Invertible Diffusion Models for Compressed Sensing

Bin Chen, Zhenyu Zhang, Weiqi Li, Chen Zhao, Jiwen Yu, Shijie Zhao, Jie Chen, and Jian Zhang, *Member, IEEE*

***Abstract*—While deep neural networks (NNs) significantly advance image compressed sensing (CS) by improving reconstruction quality, the necessity of training current CS NNs from scratch constrains their effectiveness and hampers rapid deployment. Although recent methods utilize pre-trained diffusion models for image reconstruction, they struggle with slow inference and restricted adaptability to CS. To tackle these challenges, this paper proposes Invertible Diffusion Models (IDM), a novel efficient, end-to-end diffusion-based CS method. IDM repurposes a large-scale diffusion sampling process as a reconstruction model, and fine-tunes it end-to-end to recover original images directly from CS measurements, moving beyond the traditional paradigm of one-step noise estimation learning. To enable such memory-intensive end-to-end fine-tuning, we propose a novel two-level invertible design to transform both 1) multi-step sampling process and 2) noise estimation U-Net in each step into invertible networks. As a result, most intermediate features are cleared during training to reduce up to 93.8% GPU memory. In addition, we develop a set of lightweight modules to inject measurements into noise estimator to further facilitate reconstruction. Experiments demonstrate that IDM outperforms existing state-of-the-art CS networks by up to 2.64 dB in PSNR. Compared to the recent diffusion-based approach DDNM, our IDM achieves up to 10.09 dB PSNR gain and 14.54 times faster inference.**

***Index Terms*—Compressed sensing, diffusion models, invertible neural networks, compressive imaging, and image reconstruction.**

I. Introduction

COMPRESSED sensing (CS) [1], [2] is a novel signal acquisition paradigm that surpasses the Nyquist-Shannon theorem limit [3]. It has inspired a range of imaging applications, including single-pixel imaging (SPI) [4], magnetic resonance imaging (MRI) [5], computational tomography (CT) [6], [7], and snapshot compressive imaging (SCI) [8], [9], [10], [11], [12], [13]. In this paper, we focus on natural image CS reconstruction, aiming to recover the original image $\mathbf{x} \in \mathbb{R}^N$ from its linear measurements $\mathbf{y} \in \mathbb{R}^M$. These measurements are projected through a random sampling matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$ with a CS ratio $\gamma = M/N$, using $\mathbf{y} = \mathbf{A}\mathbf{x}$. A low γ value, where $M \ll N$, is generally preferred for the appealing benefits such as lower energy consumption and shorter signal acquisition time. However, this also introduces the challenge of reconstructing $\mathbf{x}$ from limited information $\{\mathbf{y}, \mathbf{A}, \gamma\}$ due to the ill-posed nature of this inverse problem.

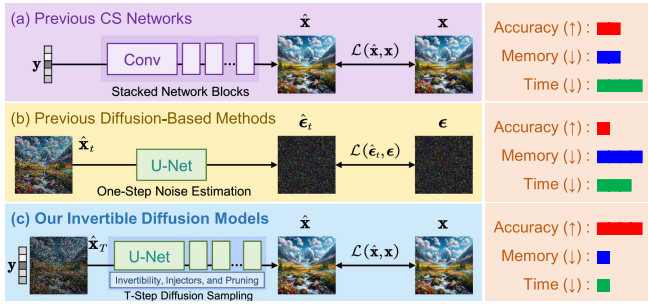

Fig. 1. Proposed IDM compared to previous methods. (a) Conventional NN-based works [23] develop and train new CS NN architectures from scratch, limiting their ability to achieve higher performance within a short timeframe for rapid deployment. (b) Traditional diffusion-based image reconstruction methods [24] train a one-step noise estimation U-Net and use it as an off-the-shelf NN module for iterative sampling. This estimator lacks awareness of the entire recovery process from measurement to image, reducing its adaptability to CS. (c) Our invertible diffusion models (IDM) fine-tune a large-scale, pre-trained diffusion sampling process to directly predict original images from CS measurements end-to-end, significantly improving performance while reducing the required sampling steps (Contribution 1). We further make the sampling process and noise estimation U-Net invertible, adding measurement injectors into our pruned noise estimation U-Net, resulting in a substantial performance boost while greatly reducing training GPU memory and runtime (Contributions 2 and 3). Here, (a), (b), and (c) correspond to (12), (1), and (9) in Table V, respectively. Please refer to Section IV-C for more details.

In the field of natural image CS reconstruction, deep neural networks (NNs) demonstrate greater effectiveness than traditional optimization-based methods [14], [15], [16] in terms of accuracy and efficiency. Early CS NNs [17], [18] treat CS recovery as a de-aliasing problem, achieving fast, non-iterative reconstruction with a single forward pass through NN layers. Techniques like deep algorithm unrolling [19], plug-and-play (PnP) [20], [21], and regularization by denoising (RED) [22] effectively separate measurement consistency enforcement from image prior learning, balancing performance and interpretability. However, as shown in Fig. 1(a), these methods typically require designing and training new NN architectures from scratch, a time-consuming process often resulting in suboptimal performance.

Recently, diffusion models [25] have been widely leveraged for image reconstruction in many works [24], [26], [27], [28],

Received 15 May 2024; revised 22 January 2025; accepted 26 January 2025. Date of publication 5 February 2025; date of current version 3 April 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62372016 and in part by Guangdong Provincial Key Laboratory of Ultra High Definition Immersive Media Technology under Grant 2024B1212010006. Recommended for acceptance by R. Giryes. *(Corresponding author: Chen Zhao.)*

Bin Chen, Zhenyu Zhang, Weiqi Li, Jiwen Yu, Jie Chen, and Jian Zhang are with the School of Electronic and Computer Engineering, Peking University, Shenzhen 518055, China (e-mail: chenbin@stu.pku.edu.cn; zhenyuzhang@pku.edu.cn; liweiqi@stu.pku.edu.cn; yujiwen@stu.pku.edu.cn; jiechen2019@pku.edu.cn; zhangjian.sz@pku.edu.cn).

Chen Zhao is with the King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia (e-mail: chen.zhao@kaust.edu.sa).

Shijie Zhao is with the ByteDance Inc, Shenzhen 518055, China (e-mail: zhaoshijie.0526@bytedance.com).

Code is available at https://github.com/Guaishou74851/IDM.

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2025.3538896, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2025.3538896

[29], [30] by utilizing pre-trained generative denoising priors. These approaches iteratively sample an image estimate $\hat{\mathbf{x}}$ through steps from the posterior distribution $p(\mathbf{x}|\mathbf{y})$, achieving impressive measurement-conditioned synthesis [26]. Similar to PnP methods, current diffusion-based approaches focus on learning a noise estimation network [24] or using pre-trained ones to address the image prior subproblems.

However, these methods often require extensive tuning of hyperparameters, such as step size, regularization coefficient, and noise level. Furthermore, as Fig. 1(b) shows, their NN backbones are mainly trained for one-step noise estimation, not directly optimized for the full reconstruction process from CS measurements $\mathbf{y}$ to the target original image $\mathbf{x}$. Additionally, previous diffusion-based inverse problem solvers [31], [32], [33], [34] can require numerous iterations, ranging from 10 to 1000, to achieve satisfactory results. Approaches using latent diffusion models, like stable diffusion (SD) [35], [36], typically involve multi-stage training [37] or frequent transitions between image and latent spaces via deep VAE encoders and decoders during sampling, reducing the efficiency of CS imaging systems.

To address these challenges, in this paper, we propose **I**nvertible **D**iffusion **M**odels (**IDM**) for image CS reconstruction, as shown in Fig. 1(c). Moving beyond the limitations of one-step noise estimation learning in existing methods, our IDM establishes a novel end-to-end framework, training directly to align a large-scale, pre-trained diffusion sampling process with the ideal reconstruction mapping $\mathbf{y} \mapsto \mathbf{x}$. This alignment ensures that all diffusion parameters and weights of noise estimation network are specifically optimized for image CS reconstruction, achieving significant performance gains and eliminating the need for a great deal of sampling steps. However, training such large diffusion models end-to-end demands high GPU memory, making it nearly infeasible on standard GPUs. To address this, we leverage the memory efficiency of invertible NNs [38] and propose a novel two-level invertible design. This design introduces auxiliary connections into (1) the overall diffusion sampling framework across multiple steps and (2) noise estimation U-Net within each step, transforming both into invertible networks. This allows IDM to clear most intermediate features during the forward pass and recompute them in back-propagation, significantly reducing GPU memory consumption.

Compared with previous end-to-end CS NNs that require training from scratch, IDM leverages pre-trained diffusion models, adapting them to CS with minimal fine-tuning. Additionally, we introduce lightweight NN modules, called injectors, that directly integrate the information of measurement $\mathbf{y}$ into the deep features of the noise estimation U-Net at various spatial scales to enhance reconstruction. Our method harnesses the power of pre-trained diffusion models to improve CS performance, establishing a new path for the customized application of large-scale diffusion models in image reconstruction. In summary, our contributions are:

1) We propose IDM, an efficient, end-to-end diffusion-based CS method. Unlike previous one-step diffusion noise estimation, IDM fine-tunes a large-scale sampling process for direct recovery from CS measurements end-to-end, improving performance by up to 4.34 dB in PSNR with a 98% reduction in step number and over 25 times faster inference.

2) We introduce a novel two-level invertible design for both the diffusion sampling framework and noise estimation U-Net, reducing fine-tuning memory by up to 93.8%.

3) We develop a set of lightweight injector modules that integrate CS measurements and sampling matrix into the noise estimation U-Net to enhance CS reconstruction. These injectors achieve over 2 dB PSNR improvements with only 0.02 M extra parameters, compared to over 100 M of U-Net.

4) We evaluate our IDM, as shown in Fig. 1(c), across four typical tasks: natural image CS, inpainting, accelerated MRI, and sparse-view CT. IDM achieves a new state-of-the-art (SOTA), surpassing existing CS NNs by up to 2.64 dB in PSNR, with up to 10.09 dB PSNR gain and 14.54 times faster inference compared to diffusion-based method DDNM [39].

II. Related Work

A. Deep End-to-End Learned Image CS Networks

Foundational NN-based CS research [17], [40] pioneered the use of fully connected layers to decode measurements into images. The introduction of convolutions [18] and self-attention layers [41] led to more effective architectures like residual blocks [42] and Transformers [43], which capture both local and long-range dependencies in images. This popularized expressive NN designs such as hierarchical and non-local architectures [44], [45]. Recent works [46], [47] leverage measurement information $\{\mathbf{y}, \mathbf{A}, \gamma\}$ to build physics-informed CS NNs. Among these methods, deep unrolling [48], which reinterprets truncated optimization inferences as iterative NNs, has established a new paradigm in this field. In comparison, our work introduces powerful diffusion priors, avoids resource-intensive training from scratch, and enhances the efficiency of CS recovery learning.

B. Diffusion-Based Image Reconstruction

Denoising diffusion models, particularly the denoising diffusion probabilistic models (DDPM) [49], have emerged as effective generative priors for inverse imaging problems. DDPM employs a T-step noising process $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$, which can often be equivalently expressed as $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$ by utilizing the variance-preserving stochastic differential equation (VP-SDE) formulation [50]. Here, $\mathbf{x}_t$ represents the sampled, scaled, and noisy image at step t, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$ is random Gaussian noise, $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$ determines the scaling factors with $\alpha_t = 1 - \beta_t$ and noise schedule $\beta_t \in [0, 1]$ ($\beta_0 = 0$), while $\mathbf{x}_0$ is sampled from the clean image distribution $p(\mathbf{x})$. Using a pre-trained NN $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ that regresses $\min_{\boldsymbol{\Theta}} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\Theta}}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|_2^2$ with learnable parameter set $\boldsymbol{\Theta}$, and given a starting point $\hat{\mathbf{x}}_T$, the denoising diffusion implicit model (DDIM) [51] enables accelerated generation of images via a deterministic sampling strategy $\hat{\mathbf{x}}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_{0|t} + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\boldsymbol{\epsilon}}_t$, where $\hat{\mathbf{x}}_{0|t} = (\hat{\mathbf{x}}_t - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}_t)/\sqrt{\bar{\alpha}_t}$ denotes the current denoised image, and $\hat{\boldsymbol{\epsilon}}_t = \boldsymbol{\epsilon}_{\boldsymbol{\Theta}}(\hat{\mathbf{x}}_t, t)$ is the estimation of added noise.

Recent research [24], [52] regards image reconstruction as the task of measurement-conditioned image generation. Zero-shot diffusion solvers for inverse problems [53] have leveraged the frameworks of singular value decomposition (SVD) [32], manifold constraints [54], posterior sampling [34], range-nullspace decomposition (RND) [39], and pseudoinverse [55] for recovery of different degradation operators. To our knowledge, no diffusion models are specifically designed for natural image CS. While many methods can be applied to CS, their noise estimation U-Nets are pre-trained for single-step noise estimation (see Fig. 1(b)). Our work overcomes this limitation by aligning diffusion sampling with CS targets via end-to-end learning, improving performance and reducing the required number of steps.

C. Invertible Neural Networks for Vision Tasks

Invertible NNs, mathematically invertible functions, are a type of NN that can reconstruct the input from the output. Initially, they were utilized for image generation [56], [57]. Later research found that, unlike non-invertible NNs which cache all intermediate activations for gradient computation, invertible NNs allow most features to be freed up and recomputed as needed, reducing memory requirements [58]. Various invertible architectures have since been proposed, including convolutional, recurrent, and graph NNs [59], [60], [61], as well as Transformers [38], [62]. Their applications extend to image reconstruction [63], [64] and image editing [65], with on-the-fly optimization in the latent space of autoencoders [66]. The memory-efficient nature of invertible NNs is especially advantageous for training large diffusion models end-to-end. In this work, we propose a novel two-level invertible design for end-to-end fine-tuning of a large-scale pre-trained diffusion sampling process at minimal memory cost, making it feasible with limited GPU resources.

III. Method

A. Preliminary

Denoising Diffusion Null Space Model (DDNM), and Its Limitations: DDNM [39] is a state-of-the-art training-free image recovery approach that employs a pre-trained noise estimator $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ as generative prior and solves the CS problem by iterating through the following three DDIM substeps:

$$\hat{\mathbf{x}}_{0|t} = \left[\hat{\mathbf{x}}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}(\hat{\mathbf{x}}_t, t)\right]/\sqrt{\bar{\alpha}_t}, \tag{1}$$

$$\bar{\mathbf{x}}_{0|t} = \mathbf{A}^{\dagger}\mathbf{y} + (\mathbf{I}_N - \mathbf{A}^{\dagger}\mathbf{A})\hat{\mathbf{x}}_{0|t}, \tag{2}$$

$$\hat{\mathbf{x}}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\bar{\mathbf{x}}_{0|t} + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}(\hat{\mathbf{x}}_t, t). \tag{3}$$

In (1), the denoised image is predicted, while (2) applies the RND theory [46], [67] for ensuring measurement consistency $\mathbf{A}\bar{\mathbf{x}}_{0|t} = \mathbf{A}\mathbf{A}^{\dagger}\mathbf{y} + \mathbf{A}(\mathbf{I}_N - \mathbf{A}^{\dagger}\mathbf{A})\hat{\mathbf{x}}_{0|t} \equiv \mathbf{y}$. Finally, (3) performs a deterministic DDIM sampling step, where $\mathbf{A}^{\dagger} \in \mathbb{R}^{N\times M}$ is the pseudo-inverse of CS sampling matrix $\mathbf{A}$. However, the performance of DDNM is limited by a task shift from pre-training $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ on noise estimation in DDPM to its application in CS. The deep features within $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ are not necessarily optimized for mapping $\mathbf{y}$ to $\mathbf{x}$. Similar challenges exist in other diffusion-based methods, including zero-shot solvers [34] and conditional models [24], [52].

B. Learn Diffusion Sampling End-to-End for CS

Diffusion-Based End-to-End CS Learning Framework: We hypothesize that directly fitting the steps of DDNM to the recovery mapping $\mathbf{y} \mapsto \mathbf{x}$ end-to-end can mitigate task shift issues and improve final performance. To achieve this, we construct a DDNM-based CS reconstruction framework, repurposing the sampling process of DDNM as a T-layer NN $\mathcal{F} = \mathcal{F}_1 \circ \mathcal{F}_2 \circ \cdots \mathcal{F}_T$, where each layer $\hat{\mathbf{x}}_{t-1} = \mathcal{F}_t(\hat{\mathbf{x}}_t; \mathbf{y}, \mathbf{A})$ denotes a single sampling step encompassing (1)-(3), with $\circ$ representing the composition of sequential layers. Our IDM framework fine-tunes both the diffusion parameters $\{\alpha_t\}$ and the pre-trained weights in $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$. As Fig. 2 illustrates, it optimizes deep features to minimize the difference between estimation $\hat{\mathbf{x}} = \hat{\mathbf{x}}_0 = \mathcal{F}(\hat{\mathbf{x}}_T; \mathbf{y}, \mathbf{A})$ and the ground truth $\mathbf{x}$ using an L_1 loss $\mathcal{L}(\hat{\mathbf{x}}, \mathbf{x}) = \frac{1}{N}\|\hat{\mathbf{x}} - \mathbf{x}\|_1$ and standard back-propagation [68]. This approach enables comprehensive adaptability to the full CS recovery process, not limited to noise estimation, enhancing performance, and introducing new gains orthogonal to developments in solvers and text prompts [69].

Initialization Strategy of $\hat{\mathbf{x}}_T$ for CS Reconstruction: Unlike typical diffusion models that sample from random noise or employ NNs to estimate an initial result [70], we propose a simple yet effective initialization of image by calculating an expectation $\hat{\mathbf{x}}_T = \mathbb{E}_{\boldsymbol{\epsilon}}[\sqrt{\bar{\alpha}_T}\hat{\mathbf{x}}'_0 + \sqrt{1-\bar{\alpha}_T}\boldsymbol{\epsilon}] = \sqrt{\bar{\alpha}_T}\mathbf{A}^{\dagger}\mathbf{y}$ using back-projection $\hat{\mathbf{x}}'_0 = (\mathbf{A}^{\dagger}\mathbf{y}) \in \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$. This strategy enhances reconstruction quality at low computational cost, outperforming noise-initialized counterparts.

C. Two-Level Invertible Design for Memory Efficiency

Our end-to-end training scheme improves performance but can challenge the memory capacity of GPUs, especially with increased step number T and a large noise estimator $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ due to the substantial footprint of intermediate features cached for back-propagation. For instance, it is infeasible to directly train a diffusion sampling process with $T = 3$ and the SD v1.5 U-Net backbone end-to-end on 4 NVIDIA A100 GPUs with 80 GB memory. To reduce memory consumption, we leverage the memory efficiency of invertible NNs and propose a novel two-level invertible design, making both (1) the multi-step sampling framework and (2) the noise estimation NN invertible through a new "wiring"[1] technique.

First-Level Invertibility for Multi-Step Diffusion Sampling Framework: Inspired by [38], [72], we propose enabling invertibility by introducing new connections into the sampling steps, as shown in Fig. 3. Each connection transmits the input $\hat{\mathbf{x}}_t$ from one step $\mathcal{F}_t$ to the output $\hat{\mathbf{x}}_{t-2}$ of the next step $\mathcal{F}_{t-1}$. We use two learnable weighting scalars u_t and v_t, such that $u_t + v_t = 1$, to scale the output of each step and the transmitted data, respectively, and fuse them to obtain $\hat{\mathbf{x}}_{t-1}$. An auxiliary

[1]Throughout this paper, "wiring" refers to the process of integrating auxiliary connections into a pre-given NN architecture.

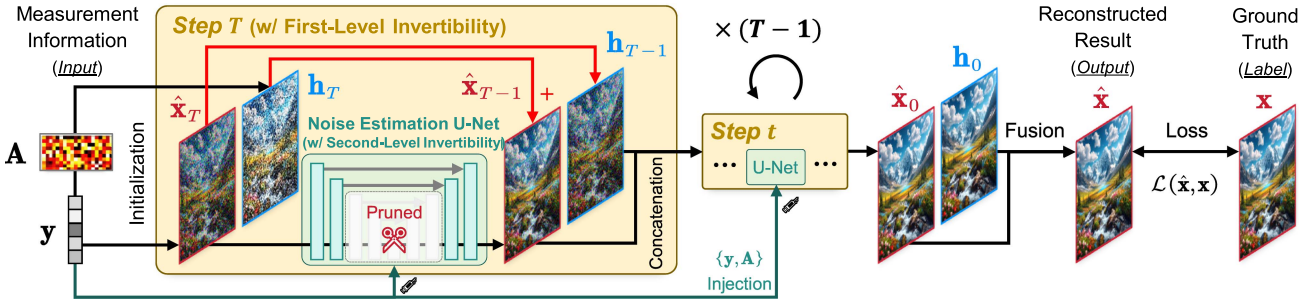

Fig. 2. Illustration of our proposed IDM framework. It receives an initial image estimate $\hat{\mathbf{x}}_T$ and learns T diffusion sampling steps for end-to-end recovery. Auxiliary connections (shown as red arrows) enable invertibility and facilitate the reuse of powerful large-scale pre-trained SD models.

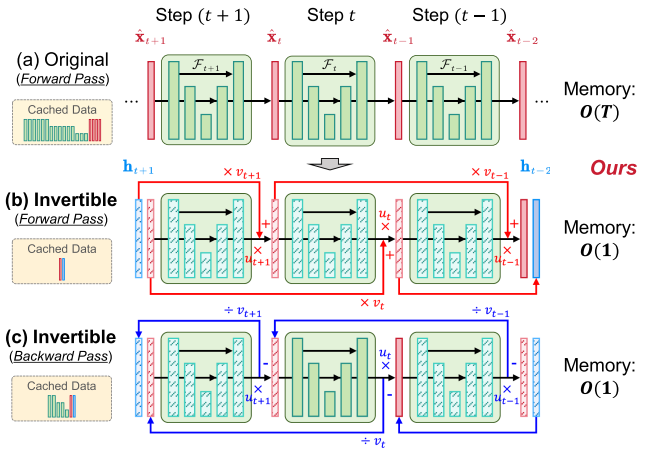

Fig. 3. Illustration of our wiring technique, exemplified with three diffusion sampling steps and a three-scale noise estimation U-Net. Here, light-colored rectangles with diagonal dashed lines represent images/features that are cleared, while dark-colored, empty rectangles indicate images/features that must be preserved. (a) The original non-invertible forward pass caches all inputs, features, and outputs, causing memory usage to increase linearly with the step number. (b) We add connections to construct invertible layers, reducing memory usage to a constant level. (c) During back-propagation, the necessary intermediate inputs/features are sequentially recomputed and cleared to obtain gradients from the last to the first step. Our wired sampling framework is equivalent to the original setup in (a) when $u_t = 1$ and $v_t = 0$.

variable $\mathbf{h}_t$ is introduced to transform each layer into an invertible one, allowing recovery of its input $\{\hat{\mathbf{x}}_t, \mathbf{h}_t\}$ from a given output $\{\hat{\mathbf{x}}_{t-1}, \mathbf{h}_{t-1}\}$ through a second pathway. Mathematically, the forward and inverse computations of each wired layer are formulated as:

$$\text{Forward:} \quad \begin{cases} \hat{\mathbf{x}}_{t-1} = u_t \mathcal{F}_t(\hat{\mathbf{x}}_t; \mathbf{y}, \mathbf{A}) + v_t \mathbf{h}_t, \\ \mathbf{h}_{t-1} = \hat{\mathbf{x}}_t, \end{cases} \tag{4}$$

$$\text{Inverse:} \quad \begin{cases} \hat{\mathbf{x}}_t = \mathbf{h}_{t-1}, \\ \mathbf{h}_t = [\hat{\mathbf{x}}_{t-1} - u_t \mathcal{F}_t(\hat{\mathbf{x}}_t; \mathbf{y}, \mathbf{A})]/v_t. \end{cases} \tag{5}$$

To maintain the dimensional consistency in the framework's input and output after wiring, we introduce two learnable scalars, w_T and w_0, at the beginning and end, computing the input and output as $\mathbf{h}_T = w_T \hat{\mathbf{x}}_T$ and $\hat{\mathbf{x}} = \hat{\mathbf{x}}_0 + w_0 \mathbf{h}_0$, respectively. As illustrated in Fig. 3 (b) and (c), in the forward pass, we cache only the final output of the wired layers, freeing up all intermediate features. After computing the loss function $\mathcal{L}$, we execute standard back-propagation layer-by-layer, from $\{\hat{\mathbf{x}}_0, \mathbf{h}_0\}$ back to $\{\hat{\mathbf{x}}_T, \mathbf{h}_T\}$. For layer t, we jointly recalculate its inputs, necessary features, and parameter gradients based on the outputs $\hat{\mathbf{x}}_{t-1}$ and $\mathbf{h}_{t-1}$, as well as their respective partial derivatives $(\partial\mathcal{L}/\partial\hat{\mathbf{x}}_{t-1})$ and $(\partial\mathcal{L}/\partial\mathbf{h}_{t-1})$ with respect to the loss function $\mathcal{L}$. We implement efficient memory management by clearing all cached intermediate features and outputs as we move through each layer from t back to $(t+1)$.[2]

Consequently, the memory of cached images and features in the framework is reduced from linear complexity $\mathcal{O}(T)$ to constant complexity $\mathcal{O}(1)$ when the sampling substeps and the architecture of noise estimation NN $\boldsymbol{\epsilon}_\Theta$ are pre-determined. The developed wiring technique offers two advantages. First, it can be applied to arbitrary diffusion models and solvers without the need to design new NNs. Second, we reuse the pre-trained weights of $\boldsymbol{\epsilon}_\Theta$ for fine-tuning, with appropriate settings of u_t and v_t. Our approach thereby leads to considerable savings in memory, time, and computation for improving reconstruction performance.

Second-Level Invertibility for Noise Estimator in Each Step: A U-Net architecture [73] is generally utilized as noise estimator $\boldsymbol{\epsilon}_\Theta$ in diffusion solvers. It includes multiple spatial scales with main and skip branches in both its down- and up-sampling blocks. These blocks contain many complex, equidimensional transformations [35] like residual connections and self-attention layers that are memory-intensive. To further improve memory efficiency, we extend our wiring technique to U-Net blocks in each step. As depicted in Fig. 4 (a), we group and wire consecutive transformation blocks within each down-/up-sampling block to make them invertible. During training, we clear all input and intermediate activations from memory for these grouped blocks, while preserving only the features for the first/last convolutions and skip branches for back-propagation. This second-level invertible design further reduces memory usage, making IDM training feasible on standard consumer-grade GPUs.

D. Inject Measurement Physics Into Noise Estimator

Effective utilization of physics information $\{\mathbf{y}, \mathbf{A}\}$ is critical for CS reconstruction. Using it solely to initialize $\hat{\mathbf{x}}_T$ and in the

[2]A code snippet is provided in Section B.3 in **Supplementary Material**.

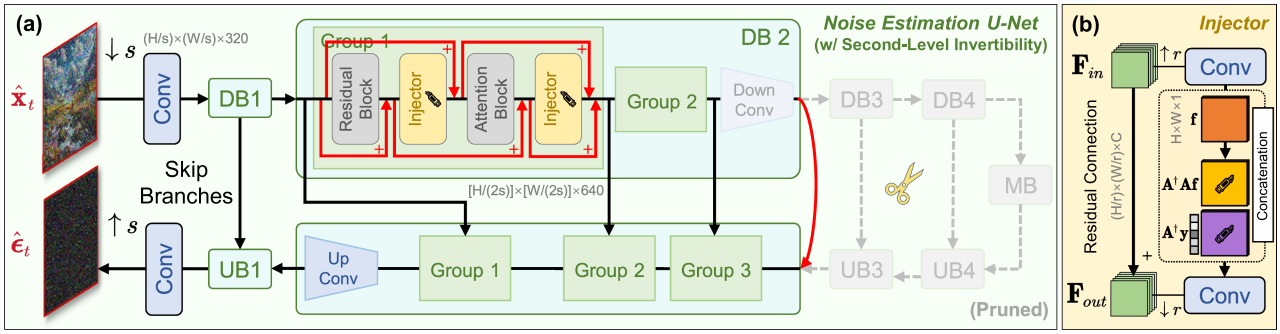

Fig. 4. Illustration of our modified noise estimation U-Net ϵ_Θ for image CS reconstruction tasks, based on the SD v1.5 models [35], [36]. (a) Injectors are added behind each residual and attention block and grouped within downsampling, upsampling, and middle blocks (marked as DB, UB, and MB) for invertibility. Our method is orthogonal to and compatible with network pruning [71] for enhanced efficiency. (b) Each injector learns to fuse measurement physics $\{\mathbf{y}, \mathbf{A}\}$ into deep features using convolutions and residual connections.

RND-based step in (2) limits reconstruction quality in complex scenarios (as shown in Fig. 9 (5) versus (9)), as it is challenging for the NN layers in ϵ_Θ to learn the physics. Therefore, it is necessary to inject the measurement physics directly into the layers. Simply appending the measurement at the input layer of ϵ_Θ in each step, as in [24], can yield unsatisfactory results, as the information of sampling matrix $\mathbf{A}$ and measurement $\mathbf{y}$ is difficult to retain in deep features.

To address this, we propose a series of injectors inspired by the success of physics-informed NNs (see Section II-A). As illustrated in Fig. 4, each injector is positioned behind every grouped block within ϵ_Θ. For an intermediate deep feature $\mathbf{F}_{in} \in \mathbb{R}^{(H/r)\times(W/r)\times C}$ in U-Net, our injector fuses measurement physics information $\{\mathbf{y}, \mathbf{A}\}$ and $\mathbf{F}_{in}$ as follows:

$$\mathbf{f} = \text{Conv}_1\left((\mathbf{F}_{in})_{\uparrow r}\right) \in \mathbb{R}^{H\times W\times 1}\ (N = H \times W), \quad (6)$$

$$\mathbf{F}_{out} = \mathbf{F}_{in} + \left(\text{Conv}_2\left(\left[\mathbf{f}, \mathbf{A}^\dagger\mathbf{A}\mathbf{f}, \mathbf{A}^\dagger\mathbf{y}\right]\right)\right)_{\downarrow r}. \quad (7)$$

Here, Conv_1 and Conv_2 denote two 3×3 convolutions that transition data between feature and image domains, $(\cdot)_{\uparrow/\downarrow r}$ is a Pixel-Shuffle/-Unshuffle layer with scaling ratio r, and $[\mathbf{f}, \mathbf{A}^\dagger\mathbf{A}\mathbf{f}, \mathbf{A}^\dagger\mathbf{y}] \in \mathbb{R}^{H\times W\times 3}$ is a channel-wise concatenation of the intermediate image space data $\mathbf{f}$, its interaction with the CS sampling matrix $\mathbf{A}^\dagger\mathbf{A}\mathbf{f}$, and the back-projected measurement $\mathbf{A}^\dagger\mathbf{y}$. These lightweight injectors are jointly wired and learned within each block group in ϵ_Θ, establishing direct pathways that fuse the measurement information and features, effectively enhancing final recovery quality.

E. Discussion

Our proposed method reinterprets the diffusion framework DDNM for image CS reconstruction, moving beyond the existing paradigm of posterior sampling [34], [89]. In this section, we discuss the relationship between our IDM method and previous diffusion-based approaches that employ posterior sampling, as well as the connections and differences between IDM and algorithm unrolling [48] techniques.

1) Relationship With Previous Diffusion-Based Methods: Traditional diffusion models based on DDPM [49] often perform image generation and reconstruction by modeling the reverse diffusion process through posterior sampling. These models solve stochastic differential equations (SDEs) or ordinary differential equations (ODEs) and gradually denoise a randomly initialized noise to obtain a sample from the desired data distribution, typically requiring a large number of iterative steps (e.g., hundreds or thousands). In contrast, IDM repurposes the diffusion sampling process as an end-to-end deterministic mapping tailored specifically for image CS reconstruction. By fine-tuning a pre-trained diffusion model, we directly learn the mapping from measurement $\mathbf{y}$ to the original image $\mathbf{x}$ using a limited number of sampling steps (e.g., $T = 3$). To be specific, this approach diverges from previous diffusion methods in four key aspects:

First, our IDM method eliminates the randomness in diffusion models by initializing the sampling process with a deterministic estimate, $\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$. This deterministic initialization improves performance, as demonstrated in our ablation studies (see Table V, variant (8) versus (9)).

Second, our approach fine-tunes all parameters of the diffusion model, including the noise estimation network ϵ_Θ, directly on the CS reconstruction task using an L_1 loss. This end-to-end training aligns the model's capacity with CS-specific requirements, optimizing both diffusion sampling process and noise estimator for accurate reconstruction.

Third, by adapting the sampling process of DDNM to CS and leveraging powerful pre-trained SD [36] models, our method achieves high-quality reconstruction with significantly fewer steps (e.g., $T = 3$) compared to hundreds or thousands of steps commonly used in previous methods.

Fourth, due to its deterministic nature and limited diffusion sampling steps, our IDM may not strictly align with the interpretation of solving SDEs or ODEs via posterior sampling. Instead, it functions as an iterative reconstruction model optimized specifically for image CS. Although our method may not fully adhere to the theoretical framework of posterior sampling, we focus on practical gains in performance and efficiency for CS reconstruction. The substantial performance improvements achieved by our method justify this approach, as shown by our experimental results.

TABLE I
COMPARISON OF AVERAGE PSNR (DB, ↑) (± STD) ACROSS VARIOUS DEEP END-TO-END LEARNED CS METHODS ON THE LUMINANCE COMPONENT OF TEST IMAGES

Method	Test Set	Set11			CBSD68		
	CS Ratio γ	10%	30%	50%	10%	30%	50%
ReconNet (CVPR 2016) [18]		24.08 (±2.29)	29.46 (±2.32)	32.76 (±2.19)	23.92 (±3.32)	27.97 (±3.74)	30.79 (±3.77)
ISTA-Net$^+$ (CVPR 2018) [74]		26.49 (±3.15)	33.70 (±3.14)	38.07 (±3.03)	25.14 (±3.97)	30.24 (±4.66)	33.94 (±4.91)
CSNet$^+$ (TIP 2019) [75]		28.34 (±3.19)	34.30 (±3.03)	38.52 (±3.04)	27.04 (±3.84)	31.60 (±4.18)	35.27 (±4.24)
SCSNet (CVPR 2019) [44]		28.52 (±3.15)	34.64 (±3.02)	39.01 (±3.10)	27.14 (±3.88)	31.72 (±4.26)	35.62 (±4.32)
OPINE-Net$^+$ (JSTSP 2020) [76]		29.81 (±3.24)	35.99 (±2.87)	40.19 (±2.90)	27.66 (±4.25)	32.38 (±4.65)	36.21 (±4.74)
DPA-Net (TIP 2020) [77]		27.66 (±3.37)	33.60 (±2.98)	-	25.33 (±4.12)	29.58 (±4.31)	-
MAC-Net (ECCV 2020) [78]		27.92 (±3.05)	33.87 (±3.01)	37.76 (±3.07)	25.70 (±4.16)	30.10 (±4.74)	33.37 (±4.05)
RK-CCSNet (ECCV 2020) [79]		-	-	38.01 (±3.22)	-	-	34.69 (±4.37)
ISTA-Net^{++} (ICME 2021) [80]		28.34 (±3.34)	34.86 (±2.93)	38.73 (±2.92)	26.25 (±4.25)	31.10 (±4.66)	34.85 (±4.68)
COAST (TIP 2021) [81]		30.02 (±3.43)	36.33 (±2.80)	40.33 (±2.88)	27.76 (±4.35)	32.56 (±4.69)	36.34 (±4.74)
AMP-Net (TIP 2021) [82]		29.40 (±3.05)	36.03 (±2.65)	40.34 (±2.85)	27.71 (±4.06)	32.72 (±4.46)	36.72 (±4.38)
MADUN (ACM MM 2021) [83]		29.89 (±3.24)	36.90 (±2.76)	40.75 (±3.05)	28.04 (±4.27)	33.07 (±4.74)	36.99 (±4.72)
DGUNet$^+$ (CVPR 2022) [84]		30.92 (±3.23)	-	41.24 (±3.42)	27.89 (±4.26)	-	36.86 (±4.69)
MR-CCSNet$^+$ (CVPR 2022) [85]		-	-	39.27 (±3.31)	-	-	35.45 (±4.49)
FSOINet (ICASSP 2022) [86]		30.44 (±3.27)	37.00 (±2.72)	41.08 (±3.16)	28.12 (±4.35)	33.15 (±4.74)	37.18 (±4.76)
CASNet (TIP 2022) [87]		30.31 (±3.49)	36.91 (±2.83)	40.93 (±3.30)	28.16 (±4.34)	33.05 (±4.65)	36.99 (±4.74)
TransCS (TIP 2022) [88]		29.54 (±2.92)	35.62 (±2.56)	40.50 (±3.00)	27.76 (±4.18)	32.43 (±4.48)	36.65 (±4.71)
OCTUF$^+$ (CVPR 2023) [33]		30.70 (±3.33)	37.35 (±2.83)	41.36 (±3.23)	28.18 (±4.40)	33.23 (±4.77)	37.28 (±4.84)
CSformer (TIP 2023) [43]		30.66 (±2.96)	-	41.04 (±3.05)	28.11 (±4.17)	-	36.75 (±4.47)
PRL-PGD$^+$ (IJCV 2023) [23]		31.70 (±3.00)	37.89 (±2.91)	41.78 (±3.27)	28.50 (±4.51)	33.41 (±4.79)	37.37 (±4.80)
IDM (Ours, $T = 3$)		**32.92** (±3.34)	**38.85** (±3.28)	**42.48** (±3.28)	**28.69** (±5.19)	**34.67** (±5.57)	**39.57** (±5.85)

Method	Test Set	Urban100			DIV2K		
	CS Ratio γ	10%	30%	50%	10%	30%	50%
ReconNet (CVPR 2016) [18]		20.71 (±3.69)	25.15 (±4.38)	28.15 (±4.61)	24.41 (±3.91)	29.09 (±4.32)	32.15 (±4.32)
ISTA-Net$^+$ (CVPR 2018) [74]		22.81 (±4.85)	29.83 (±6.27)	34.33 (±6.54)	26.30 (±4.83)	32.65 (±5.37)	36.88 (±5.41)
CSNet$^+$ (TIP 2019) [75]		23.96 (±4.24)	29.12 (±5.10)	32.76 (±5.38)	28.23 (±4.82)	33.63 (±5.22)	37.59 (±5.22)
SCSNet (CVPR 2019) [44]		24.22 (±4.36)	29.41 (±5.19)	33.31 (±5.37)	28.41 (±4.84)	33.85 (±5.31)	37.97 (±5.18)
OPINE-Net$^+$ (JSTSP 2020) [76]		25.90 (±5.31)	31.97 (±5.97)	36.28 (±5.95)	29.26 (±5.07)	35.03 (±5.42)	39.27 (±5.34)
DPA-Net (TIP 2020) [77]		24.00 (±4.98)	29.04 (±5.24)	-	27.09 (±4.92)	32.37 (±5.35)	-
MAC-Net (ECCV 2020) [78]		23.71 (±5.18)	29.03 (±5.67)	33.10 (±5.87)	26.72 (±4.83)	32.23 (±5.19)	35.40 (±4.94)
RK-CCSNet (ECCV 2020) [79]		-	-	33.39 (±5.63)	-	-	37.32 (±5.29)
ISTA-Net^{++} (ICME 2021) [80]		24.95 (±5.60)	31.50 (±5.80)	35.58 (±5.59)	27.82 (±5.04)	33.74 (±5.15)	37.78 (±4.94)
COAST (TIP 2021) [81]		26.17 (±5.53)	32.48 (±6.00)	36.56 (±5.95)	29.46 (±5.18)	35.32 (±5.43)	39.43 (±5.30)
AMP-Net (TIP 2021) [82]		25.32 (±5.01)	31.63 (±5.75)	35.91 (±5.71)	29.08 (±5.09)	35.41 (±5.30)	39.46 (±5.15)
MADUN (ACM MM 2021) [83]		26.23 (±5.17)	33.00 (±5.89)	36.69 (±5.81)	29.62 (±4.92)	36.04 (±5.47)	40.06 (±5.33)
DGUNet$^+$ (CVPR 2022) [84]		27.42 (±5.28)	-	37.13 (±5.46)	30.25 (±5.29)	-	40.33 (±5.42)
MR-CCSNet$^+$ (CVPR 2022) [85]		-	-	34.40 (±5.63)	-	-	38.16 (±5.40)
FSOINet (ICASSP 2022) [86]		26.87 (±5.52)	33.29 (±5.83)	37.25 (±5.80)	30.01 (±5.23)	36.03 (±5.52)	40.38 (±5.41)
CASNet (TIP 2022) [87]		26.85 (±5.52)	32.85 (±5.94)	36.94 (±5.99)	30.01 (±5.30)	35.90 (±5.56)	40.14 (±5.45)
TransCS (TIP 2022) [88]		25.82 (±4.87)	31.18 (±5.53)	36.64 (±5.77)	29.09 (±4.79)	34.76 (±5.28)	39.75 (±5.40)
OCTUF$^+$ (CVPR 2023) [33]		27.28 (±5.70)	33.87 (±5.82)	37.82 (±5.75)	30.17 (±5.27)	36.25 (±5.55)	40.61 (±5.45)
CSformer (TIP 2023) [43]		27.13 (±4.91)	-	37.04 (±5.03)	29.96 (±4.84)	-	40.04 (±4.95)
PRL-PGD$^+$ (IJCV 2023) [23]		28.77 (±5.76)	34.73 (±5.73)	38.57 (±5.66)	30.75 (±5.40)	36.64 (±5.60)	40.97 (±5.48)
IDM (Ours, $T = 3$)		**31.41** (±5.89)	**36.76** (±5.53)	**40.33** (±5.62)	**31.07** (±5.91)	**36.98** (±5.64)	**41.15** (±5.40)

Throughout this paper, the best and second-best results of each case are highlighted in bold red and underlined blue, respectively.

2) Relationship With Deep Algorithm Unrolling: Deep algorithm-unrolling NNs [19], [48] transforms iterative optimization algorithms into trainable NN architectures by unrolling a fixed number of iterations into NN layers. Each layer corresponds to one iteration of the original algorithm, and the entire NN is trained end-to-end to enhance performance on specific tasks such as image reconstruction. Our method shares similarities with deep unrolling, as we interpret the iterative steps of the diffusion sampling process as layers of a reconstruction model, which are then fine-tuned end-to-end for the CS task. However, our method differs from algorithm unrolling in the following four aspects:

First, unlike traditional algorithm-unrolled NNs that are trained from scratch with randomly initialized weights, IDM leverages the large-scale pre-trained diffusion model SD v1.5. This enables us to capitalize on the rich representations learned from extensive datasets, providing a strong foundation that enhances reconstruction performance.

Second, traditional unrolling approaches are based on specific optimization algorithms (e.g., ISTA, ADMM) and inherit their convergence properties and interpretability. In contrast, our method is rooted in the diffusion framework, inherently different from conventional optimization algorithms. This shift allows us to explore new architectures and training strategies that are not constrained by the limitations of traditional optimization-based methods.

Third, to address the computational challenges of training large-scale diffusion models end-to-end, we introduce a novel

TABLE II
COMPARISON OF AVERAGE PSNR (DB, ↑) AND SSIM (↑) (± STD) ACROSS EIGHT DIFFUSION-BASED METHODS ON THREE-CHANNEL RGB TEST IMAGES

Method	Test Set	Urban100 (100 RGB images of size 256×256)		
	CS Ratio γ	10%	30%	50%
DDRM (NeurIPS 2022) [32]		19.16 (±2.86)/0.4348 (±0.1185)	28.91 (±3.96)/0.8518 (±0.0717)	33.61 (±3.78)/0.9365 (±0.0342)
ΠGDM (ICLR 2023) [55]		20.09 (±3.67)/0.5089 (±0.1818)	26.70 (±4.60)/0.7925 (±0.1164)	29.75 (±4.01)/0.8724 (±0.0904)
DPS (ICLR 2023) [34]		17.12 (±4.26)/0.3270 (±0.2110)	18.47 (±4.48)/0.3891 (±0.2229)	19.21 (±4.57)/0.4308 (±0.2277)
DDNM (ICLR 2023) [39]		20.76 (±4.51)/0.4682 (±0.1727)	28.76 (±4.98)/0.8284 (±0.0948)	32.86 (±4.87)/0.9164 (±0.0539)
GDP (CVPR 2023) [90]		20.74 (±4.73)/0.5075 (±0.1686)	24.81 (±4.31)/0.7086 (±0.1338)	26.12 (±3.62)/0.7711 (±0.1170)
PSLD (NeurIPS 2023) [89]		19.43 (±5.04)/0.4054 (±0.2311)	22.42 (±4.15)/0.6399 (±0.1859)	22.78 (±3.90)/0.6882 (±0.1656)
SR3 (TPAMI 2023) [24]		18.90 (±3.05)/0.5023 (±0.1467)	21.37 (±3.08)/0.6428 (±0.1200)	23.12 (±2.99)/0.7233 (±0.0981)
IDM (Ours, $T = 3$)		**30.85** (±5.32)/**0.8970** (±0.0855)	**35.78** (±5.05)/**0.9570** (±0.0432)	**39.16** (±5.11)/**0.9771** (±0.0244)
Method	Test Set	DIV2K (100 RGB images of size 256×256)		
	CS Ratio γ	10%	30%	50%
DDRM (NeurIPS 2022) [32]		20.91 (±3.53)/0.4323 (±0.1232)	28.91 (±3.93)/0.7852 (±0.0815)	33.68 (±3.72)/0.9057 (±0.0391)
ΠGDM (ICLR 2023) [55]		22.46 (±3.71)/0.5551 (±0.1515)	28.01 (±3.74)/0.7714 (±0.1058)	30.79 (±3.47)/0.8497 (±0.0797)
DPS (ICLR 2023) [34]		19.47 (±4.36)/0.4222 (±0.1827)	20.78 (±4.35)/0.4652 (±0.1809)	21.37 (±4.30)/0.4884 (±0.1785)
DDNM (ICLR 2023) [39]		22.18 (±4.30)/0.4383 (±0.1435)	28.50 (±4.90)/0.7422 (±0.1110)	32.74 (±4.88)/0.8726 (±0.0629)
GDP (CVPR 2023) [90]		25.17 (±4.44)/0.6334 (±0.1274)	27.75 (±2.85)/0.7664 (±0.0887)	28.47 (±1.53)/0.8029 (±0.0682)
PSLD (NeurIPS 2023) [89]		21.30 (±4.35)/0.4427 (±0.1683)	23.87 (±4.06)/0.6473 (±0.1501)	24.41 (±3.83)/0.7124 (±0.1200)
SR3 (TPAMI 2023) [24]		20.10 (±2.93)/0.4794 (±0.1415)	22.07 (±3.23)/0.5689 (±0.1282)	23.63 (±2.77)/0.6518 (±0.1106)
IDM (Ours, $T = 3$)		**31.22** (±5.43)/**0.8581** (±0.0888)	**36.83** (±5.17)/**0.9482** (±0.0382)	**40.81** (±4.96)/**0.9755** (±0.0200)

TABLE III
COMPARISON OF FID (↓) AND LPIPS (↓) ACROSS EIGHT DIFFUSION-BASED METHODS ON THREE-CHANNEL RGB BENCHMARK IMAGES

Method	Test Set	Urban100 (100 RGB images of size 256×256)			DIV2K (100 RGB images of size 256×256)		
	CS Ratio γ	10%	30%	50%	10%	30%	50%
DDRM (NeurIPS 2022) [53]		184.95/0.4844	50.71/0.1495	21.76/0.0690	264.46/0.5428	93.66/0.2484	39.78/0.1211
ΠGDM (ICLR 2023) [55]		209.43/0.4873	108.90/0.2650	76.93/0.1928	254.26/0.5322	165.39/0.3511	134.47/0.2809
DPS (ICLR 2023) [34]		313.62/0.6876	281.52/0.6238	271.42/0.5838	317.67/0.6797	300.14/0.6355	299.35/0.6090
DDNM (ICLR 2023) [39]		212.84/0.5208	56.60/0.2001	23.74/0.1067	269.37/0.5638	111.64/0.3143	52.42/0.1819
GDP (CVPR 2023) [90]		211.52/0.4636	118.38/0.3005	88.00/0.2383	202.66/0.4142	116.52/0.2894	95.81/0.2448
PSLD (NeurIPS 2023) [89]		263.62/0.5749	157.69/0.3805	128.81/0.3277	258.68/0.5768	191.54/0.4259	164.17/0.3676
SR3 (TPAMI 2023) [24]		210.77/0.4493	144.52/0.3406	103.67/0.2724	255.78/0.5024	193.18/0.4116	160.33/0.3548
IDM (Ours, $T = 3$)		**42.72**/**0.1253**	**15.46**/**0.0566**	**6.47**/**0.0296**	**75.44**/**0.1997**	**27.29**/**0.0886**	**10.69**/**0.0432**

TABLE IV
COMPARISON OF AVERAGE INFERENCE TIME AND STORAGE ACROSS DIFFUSION-BASED METHODS ON AN NVIDIA A100 GPU FOR A 256×256 RGB IMAGE

Method	DDRM [32]	ΠGDM [55]	DPS [34]	DDNM [39]	GDP [90]	PSLD [89]	SR3 [24]	**IDM (Ours)**
Inference Time (s, ↓)	9.28	18.25	19.02	9.16	21.35	233.10	35.62	**0.63**
Model Size (GB, ↓)	2.1	2.1	2.1	2.1	2.1	4.3	**0.4**	**0.4**

TABLE V
ABLATION STUDY ON OUR CONTRIBUTIONS AND DEVELOPED TECHNIQUES FOR ENHANCEMENT ON PERFORMANCE AND EFFICIENCY

	Method	Arch. of ϵ_Θ	E2E	Inv.	Reu.	Inj.	Pru.	Initialization	PSNR (↑)	Mem. (↓)	Tra. t (↓)	NFEs (↓)	Inf. t (↓)	Size (↓)
(1)	DDNM*	SD v1.5	✘	✘	✘	✘	✘	$\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$	22.86/25.13	22.3	417.3	100	42.30	3.4
(2)	DDNM*		✘	✘	✔	✘	✘	$\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$	24.28/26.25	22.3	40.6	100	42.37	3.4
(3)	IDM*		✔	✘	✘	✘	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	26.17/28.09	34.6	411.1	2	1.72	3.4
(4)	IDM*		✔	✔	✘	✘	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	26.21/28.14	13.4	820.7	2	1.69	3.4
(5)	IDM*		✔	✘	✔	✘	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	26.57/28.33	34.6	12.4	2	1.69	3.4
(6)	IDM*		✔	✔	✔	✘	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	26.52/28.31	13.4	24.5	2	1.68	3.4
(7)	IDM*		✔	✔	✔	✔	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	**29.53**/**30.41**	13.4	26.4	2	1.77	3.4
(8)	IDM*		✔	✔	✔	✔	✔	$\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$	27.21/29.12	5.0	8.3	2	0.38	**0.4**
(9)	**IDM (Ours)**		✔	✔	✔	✔	✔	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	29.44/30.37	**4.9**	8.0	2	0.38	**0.4**
(10)	DDNM	UIDM	✘	✘	✔	✘	✘	$\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$	20.76/22.18	N/A	N/A	100	9.16	2.1
(11)	IDM*		✔	✔	✔	✘	✘	$\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$	24.45/26.52	18.4	**4.2**	2	**0.33**	2.1
(12)	PRL-PGD$^+$	N/A	✔	✘	✘	✘	✘	$\hat{\mathbf{X}}^{(0)} = \text{Conv}(\mathbf{A}^\dagger\mathbf{y})$	26.74/28.32	8.6	652.4	N/A	0.95	0.8

Arch.: Architecture choice of the noise estimator. E2E: End-to-end training. Inv.: Invertibility. Reu.: Reuse of pre-trained NN weights. Inj.: Injectors. Pru.: Pruning. PSNR: Average PSNR (dB) for RGB image CS on Urban100/DIV2K at $\gamma = 10\%$. Mem.: GPU memory usage (GB). Tra. t: Training time (hours) to convergence. NFEs: Neural function evaluations. Inf. t: Inference time per image (seconds). Size: Storage size (GB). X*: Modified version of method "X". UIDM: Unconditional image diffusion model [92]. N/A: Not applicable.

two-level invertible design that reduces memory consumption and enables model's efficient training on standard GPUs. This innovation is specific to our approach and is not commonly found in traditional deep unrolling methods.

Fourthly, by reusing and fine-tuning the pre-trained SD models, our method can be quickly adapted to different inverse problems like image CS, accelerated MRI, and sparse-view CT with minimal training effort. This scalability is advantageous compared to unrolled NNs, which often require extensive re-training or redesign when applied to new tasks.

Overall, while IDM shares the iterative nature of deep unrolling, the use of diffusion models and their benefits distinguish our method. We bridge the gap between diffusion-based generative modeling and inverse problem-solving in a new way, showing that diffusion models can be effectively adapted and fine-tuned for high-quality CS reconstruction.

IV. Experiment[3]

A. Implementation Details

Architectural Customization: Moving beyond the original DDNM [39] approach based on an unconditional image diffusion model[4] [91] pre-trained on the ImageNet dataset [92], our method reuses the noise estimation NN of SD v1.5[5] pre-trained on the LAION dataset [93], yielding improved performance after fine-tuning (see Table V (6) versus (11)). We adapt this noise estimator to IDM by introducing PixelUnshuffle and PixelShuffle [94] layers with scaling ratio $s = 2$ at the first and last convolutions to match the four-channel data format of SD. As illustrated in Fig. 4 (a), we further simplify the U-Net by removing its time embeddings, cross-attention layers,[6] and the final three scales to balance efficiency and performance. Notably, this customized framework differs from other zero-shot, plug-and-play, or conditional models, as it allows rapid end-to-end fine-tuning of all reused weights for CS, which is advantageous for applications where the prior of the modified, pre-trained diffusion model does not fully align with the target task.

All Learnable Parameters in IDM are jointly fine-tuned end-to-end, including the shared, wired U-Net $\boldsymbol{\epsilon}_{\boldsymbol{\Theta}}$ and its internal weighting factors (i.e., v_is with $u_i \equiv 1 - v_i$) across the T sampling steps, the non-shared and step-specific diffusion parameters $\{\alpha_t\}$, weighting scalars $\{v_t\}$ with $u_t \equiv 1 - v_t$ for our invertible diffusion sampling pipeline, the scaling factors w_T and w_0, as well as the parameters of our designed $(20T)$ injectors. The values of scalars α_t, v_t, v_i, w_T, and w_0 are initialized to 0.5, 0.5, 0.5, 1.0, and 0.0, respectively. All experiments employ the Adam [95] optimizer for training.

B. Comparison With State-of-the-Arts

Setup: IDM is compared against twenty end-to-end learned and eight diffusion-based approaches for natural image CS tasks.

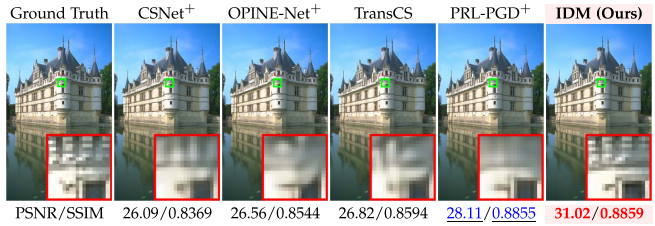

Fig. 5. Comparison of CS recovery results among various end-to-end learned methods on "test_03" image from CBSD68 at $\gamma = 10\%$.

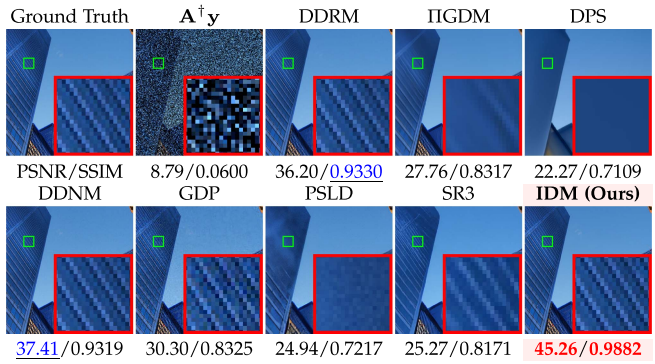

Fig. 6. Comparison of CS recovery results among various diffusion-based methods on "img_012" image from Urban100 at $\gamma = 50\%$.

Training data pairs $\{(\mathbf{y}, \mathbf{x})\}$ are generated from the randomly cropped 256×256 patches of the Waterloo exploration database (WED) [20], [23], [96], [97] using structurally random, orthonormalized i.i.d. block-based Gaussian matrices $\mathbf{A}$ of a fixed block size 32×32 [98], [99], [100], [101], [102]. Fine-tuning IDM with $T = 3$ and a batch size of 32 for 50000 iterations on 4 NVIDIA A100 (80 GB) GPUs and PyTorch [103] takes three days, with a starting learning rate of 0.0001, halved every 10000 iterations. Four benchmarks: Set11 [18], CBSD68 [104], and 256×256 center-cropped images from Urban100 [105] and DIV2K [106] are employed. For RGB images, we compressively sample each R, G, and B channel separately, and multiply the input and output channel numbers of the first and last convolutions in U-Net and injectors by 3. Peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [107] are employed as metrics. Results of existing methods are sourced from original publications, or obtained via careful hyperparameter tuning and re-training. In cases where replicable algorithms are not available, we indicate this in our tables using a "-" symbol. Especially, for SR3, we adopt $\mathbf{A}^\dagger\mathbf{y}$ as its conditional input.

Results: Table I shows that our IDM outperforms other end-to-end learned CS NNs, notably exceeding PRL-PGD$^+$ by average PSNR margins of 0.96 dB, 1.22 dB, 2.14 dB, and 0.48 dB on the four sets. Considering the test sample numbers, our IDM's PSNR improvements on CBSD68 and Urban100 are generally statistically significant, while the gains on Set11 and DIV2K are less statistically significant. As depicted in Fig. 5, IDM recovers accurate and high-fidelity details, particularly on building edges and windows. In contrast, four other competing CS NNs tend to produce oversmoothed and blurry building textures. IDM effectively reduces artifacts and achieves the highest PSNR,

[3]Please Refer to Section C in the **Supplementary Material** for Our More Experimental Results and Analyses.

[4]https://github.com/openai/guided-diffusion

[5]https://huggingface.co/stable-diffusion-v1-5/stable-diffusion-v1-5

[6]In this study, for diffusion models using the SD U-Net architecture that retain text input, embedding, and cross-attention layers, a null text prompt (i.e., "") is used as the default condition. For IDM, which removes these components entirely, no text input is applied.

Fig. 7. Comparison of CS reconstruction results on two images named "barbara" **(top)** and "Parrots" **(bottom)** from Set11 at CS ratio $\gamma = 10\%$.

surpassing the second-best end-to-end image CS approach PRL-PGD$^+$ by 2.91 dB. The superiority of IDM is further visually shown in Fig. 7. Competing methods can either oversmooth details or produce incorrect patterns with noticeable artifacts. In contrast, IDM achieves accurate and vivid details on the garment, and sharp, artifact-free strips around parrot's eye.

Table II shows the advantage of IDM in PSNR and SSIM, outperforming the second-best contenders by a notable average margin of 7.27 dB and 0.1652. Such an improvement is statistically significant and can be attributed to the ability of our method to address the shortcomings of generative priors in maintaining data fidelity and the insufficient adaptation to CS tasks of their noise estimation networks. Table III compares the perceptual quality of images from diffusion-based approaches, using Fréchet inception distance (FID) [108] and the learned perceptual image patch similarity (LPIPS) [109]. Our method achieves significant superiority, with 15-127 decreases in FID and 0.04-0.22 reductions in LPIPS. As depicted in Fig. 6, while DDNM and GDP can synthesize high-quality patterns in the areas of building windows, they also generate noise-like artifacts in the regions of sky and windows. Although ΠGDM, DPS, PSLR, and SR3 reconstruct the basic shapes and structures of the original image, they struggle to recover detailed features of the building facade, which can be observed in the zoomed-in area. As Fig. 8 shows, Π GDM, DPS, PSLD, and SR3 often result in oversmoothed and blurry outputs. DDNM and GDP can introduce noise-like artifacts despite capturing basic structures. Our IDM reliably produces high-quality and precise reconstructions of the intricate patterns of lines. Notably, SR3 requires 240 hours and 75 GB of memory per GPU for training. Our IDM stands out by necessitating only 76 hours and 15 GB memory per GPU, delivering high-quality, artifact-free images using merely 3 NFEs for the noise estimator—a stark contrast to the $\geq$100 NFEs required by competing methods. This results in an inference speed-up of approximately 15-370 times, as reported in Table IV. These observations validate the effectiveness of our end-to-end fine-tuning framework for diffusion sampling learning, and wiring technique for two-level invertibility and reuse.

C. Ablation Study and Analysis

Setup: We evaluate scaled-down IDM variants trained on an NVIDIA A100 GPU with $T = 2$, a batch size of 4, and a patch size of 128in this section. Results are in Table V and Figs. 9–11. Two SOTAs: DDNM [39] and PRL-PGD$^+$ [23] re-trained for RGB image CS, are included as references in (10) and (12). We also train two measurement-conditioned SD-based DDNM variants in (1) and (2) [24], [35].

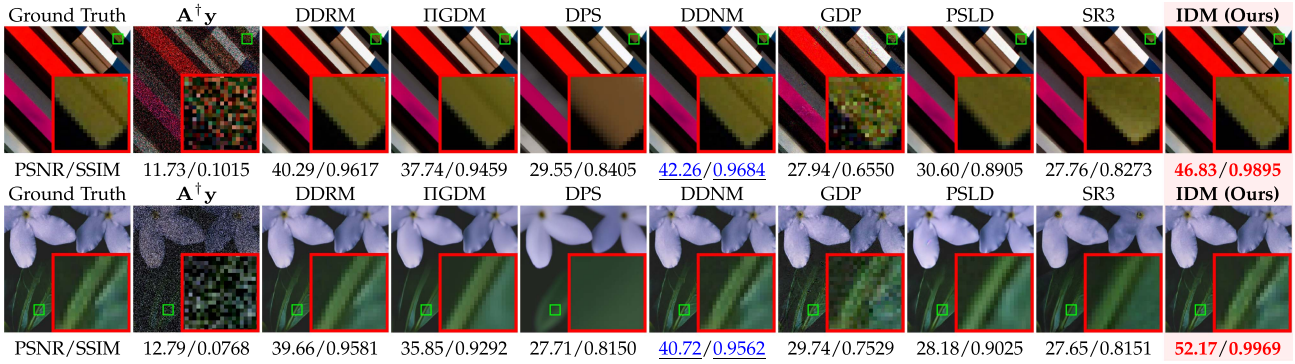

Fig. 8. Comparison of CS recovery results on two images named "img_081" from Urban100 **(top)** and "0877" from DIV2K **(bottom)** at $\gamma = 50\%$.

TABLE VI
EXTENSION OF OUR INVERTIBLE DESIGN AND INJECTOR MODULES TO TWO END-TO-END CS NNS: RK-CCSNET AND CSNET+ ON SET11 AT $\gamma = 50\%$

Method	PSNR (dB, ↑)	SSIM (↑)	Memory (GB, ↓)	Parameter Number (M, ↓)
RK-CCSNet (ECCV 2020) [79]	38.03	0.9731	1.3	**0.631**
RK-CCSNet (w/ Invertible Design)	38.07	0.9735	**0.6**	**0.631**
RK-CCSNet (w/ Injectors)	38.91	0.9762	1.5	0.633
RK-CCSNet (w/ Both Invertible Design and Injectors)	**38.95**	**0.9763**	**0.6**	0.633

Method	PSNR (dB, ↑)	SSIM (↑)	Memory (GB, ↓)	Parameter Number (M, ↓)
CSNet$^+$ (TIP 2019) [75]	38.53	0.9750	3.7	**1.419**
CSNet$^+$ (w/ Invertible Design)	38.55	0.9750	**2.2**	**1.419**
CSNet$^+$ (w/ Injectors)	39.65	0.9802	4.3	1.421
CSNet$^+$ (w/ Both Invertible Design and Injectors)	**39.77**	**0.9806**	**2.2**	1.421

TABLE VII
AVERAGE PSNR (DB, ↑) OF IDM AND STATE-OF-THE-ART CS NN PRL-PGD$^+$ TO DIFFERENT CS RATIOS ON SET11

Method	$\gamma = 30\%$ (seen)	$\gamma = 50\%$ (seen)	$\gamma = 50\%$ (unseen)
PRL-PGD$^+$ (IJCV 2023) [23]	37.89 (specific to 30%)	41.78 (specific to 50%)	28.85 (specific to 30%, applied to unseen 50%)
IDM (Ours)	**38.85** (specific to 30%)	**42.48** (specific to 50%)	**41.60** (specific to 30%, applied to unseen 50%)

The results correspond to Table I.

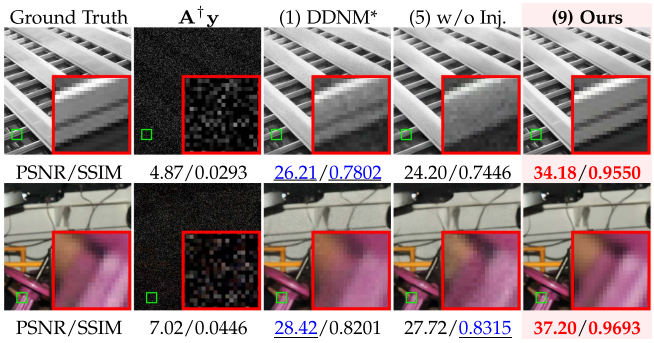

Fig. 9. Comparison of CS recovery results from the method variants of (1), (5), and (9) in Table V on the two images named "img_042" and "0888" from Urban100 **(top)** and DIV2K **(bottom)** at CS ratio $\gamma = 10\%$.

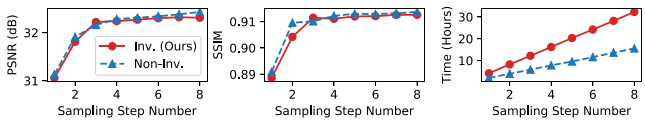

Fig. 10. Comparison of different IDM variants with $1 \leq T \leq 8$. We evaluate PSNR (↑) and SSIM (↑) for the luminance component image CS on Set11 at $\gamma = 10\%$ **(left & middle)** and training time cost (↓) **(right)**. Parameter number of model is calculated as $(115.254 + 0.006T)$M.

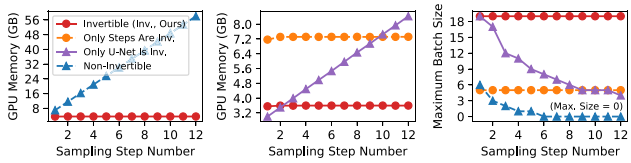

Fig. 11. Comparison of different IDM variants with $1 \leq T \leq 12$. We evaluate memory use (↓) on A100 (80 GB) GPU **(left)**, and maximum batch size (↑) for training on 1080Ti (11 GB) GPU **(right)**. Notably, our IDM reduces memory requirement by 93.8% (≈15/16) at $T = 12$.

Effect of End-to-End DDNM Sampling Learning: Comparisons (1) versus (3), (2) versus (5), and (10) versus (11) in Table V exhibit that our end-to-end DDNM sampling learning provides substantial PSNR improvements of 2.06-4.34 dB. It also decreases the number of required steps by 98% and boosts inference speed by 27.55-29.50 times. Notably, the method variant (11) needs only 4.2 hours of fine-tuning, validating the efficiency and rapid deployment capability of our IDM.

Effect of Two-Level Invertible Network Design, and Reuse of Pretrained Model Weights: Comparisons of (3) versus (4) and (5) versus (6) in Table V show that our wired invertible models achieve a 61.3% reduction in memory usage while effectively maintaining recovery performance. Fig. 11 demonstrates that our two-level wiring scheme consistently keeps memory usage within 3-4 GB and scales the maximum batch size by a factor of 3, significantly improving overall training efficiency on modern GPUs [62].

As illustrated in Fig. 10 (right), our enabled invertibility (Inv.) can maintain competitive reconstruction quality but doubles training time due to recomputation during back-propagation. However, we mitigate this by reusing pre-trained models. Comparisons of (1) versus (2), (3) versus (5), and (4) versus (6) in Table V reveal that weight reuse yields 0.17-1.42 dB PSNR improvements and reduces training time by 90.3-97.0%, enabling us to train an unpruned IDM within one day. These results underscore the critical importance of our "wiring + reuse" strategy for recovery. Furthermore, as evidenced by (6) versus (11), reusing advanced models such as SD v1.5 further enhances CS performance.

In particular, we emphasize that scaling up our model back to the default settings of $T = 3$, a batch size of 32, and a patch size of 256 makes fine-tuning a non-invertible IDM variant of (9) even impossible on 4 A100 (80 GB) GPUs due to its excessive memory requirement. We also find that the maximum batch size on such a configuration is actually reduced from 184 (Inv.) to 20 (Non-Inv.), consuming a peak memory of 70.08 GB per GPU and resulting in a decrease of 0.3–0.5 dB in final PSNR. Notably, with our two-level invertible models, the peak memory use per GPU is suppressed to 14.64 GB, making it manageable not only for A100 but also for other GPUs like 3090 and 4090 (24 GB). This again verifies the effectiveness of our invertible design.

Effect of Different Step Numbers T: Fig. 10 shows the relationship between T, reconstruction quality (in PSNR and SSIM), and computational cost (in time). As observed, PSNR and SSIM improve significantly when $T \leq 3$, with a rapid increase in reconstruction quality. However, when $T > 3$, the improvements taper off, and the curves plateau, indicating diminishing returns in reconstruction performance. Meanwhile, the computational cost increases linearly with T, highlighting a trade-off between quality and efficiency. Based on these observations, we strategically select $T = 3$ as the default value, as it achieves a near-optimal balance between image recovery quality and inference speed. This choice ensures the practicality of our method for real-world applications, where both accuracy and efficiency are critical.

Effect of Physics Integration via Injectors: Comparison (6) versus (7) in Table V reveals that injectors improve PSNR by 2.10-3.01 dB, requiring only an extra 1.9 hours of training and marginal $(0.005T)\%$ or $(0.006T)$M increase in parameters. Notably, our two-level wiring integrates these injectors seamlessly into the U-Net blocks, making them invertible with minimal extra memory usage. Fig. 9 demonstrates that our method (9) minimizes artifacts and decently recovers intricate textures of lines and corners, outperforming (5).

Effect of $\hat{\mathbf{x}}_T$ Initialization and Pruning Schemes: Comparison of (8) versus (9) in Table V confirms the superiority of our initialization $\hat{\mathbf{x}}_T = \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{y}$ with a PSNR improvement of 1.74 dB. Furthermore, we observe that the scheme does not significantly enhance the performance of DDNM in (1), (2), and (10), as there is no notable PSNR gain. Conversely, changing the initialization of (3) back to $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$ results in reduced PSNR 25.01/27.16 with a drop of 1.05 dB. This suggests that the initialization particularly benefits end-to-end IDM learning when using a very limited number of steps. Additionally, the comparison of (7) versus (9) indicates that NN pruning [71] effectively complements our method by reducing 0.7 billion (B) NN parameters and 3.0 GB storage size, with a minor 0.07 dB decrease in PSNR.

Summary: As a progression (1)$\rightarrow$ (3) $\rightarrow$ (4) $\rightarrow$ (6) $\rightarrow$ (7) $\rightarrow$ (9) shows, our method significantly enhances and extends previous diffusion-based image reconstruction paradigm for CS. Comparison (1) versus (9) exhibits that our IDM delivers a 5.91 dB PSNR gain, reduces training time by 98%, accelerates inference by 110 times, and cuts storage size by 88%. These benefits are achieved with an acceptable investment of 8.0 hours fine-tuning and 4.9 GB memory use. Compared to the SOTA CS NN (12), our approach (9) offers advantages in accuracy, memory and time efficiencies, and model size.

D. Extension to Other CS Networks

To further validate the generalizability of our approach, we apply our (1) invertible NN design and (2) injectors to the two well-established models: RK-CCSNet [79] and CSNet$^+$ [75]. Using the original training setting for each network and 400 images from BSDS500 [110], we evaluate their performance on Set11 at a sampling rate of 50%. Quantitative results are summarized in Table VI. For RK-CCSNet, our invertible design reduces training memory by 53.8% (from 1.3 GB to 0.6 GB) without changing parameter number, while the injectors improve PSNR by 0.88 dB and SSIM by 0.0031. Applying both two components jointly achieves the best results with a PSNR of 38.95 dB and SSIM of 0.9763. Similarly, for CSNet$^+$, our invertible design effectively reduces memory usage by 40.5%, and the injectors improve PSNR by 1.12 dB and SSIM by 0.0052. Combining both yields a PSNR of 39.77 dB and SSIM of 0.9806, demonstrating the effectiveness of our method across different CS NNs.

E. Generalization to Different CS Ratios

We evaluate the generalization ability of IDM to different CS ratios. While our current IDM implementation follows the common practice in the community of natural image CS—training a separate model for each specific task—we examine its performance when applied to CS ratios unseen during training. Specifically, we train an IDM on a 30% CS ratio (seen) and evaluate its performance on a 50% CS ratio (unseen). The results are compared against PRL-PGD$^+$ [23], the previous best-performing end-to-end learned CS NN. For completeness, we also include the results of both methods trained and tested

on the same seen ratios. As shown in Table VII, IDM achieves a PSNR of 41.60 dB when a model trained at $\gamma = 30\%$ is applied to $\gamma = 50\%$. While this performance is slightly lower than that of IDM trained specifically for $\gamma = 50\%$ (42.48 dB), it is significantly higher than the performance of PRL-PGD$^+$ on the same unseen task (28.85 dB), showing a remarkable 12.75 dB improvement. This observation demonstrates that IDM retains a strong degree of generalization ability to unseen CS ratios, which is particularly advantageous for real-world scenarios where training for all possible ratios may not be feasible.

V. Conclusion

We propose Invertible Diffusion Model (IDM), a novel efficient, end-to-end diffusion-based image CS method, which converts a large-scale, pre-trained diffusion sampling process into a two-level invertible framework for end-to-end reconstruction learning. Our method provides three advantages. First, it directly learns all diffusion model parameters utilizing the CS reconstruction objective, unlocking the full potential of diffusion networks in the recovery problem. Second, it improves the memory efficiency by making both (1) sampling steps and (2) noise estimation U-Net invertible. Third, it reuses pre-trained diffusion models to minimize fine-tuning effort. Additionally, our introduced lightweight injectors further facilitate reconstruction performance. Experiments validate that IDM outperforms existing CS NNs and diffusion-based inverse problem solvers, achieving new state-of-the-art performance with only three sampling steps.

This work offers new possibilities for enhancing image CS and general image reconstruction using large diffusion models, particularly in GPU resource-limited scenarios. Future work includes extending IDM to real CS systems, for example, fluorescence microscopy [111], [112] and interferometric imaging [113], as well as exploring non-linear compression techniques like JPEG2000 and deep autoencoders.

Limitations: While we have not found explicit evidence of overlap between our adopted test sets (Set11, CBSD68, Urban100, DIV2K) and the LAION dataset used for training the SD v1.5 model, we can not definitively rule out the possibility of overlap due to the sheer scale of the LAION dataset. This remains a potential limitation of our experiments.

References

[1] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[2] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution," *Commun. Pure Appl. Math., A J. Issued Courant Inst. Math. Sci.*, vol. 59, no. 6, pp. 797–829, 2006.

[3] C. E. Shannon, "Communication in the presence of noise," *Proc. Inst. Radio Engineers*, vol. 37, no. 1, pp. 10–21, 1949.

[4] M. F. Duarte et al., "Single-pixel imaging via compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 83–91, Mar. 2008.

[5] M. Lustig, D. L. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magn. Reson. Med.*, vol. 58, no. 6, pp. 1182–1195, 2007.

[6] G.-H. Chen, J. Tang, and S. Leng, "Prior image constrained compressed sensing (PICCS): A method to accurately reconstruct dynamic CT images from highly undersampled projection data sets," *Med. Phys.*, vol. 35, no. 2, pp. 660–663, 2008.

[7] T. P. Szczykutowicz and G.-H. Chen, "Dual energy CT using slow kVp switching acquisition and prior image constrained compressed sensing," *Phys. Med. Biol.*, vol. 55, no. 21, 2010, Art. no. 6411.

[8] X. Yuan, D. J. Brady, and A. K. Katsaggelos, "Snapshot compressive imaging: Theory, algorithms, and applications," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 65–88, Mar. 2021.

[9] Y. Fu, T. Zhang, L. Wang, and H. Huang, "Coded hyperspectral image reconstruction using deep external and internal learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3404–3420, Jul. 2022.

[10] J. Suo, W. Zhang, J. Gong, X. Yuan, D. J. Brady, and Q. Dai, "Computational imaging and artificial intelligence: The next revolution of mobile vision," *Proc. IEEE*, vol. 111, no. 12, pp. 1607–1639, Dec. 2023.

[11] Y. Cai et al., "Mask-guided spectral-wise transformer for efficient hyperspectral image reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 17502–17511.

[12] Y. Cai et al., "Coarse-to-fine sparse transformer for hyperspectral image reconstruction," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 686–704.

[13] Y. Cai et al., "Degradation-aware unfolding half-shuffle transformer for spectral compressive imaging," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 37749–37761.

[14] J. Zhang, D. Zhao, and W. Gao, "Group-based sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3336–3351, Aug. 2014.

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

[16] C. Zhao, S. Ma, J. Zhang, R. Xiong, and W. Gao, "Video compressive sensing reconstruction via reweighted residual sparsity," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 6, pp. 1182–1195, Jun. 2017.

[17] A. Mousavi, A. B. Patel, and R. G. Baraniuk, "A deep learning approach to structured signal recovery," in *Proc. IEEE Allerton Conf. Commun. Control Comput.*, 2015, pp. 1336–1343.

[18] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok, "ReconNet: Non-iterative reconstruction of images from compressively sensed measurements," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 449–458.

[19] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, Mar. 2021.

[20] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6360–6376, Oct. 2022.

[21] U. S. Kamilov, C. A. Bouman, G. T. Buzzard, and B. Wohlberg, "Plug-and-play methods for integrating physical and learned models in computational imaging: Theory, algorithms, and applications," *IEEE Signal Process. Mag.*, vol. 40, no. 1, pp. 85–97, Jan. 2023.

[22] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imag. Sci.*, vol. 10, no. 4, pp. 1804–1844, 2017.

[23] B. Chen, J. Song, J. Xie, and J. Zhang, "Deep physics-guided unrolling generalization for compressed sensing," *Int. J. Comput. Vis.*, vol. 131, no. 11, pp. 2864–2887, 2023.

[24] C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi, "Image super-resolution via iterative refinement," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 4713–4726, Apr. 2023.

[25] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8162–8171.

[26] Y. Zhu et al., "Denoising diffusion models for plug-and-play image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2023, pp. 1219–1229.

[27] J. Wang, Z. Yue, S. Zhou, K. C. Chan, and C. C. Loy, "Exploiting diffusion prior for real-world image super-resolution," *Int. J. Comput. Vis.*, vol. 132, pp. 5929–5949, 2024.

[28] G. Li et al., "Self-reference image super-resolution via pre-trained diffusion large model and window adjustable transformer," in *Proc. 31st ACM Int. Conf. Multimedia*, 2023, pp. 7981–7992.

[29] X. Lin et al. "DiffBIR: Towards blind image restoration with generative diffusion prior," 2023, arXiv: *2308.15070*.

[30] R. Wu, T. Yang, L. Sun, Z. Zhang, S. Li, and L. Zhang, "SeeSR: Towards semantics-aware real-world image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 25456–25467.

[31] K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1671–1681.

[32] B. Kawar, M. Elad, S. Ermon, and J. Song, "Denoising diffusion restoration models," in *Proc. Neural Inf. Process. Syst.*, 2022, pp. 23593–23606.

[33] J. Song, C. Mou, S. Wang, S. Ma, and J. Zhang, "Optimization-inspired cross-attention transformer for compressive sensing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 6174–6184.

[34] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion posterior sampling for general noisy inverse problems," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–30.

[35] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10674–10685.

[36] Stability.AI, 2024. [Online]. Available: https://stability.ai

[37] B. Xia et al., "DiffIR: Efficient diffusion model for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 13095–13105.

[38] C. Zhao, S. Liu, K. Mangalam, and B. Ghanem, "Re^{2}TAL: Rewiring pretrained video backbones for reversible temporal action localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 10637–10647.

[39] Y. Wang, J. Yu, and J. Zhang, "Zero-shot image restoration using denoising diffusion null-space model," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–31.

[40] M. Iliadis, L. Spinoulas, and A. K. Katsaggelos, "Deep fully-connected networks for video compressive sensing," *Digit. Signal Process.*, vol. 72, pp. 9–18, 2018.

[41] A. Vaswani et al., "Attention is all you need," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[43] D. Ye, Z. Ni, H. Wang, J. Zhang, S. Wang, and S. Kwong, "CSformer: Bridging convolution and transformer for compressive sensing," *IEEE Trans. Image Process.*, vol. 32, pp. 2827–2842, 2023.

[44] W. Shi, F. Jiang, S. Liu, and D. Zhao, "Scalable convolutional neural network for image compressed sensing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12290–12299.

[45] W. Cui, S. Liu, F. Jiang, and D. Zhao, "Image compressed sensing using non-local neural network," *IEEE Trans. Multimedia*, vol. 25, pp. 816–830, 2023.

[46] D. Chen and M. E. Davies, "Deep decomposition learning for inverse imaging problems," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 510–526.

[47] B. Chen and J. Zhang, "Practical compact deep compressed sensing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 3, pp. 1610–1626, 2025, doi: 10.1109/TPAMI.2024.3504490.

[48] J. Zhang, B. Chen, R. Xiong, and Y. Zhang, "Physics-inspired compressive sensing: Beyond deep unrolling," *IEEE Signal Process. Mag.*, vol. 40, no. 1, pp. 58–72, Jan. 2023.

[49] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 6840–6851.

[50] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–36.

[51] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–20.

[52] C. Saharia et al., "Palette: Image-to-image diffusion models," in *Proc. ACM Special Int. Group Comput. Graph. Interactive Techn. Conf.*, 2022, pp. 1–10.

[53] A. Graikos, N. Malkin, N. Jojic, and D. Samaras, "Diffusion models as plug-and-play priors," in *Proc. Neural Inf. Process. Syst.*, 2022, pp. 14715–14728.

[54] H. Chung, B. Sim, D. Ryu, and J. C. Ye, "Improving diffusion models for inverse problems using manifold constraints," in *Proc. Neural Inf. Process. Syst.*, 2022, pp. 25683–25696.

[55] J. Song, A. Vahdat, M. Mardani, and J. Kautz, "Pseudoinverse-guided diffusion models for inverse problems," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–30.

[56] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 10236–10245.

[57] J. Ho, X. Chen, A. Srinivas, Y. Duan, and P. Abbeel, "Flow++: Improving flow-based generative models with variational dequantization and architecture de?sign," in *Proc. Int. Conf. on Mach. Learn.*, 2019, pp. 2722–2730.

[58] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse, "The reversible residual network: Backpropagation without storing activations," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 2211–2221.

[59] Y. Song, C. Meng, and S. Ermon, "MintNet: Building invertible neural networks with masked convolutions," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 11004–11014.

[60] M. MacKay, P. Vicol, J. Ba, and R. B. Grosse, "Reversible recurrent neural networks," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 9043–9054.

[61] G. Li, M. Müller, B. Ghanem, and V. Koltun, "Training graph neural networks with 1000 layers," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6437–6449.

[62] K. Mangalam et al., "Reversible vision transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10830–10840.

[63] Y. Liu et al., "Invertible denoising network: A light solution for real noise removal," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13365–13374.

[64] J.-J. Huang and P. L. Dragotti, "WINNet: Wavelet-inspired invertible network for image denoising," *IEEE Trans. Image Process.*, vol. 31, pp. 4377–4392, 2022.

[65] B. Wallace, A. Gokul, and N. Naik, "EDICT: Exact diffusion inversion via coupled transformations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 22532–22541.

[66] B. Wallace, A. Gokul, S. Ermon, and N. Naik, "End-to-end diffusion latent optimization improves classifier guidance," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 7280–7290.

[67] J. Schwab, S. Antholzer, and M. Haltmeier, "Deep null space learning for inverse problems: Convergence analysis and rates," *Inverse Problems*, vol. 35, no. 2, 2019, Art. no. 025008.

[68] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[69] H. Chung, J. C. Ye, P. Milanfar, and M. Delbracio, "Prompt-tuning latent diffusion models for inverse problems," 2023, arXiv: *2310.01110*.

[70] J. Whang, M. Delbracio, H. Talebi, C. Saharia, A. G. Dimakis, and P. Milanfar, "Deblurring via stochastic refinement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16293–16303.

[71] B.-K. Kim, H.-K. Song, T. Castells, and S. Choi, "BK-SDM: Architecturally compressed stable diffusion for efficient text-to-image generation," in *Proc. Int. Conf. Mach. Learn. Workshops*, 2023, pp. 1–15.

[72] C. Zhao et al., "Dr^{2}Net: Dynamic reversible dual-residual networks for memory-efficient finetuning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 15835–15844.

[73] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervention*, 2015, pp. 234–241.

[74] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1828–1837.

[75] 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.

[76] J. Zhang, C. Zhao, and W. Gao, "Optimization-inspired compact deep compressive sensing," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 4, pp. 765–774, 2020.

[77] Y. Sun, J. Chen, Q. Liu, B. Liu, and G. Guo, "Dual-path attention network for compressed sensing image reconstruction," *IEEE Trans. Image Process.*, vol. 29, pp. 9482–9495, 2020.

[78] J. Chen, Y. Sun, Q. Liu, and R. Huang, "Learning memory augmented cascading network for compressed sensing of images," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 513–529.

[79] R. Zheng, Y. Zhang, D. Huang, and Q. Chen, "Sequential convolution and runge-kutta residual architecture for image compressed sensing," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 232–248.

[80] D. You, J. Xie, and J. Zhang, "ISTA-Net: Flexible deep unfolding network for compressive sensing," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2021, pp. 1–6.
[81] D. You, J. Zhang, J. Xie, B. Chen, and S. Ma, "COAST: Controllable arbitrary-sampling network for compressive sensing," *IEEE Trans. Image Process.*, vol. 30, pp. 6066–6080, 2021.
[82] Z. Zhang, Y. Liu, J. Liu, F. Wen, and C. Zhu, "AMP-Net: Denoising-based deep unfolding for compressive image sensing," *IEEE Trans. Image Process.*, vol. 30, pp. 1487–1500, 2021.
[83] J. Song, B. Chen, and J. Zhang, "Memory-augmented deep unfolding network for compressive sensing," in *Proc. ACM Int. Conf. Multimedia*, 2021, pp. 4249–4258.
[84] C. Mou, Q. Wang, and J. Zhang, "Deep generalized unfolding networks for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 17399–17410.
[85] Z.-E. Fan, F. Lian, and J.-N. Quan, "Global sensing and measurements reuse for image compressed sensing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8954–8963.
[86] W. Chen, C. Yang, and X. Yang, "FSOINet: Feature-space optimization-inspired network for image compressive sensing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 2460–2464.
[87] B. Chen and J. Zhang, "Content-aware scalable deep compressed sensing," *IEEE Trans. Image Process.*, vol. 31, pp. 5412–5426, 2022.
[88] M. Shen, H. Gan, C. Ning, Y. Hua, and T. Zhang, "TransCS: A transformer-based hybrid architecture for image compressed sensing," *IEEE Trans. Image Process.*, vol. 31, pp. 6991–7005, 2022.
[89] L. Rout, N. Raoof, G. Daras, C. Caramanis, A. G. Dimakis, and S. Shakkottai, "Solving linear inverse problems provably via posterior sampling with latent diffusion models," in *Proc. Neural Inf. Process. Syst.*, 2023, pp. 49960–49990.
[90] B. Fei et al., "Generative diffusion prior for unified image restoration and enhancement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 9935–9946.
[91] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," *Proc. Neural Inf. Process. Syst.*, 2021, pp. 8780–8794.
[92] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[93] C. Schuhmann et al., "LAION-5B: An open large-scale dataset for training next generation image-text models," in *Proc. Neural Inf. Process. Syst.*, 2022, pp. 25278–25294.
[94] W. Shi et al., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1874–1883.
[95] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–15.
[96] K. Ma et al., "Waterloo exploration database: New challenges for image quality assessment models," *IEEE Trans. on Image Process.*, vol. 26, no. 2, pp. 1004–1016, Feb. 2017.
[97] W. Li, B. Chen, and J. Zhang, "D3C2-Net: Dual-domain deep convolutional coding network for compressive sensing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 10, pp. 9341–9355, Oct. 2024.
[98] L. Gan, "Block compressed sensing of natural images," in *Proc. IEEE Int. Conf. Digit. Signal Process.*, 2007, pp. 403–406.
[99] T. T. Do, L. Gan, N. H. Nguyen, and T. D. Tran, "Fast and efficient compressive sensing using structurally random matrices," *IEEE Trans. Signal Process.*, vol. 60, no. 1, pp. 139–154, Jan. 2012.
[100] A. Adler, D. Boublil, and M. Zibulevsky, "Block-based compressed sensing of images via deep learning," in *Proc. IEEE 19th Int. Workshop Multimedia Signal Process.*, 2017, pp. 1–6.
[101] Z. Chen et al. "Deep-learned regularization and proximal operator for image compressive sensing," *IEEE Trans. Image Process.*, vol. 30, pp. 7112–7126, 2021.
[102] B. Chen, X. Zhang, S. Liu, Y. Zhang, and J. Zhang, "Self-supervised scalable deep compressed sensing," *Int. J. Comput. Vis.*, vol. 133, pp. 688–723, 2024.
[103] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.
[104] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2001, pp. 416–423.
[105] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5197–5206.
[106] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 126–135.
[107] 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.
[108] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 6629–6640.
[109] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.
[110] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.
[111] J. W. Lichtman and J.-A. Conchello, "Fluorescence microscopy," *Nat. Methods*, vol. 2, no. 12, pp. 910–919, 2005.
[112] S. Liu et al., "Deep learning-enhanced snapshot hyperspectral confocal microscopy imaging system," *Opt. Express*, vol. 32, no. 8, pp. 13918–13931, 2024.
[113] H. Sun and K. L. Bouman, "Deep probabilistic imaging: Uncertainty quantification and multi-modal solution characterization for computational imaging," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 2628–2637.

Bin Chen received the BS degree from the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China, in 2021. He is currently working toward the PhD degree in computer applications technology with the School of Electronic and Computer Engineering, Peking University, Shenzhen, China. His research interests include image compressive sensing and super-resolution.

Zhenyu Zhang received the BE degree in software engineering from Beijing Jiaotong University, Beijing, China, in 2019. He is currently working toward the master's degree in computer application technology with Peking University, Shenzhen, China. His research interests include image restoration, enhancement, and video processing.

Weiqi Li received the BE degree from the school of Software Engineering, Tongji University, Shanghai, China, in 2022. He is currently working toward the master's degree in computer applications technology with Peking University, Shenzhen, China. His research interests include image super-resolution, compressive sensing, and computer vision.

Chen Zhao received the PhD degree from Peking University (PKU), China in 2016, and studied in University of Washington (UW), from 2012 to 2013. She is currently a research scientist with the King Abdullah University of Science and Technology (KAUST), Saudi Arabia. She did an internship with the National Institute of Informatics (NII), Japan and worked as a research assistant with the Hong Kong University of Science and Technology (HKUST), China in 2016. Her research interests include image/video processing, image/video compression, image/video compressive sensing, and video understanding. On these topics, she has published more than 50 papers in representative journals and conferences and has received more than 3800 citations. She has led the team to win the firstplace in 6 video understanding challenges in CVPR 2024, CVPR 2023 and ECCV 2022 respectively. She was the recipient of the Best Paper Award in CVPR workshop 2023, the Best Paper Nomination in CVPR 2022, and the Best Paper Award in NCMT 2015.

Jiwen Yu received the bachelor's degree from the School of Computer Science, Northwestern Polytechnical University in Xi'an, China, in 2021. He is currently working toward the master's degree in computer application technology with the Peking University in Shenzhen, China. His research interests include image restoration, diffusion models, and computer vision.

Shijie Zhao received the bachelor's and doctoral degrees from the School of Mathematics, Zhejiang University, and the master's degree from Imperial College London. Currently, he leads the Video Processing and Enhancement team with ByteDance's Multimedia Lab. His main areas of research include video enhancement, low-level vision, and video compression.

Jie Chen received the MSc and PhD degrees from the Harbin Institute of Technology, China, in 2002 and 2007, respectively. He joined as a Faculty member with the Peking University, in 2019, where he is currently an associate professor with the School of Electronic and Computer Engineering. Since 2018, he has been working with the Peng Cheng Laboratory, China. From 2007 to 2018, he worked as a senior researcher with the Center for Machine Vision and Signal Analysis, University of Oulu, Finland. In 2012 and 2015, he visited the Computer Vision Laboratory, University of Maryland, and the School of Electrical and Computer Engineering, Duke University, respectively. His research interests include deep learning, computer vision, large language models, and Al4Science. He was the co-chair of International Workshops at ACCV, ACM MM, CVPR, ICCV, and ECCV. He was a guest editor of special issues of *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IJCV, and Neurocomputing. He has been selected as a Finalist of the 2022 Gordon Bell Special Prize for HPC-Based COVID-19 Research. He is an associate editor of the Visual Computer.

Jian Zhang (Member, IEEE) received the PhD degree from the School of Computer Science and Technology, Harbin Institute of Technology (HIT), Harbin, China, in 2014. He is currently an associate professor and heads the Visual-Information Intelligent Learning LAB (VILLA) with the School of Electronic and Computer Engineering, Peking University (PKU), Shenzhen, China. His research interest focuses on intelligent controllable image generation, encompassing three pivotal areas: efficient image reconstruction, controllable image generation, and precise image editing. He has published more than 100 technical articles in refereed international journals and proceedings and has received more than 10000 citations. He received several Best Paper Awards at international journals/conferences. He serves as an associate editor for the *Journal of Visual Communication and Image Representation*.