maps are contractive. The following result slightly modifies the argument in [15] to show that $F_{\boldsymbol{\theta}}$ is contractive.

Theorem 1. *If the network $R_{\boldsymbol{\theta}}$ is contractive, then the map $F_{\boldsymbol{\theta}}$ is contractive, and the iterations* (6) *converge to a unique fixed-point.*

Proof. As in [15], the ADMM iterations in the forward pass are equivalent to Douglas-Rachford splitting iterations, which can be written as $\boldsymbol{y}^{(k+1)} = T(\boldsymbol{y}^{(k)})$ with

$$T = \frac{1}{2}\mathbf{I} + \frac{1}{2}\big(2\Pi_{\mathcal{S}} - \mathbf{I}\big)\Big(2\operatorname{prox}_{\frac{f}{\rho}} - \mathbf{I}\Big). \tag{11}$$

The term $\big(2\Pi_{\mathcal{S}} - \mathbf{I}\big)$ is non-expansive, since $\Pi_{\mathcal{S}}$ is a projection onto a nonempty, closed, and convex set $\mathcal{S}$. If we assume $\big(2\operatorname{prox}_{f/\rho} - \mathbf{I}\big)$ is contractive, then T becomes contractive as well. This is the case if we replace $\operatorname{prox}_{f/\rho}$ by a contractive network $R_{\boldsymbol{\theta}}$. □

The proof differs from the one in [15] in that $\Pi_{\mathcal{S}}$ in their case is a contractive map. Thus, we need to assume the last term of (11) is contractive. To make $R_{\boldsymbol{\theta}}$ contractive, as in [15], we constrain the Lipschitz constant of $R_{\boldsymbol{\theta}}$ by spectral normalization (SN) of its weights. In particular, we consider $R_{\boldsymbol{\theta}}$ to be a 6-layer CNN with 1-Lipschitz ReLU activation functions, and apply SN to each convolutional layer, bounding their spectral norm below 1.

Likewise, (10) converges to a unique fixed-point whenever the Jacobian $\partial_{\boldsymbol{x}} F_{\boldsymbol{\theta}}(\boldsymbol{x};\cdot)$, evaluated at any $\boldsymbol{x}$, has a spectral norm strictly below 1. As shown in [13], this is equivalent to $F_{\boldsymbol{\theta}}$ being contractive. Thus, a contractive $R_{\boldsymbol{\theta}}$ also ensures the convergence of (10).

4. EXPERIMENTAL RESULTS

To illustrate the effectiveness of the proposed MCNet, we focused on one imaging application: single-image super-resolution.

Experimental setup. As backbone networks for MCNet, we selected the SOTA DNNs SRCNN [5], RDN [32] and RCAN [11]. SRCNN was designed for single channel images, while RDN and RCAN support RGB color images. To make performance assessment uniform across different methods, we designed MCNet to operate on a single channel: Y from the YCbCr colorspace. Thus, MCNet takes in only the Y channel from the outputs of RDN and RCAN. We used pretrained models of these DNNs, obtained from the respective authors, and keep them fixed during the experiments. We also included super-resolved (SR) outputs obtained by bicubic interpolation, TV-TV minimization [9], and a version of the PnP-ADMM method that solves (3) with $\beta = 0$, and uses a DNCNN [15] with 17-layers as a denoiser. We tested all methods at scales 2, 3, and 4, on the benchmark datasets Set5 [33] and Set14 [34]. Low-resolution (LR) inputs are generated from original high-resolution (HR) images by bicubic interpolation. And we assess the quality of the algorithms' outputs by computing PSNR and SSIM values on the Y channel image.

Training MCNet. We selected $R_{\boldsymbol{\theta}}$ in MCNet as a DNCNN [15] with 6-layers, but without biases in the convolution layer and no batch normalization. We will refer to it as DNCNN6. We observed that increasing the number of layers significantly increases the computational cost, but does not translate into observable gains in image quality. To train MCNet, we first pretrained $R_{\boldsymbol{\theta}}$ using 4 different DNCNN6 networks at different levels of white Gaussian noise, and selected the one that performed the best on Set5 with

Table 1. Average data-fidelity $\|\mathbf{A}\widehat{\boldsymbol{x}} - \boldsymbol{b}\|_2$, where $\widehat{\boldsymbol{x}}$ is the output of an algorithm, and $\boldsymbol{b}$ the input. Values for $\times 2$ scaling on dataset Set5. The smaller the values, the better. Similar values are obtained on dataset Set14.

Method	**Scale**	$\|\mathbf{A}\widehat{\boldsymbol{x}} - \boldsymbol{b}\|_2$
Bicubic	$\times 2$	1.2652×10^{0}
PnP	$\times 2$	2.2167×10^{-5}
SRCNN[5]	$\times 2$	3.0626×10^{-1}
RDN[32]	$\times 2$	8.3431×10^{-2}
RCAN[11]	$\times 2$	8.5996×10^{-2}
TVTV [9]+SRCNN	$\times 2$	7.4337×10^{-6}
TVTV+RDN	$\times 2$	7.0079×10^{-6}
TVTV+RCAN	$\times 2$	7.1031×10^{-6}
MCNet+SRCNN	$\times 2$	2.6276×10^{-5}
MCNet+RDN	$\times 2$	4.1483×10^{-1}
MCNet+RCAN	$\times 2$	4.4271×10^{-1}

the PnP-ADMM method mentioned above. Using the selected DNCNN6, we then searched for a good initial value for parameter β using a coarse grid, by solving (3) with the ADMM iterations in (4). This pretraining of $R_{\boldsymbol{\theta}}$ and β enabled training MCNet successfully in a few number of epochs, 40. For training, we followed the methodology in [15]. In particular, the contractiveness of $R_{\boldsymbol{\theta}}$ was ensured by applying SN to all layers. MCNet was trained on image patches of the dataset T91 and validated on Set5. Adam algorithm [35] was applied with an initial learning-rate 10^{-4}, which was reduced to 10^{-5}. Finally, we mention that MCNet+SRCNN (i.e., MCNet using SRCNN as a backbone) was trained with $\varepsilon = 0$, whereas MCNet+RDN and MCNet+RCAN were trained with $\varepsilon = 0.1$ [cf. the definition of $\mathcal{S}$ after (2)]. In fact, we observed that decreasing ε slightly decreases the performance of MCNet. Thus, there is a trade-off between ensuring measurement consistency and obtaining good image quality. Other details on the training and code to replicate our experiments will be available online.[1].

Results. Tables 1 and 2 present the results of our experiments, with our method being denoted as MCNet+Fixed-DNN. Table 1 shows the average data fidelity, $\|\mathbf{A}\widehat{\boldsymbol{x}} - \boldsymbol{b}\|$, of several methods evaluated on Set5. It can be observed that methods based on constrained optimization, including MCNet, have consistency values much better than SOTA, end-to-end DNNs. Specifically, MCNet+SRCNN, which uses $\varepsilon = 0$, yields a consistency value akin to TV-TV and PnP, whereas when we relax ε to 0.1, consistency values are similar to end-to-end networks.

Better measurement consistency, however, does not always translate into better image quality, as shown in Table 2, which depicts PSNR and SSIM values. For example, for $\times 4$ scaling (3rd block), PnP is outperformed by SRCNN. Yet, the table shows that, in all but one case, MCNet outperforms all the other methods, sometimes by a large margin.

Table 2. Average PSNR (SSIM) in dB on datasets Set5 and Set14, for $\times 2$, $\times 3$, and $\times 4$ scaling. The higher the values, the better. The table compares 11 different methods and best values are highlighted.

Method	**Scale**	**Set5**	**Set14**
Bicubic	$\times 2$	33.6890 (0.9375)	30.2327 (0.8834)
PnP	$\times 2$	**37.4337 (0.9814)**	32.8681 (0.9496)
SRCNN	$\times 2$	36.6530 (0.9597)	32.5932 (0.9169)
TVTV+SRCNN	$\times 2$	36.7813 (0.9607)	32.7086 (0.9185)
MCNet+SRCNN	$\times 2$	37.2272 (0.9623)	**32.8980 (0.9195)**
Bicubic	$\times 3$	30.8867 (0.8941)	27.8792 (0.8110)
PnP	$\times 3$	33.6468 (0.9650)	29.9157 (0.9077)
SRCNN	$\times 3$	33.3614 (0.9323)	29.9630 (0.8540)
TVTV+SRCNN	$\times 3$	33.4805 (0.9343)	30.0659 (0.8569)
MCNet+SRCNN	$\times 3$	**34.0458 (0.9401)**	**30.2655 (0.8604)**
Bicubic	$\times 4$	28.4337 (0.8234)	25.9724 (0.7260)
PnP	$\times 4$	30.2903 (0.9262)	27.4729 (0.8539)
SRCNN	$\times 4$	30.3270 (0.8726)	27.6146 (0.7724)
TVTV+SRCNN	$\times 4$	30.4829 (0.8771)	27.7241 (0.7770)
MCNet+SRCNN	$\times 4$	**31.0690 (0.8888)**	**27.9799 (0.7839)**
RDN	$\times 2$	38.0434 (0.9657)	33.8416 (0.9277)
TVTV+RDN	$\times 2$	38.0449 (0.9657)	33.8351 (0.9278)
MCNet+RDN	$\times 2$	**38.1246 (0.9657)**	**33.8765 (0.9277)**
RDN	$\times 3$	35.2322 (0.9487)	31.1424 (0.8749)
TVTV+RDN	$\times 3$	35.2339 (0.9487)	31.1510 (0.8749)
MCNet+RDN	$\times 3$	**35.2808 (0.9486)**	**31.1903 (0.8749)**
RDN	$\times 4$	32.3311 (0.9069)	28.8365 (0.8050)
TVTV+RDN	$\times 4$	32.3399 (0.9069)	28.8407 (0.8047)
MCNet+RDN	$\times 4$	**32.3452 (0.9068)**	**28.8458 (0.8050)**
RCAN	$\times 2$	38.1996 (0.9662)	34.2138 (0.9304)
TVTV+RCAN	$\times 2$	38.2041 (0.9662)	34.1978 (0.9304)
MCNet+RCAN	$\times 2$	**38.2838 (0.9662)**	**34.2512 (0.9304)**
RCAN	$\times 3$	35.4484 (0.9501)	31.3028 (0.8774)
TVTV+RCAN	$\times 3$	35.4548 (0.9501)	31.3090 (0.8772)
MCNet+RCAN	$\times 3$	**35.5024 (0.9500)**	**31.3463 (0.8773)**
RCAN	$\times 4$	32.6554 (0.9099)	28.9400 (0.8076)
TVTV+RCAN	$\times 4$	**32.6655 (0.9099)**	28.9579 (0.8073)
MCNet+RCAN	$\times 4$	32.6653 (0.9094)	**28.9782 (0.8073)**

5. CONCLUSIONS

We proposed a framework to transform any existing DNN for solving inverse problems into a measurement-consistent network, when the acquisition model is known precisely. The core of the framework is an implicit layer defined by a constrained optimization problem, which is solved by an ADMM algorithm with learnable parameters. Extensive experimental results on a single-image super-resolution problem show superior performance with respect to end-to-end DNNs, plug-and-play, and other methods. Although the proposed method requires solving an optimization problem during training and inference, the additional computational cost is justified in applications that require precise fidelity to measurements, for example, in areas in medicine, astronomy or defence. Future work includes exploring different applications and different algorithmic choices, e.g., using a shallow CNN instead of an ℓ_2-norm to integrate the output of the fixed-DNN into our method.

[1]https://github.com/mouryarahul/SR-MCNet

6. REFERENCES

[1] Patrick L Combettes and Valérie R Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling and Simulation*, vol. 4, pp. 1168–1200, 2005.

[2] Amir Beck and Marc Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[3] Rahul Mourya, Loïc Denis, Jean-Marie Becker, and Eric Thiébaut, "Augmented lagrangian without alternating directions: Practical algorithms for inverse problems in imaging," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 1205–1209.

[4] T M Nimisha and et al, "Blur-invariant deep learning for blind-deblurring," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4762–4770.

[5] Chao Dong and et al., "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, pp. 295–307, 2 2016.

[6] C. Qin and et al, "Convolutional recurrent neural networks for dynamic MR image reconstruction," *IEEE Trans. Med. Imaging*, vol. 38, no. 1, pp. 280–290, 2019.

[7] Anuroop Sriram and et al., "End-to-end variational networks for accelerated mri reconstruction," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Cham, 2020, pp. 64–73, Springer International Publishing.

[8] Vegard Antun and et al., "On instabilities of deep learning in image reconstruction and the potential costs of ai," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30088–30095, 2020.

[9] Marija Vella and João F.C. Mota, "Robust single-image super-resolution via cnns and tv-tv minimization," *IEEE Transactions on Image Processing*, vol. 30, pp. 7830–7841, 2021.

[10] Marija Vella and João F.C. Mota, "Overcoming measurement inconsistency in deep learning for linear inverse problems: Applications in medical imaging," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 8113–8117.

[11] Yulun Zhang and et al, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European conference on computer vision (ECCV)*, 7 2018, pp. 286–301.

[12] Olaf Ronneberger and et al, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Cham, 2015, pp. 234–241, Springer International Publishing.

[13] Davis Gilton, Greg Ongie, and Rebecca Willett, "Deep equilibrium architectures for inverse problems in imaging," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 1123–1133, 2021.

[14] Weisheng Dong and et al., "Denoising prior driven deep neural network for image restoration," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 10, pp. 2305–2318, 2018.

[15] Ernest Ryu and et al., "Plug-and-play methods provably converge with properly trained denoisers," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5546–5557.

[16] Karol Gregor and Yann LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the 27th international conference on international conference on machine learning*, 2010, pp. 399–406.

[17] Singanallur V. Venkatakrishnan, Charles A. Bouman, and Brendt Wohlberg, "Plug-and-play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013 - Proceedings*, 2013, pp. 945–948.

[18] Kostadin Dabov and et al., "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on image processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[19] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, pp. 3142–3155, 8 2017.

[20] Hemant K. Aggarwal, Merry P. Mani, and Mathews Jacob, "Modl: Model-based deep learning architecture for inverse problems," *IEEE Transactions on Medical Imaging*, vol. 38, pp. 394–405, 2 2019.

[21] J. Schempler, 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. Imaging*, vol. 37, no. 2, pp. 491–503, 2018.

[22] G. Oh, B. Sim, H. Chung, L. Sunwoo, and J. C. Ye, "Unpaired deep learning for accelerated MRI using optimal transport driven CycleGAN," *IEEE Trans. Comput. Imaging*, vol. 6, pp. 1285–1296, 2020.

[23] Marija Vella and Joao F.C. Mota, "Single image super-resolution via cnn architectures and tv-tv minimization," in *Proceedings of the British Machine Vision Conference (BMVC)*, Kirill Sidorov and Yulia Hicks, Eds. September 2019, pp. 219.1–219.12, BMVA Press.

[24] Antonin Chambolle and Pierre-Louis Lions, "Image recovery via total variation minimization and related problems," *Numerische Mathematik*, vol. 76, pp. 167–188, 1997.

[25] Marija Vella and et al., "Enhanced hyperspectral image super-resolution via rgb fusion and tv-tv minimization," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3837–3841.

[26] Shaojie Bai, J Zico Kolter, and Vladlen Koltun, "Deep equilibrium models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[27] Shaojie Bai, Vladlen Koltun, and J Zico Kolter, "Multiscale deep equilibrium models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5238–5250, 2020.

[28] Fangda Gu and et al., "Implicit graph neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11984–11995, 2020.

[29] Renjie Liao and et al., "Reviving and improving recurrent back-propagation," in *Proceedings of the 35th International Conference on Machine Learning*, Jennifer Dy and Andreas Krause, Eds. 10–15 Jul 2018, vol. 80 of *Proceedings of Machine Learning Research*, pp. 3082–3091, PMLR.

[30] B. Amos and J. Z. Kolter, "OptNet: Differentiable optimization as a layer in neural networks," in *Inter. Conf. Machine Learning*, 2017, pp. 1–13.

[31] Homer F. Walker and Peng Ni, "Anderson acceleration for fixed-point iterations," *SIAM Journal on Numerical Analysis*, vol. 49, pp. 1715–1735, 2011.

[32] Yulun Zhang and et al, "Residual dense network for image super-resolution," in *CVPR*, 2018, pp. 2472–2481.

[33] Marco Bevilacqua and et al, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *Proceedings of the British Machine Vision Conference*, 2012, pp. 135.1–135.10.

[34] Roman Zeyde and et al, "On single image scale-up using sparse-representations," in *Curves and Surfaces*, Berlin, Heidelberg, 2012, pp. 711–730.

[35] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, Yoshua Bengio and Yann LeCun, Eds., 2015.