Robust Single-Image Super-Resolution via CNNs and TV-TV Minimization

Marija Vella and João F. C. Mota

Abstract—**Single-image super-resolution is the process of increasing the resolution of an image, obtaining a high-resolution (HR) image from a low-resolution (LR) one. By leveraging large training datasets, convolutional neural networks (CNNs) currently achieve the state-of-the-art performance in this task. Yet, during testing/deployment, they fail to enforce consistency between the HR and LR images: if we downsample the output HR image, it never matches its LR input. Based on this observation, we propose to post-process the CNN outputs with an optimization problem that we call *TV-TV minimization*, which enforces consistency. As our extensive experiments show, such post-processing not only improves the quality of the images, in terms of PSNR and SSIM, but also makes the super-resolution task robust to operator mismatch, i.e., when the true downsampling operator is different from the one used to create the training dataset.**

Index Terms—**Image super-resolution, image reconstruction, convolutional neural networks. (CNNs), ℓ_1-ℓ_1 minimization, prior information.**

I. Introduction

In SCIENCE and engineering, images acquired by sensing devices often have resolution well below the desired one. Common reasons include physical constraints, as in astronomy or biological microscopy, and cost, as in consumer electronics or medical imaging. Creating high-resolution (HR) images from low-resolution (LR) ones, a task known as *super-resolution* (SR), can therefore be extremely useful in these areas; it enables, for example, the identification of structures or objects that are barely visible in LR images. Doing so, however, requires inferring values for the unobserved pixels, which cannot be done without making assumptions about the class of images to super-resolve and their acquisition process.

Classical interpolation algorithms assume that the missing pixels can be inferred by linearly combining neighboring pixels via, e.g., the application of filters [3], [4], or by preserving the statistics of the image gradient from the LR to the HR image [5]. Reconstruction-based methods, on the other hand, assume that images have sparse representations in some domain, e.g., sparse gradients [6]–[12]. More recently, data-driven methods have become very popular; their main assumption is that image features can be learned from training data, via dictionaries [13], [14] or via convolutional neural networks (CNNs) [15]–[17].

CNNs were first applied to image SR in the seminal work [15] and have ever since remained the state-of-the-art, both in terms of reconstruction performance and computational complexity (during deployment/testing). By relying on vast databases of images for training, such as ImageNet [18] or T91 [13], they can effectively learn to map LR images/patches to HR images/patches. Although training a CNN can take several days, applying it to an image (what is typically called the testing phase) takes a few seconds or even sub-seconds. Despite these advantages, the knowledge that CNNs extract from data is never made explicit, making them hard to adapt to new scenarios: for example, simply changing the scaling factor or the sampling model, e.g., from bicubic to point sampling, almost always requires retraining the entire network. More conspicuously, however, is that during testing SR CNNs fail to guarantee the consistency between the reconstructed HR image and the input LR image, effectively ignoring precious "measurement" information, as we will illustrate shortly. Ignoring such information makes CNNs prone to generalization errors and, as a consequence, also less robust.

Curiously, adaptability and measurement consistency are the main features of classical reconstruction-based methods, which consist of algorithms designed to solve an optimization problem. To formulate such an optimization problem, one has to explicitly encode the measurement model and the assumptions about the class of images to be super-resolved. Although this explicit encoding confers reconstruction-based methods great adaptability and flexibility, it naturally limits the complexity of the assumptions, which is one of the reasons why reconstruction-based methods are outperformed by data-driven methods (CNNs). This motivates our problem:

Can we design SR algorithms that learn from large quantities of data and, at the same time, are easily adaptable to new scenarios and guarantee measurement consistency during the testing phase? In other words, can we design algorithms that have the advantages of both data-driven and model-based methods?

A. Lack of Consistency by CNNs

Before summarizing our method, we describe how CNNs fail to enforce consistency between the reconstructed HR image and the input LR image. Although we illustrate this

Manuscript received December 4, 2020; revised March 30, 2021; accepted August 16, 2021. Date of publication September 10, 2021; date of current version September 16, 2021. This work was supported in part by U.K.'s Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/T026111/1 and Grant EP/S000631/1, in part by the Ministry of Defense (MOD) University Defence Research Collaboration, and in part by EPSRC under Grant EP/S018018/1. Part of this work has been presented in [1] [DOI: 10.5244/C.33.219], [2]. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hairong Qi. *(Corresponding author: João F. C. Mota.)*

The authors are with the School of Engineering and Physical Sciences, Heriot-Watt University, Edinburgh EH14 4AS, U.K. (e-mail: mv37@hw.ac.uk; j.mota@hw.ac.uk).

Digital Object Identifier 10.1109/TIP.2021.3108907

phenomenon here for the specific SRCNN network [15], more systematic experiments can be found in Section IV. The top-left corner of Fig. 1 shows a ground truth (GT) image $X^\star \in \mathbb{R}^{M\times N}$, which the algorithms have access to during training, but not during testing. We will represent $X^\star$ by its column-major vectorization $x^\star = \text{vec}(X^\star) \in \mathbb{R}^n$, where $n = M \cdot N$. The GT image $x^\star$ is downsampled via a linear operator represented by $A \in \mathbb{R}^{m\times n}$ into a LR image $b := Ax^\star \in \mathbb{R}^m$. In this specific example, $n = 240,000$ and $m = 15,000$, i.e., $x^\star$ is downsampled by a factor of 4, and A implements bicubic downsampling. The goal is to reconstruct $x^\star$ from b, i.e., to super-resolve b. The figure illustrates that CNN methods, in particular [15], reconstruct an image w that does not necessarily satisfy $Aw = b$, even though we know that $Ax^\star = b$. Specifically, SRCNN [15] outputs an image $w = \text{vec}(W)$ (top-right corner) very close to $x^\star$ (22.73 dBs in PSNR) but that fails to satisfy $Aw = b$ with enough precision: $||b - Aw||_2 \simeq 0.53$. We point out that A represents bicubic downsampling, which was what the authors of [15] assumed during the training of SRCNN.

Although the data consistency (DC) problem has been given importance in other fields, only a few SISR techniques have tackled it. Existing supervised methods mainly rely on modifying the network structure [19], [20] or learn pretrained-denoisers and plug them in a model-based algorithm [21], [22].

B. Our Approach

Our algorithm can be viewed as a post-processing step that takes as input the CNN image w and the LR image b, and reconstructs a HR image $\hat{x} \in \mathbb{R}^n$ that is similar to w but, in contrast to it, satisfies $A\hat{x} = b$ (bottom-right corner of Fig. 1). As a result, the images created by our method almost always have better quality than w, in terms of PSNR and SSIM. In addition, our method confers robustness to the SR task, even when the operator A used to generate the training data differs from the one used during testing.

We integrate b and w via an algorithm that solves an optimization problem that we call *TV-TV minimization*. The problem enforces the reconstructed image to have a small number of edges, a property captured by a small TV-norm, and also to not differ much from w, as measured again by the TV-norm. Naturally, it also imposes the constraint $A\hat{x} = b$.

C. Contributions

We summarise our contributions as follows:

1) We introduce a framework that has the advantages of learning-based and reconstruction-based methods. Like reconstruction methods, it is adaptable, flexible, and enforces measurement consistency. At the same time, it retains the excellent performance of learning methods.
2) We integrate learning and reconstruction-based methods via a TV-TV minimization problem. Although we have no specific theoretical guarantees for it, existing theory for a related, simpler problem (ℓ_1-ℓ_1 minimization) provides useful insights about how to tune a regularization parameter. This makes our algorithm easy to deploy, since there are virtually no parameters to tune.
3) We propose an algorithm based on the alternating direction method of multipliers (ADMM) [23] to solve the TV-TV minimization problem. In contrast with most SR methods, which process image patches independently, our algorithm processes full images at once. It also easily adapts to different degradations and scaling factors.
4) We conduct extensive experiments that illustrate not only the robustness of our algorithm under different degradation operators, but also how it systematically improves (in terms of PSNR and SSIM) the outputs of state-of-the-art SR networks, such as EDSR [24] and RCAN [25].

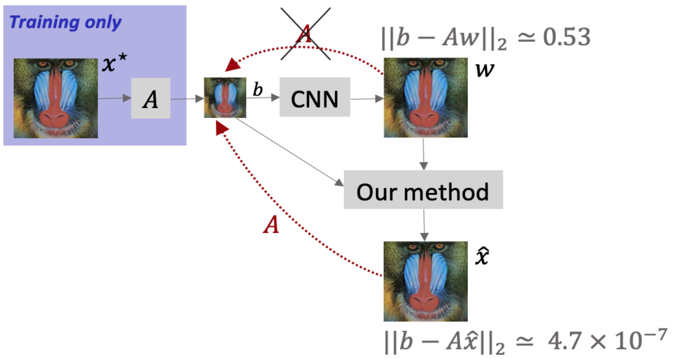

Training only
$x^\star$
A
b
CNN
A
$||b - Aw||_2 \simeq 0.53$
w
Our method
A
$\hat{x}$
$||b - A\hat{x}||_2 \simeq 4.7 \times 10^{-7}$

Fig. 1. Illustration of the lack of measurement consistency by CNNs during *testing*: when the output image w is downsampled using A, it typically differs significantly from b. Our method takes in both w and b, and fixes this problem.

We highlight the following differences with respect to our previous work in [1], [2]. We now explore and illustrate with experiments the underlying reason why our framework improves the output of state-of-the-art SR CNNs. We also describe how the proposed optimization problem can be solved efficiently using ADMM; in fact, the algorithms used in [1], [2] were different and less efficient than the algorithm we present here. Our experiments are also much more extensive: they consider different sampling operators to illustrate robustness to operator mismatch, and include many more algorithms, e.g., FSRCNN [26], VDSR [17], LapSRN [27], SRMD [21], IRCNN [28], and the PSNR oriented network presented in [16] which we refer to in this paper as $\text{ESRGAN}_{\text{PSNR}}$.

D. Organization

Section II summarizes prior work on SR algorithms, and Section III describes the proposed framework and optimization scheme. Section IV then reports our experimental results, and Section V concludes the paper.

II. Related Work

SR schemes are often labeled as interpolation, reconstruction, or data-driven. Interpolation methods infer the missing pixels by locally applying an interpolation function such as the bicubic or bilinear filter [3], [4]. As they have been surpassed by both reconstruction and learning SR algorithms, we will limit our review to the latter.

A. Reconstruction-Based SR

Reconstruction-based schemes view SR as an image reconstruction problem and address it by formulating an optimization problem. In general, the optimization problem has two terms: a DC term that encodes assumptions about the acquisition process, usually that $Ax \simeq b$ (where x is the optimization variable), and a regularization term on x that encodes assumptions about the class of images. Different methods differ mostly on the image assumptions.

1) Image Assumptions: Reconstruction SR methods encode assumptions about the images by penalizing in the optimization problem measures of complexity. These reflect the empirical observation that natural images have parsimonious representations in several domains. Examples include sparsity in the wavelet domain [29], sparsity of image patches in the DCT domain [30] and, as we will explore shortly in more detail, sparsity of image gradients [6]–[12]. Since sparsity is well captured by the ℓ_1-norm, the resulting optimization problem is typically convex and can be solved efficiently. A more challenging assumption is multi-scale recurrence [24], [31], which captures the notion that patches of natural images occur repeatedly across the image.

2) Total Variation: In natural images, the number of pixels that correspond to an edge, i.e., a transition between different objects, is a small percentage of the total number of pixels. This can be measured by the total variation (TV) of the image [6]. Although TV was initially defined in the context of partial differential equations, there has been work that discretizes the differential equations [9], [10] or that directly defines TV in the discrete setting [11], [32]. Although there are several definitions of discrete TV, the most popular are the *isotropic TV*, which consists of the sum (over all pixels) of the ℓ_2-norms of the vectors containing the horizontal and vertical differences at each pixel, and the *anisotropic TV*, which is similar to isotropic TV but with the ℓ_2-norms replaced by the ℓ_1-norm. Both definitions yield convex, yet nondifferentiable, functions. Many algorithms have been proposed to solve problems involving discrete TV, including primal-dual methods [8], [33], and proximal and gradient-based schemes [11], [12].

The concept of TV has been used in many imaging tasks, from denoising [6], [11] to SR [9], [10]. For example, [9] discretizes a differential equation relating variations in the values of pixels to the curvature of level sets, while enforcing fidelity to the LR image.

3) Back-Projection: Back-projection (BP) is an iterative algorithm originally proposed for multi-image SR [34]. It minimizes the reconstruction error and then projects the outputs back to the GT image to adjust its intensity. Although this improves the outputs, it is prone to ringing and chessboard artifacts.

B. Learning-Based Algorithms

Learning-based algorithms typically consist of two stages: *training*, in which a map from LR to HR patches is learned from a database of training images, and *testing*, in which the learned map is applied to super-resolve an unseen image.

1) Dictionary Learning: In dictionary learning, also known as sparse coding, patches of HR images are assumed to have a sparse representation on an over-complete dictionary, which is learned from training images. For example, [13] uses training images to learn dictionaries for LR and HR patches while constraining corresponding patches to have the same coefficients. Other schemes use similar concepts, but require no training data at all. For example, [14] uses self-similarity to learn the LR-HR map without any external database of images.

2) CNN-Based Methods: The advent of deep learning and the availability of large image datasets inspired the application of CNNs to SR. Currently, they surpass any reconstruction- or interpolation-based method both in reconstruction performance and in execution time (during testing). The first CNN for SR was proposed by [15]; although its design was inspired by dictionary learning methods, the proposed architecture set a new standard for SR performance.

SR networks can be classified as direct or progressive. In direct networks, the LR image is first upscaled, typically via bicubic interpolation, to the required spatial resolution, and then is fed to a CNN, as in [21], [24], [25]. In this case, the CNN thus learns how to deblur the upscaled image. As previously mentioned, CNN architectures need to be retrained every time we change the scaling factor. To overcome this, [35] repeatedly applied a recursive convolutional layer to obtain the super-resolved image. However, since the LR input is blurry, the CNN outputs a HR image lacking fine details. Inspired by this observation, [36] proposed the SRGAN, which produces photo-realistic HR images, even though they do not yield the best PSNR. As direct networks operate on high-dimensional images, their training is computationally expensive [37].

Progressive networks, in contrast, have reduced training complexity, as they directly process LR images. Specifically, the upsampling step, performed using sub-pixel or transposed convolution [37], is applied at the end of the network. For instance, LapSRN [27] used the concept of Laplacian pyramids, in which each network level is trained to predict residuals between the upscaled images at different levels in the pyramid.

In spite of achieving state-of-the-art performance, CNNs for SR suffer from two major shortcomings: as already illustrated, they fail to guarantee the consistency between the LR and HR image during testing, and the trained network applies only to a unique scaling factor and degradation function. Most of the CNNs, e.g. [15], [21], [26], are trained by solving

$$\underset{\theta}{\text{minimize}} \;\; \frac{1}{T}\sum_{t=1}^{T} \big\| f_\theta(Ax^{(t)}) - x^{(t)} \big\|_2^2,$$

where $x^{(t)}$ represents (the vectorization of) the tth image in the training set, A the bicubic sampling operator, and $f_\theta(\cdot)$ a CNN parameterized by θ (i.e., weights and biases of the neural connections). Most CNNs are trained with images that have been downsampled with a bicubic filter. Thus, their performance degrades when the true downsampling operator is different. Indeed, during testing, the true degradation is unknown.

The work in [21] addresses this problem by designing a network that deals with different degradations by accepting as input both the blur kernel and the noise level.

C. Plug-and-Play Methods

A different line or work blends learning- and model-based methods. The main observation is that, when solving linear inverse problems, proximal-based algorithms separate the operations of measurement consistency and problem regularization (using prior knowledge). The latter usually consists of a simple operation, like soft-thresholding, which encodes the assumptions about the target image and which can be viewed as a denoising step. Given its independence from the measurements, such operation can be replaced by a more complex function, such as a CNN. The resulting algorithms are versatile, as the measurement operator can be easily modified. Most work in this area, however, has focused on compressed sensing, in which the measurement operator is typically a dense random matrix; see [38]–[40].

The One-Net [41], for example, replaces the proximal operator associated to image regularization in an ADMM algorithm with a CNN trained on a large database. The experiments in [41] considered SR, but the resulting network does not perform as well as current leading CNN-based methods.

The pioneering work in [42] takes this idea further and proposes a scheme that requires no training at all. There, an untrained CNN is used as a prior. Specifically, a linear inverse problem is reparameterized as a function of the weights of a CNN whose input is a noisy/corrupted image and whose output is the denoised/reconstructed image. Such reparameterization provides a type of regularization.

The work in [43] combined this idea with regularization by denoising and used ADMM to solve the resulting linear inverse problem.

These algorithms require no training and can be adapted to different measurement operators. However, they can be slow, as each iteration requires some backpropagation iterations on the CNN. And, when applied to SR, they are still outperformed by training-based CNN architectures.

Another work embeds DC layers in the network. For example, Haris *et.al.* [19] extends the BP algorithm by using consecutive up and down sampling blocks to create the HR feature map.

III. Proposed Framework

A. Main Model and Assumptions

We aim to reconstruct the vectorized version of an HR image $x^\star \in \mathbb{R}^n$ from a LR image $b \in \mathbb{R}^m$, with $m < n$. We assume that these quantities are linearly related:

$$b = Ax^\star, \tag{1}$$

where $A \in \mathbb{R}^{m\times n}$ represents the downsampling operator. The model in (1) is often used in reconstruction-based and dictionary learning algorithms [10], [13], [29], even though many methods also consider additive noise: $b = Ax^\star + \epsilon$, where ϵ is a Gaussian random vector [12], [41], [44]–[46].

More interesting, however, is that CNN-based methods implicitly assume the model in (1), although that is rarely acknowledged. In particular, all the SR networks we know of (e.g., [15], [16], [26], [27]) are trained with HR images that are downsampled according to (1), where A implements bicubic downsampling. We next discuss other possible choices for A.

1) Common Choices for A: Different instances of $A \in \mathbb{R}^{m\times n}$ in (1) have been assumed in the SR literature:

- *Simple Subsampling*: A contains equispaced rows of the identity matrix $I_n \in \mathbb{R}^{n\times n}$, i.e., each row of A is a canonical vector $(0, \dots, 0, 1, 0, \dots, 0)$. This operator is simple to implement, but often introduces aliasing.
- *Bicubic*: $A = S \cdot B$, where S is a simple subsampling operator, and B is a bicubic filter. It is the operator of choice for processing training data for CNNs.
- *Box-Averaging*: if the scaling factor is s, then each row of A contains s^2 nonzero elements, equal to $1/s^2$, in positions corresponding to a neighborhood of a pixel. In other words, box-averaging replaces each block of $s \times s$ pixels by their average. Although simpler than the bicubic operator, it does not introduce the aliasing that simple subsampling does; see, e.g., [41].

In our experiments, we will mostly instantiate A as a bicubic operator. The reason is that most SR CNNs assume this operator during training. Simple subsampling and box-averaging will be used to illustrate how our post-processing scheme adds robustness to operator mismatch, i.e., when A is different during training and testing.

2) Assumptions: We estimate $x^\star \in \mathbb{R}^n$ from $b \in \mathbb{R}^m$ by taking into account two possibly conflicting assumptions:

1) $x^\star$ has a small TV; a property that indicates the reconstructed image is close to the GT. For example, the average TV norm of the BSD100 dataset is 0.08.
2) $x^\star$ is also close to the prior information w, the image returned by a learning-based method (CNN), where the notion of distance is also measured by TV.

For a given vectorization $x \in \mathbb{R}^n$ of an image $X \in \mathbb{R}^{M\times N}$, the anisotropic 2D TV (semi-)norm is defined as [7], [11]

$$\|x\|_{\mathrm{TV}} := \sum_{i=1}^{M}\sum_{j=1}^{N} \left|v_{ij}^\top x\right| + \left|h_{ij}^\top x\right| = \left\| \begin{bmatrix} V \\ H \end{bmatrix} x \right\|_1 = \|Dx\|_1. \tag{2}$$

In (2), $v_{ij} \in \mathbb{R}^n$ and $h_{ij} \in \mathbb{R}^n$ extract the vertical and horizontal differences at pixel (i, j) of X. By concatenating v_{ij} (resp. h_{ij}) as rows of $V \in \mathbb{R}^{n\times n}$ (resp. $H \in \mathbb{R}^{n\times n}$), we obtain the representation in (2), where $\|\cdot\|_1$ is the ℓ_1-norm (sum of absolute values). And the matrix $D \in \mathbb{R}^{2n\times n}$ in (2) is the vertical concatenation of V and H. We assume periodic boundaries, so that both V and H are circulant. As circulant matrices are diagonalizable by the DFT, matrix-vector products by both V and H can be computed via the FFT in $O(n \log n)$ time.

B. Our Framework

The framework we propose is shown schematically in Fig. 2. It starts by super-resolving b into $w \in \mathbb{R}^n$ with a base method,

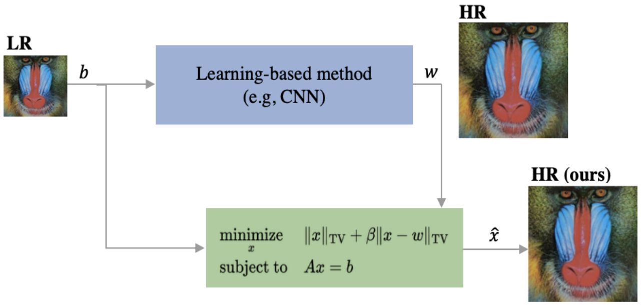


Fig. 2. Our framework: the low-resolution image b and the image w super-resolved by a CNN are fed into the TV-TV minimization problem which, in turn, obtains a high-resolution image $\widehat{x}$ with better quality.

which we assume is implemented by a CNN due to their current outstanding performance. As explained in Section I, CNNs fail to enforce measurement consistency during testing, i.e., $Aw \neq b$ for any matrix A that is assumed to implement the downsampling operation.

We propose to use an additional block that takes in both the HR output w of the CNN and the LR image b, and creates another HR image $\widehat{x}$. The block implements what we call TV-TV minimization, which enforces measurement consistency while guaranteeing that assumptions 1) and 2) are met.

1) TV-TV Minimization: Given the LR image b and a HR image w, *TV-TV minimization* consists of

$$\begin{aligned} &\underset{x}{\text{minimize}}\ \|x\|_{\mathrm{TV}} + \beta\|x-w\|_{\mathrm{TV}} \\ &\text{subject to}\ Ax = b, \end{aligned} \tag{3}$$

where $x \in \mathbb{R}^n$ is the optimization variable, and $\beta \geq 0$ tradeoffs between assumptions 1) and 2). Indeed, the first term in the objective of (3) encodes assumption 1), the second term assumption 2), and the constraints enforce measurement consistency. Robustness of our method is achieved by the framework itself. CNNs, being purely data-driven, suffer from generalization errors. A generalization error typically implies lack of consistency with the input LR image. Our method overcomes this by guaranteeing measurement consistency, and thus robustness with respect to generalization errors, via the constraints of TV-TV minimization. Of course, our assumptions 1)-2) can be easily modified to better capture the class of images to be super-resolved. We found that using TV semi-norms in the objective yielded better results. In addition, as these functions are convex, problem (3) is convex as well.

Although a problem like (3) has appeared before in [47] in the context of dynamic computed tomography (CT), the prior information w there was an image reconstructed by solving the same problem in the previous instant; see also [48]–[50]. *Our approach is conceptually different in that we use* (3) *to improve the output of a CNN-based method.*

Next, we show how TV-TV minimization relates to ℓ_1-ℓ_1 minimization, and how the theory for the latter in [48] suggests that selecting $\beta = 1$ in (3) may lead to better performance.

2) Relation to ℓ_1-ℓ_1 Minimization: Introducing an auxiliary variable $u \in \mathbb{R}^{2n}$ and defining $\overline{w} := Dw$, we rewrite (3) as

$$\begin{aligned} &\underset{u,x}{\text{minimize}}\ \|u\|_1 + \beta\|u-\overline{w}\|_1 \\ &\text{subject to}\ Ax = b, Dx \ = u. \end{aligned} \tag{4}$$

Thus, $\overline{x} := (u, x) \in \mathbb{R}^{3n}$ is the full optimization variable. Define $\overline{A} := \big[0_{m\times 2n}\ A;\ -I_{2n}\ D\big]$, and $\overline{b} := \big[b\ 0_{2n}\big]^\top$, where $0_{a\times b}$ (resp. 0_a) represents the zero matrix (resp. vector) of dimensions $a \times b$ (resp. $a \times 1$), and I_{2n} is the identity matrix in $\mathbb{R}^{2n}$. This enables us to rewrite (4) as

$$\begin{aligned} &\underset{\overline{x}}{\text{minimize}} \|G_{2n}\overline{x}\|_1 + \beta\|G_{2n}\overline{x} - \overline{w}\|_1 \\ &\text{subject to}\ \overline{A}\overline{x} = \overline{b}, \end{aligned} \tag{5}$$

where $G_{2n} \in \mathbb{R}^{2n\times 3n}$ contains the first $2n$ rows of the identity matrix I_{3n}. In other words, for a vector $v \in \mathbb{R}^{3n}$, $G_{2n}v$ represents the first $2n$ components of v. The work in [48] analyzes (5) when G_{2n} is the full identity matrix and the entries of $\overline{A}$ are drawn from a Gaussian distribution. Specifically, it provides the number of measurements required for perfect reconstruction under these assumptions. It is shown both theoretically and experimentally that the best reconstruction performance is obtained when $\beta = 1$. Although the theory in [48] cannot be easily extended[1] to (5), our experiments indicate that $\beta = 1$ still leads to the best results in our setting.

C. Algorithm for TV-TV Minimization

We now explain how to efficiently solve TV-TV minimization (3) with ADMM [23]. In contrast with the majority of SR algorithms, which operate on individual patches, our algorithm operates on full images. We do that by capitalizing on the fact that matrix-vector multiplications can be performed fast whenever the matrix is D [cf. (2)] or any of the instantiations of A mentioned in Section III-A.

ADMM: The problem that ADMM solves is

$$\begin{aligned} &\underset{y,z}{\text{minimize}}\ f(y) + g(z) \\ &\text{subject to}\ Fy + Gz = 0, \end{aligned} \tag{6}$$

where f and g are closed, proper, and convex functions, and F and G are given matrices. Associating a dual variable λ to the constraints of (6), ADMM iterates on k

$$y^{k+1} = \underset{y}{\text{argmin}}\ f(y) + \frac{\rho}{2}\big\|Fy + Gz^k + \lambda^k\big\|_2^2 \tag{7a}$$

$$z^{k+1} = \underset{z}{\text{argmin}}\ g(z) + \frac{\rho}{2}\big\|Fy^{k+1} + Gz + \lambda^k\big\|_2^2 \tag{7b}$$

$$\lambda^{k+1} = \lambda^k + Fy^{k+1} + Gz^{k+1}, \tag{7c}$$

where $\rho > 0$ is the augmented Lagrangian parameter.

1) Applying ADMM: Although there are many possible reformulations of (3) to which ADMM is applicable, they can yield different performances. Our reformulation simply adds another variable $v \in \mathbb{R}^n$ to (4) [which is equivalent to (3)]:

$$\begin{aligned} &\underset{u,x,v}{\text{minimize}}\ \|u\|_1 + \beta\|u-\overline{w}\|_1 \\ &\text{subject to}\ Ax = b,\ \ Dv = u,\ \ v = x. \end{aligned} \tag{8}$$

We establish the following correspondence between (6) and (8): we set $y = (u, x)$, $z = v$, and assign

$$f(u,x) = \|u\|_1 + \beta\|u-\overline{w}\|_1 + \mathrm{i}_{Ax=b}(x) \quad F = \begin{bmatrix} -I_{2n} & 0 \\ 0 & I_n \end{bmatrix}$$

$$g(v) = 0 \quad G = \begin{bmatrix} D \\ -I_n \end{bmatrix},$$

[1] The reason is that the matrix $\overline{A}$ is very structured and thus, even when A is assumed Gaussian, its nullspace is not uniformly distributed.

where $\mathrm{i}_{Ax=b}(x)$ is the indicator function of $Ax = b$, i.e., it evaluates to 0 if $Ax = b$, and to $+\infty$ otherwise. This means that we dualize only the last two constraints of (8), and thus λ has two components: $\lambda = (\eta, \mu) \in \mathbb{R}^{2n} \times \mathbb{R}^{n}$. The above correspondence yields closed-form solutions for the problems in (7a) and (7b) (see below). Furthermore, even though the objective of (8) is not strictly convex, the fact that F and G have full column-rank implies that the sequence (y^k, z^k) generated by ADMM (7) has a *unique* limit point, which solves (8) [51]. We now elaborate on how to solve (7a)-(7b).

2) Solving (7a)*:* Using the above correspondence, problem (7a) decouples into two independent problems that can be solved in parallel:

$$u^{k+1} = \underset{u}{\operatorname{argmin}} \ \|u\|_1 + \beta\|u - \overline{w}\|_1 + \frac{\rho}{2}\|u - s^k\|_2^2 \quad (9)$$

$$x^{k+1} = \underset{x}{\operatorname{argmin}} \ \frac{1}{2}\|x - p^k\|_2^2 \quad \text{s.t.} \ \ Ax = b, \quad (10)$$

where we defined $s^k := Dv^k + \eta^k$ and $p^k := v^k - \mu^k$.

Problem (9) decomposes further componentwise, and the solution for each component can be obtained by evaluating the respective optimality condition (via subgradient calculus). Namely, for $i = 1, \ldots, 2n$, if $\overline{w}_i \geq 0$, component u_i^{k+1} is

$$\begin{cases} s_i - \frac{1}{\rho}(\beta+1), & s_i > \overline{w}_i + \frac{1}{\rho}(\beta+1) \\ \overline{w}_i, & \overline{w}_i - \frac{1}{\rho}(\beta-1) \leq s_i \leq \overline{w}_i + \frac{1}{\rho}(\beta+1) \\ s_i + \frac{1}{\rho}(\beta-1), & -\frac{1}{\rho}(\beta-1) < s_i < \overline{w}_i - \frac{1}{\rho}(\beta-1) \\ 0, & -\frac{1}{\rho}(\beta-1) \leq s_i \leq -\frac{1}{\rho}(\beta-1) \\ s_i + \frac{1}{\rho}(\beta+1), & s_i < -\frac{1}{\rho}(\beta+1). \end{cases} \quad (11)$$

The case for $\overline{w}_i < 0$ is obtained in similar way. More details are provided in[2].

Problem (10) is the projection of p^k onto the solutions of $Ax = b$. Assuming that A has full row-rank, i.e., $AA^\top$ is invertible, (10) also has a closed-form solution:

$$x^{k+1} = p^k - A^\top(AA^\top)^{-1}(Ap - b), \quad (12)$$

whose computation has a complexity that depends on the properties of the downsampling operator A. When A is simple subsampling or the box-averaging operator, $AA^\top$ is the identity matrix I_m or a multiple of it. In that case, computing (12) requires only two matrix-vector operations which, due to the structure of A, can be implemented by indexing. In other words, there is no need to construct A explicitly.

On the other hand, when A is the bicubic operator, the inverse of $AA^\top$ can no longer be computed easily, and we solve the linear system in (10) with the conjugate gradient method. In this case, matrix-vector products can be computed in $O(n \log n)$ time using the FFT.

3) Solving (7b)*:* With our choice of g, F, and G, problem (7b) becomes

$$\begin{aligned} v^{k+1} &= \underset{v}{\operatorname{argmin}} \ \frac{1}{2}\big\|Dv - u^{k+1} + \eta^k\big\|_2^2 + \frac{1}{2}\big\|v - x^{k+1} + \mu^k\big\|_2^2 \\ &= (I_n + D^\top D)^{-1}\Big[x^{k+1} - \mu^k + D^\top(u^{k+1} - \eta^k)\Big]. \quad (13) \end{aligned}$$

Given the definition of D in (2), we have

$$\begin{aligned} I_n + D^\top D &= I_n + V^\top V + H^\top H \\ &= C_n^H\Big(I_n + \mathrm{Diag}(C_n v)^2 + \mathrm{Diag}(C_n h)^2\Big)C_n, \end{aligned}$$

where the last step uses the fact that V and H are circulant matrices and, therefore, are generated by some vectors v and h, respectively. Also, C_n denotes the DFT matrix in $\mathbb{R}^n$, and $\mathrm{Diag}(x)$ is a diagonal matrix with the entries of x in its diagonal. This representation of $I_n + D^\top D$ not only enables us to compute its inverse in closed-form (just take the inverse of the matrix in parenthesis), but also to do it without constructing any matrix explicitly.

4) Dual Updates: Finally, since λ decomposes as (η, μ), the dual variable update in (7c) becomes

$$\eta^{k+1} = \eta^k + Dv^{k+1} - u^{k+1} \quad (14a)$$

$$\mu^{k+1} = \mu^k + x^{k+1} - v^{k+1}. \quad (14b)$$

Applying ADMM (7) to the equivalent reformulation (8) of TV-TV minimization (3) therefore yields step (11) for each component of u, (12) for x, (13) for v, and (14) for the dual variables. These steps are repeated iteratively until a stopping criterion is met; we use the one suggested in [23].

IV. Experiments

We now describe our experiments. After explaining the experimental setup, we expand on the phenomenon described in Fig. 1. Then, we consider the case of operator mismatches (i.e., A is different during training and testing), and show how our framework adds significant robustness in this scenario. Finally, we report experiments on standard SR datasets. Code to replicate our experiments is available online[2].

A. Experimental Setup

1) Algorithm Parameters: Most experiments were run using the same algorithm settings, unless indicated otherwise. The hyperparameter β in (3) was generally set to 1 and in some instances set to 2. For most experiments, A was the bicubic operator via MATLAB's IMRESIZE. For ADMM, we adopted the stopping criterion in [23, §3.3.1] with $\epsilon^{\text{pri}} = \epsilon^{\text{dual}} = 0.001$, or stopped after 500 iterations. Also, we initialized $\rho = 0.5$ and adjusted it automatically using the heuristic in [23, §3.4.1].

2) Datasets: We considered the standard SR test sets Set5 [52], Set14 [53], BSD100 [54] and Urban100 [55], which contain images of animals, buildings, people, and landscapes.

3) Computational Platform: All experiments were run on Matlab using a workstation with 12 core 2.10GHz Intel Xeon Silver 4110 CPU and two NVIDIA GeForce RTX GPUs.

4) Methods Evaluated: We compared our framework against the state-of-the-art methods in Table I and also considered simple TV minimization, i.e., (3) with $\beta = 0$, using the TVAL3 solver [8]. The table shows the acronyms and references of the methods, their main technique, the scaling factors (S.F.) considered in the original papers, and the datasets used for training. Note that all methods except ESRGAN$_{\text{PSNR}}$ were

[2] https://github.com/marijavella/sr-via-CNNs-and-tvtv

TABLE I

METHODS USED IN OUR EXPERIMENTS. FOR EACH, WE SHOW THE MAIN TECHNIQUE, THE SCALING FACTORS IT CAN HANDLE AND, IF ANY, THE TRAINING DATASET

Method	Type	S.F.	Training dataset
SRCNN [15]	CNN	2, 4	ImageNet [18]
FSRCNN [26]	CNN	2, 4	T91 [13], General100 [26]
DRCN [35]	CNN	2, 4	T91 [13]
VDSR [17]	CNN	2, 4	T91 [13], BSDS200 [56]
LapSRN [27]	CNN	2, 4	T91 [13], BSDS200 [56]
SRMD [21]	CNN	2, 4	DIV2K [57], BSDS200 [56] WED [58]
RCAN [25]	CNN	2, 4	DIV2K
EDSR [24]	CNN	2, 4	DIV2K
IRCNN [28]	Plug-and-play	2, 4	ImageNet [18], WED [58], BSD500 [56]
ESRGAN$_{\text{PSNR}}$ [16]	GAN	4	DIV2K [57], Flickr2K [57], OutdoorSceneTraining [60]
DeepRED [42]	Plug-and-play	2, 4	
TVAL3 [8]	Optimization	2, 4	

TABLE II

CONSISTENCY ACHIEVED BY CNN-TYPE METHODS ($\|Aw - b\|_2$) AND BY OUR ALGORITHM ($\|A\hat{x} - b\|_2$)

Method	Image	$\|Aw - b\|_2$	$\|A\hat{x} - b\|_2$
SRCNN [15]	*Baboon*	5.29×10^{-1}	$\mathbf{4.77 \times 10^{-7}}$
	38092	4.32×10^{-1}	$\mathbf{3.54 \times 10^{-7}}$
	img$_{005}$	14.93×10^{-1}	$\mathbf{8.02 \times 10^{-7}}$
FSRCNN [26]	*Baboon*	3.26×10^{-1}	$\mathbf{4.93 \times 10^{-7}}$
	38092	2.91×10^{-1}	$\mathbf{4.07 \times 10^{-7}}$
	img$_{005}$	10.32×10^{-1}	$\mathbf{4.10 \times 10^{-7}}$
SRMD [21]	*Baboon*	4.14×10^{-1}	$\mathbf{6.85 \times 10^{-7}}$
	38092	2.39×10^{-1}	$\mathbf{9.96 \times 10^{-7}}$
	img$_{005}$	9.53×10^{-1}	$\mathbf{3.95 \times 10^{-7}}$
IRCNN [28]	*Baboon*	1.09×10^{-1}	$\mathbf{6.14 \times 10^{-7}}$
	38092	9.15×10^{-2}	$\mathbf{5.12 \times 10^{-7}}$
	img$_{005}$	5.33×10^{-1}	$\mathbf{7.15 \times 10^{-7}}$
DeepRed [43]	*Baboon*	2.62×10^{-1}	
	38092	2.06×10^{-1}	
	img$_{005}$	3.11×10^{-1}	

evaluated for 2× and 4× scaling factors since ESRGAN$_{\text{PSNR}}$ only handles 4×. The training datasets in Table I have 91 (T91), 100 (General100), 200 (BSDS200), 324 (OutdoorSceneTraining), 500 (BSDS500), 800 (DIV2K), 2650 (Flickr2K), 4744 (WED), and 396,000 (ImageNet) images.

Both during training and testing, SRCNN, DRCN and FSRCNN extract the luminance channel of the YCbCr color space, while the rest of the CNN-based methods in Table I work on all the RGB channels.

During training, the HR images are converted to LR images by applying MATLAB's IMRESIZE, as originally done in [15]. The output images for SRCNN [15] were retrieved from an online repository.[3] For the remaining methods, we generated the outputs from the available pretrained models.

5) Performance Metrics: We compared different algorithms by evaluating the PSNR (dB) and SSIM [61] on the luminance channel of the output images. We also provide sample images for qualitative evaluation.

TABLE III

OPERATOR MISMATCH EXPERIMENTS. PSNR VALUES UNDER DIFFERENT SAMPLING OPERATORS FOR A: BICUBIC, BOX FILTERING, AND SIMPLE SUBSAMPLING. WITHIN EACH BOX, THE BEST (HIGHER) VALUES ARE HIGHLIGHTED IN **BOLD**

Method	Image	Bic.	*Ours*	Box	*Ours*	Sub.	*Ours*
SRCNN [15]	*Baboon*	22.70	**22.73**	22.49	**22.53**	17.48	**19.16**
	38092	25.90	**25.95**	25.69	**25.75**	20.07	**21.84**
	img$_{005}$	25.12	**25.27**	24.99	**25.21**	17.92	**19.70**
FSRCNN [26]	*Baboon*	22.79	**22.80**	22.49	**22.55**	17.38	**19.28**
	38092	26.03	**26.05**	25.64	**25.73**	20.00	**21.94**
	img$_{005}$	25.81	**25.85**	25.12	**25.34**	17.79	**19.72**
SRMD [21]	*Baboon*	22.90	**22.91**	22.52	**22.59**	16.95	**18.98**
	38092	26.20	**26.21**	25.62	**25.73**	19.39	**21.50**
	img$_{005}$	26.56	**26.61**	25.59	**25.89**	17.36	**19.30**
IRCNN [28]	*Baboon*	**22.76**	**22.76**	**22.51**	**22.51**	17.41	**19.45**
	38092	**26.09**	**26.09**	25.77	**25.78**	20.54	**22.40**
	img$_{005}$	26.18	**26.21**	**25.86**	**25.86**	18.23	**19.64**
TVAL3 [8]	*Baboon*	22.40		22.27		20.83	
	38092	25.59		25.00		21.35	
	img$_{005}$	24.29		22.54		17.28	

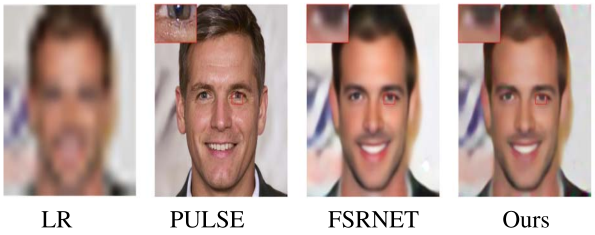


Fig. 3. Result on a sample image from CelebA-HQ.

TABLE IV

PSNR AND NIQE FOR THE SAMPLE IMAGE IN FIG. 3

Method	PSNR	NIQE
PULSE [62]	18.83	3.16
FSRNET [63]	20.80	5.87
FSRNET + TVTV (Ours)	23.58	5.73

B. Measurement Inconsistency of CNNs

We show that the phenomenon illustrated in Fig. 1 for SRCNN [15] occurs not only for this network, but is pervasive. That is, CNNs for SR fail to enforce measurement consistency (1) during testing. We chose three images for this purpose: *Baboon* from Set14, *38092* from BSD100, and *img*$_{005}$ from Urban100. Every image is downsampled with MATLAB's IMRESIZE, which is the procedure executed for training each CNN, and the resulting LR image is fed into the network. We chose a scaling factor of 4.

Results: Table II shows the results for a subset of methods in Table I. In the 3rd column, it displays the ℓ_2-norm of the difference between the downsampled HR outputs, i.e., Aw, and the input LR image b; in the 4th column, it shows the same quantity after feeding the corresponding w (and b, cf. Fig. 2) to our method. It can be seen that our post-processing improves consistency by 6 orders of magnitude. Note that even though SRMD models various degradations without retraining, it still fails to ensure consistency. IRCNN is a plug-and-play method and, as a result, can also handle different degradation models. Although it achieves better consistency than pure CNN-based methods, it is still 5 orders of magnitude below our scheme. The last row of Table II shows the consistency of DeepRED [43].

[3] https://github.com/jbhuang0604/SelfExSR

TABLE V
AVERAGE PSNR (SSIM) RESULTS IN dB AND EXECUTION TIME IN SECONDS OF OUR METHOD USING THE REFERENCE METHODS

Dataset	**Scale**	**TVAL3** [8]	SRCNN [15]	***Ours***	**Time**	FSRCNN [26]	***Ours***	**Time**
Set5	×2	34.0315 (0.9354)	36.2772 (0.9509)	**36.5126** (**0.9535**)	18.23	36.9912 (0.9556)	**37.0368** (**0.9559**)	8.97
	×4	29.1708 (0.8349)	30.0765 (0.8525)	**30.2460** (**0.8583**)	13.93	30.7122 (0.8658)	**30.7886** (**0.8686**)	9.86
Set14	×2	31.0033 (0.8871)	32.1245 (0.9028)	**32.2793** (**0.9055**)	11.57	32.6516 (0.9089)	**32.6880** (**0.9091**)	8.92
	×4	26.6742 (0.7278)	27.1808 (0.7410)	**27.2952** (**0.7476**)	8.21	27.6179 (0.7550)	**27.6794** (**0.7569**)	9.04
BSD100	×2	30.1373 (0.8671)	31.1087 (0.8835)	**31.2148** (**0.8864**)	6.79	31.5075 (0.8905)	**31.5229** (**0.8907**)	4.21
	×4	26.3402 (0.6900)	26.7027 (0.7018)	**26.7793** (**0.7082**)	5.30	26.9675 (0.7130)	**26.9966** (**0.7146**)	4.64
Urban100	×2	27.5143 (0.8728)	28.6505 (0.8909)	**28.8219** (**0.8935**)	25.34	29.8734 (0.9010)	**29.8926** (**0.9013**)	27.75
	×4	23.7529 (0.6977)	24.1443 (0.7047)	**24.2308** (**0.7110**)	24.12	24.6196 (0.7270)	**24.6522** (**0.7291**)	25.08

Dataset	**Scale**	**TVAL3** [8]	DRCN [35]	***Ours***	**Time**	VDSR [17]	***Ours***	**Time**
Set5	×2	34.0315 (0.9354)	37.6279 (0.9588)	**37.6697** (**0.9591**)	20.40	37.5295 (**0.9587**)	**37.5397** (0.9585)	65.91
	×4	29.1708 (0.8349)	31.5344 (0.8854)	**31.5660** (**0.8857**)	14.25	31.3485 (**0.8838**)	**31.3696** (0.8836)	42.38
Set14	×2	31.0033 (0.8871)	33.0585 (0.9121)	**33.1033** (**0.9129)**	8.92	33.0527 (0.9127)	**33.0906** (**0.9128**)	20.05
	×4	26.6742 (0.7278)	28.0269 (0.7673)	**28.0551** (**0.7679**)	8.57	28.0152 (0.7678)	**28.0375** (**0.7679**)	19.25
BSD100	×2	30.1373 (0.8671)	31.8536 (0.8942)	**31.8722** (**0.8952**)	26.23	31.9078 (**0.8960**)	**31.9071** (**0.8960**)	106.92
	×4	26.3402 (0.6900)	27.2364 (0.7233)	**27.2491** (**0.7239**)	24.23	26.8776 (0.7093)	**27.2342** (**0.7228**)	109.86

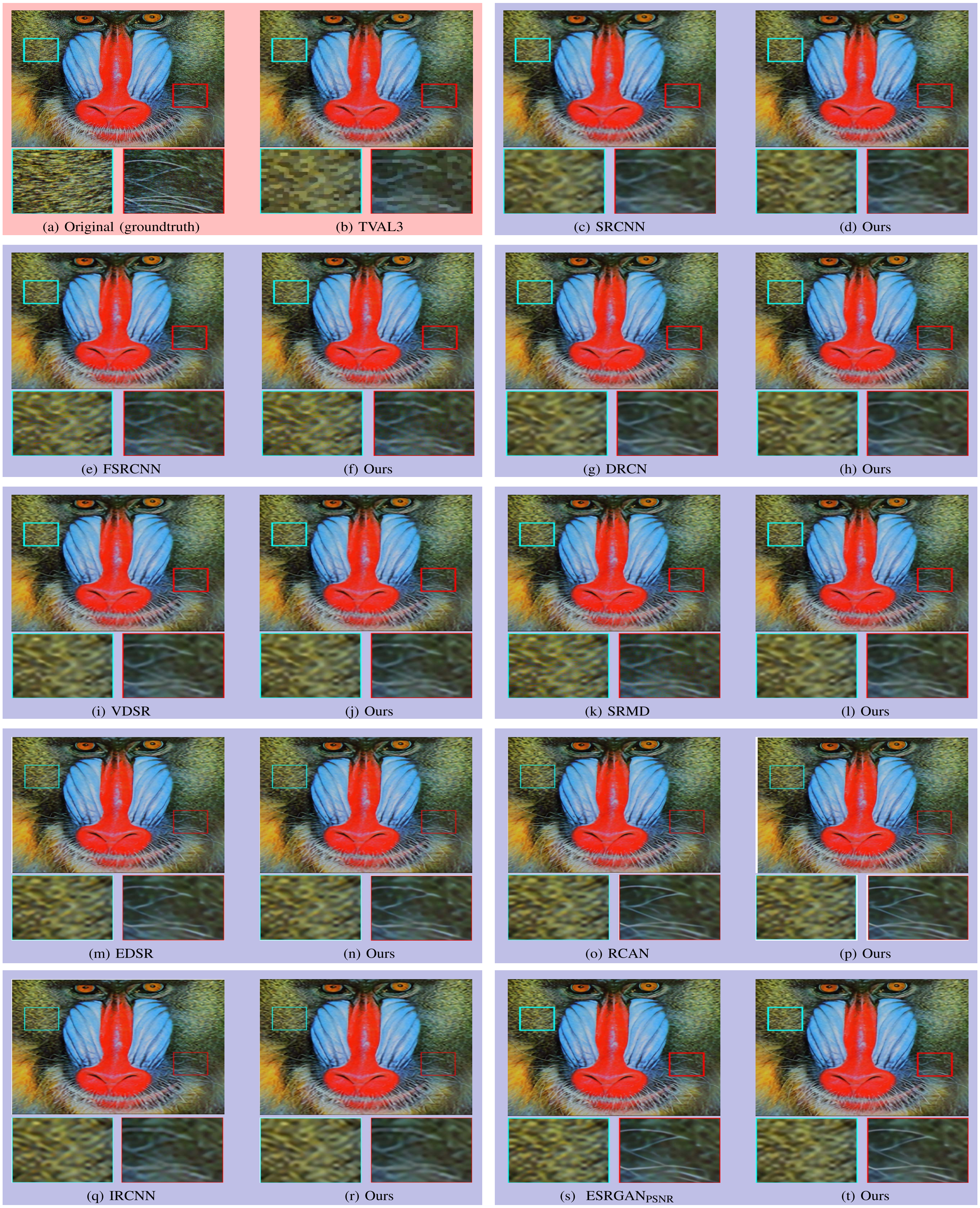
Dataset	**Scale**	**TVAL3** [8]	LapSRN [27]	***Ours***	**Time**	SRMD [21]	***Ours***	**Time**
Set5	×2	34.0315 (0.9354)	37.7008 (0.9590)	**37.7132** (**0.9592**)	52.80	37.4496 (0.9579)	**37.5859** (**0.9588**)	51.96
	×4	29.1708 (0.8349)	31.7181 (0.8891)	**31.7428** (**0.8894**)	49.45	31.5750 (0.8853)	**31.6695** (**0.8869**)	37.60
	×8	21.7360 (0.6360)	26.3314 (**0.7548**)	**26.3876** (0.7544)	42.50			
Set14	×2	31.0033 (0.8871)	33.2518 (0.9138)	**33.2618** (**0.9141**)	53.57	32.9460 (0.9126)	**33.2139** (**0.9140**)	51.43
	×4	26.6742 (0.7278)	28.2533 (0.7730)	**28.2682** (**0.7731**)	36.43	28.0833 (0.7721)	**28.2229** (**0.7725**)	36.42
	×8	20.8616 (0.5676)	24.5643 (**0.6266**)	**24.5946** (**0.6266**)	24.54			
BSD100	×2	30.1373 (0.8671)	32.0214 (0.8970)	**32.0281** (**0.8975**)	19.56	31.8722 (0.8953)	**31.9032** (**0.8960**)	23.71
	×4	26.3402 (0.6900)	27.4164 (0.7296)	**27.4298** (**0.7298**)	18.12	27.3350 (0.7273)	**27.3608** (**0.7283**)	20.94
	×8	20.1400 (0.5575)	24.6495 (**0.5887**)	**24.6776** (**0.5887**)	19.32			
Urban100	×2	27.9935 (0.8742)	31.1319 (0.9180)	**31.1464** (**0.9183**)	115.23	30.8799 (0.9146)	**30.9314** (**0.9153**)	108.32
	×4	23.7529 (0.6977)	25.5026 (**0.7661**)	**25.5152** (**0.7661**)	104.23	25.3494 (0.7605)	**25.3889** (**0.7614**)	102.15
	×8	18.8295 (0.5436)	22.0547 (**0.5956**)	**22.0788** (0.5950)	103.29			

Dataset	**Scale**	**TVAL3** [8]	EDSR [24]	***Ours***	**Time**	RCAN [25]	***Ours***	**Time**
Set5	×2	34.0315 (0.9354)	37.9022 (0.9594)	**37.9198** (**0.9597**)	61.84	38.1819 (0.9604)	**38.2121** (**0.9608**)	58.22
	×4	29.1708 (0.8349)	32.0726 (0.8927)	**32.0968** (**0.8931**)	43.10	32.6003 (0.8991)	**32.6137** (**0.8992**)	38.51
	×8	21.7360 (0.6360)				27.2985 (**0.7866**)	**27.3264** (0.7864)	41.36
Set14	×2	31.0033 (0.8871)	33.4433 (0.9162)	**33.4862** (**0.9167**)	52.29	33.9896 (0.9203)	**34.0168** (**0.9207**)	32.15
	×4	26.6742 (0.7278)	28.4719 (0.7790)	**28.4949** (**0.7797**)	36.43	28.7596 (0.7866)	**28.7898** (**0.7869**)	18.64
	×8	20.8616 (0.5676)				25.1276 (**0.6479**)	**25.1697** (0.6474)	22.31
BSD100	×2	27.9935 (0.8742)	32.1323 (0.8986)	**32.1423** (**0.8989**)	21.34	32.3825(0.9019)	**32.3912** (**0.9022**)	21.67
	×4	26.3402 (0.6900)	27.5479 (0.7349)	**27.5579** (**0.7354**)	16.02	27.7547 (0.7428)	**27.7766**(**0.7429**)	19.65
	×8	20.1400 (0.5575)				24.9745 (**0.6050**)	**24.9982** (0.6047)	18.58
Urban100	×2	27.9935 (0.8742)	32.6128 (0.9152)	**32.6248** (**0.9153**)	105.23	33.0248 (0.9327)	**33.0488** (**0.9328**)	106.54
	×4	26.3402 (0.6900)	26.0311 (0.7841)	**26.0434** (**0.7842**)	102.94	26.8132 (0.8075)	**26.8251** (**0.8080**)	107.34
	×8	18.8295 (0.5436)				23.0010 (**0.6445**)	**23.0183** (0.6438)	108.26

Dataset	**Scale**	**TVAL3** [8]	IRCNN [28]	***Ours***	**Time**	ESRGAN$_{\text{PSNR}}$ [16]	***Ours***	**Time**
Set5	×2	34.0315 (0.9354)	37.3436 (0.9572)	**37.3712** (**0.9576**)	52.40			
	×4	29.1708 (0.8349)	30.9995 (0.8778)	**31.0082** (**0.8780**)	37.20	32.7072 (0.9001)	**32.7227** (**0.9002**)	37.33
Set14	×2	31.0033 (0.8871)	32.8573 (0.9105)	**32.8947** (**0.9109**)	51.43			
	×4	26.6742 (0.7278)	27.7195 (0.7614)	**27.7454** (**0.7617**)	37.50	28.8920 (0.7893)	**28.9151** (**0.7895**)	36.86
BSD100	×2	31.0033 (0.8871)	31.6543 (0.8918)	**31.6745** (**0.8923**)	25.04			
	×4	26.3402 (0.6900)	27.0848 (0.7188)	**27.0920** (**0.7191**)	20.79	27.8332 (0.7447)	**27.8531** (**0.7452**)	22.14
Urban100	×2	31.0033 (0.8871)	30.0623 (0.9105)	**30.0899** (**0.9108**)	101.32			
	×4	23.7529 (0.6977)	24.8913 (0.7395)	**24.9041** (**0.7396**)	101.27	27.0270 (**0.8146**)	**27.0308** (0.8142)	108.86

C. Robustness to Operator Mismatch

As previously stated, most SR CNNs are trained by downsampling a HR into a LR image using the bicubic operator. If, during testing, A is different from the bicubic operator then, as we will see, there can be a serious drop in performance. This may indeed limit the applicability of CNNs in real-life scenarios where the required time and computation resources might not be available. Our approach, however, mitigates this effect and adds robustness to the SR task. We considered the same images and methods as in Table II, with DeepRed replaced by TVAL3, and considered the operators for A described in Section III-A: bicubic, box averaging, and simple subsampling.



(a) Original (groundtruth) (b) TVAL3 (c) SRCNN (d) Ours

(e) FSRCNN (f) Ours (g) DRCN (h) Ours

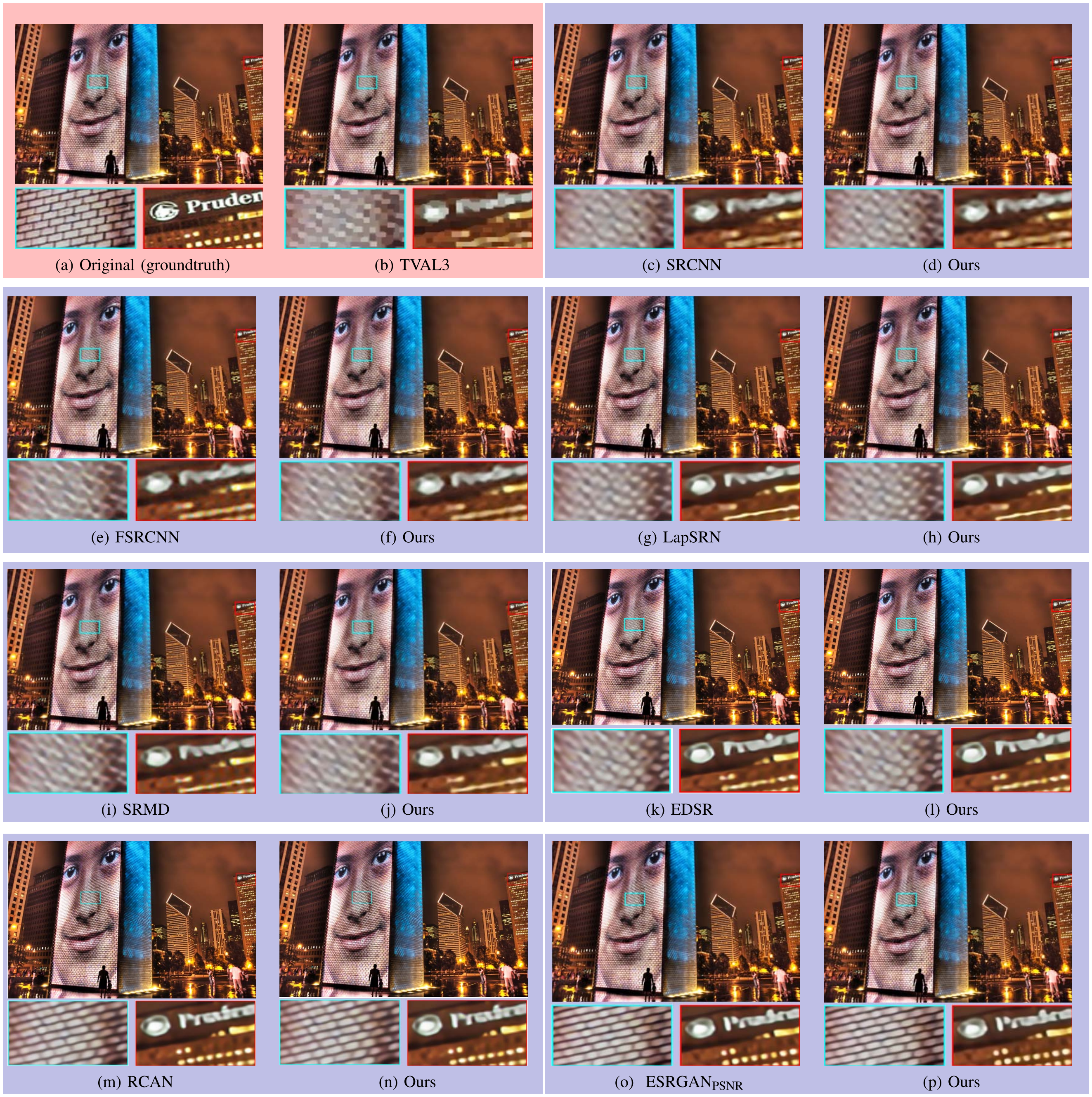
(i) VDSR (j) Ours (k) SRMD (l) Ours

(m) EDSR (n) Ours (o) RCAN (p) Ours

(q) IRCNN (r) Ours (s) $\text{ESRGAN}_{\text{PSNR}}$ (t) Ours

Fig. 4. Results on *Baboon* (Set14) for 4×. Each shaded area (except the top-left) shows the output of a learning-based algorithm and of our method.

Results: Each shaded box in Table III shows, for each subsampling operator, the PSNR values obtained by a given method, and by subsequently processing its output with our scheme. While all methods perform the best under bicubic subsampling, there is a performance drop for box filtering, and an even larger drop for simple subsampling. Note that our method systematically improves the output of all the networks, even for bicubic subsampling. And while the improvement is of less than 1dB for bicubic subsampling, it averages around 2dBs for simple subsampling. Indeed, the performance of the



(a) Original (groundtruth) (b) TVAL3 (c) SRCNN (d) Ours

(e) FSRCNN (f) Ours (g) LapSRN (h) Ours

(i) SRMD (j) Ours (k) EDSR (l) Ours

(m) RCAN (n) Ours (o) ESRGAN$_{\text{PSNR}}$ (p) Ours

Fig. 5. Results on *img076* (Urban100) for 4×. Each shaded area (except the top-left) shows the output of a learning-based algorithm and of our method.

CNNs for this case drops so much that there is a large margin for improvement. Interestingly, TVAL3, which solves (3) with $\beta = 0$, is the worst method for bicubic subsampling, but approaches the performance of CNNs for box averaging and, besides ours, becomes the best for simple subsampling. Hence, this illustrates that reconstruction-based methods can be more robust and adaptable than CNN architectures.

D. Standard Datasets With Bicubic Downsampling

We also conducted more systematic experiments using the standard datasets Set5, Set14, BSD100, and Urban100, under different scaling factors and using bicubic downsampling only.

1) Quantitative Results: Table V displays the average PSNR and SSIM, as well as the average execution time of our method (in seconds), for 2×, and 4× scaling factors. Each shaded area shows the performance of a given (learning-based) reference method, the performance of our scheme applied to the output of that reference method, and the average execution time (of our method). For easy comparison, the values for TVAL3 occur repeatedly in different vertical sub-blocks of the table. Note that since ESRGAN$_{\text{PSNR}}$ was designed specifically for 4× upsampling, we do not present its values for other scaling factors. Most results for our method were generated with $\beta = 1$ and in some instances with $\beta = 2$ in (3).

An obvious pattern in the table is that our method consistently improves the outputs of all the methods in terms of PSNR and SSIM, except in a small subset of cases. The improvements range between 0.0038 and 0.3566 dB. One of the exceptions occurs for 4× upsampling with ESRGAN$_{\text{PSNR}}$. In that case, ESRGAN$_{\text{PSNR}}$ has always better SSIM than our method, even though the opposite happens for the PSNR. As expected, TVAL3 had the worst performance overall and was surpassed by all learning-based methods.

A drawback of our method, however, is its possibly long execution time. We recall that of all downsampling operators mentioned in Section III-A, the most computationally complex is bicubic downsampling, as considered in these experiments. The timing values in Table V are average values: they report the total execution time of our algorithm over all the images of the corresponding dataset divided by the number of images. While in some cases our algorithm took an average of 4 sec (FSRCNN, BSD100, 2×), in others it took more than 108 sec (SRMD, Urban100, 2×). In fact, for the Urban100 dataset, because of the large size of its images (1024 × 644), we had to reduce the number of simultaneous threads to prevent the GPUs from overflowing. For reference, for (SRCNN, BSD100, 4×), our method takes an average of 9 sec when we use simple subsampling. This is roughly half the execution time it takes for bicubic downsampling.

2) Qualitative Results: Figures 4 and 5 depict the output images of all the algorithms (except IRCNN) for the test images *baboon* from Set14, and *img067* from Urban100. All super-resolved images exhibit blur and loss of information compared with the GT images in Figs. 4a-5a. And as our scheme builds upon the outputs of other methods, it also inherits some of their artifacts. It is difficult to visually assess differences between the outputs of the algorithms and of our method, in part because the improvements, as measured by the PSNR, are relatively small. Yet, as our experiments show, our scheme not only systematically improves the outputs of CNN-based methods, but also adds significant robustness to operator mismatch.

3) Face Hallucination: We also perform a small experiment for face SR. Conventional face SR networks such as [63] work on the same principles of the methods previously considered: they obtain an HR image from a LR image. Recent methods use generative adversarial networks (GANs) which are able to produce photo-realistic images and also allow the use of large downscaling factors. The work in [62] proposes the PULSE algorithm, which produces sharp faces mapping to the correct LR input. In contrast to FSRNet, this network aims to lower the Naturalness Image Quality Evaluator (NIQE) score rather than improve the PSNR. Figure 3 shows the different outputs obtained on a sample image from CelebA-HQ [64] for a scaling factor of 8, while Table IV shows their respective PSNR and NIQE score. These results show that FSRNET and TV-TV minimization obtain better PSNR scores, but PULSE is able to obtain impressive NIQE scores, i.e., more realistic outputs.

V. Conclusion

We proposed a framework for single-image SR that blends model- and learning-based (e.g., CNN) techniques. As a result, our framework enables solving the consistency problem that CNNs suffer from, namely that downsampled output (HR) images fail to match the input (LR) images. Our experiments show that enforcing such consistency not only systematically improves the quality of the output images of CNNs, but also adds robustness to the SR task. At the core of our framework is a problem that we call TV-TV minimization and which we solve with an ADMM-based algorithm. Possible lines of future research include designing loss functions that enforce consistency during training and unrolling the proposed algorithm with a neural network.

References

[1] M. Vella and J. F. C. Mota, "Single image super-resolution via CNN architectures and TV-TV minimization," in *Proc. BMVC*, 2019, pp. 1–12, doi: 10.5244/C.33.219.

[2] M. Vella and J. F. C. Mota, "Robust super-resolution via deep learning and TV priors," in *Proc. SPARS*, 2019, pp. 1–3.

[3] T. Blu, P. Thévenaz, and M. Unser, "Linear interpolation revitalized," *IEEE Trans. Image Process.*, vol. 13, no. 5, pp. 710–719, May 2004.

[4] R. G. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-29, no. 6, pp. 1153–1160, Dec. 1981.

[5] R. Fattal, "Image upsampling via imposed edge statistics," *ACM Trans. Graph.*, vol. 26, no. 3, p. 95, Jul. 2007.

[6] S. Osher and L. I. Rudin, "Feature-oriented image enhancement using shock filters," *SIAM J. Numer. Anal.*, vol. 27, no. 4, pp. 919–940, 1990.

[7] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D, Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, 1992.

[8] C. Li, W. Win, H. Jing, and Y. Zhang, "An efficient augmented Lagrangian method with applications to total variation minimization," *Comput. Optim. Appl.*, vol. 56, no. 3, pp. 507–530, Dec. 2013.

[9] S. Sun, D. Xue, and D. Chen, "Image magnification using fractional order level set reconstruction," in *Proc. 25th Chin. Control Decis. Conf. (CCDC)*, May 2013, pp. 333–334.

[10] H. A. Aly and E. Dubois, "Image up-sampling using total-variation regularization with a new observation model," *IEEE Trans. Image Process.*, vol. 14, no. 10, pp. 1647–1659, Oct. 2005.

[11] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, no. 1, pp. 89–97, 2004.

[12] S. Becker, J. Bobin, and E. J. Candès, "NESTA: A fast and accurate first-order method for sparse recovery," *SIAM J. Imag. Sci.*, vol. 4, no. 1, pp. 1–39, 2011.

[13] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[14] A. Singh, F. Porikli, and N. Ahuja, "Super-resolving noisy images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2846–2853.

[15] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. ECCV*, 2014, pp. 184–199.

[16] X. Wang *et al.*, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Proc. ECCVW*, Sep. 2018, pp. 1–16.

[17] J. Kim, J. Lee, and K. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. CVPR*, 2016, pp. 1646–1654.

[18] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, Jun. 2009, pp. 248–255.

[19] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1664–1673.

[20] B. Ghojogh, F. Karray, and M. Crowley, "Backprojection for training feedforward neural networks in the input and feature spaces," in *Proc. ICIAR*, 2020, pp. 16–24.

[21] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3262–3271.

[22] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," *IEEE Trans. Signal Process.*, vol. 4, no. 7, pp. 932–946, Jul. 1995.

[23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[24] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 136–144.

[25] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proc. ECCV*, Sep. 2018, pp. 286–301.

[26] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *Proc. ECCV*, 2016, pp. 391–407.

[27] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep Laplacian pyramid networks for fast and accurate super-resolution," in *Proc. CVPR*, Jul. 2017, pp. 624–632.

[28] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3929–3938.

[29] S. Mallat and G. Yu, "Super-resolution with sparse mixing estimators," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2889–2900, Nov. 2010.

[30] W. Zhang and W.-K. Cham, "Hallucinating face in the DCT domain," *IEEE Trans. Image Process.*, vol. 20, no. 10, pp. 2769–2779, Oct. 2011.

[31] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 349–356.

[32] L. Condat, "Discrete total variation: New definition and minimization," *SIAM J. Imag. Sci.*, vol. 10, no. 3, pp. 1258–1290, 2017.

[33] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM J. Imag. Sci.*, vol. 2, no. 2, pp. 323–343, 2009.

[34] M. Irani and S. Peleg, "Improving resolution by image registration," *CVGIP, Graph. Models Image Process.*, vol. 53, no. 3, pp. 231–239, May 1991.

[35] J. Kim, J. Lee, and K. Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proc. CVPR*, 2016, pp. 1637–1645.

[36] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. CVPR*, 2017, pp. 105–114.

[37] W. Shi *et al.*, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. CVPR*, 2016, pp. 1874–1883.

[38] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proc. CVPR*, 2018, pp. 1828–1837.

[39] 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. (CVPR)*, Jun. 2016, pp. 449–458.

[40] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," in *Proc. ICML*, 2017, pp. 537–546.

[41] J. H. R. Chang, C. Li, B. Póczos, B. V. K. V. Kumar, and A. C. Sankaranarayanan, "One network to solve them all? solving linear inverse problems using deep projection models," in *Proc. ICCV*, 2017, pp. 5889–5898.

[42] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9446–9454.

[43] G. Mataev, P. Milanfar, and M. Elad, "DeepRED: Deep image prior powered by RED," in *Proc. ICCVW*, Oct. 2019, pp. 1–10.

[44] F. Shi, J. Cheng, L. Wang, P.-T. Yap, and D. Shen, "LRTV: MR image super-resolution with low-rank and total variation regularizations," *IEEE Trans. Med. Imag.*, vol. 34, no. 12, pp. 2459–2466, Dec. 2015.

[45] J. Li, J. Wu, H. Deng, and J. Liu, "A self-learning image super-resolution method via sparse representation and non-local similarity," *Neurocomputing*, vol. 184, pp. 196–206, Apr. 2016.

[46] G. Peyré, S. Bougleux, and L. Cohen, "Non-local regularization of inverse problems," in *Proc. ECCV*, 2008, pp. 57–68.

[47] 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.

[48] J. F. C. Mota, N. Deligiannis, and M. R. D. Rodrigues, "Compressed sensing with prior information: Strategies, geometry, and bounds," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4472–4496, Jul. 2017.

[49] L. Weizman, Y. C. Eldar, and D. Ben-Bashat, "Reference-based MRI," *Med. Phys.*, vol. 43, no. 10, pp. 5357–5369, Oct. 2016.

[50] J. F. C. Mota, N. Deligiannis, A. C. Sankaranarayanan, V. Cevher, and M. R. D. Rodrigues, "Adaptive-rate reconstruction of time-varying signals with application in compressive foreground extraction," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3651–3666, Jul. 2016.

[51] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "A proof of convergence for the alternating direction method of multipliers applied to polyhedral-constrained functions," 2011, *arXiv:1112.2295*. [Online]. Available: https://arxiv.org/abs/1112.2295

[52] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L.-A. Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *Proc. Brit. Mach. Vis. Conf.*, 2012, p. 135.

[53] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Curves and Surfaces*. Springer, 2012, pp. 711–730.

[54] 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. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2, Jul. 2001, pp. 416–423.

[55] J. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. CVPR*, 2015, pp. 5197–5206.

[56] P. Arbeláez, 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.

[57] 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 (CVPRW)*, Jul. 2017, pp. 1122–1131.

[58] K. Ma *et al.*, "Waterloo exploration database: New challenges for image quality assessment models," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 1004–1016, Feb. 2017.

[59] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2472–2481.

[60] X. Wang, K. Yu, C. Dong, and C. C. Loy, "Recovering realistic texture in image super-resolution by deep spatial feature transform," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 606–615.

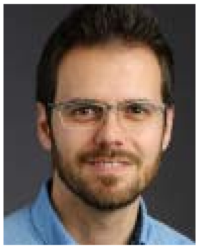
[61] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[62] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "PULSE: Self-supervised photo upsampling via latent space exploration of generative models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2437–2445.

[63] Y. Chen, Y. Tai, X. Liu, C. Shen, and J. Yang, "FSRNet: End-to-end learning face super-resolution with facial priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2492–2501.

[64] T. Karrass, T. Aila, S. Laine, N. Ravi, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability and variation," in *Proc. ICLR*, 2018, pp. 1–26.

Marija Vella received the B.Eng. degree from the University of Malta in 2017 and the M.S. degree in quantitative finance and mathematics from Heriot-Watt University, Edinburgh, in 2018, where she is currently pursuing the Ph.D. degree in electrical engineering. Her research focuses on developing new techniques utilizing optimization and machine learning methods for more reliable computational imaging.



João F. C. Mota received the M.Sc. and Ph.D. degrees in electrical and computer engineering from the Technical University of Lisbon in 2008 and 2013, respectively, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University in 2013. He is currently an Assistant Professor of signal and image processing with Heriot-Watt University, Edinburgh. His research interests include theoretical and practical aspects of high-dimensional data processing, inverse problems, optimization theory, machine learning, data science, and distributed information processing and control. He was a recipient of the 2015 IEEE Signal Processing Society Young Author Best Paper Award.