Learning Dual Priors for JPEG Compression Artifacts Removal

Xueyang Fu[1], Xi Wang[1], Aiping Liu[1], Junwei Han[2], Zheng-Jun Zha[1*]
[1]University of Science and Technology of China, China
[2]Northwestern Polytechnical University, China
{xyfu, aipingl, zhazj}@ustc.edu.cn, wangxxi@mail.ustc.edu.cn, jhan@nwpu.edu.cn

Abstract

Deep learning (DL)-based methods have achieved great success in solving the ill-posed JPEG compression artifacts removal problem. However, as most DL architectures are designed to directly learn pixel-level mapping relationships, they largely ignore semantic-level information and lack sufficient interpretability. To address the above issues, in this work, we propose an interpretable deep network to learn both pixel-level regressive prior and semantic-level discriminative prior. Specifically, we design a variational model to formulate the image de-blocking problem and propose two prior terms for the image content and gradient, respectively. The content-relevant prior is formulated as a DL-based image-to-image regressor to perform as a de-blocker from the pixel-level. The gradient-relevant prior serves as a DL-based classifier to distinguish whether the image is compressed from the semantic-level. To effectively solve the variational model, we design an alternating minimization algorithm and unfold it into a deep network architecture. In this way, not only the interpretability of the deep network is increased, but also the dual priors can be well estimated from training samples. By integrating the two priors into a single framework, the image de-blocking problem can be well-constrained, leading to a better performance. Experiments on benchmarks and real-world use cases demonstrate the superiority of our method to the existing state-of-the-art approaches.

1. Introduction

With the rapid development of consumer devices (*e.g.*, digital cameras and smartphones) and wireless network, the number of images and videos has achieved explosive growth, which has brought more pressure and challenges to storage and transmission systems. To save the storage capacity and transmission bandwidth, captured images and videos are usually compressed to reduce information redundancy. Lossy compression algorithms, *e.g.*, Joint Photographic Experts Group (JPEG) [43] and High Efficiency Video Coding (HEVC) [41], have been widely explored to achieve this goal. However, due to the inevitable signal loss during compression, these compression algorithms usually generate visually unpleasing compression artifacts. These artifacts not only decrease the visual quality, but also degrade the performance of downstream computer vision systems, especially at high compression ratios. Therefore, removing compression artifacts is an important post-processing task and has attracted more attention in recent years [20,30]. We refer the reader to review articles [28,30] for more details. In this paper, we focus on alleviating still image degradation caused by JPEG compression, which is one of the most prevalent compression standards.

JPEG compression first applies the discrete cosine transformation (DCT) on 8×8 pixel blocks. Then, these DCT coefficients are coarsely quantized to remove high-frequency details to save space. Due to the independent processing on each pixel block and the removal of high-frequency details, compressed images usually suffer from blocking and blurring artifacts. In addition, using a large quantization step, banding artifacts will appear in smooth areas. Some recent studies have proposed methods to remove undesirable JPEG compression artifacts. According to the design mechanism, these methods can be roughly classified into two categories: model-based methods and deep learning (DL)-based methods. Early model-based works perform filtering to remove compression artifacts. For instance, Foi *et al.* [12] propose a shape-adaptive DCT filtering method for compression artifacts reduction. On the other hand, since multiple latent clear versions can be estimated from a single compressed image, this task is essentially an ill-posed inverse problem, which requires prior knowledge to constrain it. Along this research direction, many researchers formulate this problem as a minimization of a variational model with favor-

*Corresponding author: Zheng-Jun Zha. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0105702; in part by the National Natural Science Foundation of China (NSFC) under Grants U19B2038 and 61901433; in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2019-025; and in part by the USTC Research Funds of the Double First-Class Initiative under Grant YD2100002003.

able prior terms. Based on the maximum a posterior (MAP) framework, many prior models, *e.g.*, quantization step [55], sparse representation [3] and low rank [59], have been developed. Although these model-driven methods have shown good performance, the representation abilities of handcrafted priors are limited, which leads to unstable results when processing compressed images with complex structures.

In the past few years, DL-based methods have achieved significant progress of JEPG artifacts removal [8, 57, 63]. Due to the powerful nonlinear capacity [16, 21, 22, 29, 53] and huge amounts of training data, these methods can learn the inverse mapping of compression degradations, and thus produce better results than model-driven methods. However, most of current DL-based methods adopt feed-forward networks to directly predict clear images, making them like black boxes and lack interpretability. In addition, since these DL-based methods only learn pixel-level mappings, semantic-level information is not fully explored and exploited, which further limits their performance improvement.

Different from these methods, we propose an interpretable deep network by combining advantages of both the model-based methods and data-driven DL models. Specifically, we introduce an effective algorithm with DL to learn a pixel-level regressive prior for image content and a semantic-level discriminative prior for image gradient, respectively. First, we model the content-relevant prior as an image-to-image regressor to perform de-blocking, and design the gradient-relevant prior as binary classifier to distinguish whether the image is compressed. Then, the image de-blocking problem is formulated as a minimization of a variational model with the two proposed priors. To effectively solve the model, we design an alternating minimization scheme based on the gradient descent technique and half-quadratic splitting method. Finally, the iterative algorithm is unfolded into a deep network architecture, in which the two priors can be automatically learned through an effective network training strategy. We show that our method is able to predict visually pleasing de-blocked images while removing undesirable JPEG artifacts sufficiently. The contributions of this work are as follows:

- We propose two effective priors to describe the image content and image gradient from the pixel-level and semantic-level, respectively. By using the two priors as the regularizer, we introduce a new variational model for the JPEG compression artifacts removal.

- We propose an alternating minimization algorithm, which is based on the gradient descent technique and half-quadratic splitting method, to solve the variational model. By unfolding the algorithm, we design a new deep network architecture for the image de-blocking problem. In this way, the two proposed priors can be automatically estimated from training samples. In addition, since the feed-forward process mimics the processing flow of the alternating minimization algorithm, the interpretability of the deep model is increased.

- We collect a new dataset containing compressed/clear image pairs based on the popular online social software *WeChat*. This dataset aims to complement the existing *Twitter* dataset [8] to serve the relevant research communities. Extensive experiments show that our proposed network performs favorably against state-of-the-arts on both benchmarks and real-world use cases.

2. Related work

2.1. Model-based methods

In early studies, image filtering technologies are widely explored to explicitly remove compression artifacts. For instance, a quadratic programming technology with an adaptive is proposed in [36] to remove blocking artifacts and preserve image details. An adaptive neighborhood for smoothing and conducting post-filtering in shifted windows is introduced in method [54]. Yoo *et al*. [49] utilize group-based filtering to improve the correlation between image blocks and reduce blocking artifacts. Foi *et al*. [12] achieve both the image de-noising and de-blocking by conducting filtering in the shape-adaptive DCT domain.

On the other hand, prior knowledge also plays a vital role in this task since it is an ill-posed inverse problem. Many researchers make great effort to explore effective priors to constrain the solution space. As one of the most important priors, the quantization step can be utilized to estimate the range of the DCT coefficients of the clear image to constrain the de-blocked result [35, 55]. Other image priors, *e.g.*, sparse representation [3, 4, 40, 52], low rank [59], non-local self-similarity [60] and graph [34], are also explored and exploited. Li *et al*. [27] combine image decomposition algorithm and sparse prior to achieve both JPEG artifacts removal and image enhancement. Liu *et al*. [35] improve image de-blocking by exploiting the sparsity in both the image and DCT domains. In [37], a graph-based low-rank prior is introduced to reflect the manifold structures of image patches. Zhang et al. [59] reduce compression artifacts by exploring the non-local similarity in the DCT domain. Liu *et al*. [34] propose a graph smoothness prior to jointly reduce compression artifacts and improve contrast based on the Retinex theory. Although these model-based methods are flexible and have good interpretability, they usually have limited representation capabilities with handcrafted filters and priors.

2.2. Deep learning-based methods

In the past few years, deep learning has made breakthrough progress in JPEG compression artifacts removal.

Due to the powerful nonlinear representation ability, DL-based methods usually have better performance than model-based ones. Dong *et al*. [8] introduce the first DL-based method by designing a four-layer CNNs architecture, which makes a breakthrough progress in compression artifacts removal. Inspired by the residual learning, several deep network architectures are well designed for JPEG compression artifacts removal and relevant restoration tasks [18, 32, 42, 57, 62, 63]. Fan *et al*. [11] construct a decouple learning framework by incorporating different parameterized image operators for various image restoration tasks. To produce visually pleasing results, the generative adversarial networks (GANs) are utilized to capture the underlying data distribution to generate vivid image textures [14, 15, 20]. Yoo *et al*. [48] achieve image de-blocking by estimating frequency distribution of local patches. Kim *et al*. [23] propose a pseudo-blind de-blocking method, in which the quality factor is estimated for both blind and non-blind de-blocking. Many dual domain learning-based methods [5,10,19,61] are also introduced by considering the DCT-relevant prior.

Recently, many researchers attempt to combine both the domain knowledge and deep learning for various image restoration tasks. Wang *et al*. [45] build a dual domain network by using the DCT-pixel domain sparse coding and the learned iterative shrinkage thresholding algorithm. Chen *et al*. [6] design a deep network based on the classic iterative nonlinear reaction diffusion for effective and efficient image restoration. In [7, 13], the authors adopt the classic convolutional sparse coding to solve the image de-blocking. Yang *et al*. [46] use sparsely sampled measurements for image reconstruction by combining compressive sensing theory and deep learning. Under the alternative minimization framework, deep CNNs are also utilized to learn priors and perform as the regularizer [9,31,33,56,58]. Our method shares these similar spirits, but different from the above approaches that only learn pixel-level mapping relationships, we further introduce semantic-level information to better handle the JPEG compression artifacts removal.

3. Methodology

3.1. Motivation

In general, JPEG artifacts removal aims to obtain the clear image $\mathbf{x}$ from its compressed observation $\mathbf{y} = \mathbf{x} + \mathbf{v}$, where $\mathbf{v}$ contains compression artifacts and residual image content. Since this task is an ill-posed inverse problem, from a Bayesian perspective, the clear image can be obtained by solving a MAP problem:

$$\arg\max_{\mathbf{x}} \log p(\mathbf{y}\,|\mathbf{x}\,) + \log p(\mathbf{x}), \tag{1}$$

where $\log p(\mathbf{y}\,|\mathbf{x}\,)$ and $\log p(\mathbf{x})$ denote the data likelihood and the prior terms, respectively. Formally, by performing a negative logarithmic transformation, Equation (1) can be reformulated as an energy minimization variational model:

$$\arg\max_{\mathbf{x}} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda f(\mathbf{x}), \tag{2}$$

where $f(\mathbf{x})$ denotes the regularizer associated with the prior $\log p(\mathbf{x})$, and λ is a trade-off parameter. It is clear that the regularizer plays a vital role in obtaining high-quality solutions. In model-based optimization methods, many regularizers have been explored, *e.g*., low rank [59] and nonlocal self-similarity [60]. While these methods are usually time-consuming with handcrafted priors that are not powerful enough for good performance. Therefore, with the powerful nonlinear capabilities, deep unfolding networks [7,13] have been explored to extract the priors from training samples.

Albeit improvement in interpretability and performance, existing deep unfolding methods only design pixel-level regression networks for the regularization term, semantic-level information is not fully utilized. Affected by compression artifacts (*e.g*., blocking, blurring and banding), the overall quality of the JPEG compressed image will be significantly lower than its clear counterpart. In other words, it should be easy to distinguish whether the image has been compressed from the semantic-level. This observation motivates us to introduce a semantic-level discriminative prior to complement existing methods for image de-blocking.

In addition, according to our domain knowledge, most compression artifacts have a greater impact on the high-frequency part than the low-frequency part. As shown in Figure 1, due to the inherent sparsity of gradients, the histogram on the gradient domain has stronger regularity than the image domain. Moreover, the histogram of the compressed gradients $\nabla\mathbf{y}$ is much sparser than the clear gradients $\nabla\mathbf{x}$, as shown in Figure 1. This is because the quantization intervals are much larger in high-frequency part than those in low-frequency part, which results in significant changes in the high-frequency part before and after compression. Therefore, we argue that using the high-frequency part of the image can provide better discriminative information. Based on the above observation, we design a semantic-level prior and apply it to the image gradient, which is the most commonly used high-frequency image information [38]. By adding the semantic-level discriminative prior to Equation (2), our final variational model is:

$$\arg\max_{\mathbf{x}} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda_1 f_p(\mathbf{x}) + \lambda_2 f_s(\nabla\mathbf{x}), \tag{3}$$

where ∇ is the differential operator. $f_p(\cdot)$ and $f_s(\cdot)$ mean the regularizers to deliver the pixel-level regressive prior and semantic-level discriminative prior, respectively.

3.2. Optimization

To construct a step-by-step corresponding deep unfolding network architecture for Equation (3), we first design an

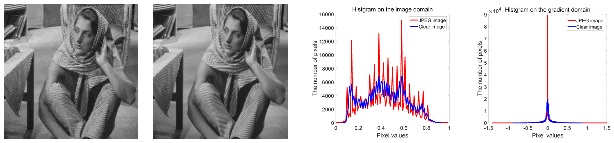


Figure 1: Statistical histogram. From left to right: $\mathbf{y}$; $\mathbf{x}$; histograms of $\mathbf{y}$ and $\mathbf{x}$; histograms of $\nabla\mathbf{y}$ and $\nabla\mathbf{x}$.

effective alternating minimization algorithm to obtain the unfolding inference. By introducing two auxiliary variables $\mathbf{u}$ and $\mathbf{g}$, Equation (3) can be rewritten as a non-constrained optimization problem:

$$\arg\max_{\mathbf{x},\mathbf{u},\mathbf{g}} \|\mathbf{y}-\mathbf{x}\|_2^2 + \alpha\|\mathbf{u}-\mathbf{x}\|_2^2 + \beta\|\mathbf{g}-\nabla\mathbf{x}\|_2^2 + \lambda_1 f_p(\mathbf{u}) + \lambda_2 f_s(\mathbf{g}), \tag{4}$$

where α and β are penalty parameters. Equation (4) can be addressed by alternatively solving sub-problems:

$$\arg\max_{\mathbf{u}} \alpha\|\mathbf{u}-\mathbf{x}_{k-1}\|_2^2 + \lambda_1 f_p(\mathbf{u}), \tag{5}$$

$$\arg\max_{\mathbf{g}} \beta\|\mathbf{g}-\nabla\mathbf{x}_{k-1}\|_2^2 + \lambda_2 f_s(\mathbf{g}), \tag{6}$$

$$\arg\max_{\mathbf{x}} \|\mathbf{y}-\mathbf{x}\|_2^2 + \alpha\|\mathbf{u}_k-\mathbf{x}\|_2^2 + \beta\|\mathbf{g}_k-\nabla\mathbf{x}\|_2^2, \tag{7}$$

where $k = 1, 2, ..., K$ is the iteration. Below, we detail the updates for each sub-problem.

1) **Solving u** The sub-problem (5) is a proximity operator of $f_p(\mathbf{u})$ and corresponds to de-block the image $\mathbf{x}_{k-1}$. The solution can be expressed as:

$$\mathbf{u}_k = deblocker(\mathbf{x}_{k-1}), \tag{8}$$

where $deblocker(\cdot)$ can be arbitrary image de-blocking algorithms. In this paper, we design a deep convolutional neural networks to perform the de-blocker. In this way, complicated image content-relevant priors can be directly learned from training data without manual design.

2) **Solving g** Since we want to introduce semantic-level discriminative information for image de-blocking, $f_s(\mathbf{g})$ is designed as a binary classifier since classification is the most fundamental semantic-related analysis. Therefore, unlike solving $\mathbf{u}$ that directly deploys an image-to-image regression network, we follow Li *et al.* [26] and adopt the back-propagation to compute the derivative of $f_s(\mathbf{g})$. The solution for solving $\mathbf{g}$ is:

$$\mathbf{g}_{d,k}^{(j)} = \mathbf{g}_{d,k}^{(j-1)} - \eta\left[\beta(\mathbf{g}_{d,k}^{(j-1)} - \nabla_d\mathbf{x}_{k-1}) + \lambda_2\frac{\partial f_s(\mathbf{g}_{d,k}^{(j-1)})}{\partial\mathbf{g}_{d,k}^{(j-1)}}\right], \tag{9}$$

where $d \in \{h, v\}$ are the horizontal and vertical directions, respectively. η is the step size, and j is the inner iteration. To fit with the overall energy minimization, we intentionally label clear images as 0 (negative) and compressed images as 1 (positive). In this way, during the optimization, Equation (9) provides semantic determining for $\mathbf{g}$ whether increasing or decreasing its value will improve the clarity, which complements the pixel-level constraints in Equation (8).

3) **Solving x** Since Equation (7) is a least squares problem, it has a closed form solution. To speed up the process, we adopt Fast Fourier Transformation (FFT) to diagonalize the differential operator so that large-matrix inversion can be avoided. By setting the first-order derivative to zero, the solution of Equation (7) is:

$$\mathbf{x}_k = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\mathbf{y}) + \alpha\mathscr{F}(\mathbf{u}_k) + \beta\left(\sum_{d\in\{h,v\}}\mathscr{F}^*(\nabla_d)\mathscr{F}(\mathbf{g}_k)\right)}{\mathscr{F}(\mathbf{I}) + \alpha\mathscr{F}(\mathbf{I}) + \beta\left(\sum_{d\in\{h,v\}}\mathscr{F}^*(\nabla_d)\mathscr{F}(\nabla_d)\right)}\right), \tag{10}$$

where $\mathbf{I}$ is the identity matrix, $\mathscr{F}$ is the FFT operator, $\mathscr{F}^*$ is the complex conjugate operator, $\mathscr{F}^{-1}$ is the inverse FFT operator, ∇_h and ∇_v are the horizontal and vertical differential operators, respectively. Since all calculations are performed pixel-wise, the update of $\mathbf{x}$ can be efficiently computed. However, if complex handcrafted de-blocker and classifier are used, the entire optimization process will be computationally expensive. Therefore, we unfold the above algorithm into a deep network to gain advantages from both model-based optimization and data-driven deep learning.

3.3. Deep unfolding network

As shown in Figure 2, our deep unfolding network contains K stages, which are intentionally designed to correspond to K iterations in the optimization algorithm. In each network stage, two auxiliary variables are first updated, and then the de-blocked image is calculated. Therefore, the question left to us now is how to design the pixel-level regularizer $f_p(\cdot)$ and the semantic-level regularizer $f_s(\cdot)$.

De-blocker $f_p(\cdot)$ To achieve an image-to-image regression for exploring pixel-level content-relevant prior, we first design a basic unit and then adopt it to construct the de-blocker. Due to different quantization steps, compression artifacts at different spatial scales will appear. Therefore, to capture both global and multi-scale local spatial information, we utilize the non-local operation [62] and dilated convolutions [50] to form the basic unit. Specifically, in each basic unit, we first deploy the non-local operation to capture global spatial information. Similar with [62], the non-local operation is performed as:

$$\mathbf{M}_{out} = \mathbf{M}_{in} + \theta(\mathbf{M}_{in})\nu(\mathbf{M}_{in})^\dagger\xi(\mathbf{M}_{in})\mathbf{W}, \tag{11}$$

where $\mathbf{M}_{in}$ and $\mathbf{M}_{out}$ are the input and output features; $\theta(\cdot)$, $\nu(\cdot)$ and $\xi(\cdot)$ are 1×1 convolutions to reduce the channel number; $\dagger$ is the transpose operation; $\mathbf{W}$ is a 1×1 convolution to perform a hidden-to-output operation. We re-ordered $(\theta\nu^\dagger)\xi$ to $\theta(\nu^\dagger\xi)$ according to the associative rule, which can greatly reduce computation complexity by

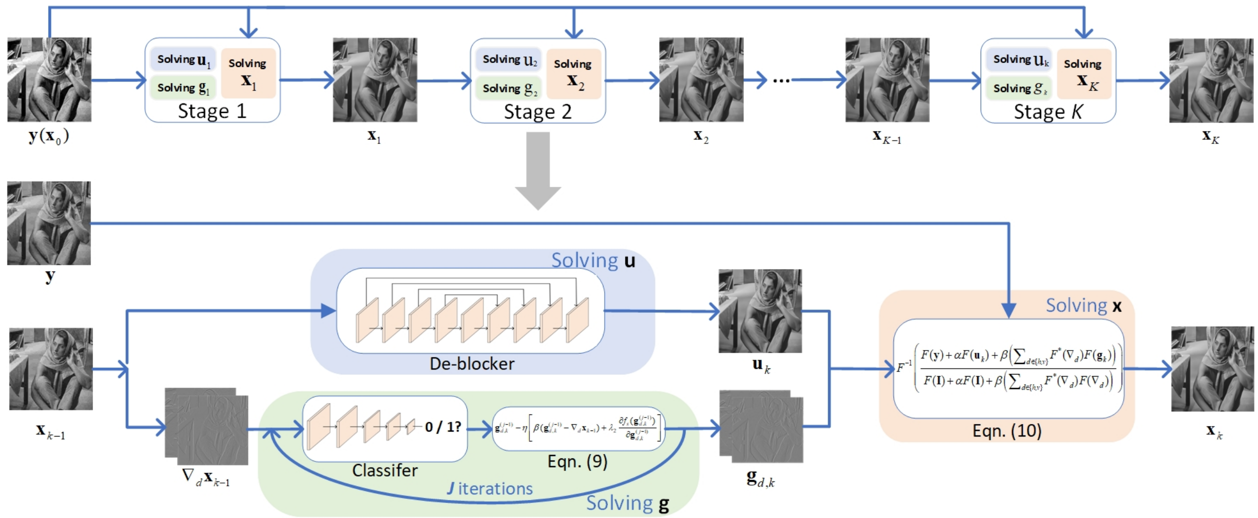


Figure 2: The framework of our network with K stages. Each stage consists of three operations to accomplish the update of $\mathbf{u}$, $\mathbf{g}$ and $\mathbf{x}$. The overall optimization process and illustration are shown on the top row and bottom row, respectively.

avoiding large matrix calculation. Then, these global spatial information is sent into three cascaded dilated convolutional layers with different dilation factors. In this way, the basic unit can capture wide-range spatial information and enable a single network to handle multiple quantization steps. Finally, we utilize the basic unit to construct the de-blocker. Note that we adopt dense structures [22] and skip-connections to avoid gradient vanishing and propagate image detail to improve the de-blocking performance.

Classifier $f_s(\cdot)$ To further utilize semantic information for image de-blocking, we build a DL-based binary classifier, which receives the image gradients as the input and outputs a single scalar, to represent the probability of being compressed. Specifically, we adopt six standard convolutional layers (with non-linear activations), one global average pooling layer, and two fully-connected layers to construct the classifier. The output scalar is processed by a sigmoid non-linear function to facilitate the binary prediction. Note that the global average pooling operation can convert a feature map into a single scalar, which allows our classifier to handle input gradients of arbitrary sizes. Moreover, since the entire classifier is differentiable, it can participate in the calculation of Equation (9) to update $\mathbf{g}$.

As shown in Figure 2, by plugging the de-blocker and classifier into the optimization, the deep unfolding network can be constructed. It should be indicated that our network has a good interpretability, *i.e.*, each network module corresponds to each step in the optimization. The de-blocker accomplishes the exploration of image content-relevant prior to achieve the function of Equation (8), which is to remove JPEG artifacts. The classifier accomplishes the exploration of image gradient-relevant prior and participates in the calculation of Equation (9) so that the gradients can be updated along the direction of 'clarity'. By taking both pixel-level and semantic-level information into consideration, the de-blocked image can be obtained by Equation (10).

3.4. Implementation details

Since our deep unfolding network contains a large number of parameters, it is impractical to manually determine these parameters. Therefore, we make all the learnable parameters be automatically learned from the training samples. We enforce the de-blocker and the classifier to share their own parameters to reduce the number of parameters and thus avoid over-fitting. For the weights in Equations (8) to (10), we let them be discriminatively learned.

In this paper, we adopt a two-stage training strategy to train our deep unfolding network. In the first stage, we only train the classifier via the binary cross entropy loss function:

$$\mathscr{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N} \hat{z}_i \log(z_i) + (1-\hat{z}_i)\log(1-z_i), \quad (12)$$

where N is the number of training samples, z_i is the output scalar of the classifier, and $\hat{z}_i$ is the label. We set $\hat{z}_i = 1$ for compressed images and $\hat{z}_i = 0$ for clear images. In the second stage, the parameters of the trained classifier are frozen, and the remaining parameters in the network are trained by using mean absolute error (MAE).

In our deep unfolding network, all convolutional kernel sizes are set as 3×3. The number of feature maps of de-blocker and classifier are 112 and 32, respectively. We set $K = 5$ and $J = 3$, and use the classic ReLU [25] as the nonlinear activation. The dilation factors are set as 1, 3 and

Table 1: PSNR | SSIM | PSNR-B values and parameter numbers comparisons. The best and the second best results are **boldfaced** and underlined. Our network achieves the best overall results with tolerable parameter numbers.

Dataset	Quality	SADCT [12]	LD [27]	PCA [40]	ARCNN [8]	TNRD [6]	DnCNN [57]
Classic5	10	28.88 | 0.8071 | 28.16	28.39 | 0.7997 | 27.59	29.32 | 0.8002 | 29.08	29.03 | 0.7929 | 28.76	29.28 | 0.7992 | 29.04	29.40 | 0.8026 | 29.13
	20	30.92 | 0.8663 | 29.75	30.30 | 0.8584 | 29.37	31.56 | 0.8584 | 31.12	31.15 | 0.8517 | 30.59	31.47 | 0.8576 | 31.05	31.63 | 0.8610 | 31.19
	30	32.14 | 0.8914 | 30.83	31.47 | 0.8830 | 30.17	32.86 | 0.8838 | 32.31	32.51 | 0.8806 | 31.98	32.78 | 0.8837 | 32.24	32.91 | 0.8861 | 32.38
LIVE1	10	28.65 | 0.8093 | 28.01	28.26 | 0.8052 | 27.68	29.01 | 0.8090 | 28.83	28.96 | 0.8076 | 28.77	29.14 | 0.8111 | 28.88	29.19 | 0.8123 | 28.90
	20	30.81 | 0.8781 | 29.82	30.19 | 0.8715 | 29.64	31.28 | 0.8746 | 30.72	31.29 | 0.8733 | 30.79	31.46 | 0.8769 | 31.04	31.59 | 0.8802 | 31.07
	30	32.08 | 0.9078 | 30.92	29.41 | 0.8960 | 29.36	32.62 | 0.9034 | 32.18	32.67 | 0.9043 | 32.22	32.84 | 0.9059 | 32.28	32.98 | 0.9090 | 32.34
BSD500	10	28.23 | 0.7780 | 27.38	28.03 | 0.7824 | 27.29	28.64 | 0.7793 | 28.01	28.56 | 0.7907 | 27.87	28.60 | 0.7926 | 27.95	28.84 | 0.8006 | 28.44
	20	30.09 | 0.8510 | 28.61	29.82 | 0.8514 | 28.43	30.73 | 0.8510 | 29.42	30.43 | 0.8594 | 29.10	30.51 | 0.8611 | 29.34	31.05 | 0.8741 | 30.29
	30	31.21 | 0.8838 | 29.34	30.87 | 0.8719 | 29.15	31.99 | 0.8840 | 30.84	31.52 | 0.8904 | 29.92	31.58 | 0.8902 | 30.02	32.36 | 0.9049 | 31.43
Twitter		27.61 | 0.7281 | 27.53	27.58 | 0.7274 | 27.49	27.71 | 0.7302 | 27.66	27.54 | 0.7295 | 27.49	27.60 | 0.7272 | 27.52	27.63 | 0.7294 | 27.54
WeChat		29.60 | 0.7995 | 29.59	29.48 | 0.7956 | 29.47	29.63 | 0.7994 | 29.60	29.30 | 0.7987 | 29.29	26.64 | 0.7908 | 26.63	29.57 | 0.7982 | 29.57
# Params ($\times 10^5$)		—	—	—	1.06	0.21	6.69

Dataset	Quality	LPIO [11]	M-Net [42]	DCSC [13]	RNAN [62]	RDN [63]	Proposed
Classic5	10	29.35 | 0.8015 | 29.04	29.69 | 0.8107 | 29.31	29.62 | 0.8270 | 29.30	29.87 | 0.8278 | 29.42	30.03 | 0.8194 | 29.59	**30.26** | **0.8403** | **30.05**
	20	31.58 | 0.8567 | 31.12	31.90 | 0.8658 | 31.29	31.81 | 0.8804 | 31.34	32.11 | 0.8693 | 31.26	32.19 | 0.8704 | 31.53	**32.40** | **0.8881** | **31.96**
	30	32.86 | 0.8835 | 32.28	32.97 | 0.8881 | 32.49	33.06 | 0.9030 | 32.49	33.38 | 0.8924 | 32.35	33.46 | 0.8932 | 32.59	**33.56** | **0.9080** | **32.90**
LIVE1	10	29.17 | 0.8119 | 28.89	29.45 | 0.8193 | 29.04	29.34 | 0.8317 | 29.01	29.63 | 0.8239 | 29.13	29.70 | 0.8252 | 29.37	**29.75** | **0.8395** | **29.51**
	20	31.52 | 0.8766 | 31.07	31.83 | 0.8846 | 31.14	31.70 | 0.8960 | 31.18	32.03 | 0.8877 | 31.12	**32.10** | 0.8886 | 31.29	32.06 | **0.9009** | **31.62**
	30	32.99 | 0.9074 | 32.31	33.07 | 0.9108 | 32.47	33.07 | 0.9218 | 32.43	33.45 | 0.9149 | 32.22	**33.54** | 0.9156 | 32.62	33.43 | **0.9254** | **32.81**
BSD500	10	28.81 | 0.7815 | 28.39	28.96 | 0.8039 | 28.56	28.95 | 0.8050 | 28.55	29.08 | 0.8054 | 28.48	29.24 | 0.8080 | 28.71	**29.48** | **0.8146** | **29.13**
	20	30.92 | 0.8551 | 30.07	31.05 | 0.8742 | 30.36	31.13 | 0.8758 | 30.41	31.25 | 0.8751 | 30.27	31.48 | 0.8789 | 30.45	**31.65** | **0.8825** | **30.96**
	30	32.31 | 0.8866 | 31.27	32.61 | 0.9072 | 31.15	32.42 | 0.9057 | 31.52	32.70 | 0.9068 | 31.33	32.83 | 0.9076 | 31.60	**32.93** | **0.9108** | **31.97**
Twitter		27.47 | 0.7333 | 27.41	27.98 | 0.7441 | 27.87	27.63 | 0.7313 | 27.43	27.43 | 0.7183 | 27.42	27.44 | 0.7190 | 27.39	**28.27** | **0.7498** | **28.26**
WeChat		28.90 | 0.8000 | 28.90	29.82 | 0.8067 | 29.82	29.58 | 0.8004 | 29.58	29.56 | 0.7996 | 29.56	29.57 | 0.8000 | 29.51	**30.16** | **0.8196** | **30.13**
# Params ($\times 10^5$)		13.94	6.67	3.21	74.09	220.03	88.21

5. To train the network, we use the Matlab JPEG encoder to generate JPEG compressed images. The JPEG quality factors (QF) are set to 10, 20 and 30. We use both the training and testing sets from BSD500 [2] as our training set. The training process is conducted on the Y channel image of YCrCb space. We randomly generate 64×64 training patch pairs with a batch size of 10. We adopt the Adam solver [24] as the optimizer, and the learning rate is fixed to 10^{-4}. We use TensorFlow [1] to implement our network. Note that we only train one single model to handle all the JPEG compression factors. We initialize $\mathbf{x}_0 = \mathbf{y}$ and $\mathbf{g}_{d,k}^{(0)} = \nabla_d \mathbf{x}_{k-1}$, and all weights in Equations (8) to (10) are initialized as 0.01.

4. Experimental results

We compare our network with three model-driven methods: Shape-Adaptive DCT (SADCT) [12] , Layer Decomposition (LD) [27] and PCA basis Learning (PCA) [40], and several deep learning-based methods: Artifacts Reduction Convolutional Neural Network (ARCNN) [8], Trainable Nonlinear Reaction Diffusion (TNRD) [6], Denoising Convolutional Neural Network (DnCNN) [57], Learning Parameterized Image Operators (LPIO) [11], Memory Network (M-Net) [42], Deep Convolutional Sparse Coding (DCSC) [13], Residual Non-local Attention Networks (R-NAN) [62] and Residual Dense Network (RDN) [63].

4.1. Comparisons on synthetic datasets

We first report the comparison results on the three widely used synthetic datasets, *i.e.*, 5 images in *Classic5* [51], 29 images in *LIVE1* [39] and 100 images in the validation set of *BSD500* [2]. We adpot the PSNR, SSIM [44], and PSNR-B [47] for quantitative evaluations. Since PSNR-B is more sensitive to blocking artifacts than PSNR and S-SIM, it is recommended [8] for use in this de-blocking problem. Table 1 reports the quantitative results and our network achieves the best overall results on all synthetic datasets. Particularly, for the *Classic5* dataset with QF = 10, the average gains of our method over the recently proposed RD-N [63] are respectively 0.23dB in PSNR, 0.0209 in SSIM, and 0.46dB in PSNR-B. When compared with the other approaches, our method is far ahead. Note that we only train one model to cover all three QFs, which substantiates the flexibility and effectiveness of our method, in diverse JPEG artifacts contained in these datasets.

In Figure 3, we show two visual comparisons and it is clear that other compared methods can effectively remove most compression artifacts but fail to recover image details. While our network can finely recover the image textures with a better visual quality. This is because our method learns the dual priors, which enables the network to simultaneously utilize the low-level pixel information and high-level semantic information. In this way, not only the image

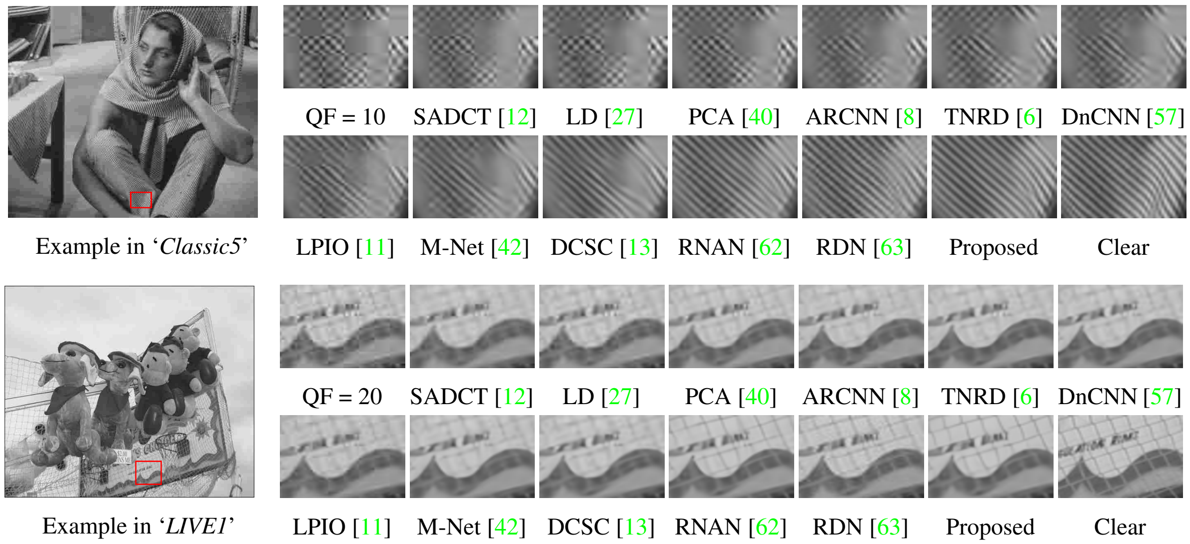


Figure 3: Visual comparisons on different synthetic datasets with different JPEG quality factors.

content-related compression artifacts are removed, but also the image gradient-related clarity is recovered.

4.2. Comparisons on real-world use cases

Online social media softwares have been widely used for message publishing and sharing. To reduce transmission and storage consumption, these platforms usually compresses and re-scales the original images on the server-side. This leads to undesired and unavoidable compression artifacts appearance when users view the posted images. To complement the existing *Twitter* dataset [8] and serve the relevant research communities on this issue in real scenarios, we manually construct a new dataset based on the popular social media *WeChat*. The dataset contains 300 images and their *WeChat*-compressed versions[1]. To avoid out-of-memory caused by excessive image resolution, we first randomly crop the images and then perform the de-blocking operation and measurement calculation. We show quantitative results in Table 1, in which our model consistently generates the best overall performance due to the effective dual priors. Figure 4 shows two visual comparisons. Due to different compression strategies, it is clear that the two compressed images contain different appearances of artifacts, and our method still achieves a superior performance than other compared methods. It is observed that our method can generate clearer results than other competing ones.

4.3. Analysis

Analysis on classifier $f_s(\cdot)$ We first visualize the derivative of the classifier at different stages in Figure 5. By referring to $\nabla \mathbf{x}_0$, the areas of large derivative values, which are normalized for visualization, of $\mathbf{g}$ are basically the same

[1]The dataset can be found at: https://xueyangfu.github.io/

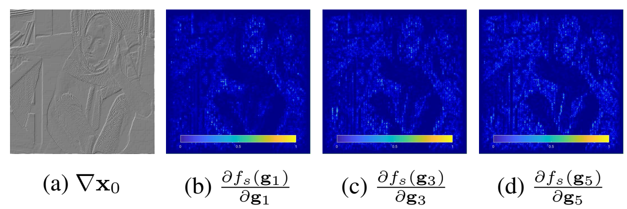


Figure 5: Visualizations of the derivative of the classifier.

as the areas of compression artifacts. Since the gradient represents the increasing direction of the function, using Equation (9) can update $\mathbf{g}$ along the direction of less artifacts.

In Figure 6, we visualize the activations of the last convolutional layer in the classifier. It is clear that our classifier can effectively distinguish clear images and their compressed versions generated from different platforms. Since we use the classifier to explore gradient-relevant priors, the differences between compressed and clear activations are mainly concentrated in areas such as image texture and structure, and edges of compression artifacts. This proves that using our gradient-relevant prior can provide the network with constraints related to 'clarity'.

We also show one visual result in Figure 7 to demonstrate the effect of $f_s(\cdot)$. It is clear that using only pixel-level $f_p(\cdot)$ can obtain a de-blocked result with blurred edges. By adding the semantic-level $f_s(\cdot)$, the edges become more clear with a better visual quality.

Analysis on de-blocker $f_p(\cdot)$ For the pixel-level de-blocker, we compare with the scenarios by using three operators. Specifically, we test the handcrafted ℓ_1 sparsity [17], the DL-based DnCNN [57], and our default $f_p(\cdot)$. Table 2 shows the quantitative comparisons on *BSD500* (QF = 10), and using our $f_p(\cdot)$ achieves the best results. Compared

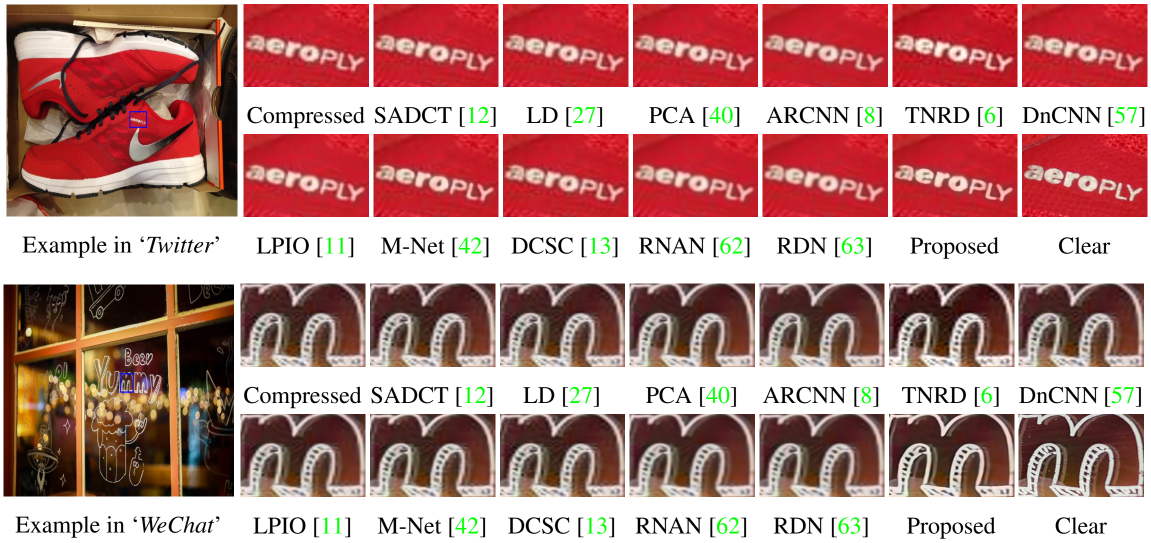
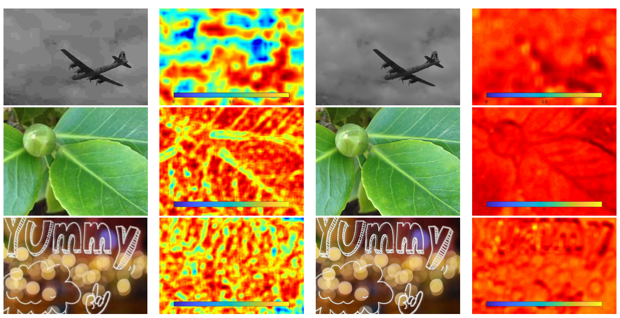


Figure 4: Visual comparisons on two real-world use cases.



(a) Compressed (b) Activ. of (a) (c) Clear (d) Activ. of (c)

Figure 6: Activations of a feature map in the classifier. From top to bottom: *BSD500*, *Twitter* and *WeChat*.

(a) Compressed (b) Without $f_s(\cdot)$ (c) With $f_s(\cdot)$

Figure 7: Effectiveness of $f_s(\cdot)$.

with the other two de-blockers, our $f_p(\cdot)$ can extract both global and multi-scale local features. This is particularly suitable for image de-blocking since compression artifacts usually have the appearance of different spatial scales.

Analysis on stage K We also analyze the effect of stage number K and show the quantitative results on *BSD500* (QF = 10) in Figure 8. Using $K = 1$ as a baseline, it is clear that the performance has an obvious improvement with 3 stages. When $K = 7$, the quantitative results show a slight decreasing trend, which may be caused by the difficulty of gradient propagation due to the increased stage. Therefore, we set $K = 5$ as the default stage number.

Table 2: Comparisons on using different de-blockers.

Proximal operators	PSNR	SSIM	PNSR-B
ℓ_1 sparsity [17]	28.18	0.7985	27.28
DnCNN [57]	28.97	0.8094	27.95
Our $f_p(\cdot)$	29.48	0.8146	29.13

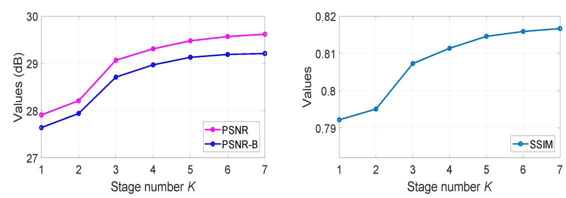


Figure 8: Quantitative results on different stage numbers K.

5. Conclusion

In this paper, we propose two data-driven priors for JPEG compression artifacts removal. Specifically, we design one pixel-level prior and one semantic-level prior to provide regressive and discriminative information, respectively. We then embed these two priors into a variational model and develop an alternative optimization algorithm to solve it. This optimization algorithm is further unfolded into a deep network, in which the dual priors can be effectively explored from training samples. Experiments on benchmarks and real-world use cases show that our method performs favorably against state-of-the-art methods.

References

[1] Martín Abadi, Paul Barham, et al. Tensorflow: A system for large-scale machine learning. In *USENIX symposium on operating systems design and implementation*, 2016. 6

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2010. 6

[3] Kristian Bredies and Martin Holler. A total variation–based jpeg decompression model. *SIAM Journal on Imaging Sciences*, 5(1):366–393, 2012. 2

[4] Huibin Chang, Michael K Ng, and Tieyong Zeng. Reducing artifacts in jpeg decompression via a learned dictionary. *IEEE Transactions on Signal Processing*, 62(3):718–728, 2013. 2

[5] Honggang Chen, Xiaohai He, Linbo Qing, Shuhua Xiong, and Truong Q Nguyen. DPW-SDNet: Dual pixel-wavelet domain deep cnns for soft decoding of jpeg-compressed images. In *CVPR Workshops*, 2018. 3

[6] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1256–1272, 2016. 3, 6, 7, 8

[7] Nathaniel Chodosh and Simon Lucey. When to use convolutional neural networks for inverse problems. In *CVPR*, 2020. 3

[8] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. Compression artifacts reduction by a deep convolutional network. In *ICCV*, 2015. 2, 3, 6, 7, 8

[9] Weisheng Dong, Peiyao Wang, Wotao Yin, Guangming Shi, Fangfang Wu, and Xiaotong Lu. Denoising prior driven deep neural network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(10):2305–2318, 2018. 3

[10] Max Ehrlich, Larry Davis, Ser-Nam Lim, and Abhinav Shrivastava. Quantization guided jpeg artifact correction. In *ECCV*, 2020. 3

[11] Qingnan Fan, Dongdong Chen, Lu Yuan, Gang Hua, Nenghai Yu, and Baoquan Chen. A general decoupled learning framework for parameterized image operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. doi:10.1109/TPAMI.2019.2925793. 3, 6, 7, 8

[12] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE Transactions on Image Processing*, 16(5):1395–1411, 2007. 1, 2, 6, 7, 8

[13] Xueyang Fu, Zheng-Jun Zha, Feng Wu, Xinghao Ding, and John Paisley. JPEG artifacts reduction via deep convolutional sparse coding. In *ICCV*, 2019. 3, 6, 7, 8

[14] Leonardo Galteri, Lorenzo Seidenari, Marco Bertini, and Alberto Del Bimbo. Deep generative adversarial compression artifact removal. In *ICCV*, 2017. 3

[15] Leonardo Galteri, Lorenzo Seidenari, Marco Bertini, and Alberto Del Bimbo. Deep universal generative adversarial compression artifact removal. *IEEE Transactions on Multimedia*, 21(8):2131–2145, 2019. 3

[16] Shanghua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip HS Torr. Res2net: A new multi-scale backbone architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):652–662, 2021. 2

[17] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *ICML*, 2010. 7, 8

[18] Shuhang Gu, Wen Li, Luc Van Gool, and Radu Timofte. Fast image restoration with multi-bin trainable linear units. In *ICCV*, 2019. 3

[19] Jun Guo and Hongyang Chao. Building dual-domain representations for compression artifacts reduction. In *ECCV*, 2016. 3

[20] Jun Guo and Hongyang Chao. One-to-many network for visually pleasing compression artifacts reduction. In *CVPR*, 2017. 1, 3

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[22] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 2, 5

[23] Yoonsik Kim, Jae Woong Soh, Jaewoo Park, Byeongyong Ahn, Hyun-Seung Lee, Young-Su Moon, and Nam Ik Cho. A pseudo-blind convolutional neural network for the reduction of compression artifacts. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(4):1121–1135, 2019. 3

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 6

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 5

[26] Lerenhan Li, Jinshan Pan, Wei-Sheng Lai, Changxin Gao, Nong Sang, and Ming-Hsuan Yang. Learning a discriminative prior for blind image deblurring. In *CVPR*, 2018. 4

[27] Yu Li, Fangfang Guo, Robby T Tan, and Michael S Brown. A contrast enhancement framework with jpeg artifacts suppression. In *ECCV*, 2014. 2, 6, 7, 8

[28] Dong Liu, Zhenzhong Chen, Shan Liu, and Feng Wu. Deep learning-based technology in responses to the joint call for proposals on video compression with capability beyond hevc. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(5):1267–1280, 2019. 1

[29] Daqing Liu, Hanwang Zhang, Feng Wu, and Zheng-Jun Zha. Learning to assemble neural module tree networks for visual grounding. In *ICCV*, 2019. 2

[30] Jiaying Liu, Dong Liu, Wenhan Yang, Sifeng Xia, Xiaoshuai Zhang, and Yuanying Dai. A comprehensive benchmark for single image compression artifact reduction. *IEEE Transactions on Image Processing*, 29:7845–7860, 2020. 1

[31] Risheng Liu, Zhiying Jiang, Xin Fan, and Zhongxuan Luo. Knowledge-driven deep unrolling for robust image layer separation. *IEEE Transactions on Neural Networks and Learning Systems*, 31(5):1653–1666, 2019. 3

[32] Risheng Liu, Long Ma, Jiaao Zhang, Xin Fan, and Zhongxuan Luo. Retinex-inspired unrolling with cooperative prior architecture search for low-light image enhancement. In *CVPR*, 2021. 3

[33] Risheng Liu, Long Ma, Yuxi Zhang, Xin Fan, and Zhongxuan Luo. Underexposed image correction via hybrid priors navigated deep propagation. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 3
[34] Xianming Liu, Gene Cheung, Xiangyang Ji, Debin Zhao, and Wen Gao. Graph-based joint dequantization and contrast enhancement of poorly lit jpeg images. *IEEE Transactions on Image Processing*, 28(3):1205–1219, 2018. 2
[35] Xianming Liu, Xiaolin Wu, Jiantao Zhou, and Debin Zhao. Data-driven soft decoding of compressed images in dual transform-pixel domain. *IEEE Transactions on Image Processing*, 25(4):1649–1659, 2016. 2
[36] Shigenobu Minami and Avideh Zakhor. An optimization approach for removing blocking effects in transform coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(2):74–82, 1995. 2
[37] Jing Mu, Ruiqin Xiong, Xiaopeng Fan, Dong Liu, Feng Wu, and Wen Gao. Graph-based non-convex low-rank regularization for image compression artifact reduction. *IEEE Transactions on Image Processing*, 29:5374–5385, 2020. 2
[38] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1-4):259–268, 1992. 3
[39] HR Sheikh. LIVE image quality assessment database release 2. *http://live. ece. utexas. edu/research/quality*, 2005. 6
[40] Qiang Song, Ruiqin Xiong, Xiaopeng Fan, Dong Liu, Feng Wu, Tiejun Huang, and Wen Gao. Compressed image restoration via artifacts-free PCA basis learning and adaptive sparse modeling. *IEEE Transactions on Image Processing*, 29:7399–7413, 2020. 2, 6, 7, 8
[41] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, 2012. 1
[42] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *ICCV*, 2017. 3, 6, 7, 8
[43] Gregory K Wallace. The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1):xviii–xxxiv, 1992. 1
[44] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6
[45] Zhangyang Wang, Ding Liu, Shiyu Chang, Qing Ling, Yingzhen Yang, and Thomas S Huang. D^3: Deep dual-domain based fast restoration of jpeg-compressed images. In *CVPR*, 2016. 3
[46] Yan Yang, Jian Sun, Huibin Li, and Zongben Xu. ADMM-CSNet: A deep learning approach for image compressive sensing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(3):521–538, 2018. 3
[47] Changhoon Yim and Alan Conrad Bovik. Quality assessment of deblocked images. *IEEE Transactions on Image Processing*, 20(1):88–98, 2010. 6
[48] Jaeyoung Yoo, Sang-ho Lee, and Nojun Kwak. Image restoration by estimating frequency distribution of local patches. In *CVPR*, 2018. 3
[49] Seok Bong Yoo, Kyuha Choi, and Jong Beom Ra. Post-processing for blocking artifact reduction based on inter-block correlation. *IEEE Transactions on Multimedia*, 16(6):1536–1548, 2014. 2
[50] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 4
[51] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, 2010. 6
[52] Zhiyuan Zha, Xin Yuan, Bihan Wen, Jiachao Zhang, Jiantao Zhou, and Ce Zhu. Image restoration using joint patch-group-based sparse representation. *IEEE Transactions on Image Processing*, 29:7735–7750, 2020. 2
[53] Zheng-Jun Zha, Daqing Liu, Hanwang Zhang, Yongdong Zhang, and Feng Wu. Context-aware visual policy network for fine-grained image captioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. doi: 10.1109/TPAMI.2019.2909864. 2
[54] Guangtao Zhai, Wenjun Zhang, Xiaokang Yang, Weisi Lin, and Yi Xu. Efficient image deblocking based on postfiltering in shifted windows. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(1):122–126, 2008. 2
[55] Jian Zhang, Ruiqin Xiong, Chen Zhao, Yongbing Zhang, Siwei Ma, and Wen Gao. CONCOLOR: Constrained non-convex low-rank model for image deblocking. *IEEE Transactions on Image Processing*, 25(3):1246–1259, 2016. 2
[56] Kai Zhang, Luc Van Gool, and Radu Timofte. Deep unfolding network for image super-resolution. In *CVPR*, 2020. 3
[57] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 2, 3, 6, 7, 8
[58] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *CVPR*, 2017. 3
[59] Xinfeng Zhang, Weisi Lin, Ruiqin Xiong, Xianming Liu, Siwei Ma, and Wen Gao. Low-rank decomposition-based restoration of compressed images via adaptive noise estimation. *IEEE Transactions on Image Processing*, 25(9):4158–4171, 2016. 2, 3
[60] Xinfeng Zhang, Ruiqin Xiong, Xiaopeng Fan, Siwei Ma, and Wen Gao. Compression artifact reduction by overlapped-block transform coefficient estimation with block similarity. *IEEE Transactions on Image Processing*, 22(12):4613–4626, 2013. 2, 3
[61] Xiaoshuai Zhang, Wenhan Yang, Yueyu Hu, and Jiaying Liu. DMCNN: Dual-domain multi-scale convolutional neural network for compression artifacts removal. In *ICIP*, 2018. 3
[62] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. Residual non-local attention networks for image restoration. In *ICLR*, 2019. 3, 4, 6, 7, 8
[63] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. doi:10.1109/TPAMI.2020.2968521. 2, 3, 6, 7, 8