IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 4, 2019, accepted January 11, 2020, date of publication January 23, 2020, date of current version January 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2968944

AGARNet: Adaptively Gated JPEG Compression Artifacts Removal Network for a Wide Range Quality Factor

YOONSIK KIM, JAE WOONG SOH, (Student Member, IEEE), AND NAM IK CHO, (Senior Member, IEEE)

Department of Electrical and Computer Engineering, INMC, Seoul National University, Seoul 08826, South Korea

Corresponding author: Nam Ik Cho (nicho@snu.ac.kr)

This work was supported in part by the Institute for Information and Communications Technology Planning and Evaluation (IITP) through the Korea government (MSIT), (Development of AI based Robot Technologies for Understanding Assembly Instruction and Automatic Assembly Task Planning) under Grant NI190004, and in part by the Samsung Electronics Company, Ltd.

ABSTRACT Most of existing compression artifacts reduction methods focused on the application for low-quality images and usually assumed a known compression quality factor. However, images compressed with high quality should also be manipulated because even small artifacts become noticeable when we enhance the compressed image. Also, the use of quality factor from the decoder is not practical because there are too many recompressed or transcoded images whose quality factor are not reliable and spatially varying. To address these issues, we propose a quality-adaptive artifacts removal network based on the gating scheme, with a quality estimator that works for a wide range of quality factor. Specifically, the estimator gives a pixel-wise quality factor, and our gating scheme generates gate-weights from the quality factor. Then, the gate-weights control the magnitudes of feature maps in our artifacts removal network. Thus, our gating scheme guarantees the proposed network to perform adaptively without changing the parameters according to the change of quality factor. Moreover, we exploit the Discrete Cosine Transform (DCT) scheme with 3D convolution for capturing both spatial and frequency dependencies of images. Experiments show that the proposed network provides better performance than the state-of-the-art methods over a wide range of quality factor. Also, the proposed method provides robust results in real-world scenarios such as the manipulation of transcoded images and videos.

INDEX TERMS Compression artifacts removal, recompression, adaptive network, convolutional neural network (CNN), DCT network, 3D convolution.

I. INTRODUCTION

There have been many deep convolutional neural networks (CNNs) [1]–[3] for reducing various kinds of noise in images and videos [4]–[12]. Specifically, many researchers trained various kinds of CNNs for the reduction of additive white Gaussian noise and also showed that the networks can be trained for other kinds of noise including the compression artifacts.

There are also some researches that focused on the compression artifacts in images and videos [13]–[20]. Most of these methods focused on the experiments for severe compression artifacts, for example, quality factors (Q) from 10 to 40 in the case of JPEG images. Both blind and non-blind approaches are considered in many works, where the non-blind approaches (assuming Q is known) usually show better performances than the blind (training a single network for the images with unknown Q). However, non-blind methods have a strong disadvantage that they need multiple networks that are specifically trained to different noise levels to cope with various compression quality factors. Hence, they generally require large memory, and there can be redundancy between the models for similar Qs.

Meanwhile, we need to consider the properties of real-world compressed images for their enhancement and denoising. First of all, the photos from our smartphones

The associate editor coordinating the review of this manuscript and approving it for publication was Yongjie Li.

and digital cameras are now over the quality factors 50[1] whereas the existing works studied the training on very low-quality factors as stated above. Moreover, when we upload our photos to SNS or send them to our friends through message applications, they are always transcoded or recompressed to different quality factors, possibly with resizing, retouching, and/or cropping. Hence, the quality factors from the decoders are not credible because the photo that we receive or see on the Internet is almost always a recompressed one. In addition, the decoder does not provide region-wise image quality. For example, when an image is compressed with a designated Q, the quality actually changes over the regions according to the texture complexity. Also, when an image is a capture of a video frame, the quality is different from region to region due to the nature of adaptive quantization in video compression methods.

In summary, we think that the problems with conventional works are as follows:

- High-quality compression factors are not considered in many works, *i.e.* most of the conventional researches are focused on the experiments on JPEG compression quality factor $Q = 10$ to 40.
- Most works employed compression quality factor from the decoder, which is not reliable and does not reflect the region-wise quality.

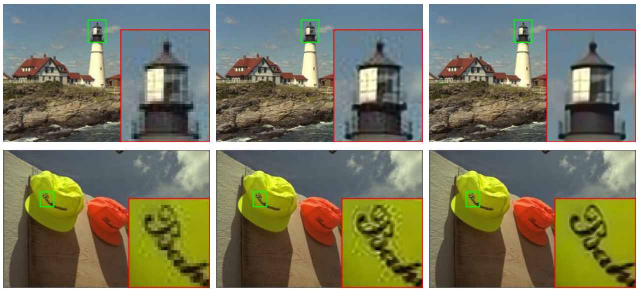


FIGURE 1. The first column shows JPEG compressed images with $Q = 80$ (high quality with 34.81 dB and 37.66 dB). The second shows their ×2 super-resolution using EDSR [21]. The third shows the results of ×2 super-resolution after the images are preprocessed by the proposed artifacts removal method.

Maybe, the high Qs are neglected in the existing works because the photos are considered good enough when Q is over 50. However, Fig. 1 shows that we still need to remove the noise resulting from the higher Q when we enhance or super-resolve the images. Specifically, small artifacts still remain even with high Q, as shown in the first column of the figure. The second column shows the super-resolution of compressed images, where we can find that the small artifacts are boosted and become noticeable. Therefore, high-quality images should also be manipulated when we enhance their contrast, resolution, and/or dynamic ranges. The third column shows that the noise is not noticeable when we preprocess the images before the super-resolution.

The blind approach or having a credible Q-estimator can be a solution to cope with the unreliable Q in real-world environments. The blind approach is to train a single network with the images from different Qs altogether. Hence, it has the advantage that it works quite robustly with uncertain Q, but it tends to produce blurry results. Assuming that Q is correctly estimated, applying a bundle of non-blind models can have better performance [22]. However, the bundle of non-blind models has a critical problem with memory and redundancy issues as stated above. In addition to the disadvantages of each approach, both blind and non-blind approaches cannot cope with spatially varying quality, because they apply the same filter to overall regions. Recently, there have been flexible methods [23]–[26] for region-adaptive enhancement, but they mostly focused on Gaussian denoising or super-resolution, which deal with a less complicated degradation model (linear) than the compression.

To address the above-stated problems of existing researches in real-world environments, we propose an adaptive noise removing system based on the gating scheme. Besides, we design a Q-estimator that generates a pixel-wise quality factor Q_{map}. Precisely, our network consists of a Q-estimator and two adaptive networks where one operates in the pixel-domain and the other in the DCT-domain (see Fig. 2). Each of the adaptive networks is a combination of a reconstruction network and a gate-weight generation network. In our method, the "gating scheme" means to control the magnitudes of feature maps of the reconstruction networks by multiplying the learned weights of the gate-weight generating network, for the given quality. The adaptive network is trained with the Q_{map} such that the gate-weight generating network produces the weights using the Q_{map} as the input. Then, the weights control the magnitudes of feature maps in the reconstruction networks. Thus, our gating scheme enables the reconstruction network to work adaptively without changing the parameters according to the change of Q. Moreover, we adopt 3-dimensional convolution in the DCT-domain reconstruction network, considering the spatial and frequency dependencies of images. In summary, the main contributions of our work are as follows:

- We propose a deep network that removes compression artifacts over a wide range of Q.
- We design a Q-estimator, which finds the spatially variant quality-level map for the given input. This is essential for dealing with compressed images and videos in the real world.
- We apply a gating scheme for the removal of the artifacts, which enables us to use a single network for a wide range of Q by controlling the magnitudes of feature maps depending on Q.
- The image reconstruction is processed in the dual-domain (pixel-domain and DCT-domain), where we adopt the 3D-convolution in the DCT-domain.
- The 3D-convolution in the DCT-domain reconstruction network enables the combination of frequency-wise and feature-wise attention according to the change of Q.

[1] We checked default JPEG compression quality factors of several smartphones and digital cameras and found that they range from 92 to 98.

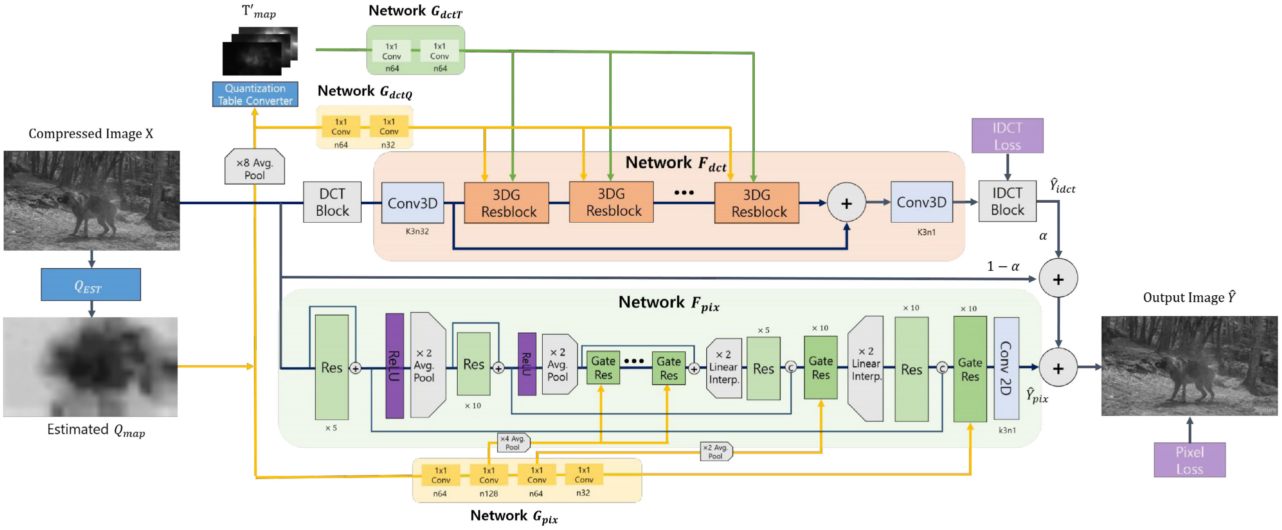


FIGURE 2. Overview of proposed system, where *K* is the 3D kernel size, *k* is the 2D kernel size, and *n* is the number of features. Some repetitive blocks are represented with '×' notation above/below each block such as ×5 and ×10. The total number of 3DG-ResBlock is 8 and that of Gate-ResBlock in 1/4 scale is 20. The final output is obtained as the weighted sum of X, $\hat{Y}_{idct}$ and $\hat{Y}_{pix}$. The weight factor α is trained along with the network parameters in the training phase and fixed at the test. It is found that α converges to 0.23 for our training dataset. The proposed system requires 10.3 M training parameters in total; precisely F_{dct}, F_{pix}, G_{dctT}, G_{dctQ}, G_{pix}, and Q_{EST} need 526K, 9104K, 11K, 8K, 15K, and 702K respectively.

- The proposed method is shown to provide better performance than the conventional ones, and also works well for the recompressed images.

II. PROPOSED ARCHITECTURE

The image compression can be modeled as

$$X = C(Y, Q), \tag{1}$$

where Y is the original (uncompressed) image, and X is the output of a compression method C with the quality factor Q. There have been many specifically or blindly trained methods to reduce the compression artifacts, which can be written as

$$\hat{Y} = F_Q(X), \tag{2}$$
$$\hat{Y} = F(X), \tag{3}$$

respectively. For removing the noise in a compressed image, the specifically trained method estimates or extracts Q from the compressed data, and then switches the model among several F_Qs that are trained for the Qs. On the other hand, the blindly trained method does not take Q as an input because it assumes that input image implicitly includes the information of Q. As mentioned in the introduction, we think that both schemes have certain limitations, and thus we propose a flexible network named as adaptively gated artifact removal network (AGARNet). Specifically, the proposed method is to design a single network that adaptively works for a wide range of Q. The proposed model can be described as

$$\hat{Y} = F(X, Q), \tag{4}$$

where F adaptively processes compressed image depending on Q. The proposed model F can take Q when it is known and there was no recompression, or it can also take $\hat{Q}$ that is estimated by a Q-estimator when we do not know the actual image quality due to recompression. We design a noise-removing system including this function, which consists of three elements: Q-estimator (Q_{EST}), gate-weight generating network (G), and reconstruction network (F). The Q_{EST} estimates the pixel-wise Q ($\hat{Q}_{map} \in R^{H\times W\times 1}$ where $H \times W$ is the spatial dimension of the input image). The G takes $\hat{Q}_{map}$ as the input and generates gate-weights, which adaptively control the activation of feature maps in F. Then, the F takes a compressed image and the output of G as input and produces the noise-reduced image.

Fig. 2 shows the details of our noise removing system, which shows that the input (compressed image) is branched into two domains, i.e., DCT and pixel-domains. For this, we design two Fs (F_{pix} for the pixel-domain and F_{dct} for the DCT-domain), and three Gs (G_{pix} for the pixel-domain and $\{G_{dctQ}, G_{dctT}\}$ for the DCT-domain). The figure also shows that the Q-estimator produces the quality map ($\hat{Q}_{map}$) from the input. Then, the $\hat{Q}_{map}$ is fed to the Gs which generate the weights that control the Fs. After the reconstructions, the results of Fs and input are linearly combined to produce the noise-reduced image. In the rest of this section, we explain the details of these networks.

A. QUALITY ESTIMATOR

We design a Q-estimator (Q_{EST}) that estimates the spatially varying compression quality factors, as shown in Table 1. The fundamental block size is 64 × 64, and we regard the *Conv*9 as a block-wise estimation result. Thus, spatially variant $\hat{Q}_{map}$ can be generated by rescaling the *Conv*9 to input image size using the bilinear interpolation method. Also, a single Q is achieved by spatially averaging the block-wise estimation result (*Conv*9). Then, the obtained single Q is tiled to input image size and fed to reconstruction networks.

TABLE 1. Proposed Estimator.

Type	Kernel / Stride Size	Output Size
Conv1,2	$3 \times 3 / 1 \times 1$	$H \times W \times 32$
Max Pool1	$4 \times 4 / 4 \times 4$	$\frac{H}{4} \times \frac{W}{4} \times 32$
Conv3,4	$3 \times 3 / 1 \times 1$	$\frac{H}{4} \times \frac{W}{4} \times 64$
Max Pool2	$2 \times 2 / 2 \times 2$	$\frac{H}{8} \times \frac{W}{8} \times 64$
Conv5,6	$3 \times 3 / 1 \times 1$	$\frac{H}{8} \times \frac{W}{8} \times 96$
Max Pool3	$4 \times 4 / 4 \times 4$	$\frac{H}{32} \times \frac{W}{32} \times 96$
Conv7,8	$3 \times 3 / 1 \times 1$	$\frac{H}{32} \times \frac{W}{32} \times 192$
Avg Pool	$2 \times 2 / 2 \times 2$	$\frac{H}{64} \times \frac{W}{64} \times 192$
Conv9	$1 \times 1 / 1 \times 1$	$\frac{H}{64} \times \frac{W}{64} \times 1$
Rescale	-	$H \times W \times 1$

B. GATE-WEIGHT GENERATING NETWORK

We propose three gate-weight generating networks (G_{pix}, G_{dctQ}, G_{dctT}), which take $\hat{Q}_{map}$ as the input and generate gate-weights for pixel-domain reconstruction network (F_{pix}) and DCT-domain reconstruction network (F_{dct}). First, G_{pix} generates three different-scales of gate-weight which can be written as

$$[g^4_{pix}, g^2_{pix}, g^1_{pix}] = G_{pix}(\hat{Q}_{map}), \tag{5}$$

where $g^s_{pix} \in R^{H/s \times W/s \times 32s}$ is the pixel-domain gate-weight that will be applied to $1/s$-scale feature map in F_{pix}. Second, to apply $\hat{Q}_{map}$ in the proposed DCT-domain, it is spatially 8×8 average pooled to be $\hat{Q}'_{map} \in R^{1 \times H/8 \times W/8 \times 1}$, because the values in 8×8 pixel-domain block are decomposed to the frequency dimension at the DCT-domain. Then, G_{dctQ} takes $\hat{Q}'_{map}$ as an input and generates g_{dctQ}. Lastly, we suppose that the quantization table (T) achieved from the corresponding Q is another important factor for image quality. Hence, we also feed the pixel-wise T (T'_{map}) converted from $\hat{Q}'_{map}$ to G_{dctT}, which generates the output g_{dctT}. Formally, generation of these gate-weights is written as

$$g_{dctQ} = G_{dctQ}(\hat{Q}'_{map}), \tag{6}$$

$$g_{dctT} = G_{dctT}(T'_{map}), \tag{7}$$

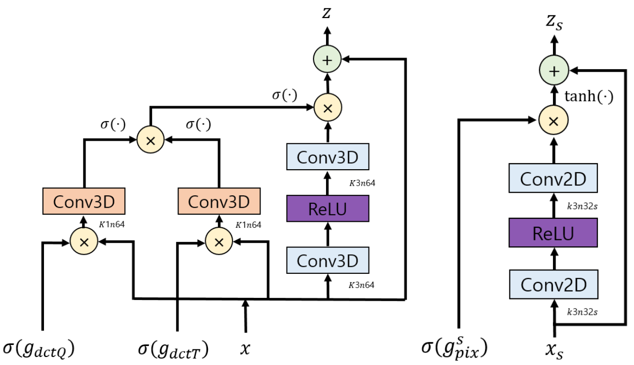
where $g_{dctQ} \in R^{H/8 \times W/8 \times 32}$ and $g_{dctT} \in R^{H/8 \times W/8 \times 64}$ are DCT-domain gate-weights that will be applied to F_{dct}.

C. DCT-DOMAIN RECONSTRUCTION NETWORK

The proposed DCT-domain reconstruction network consists of the DCT/IDCT converter, 3D Convolution, and 3D gate-residual blocks (3DG-ResBlock). The DCT transforms X into $X_{dct} \in R^{64 \times H/8 \times W/8 \times 1}$ where the first number 64 means the frequency dimension, which consists of 1 DC and 63 AC coefficients. The last number 1 represents the feature dimension. Then, X_{dct} is fed to the DCT-domain reconstruction network that is composed of 3D convolutions and proposed 3DG-ResBlock. Lastly, the output of the DCT-domain reconstruction network is transformed to the pixel-domain by applying Inverse Discrete Cosine Transform (IDCT).

1) DCT CONVERTER AND 3D CONVOLUTION

The DCT-domain methods have long been researched in image processing, and have accomplished certain improvements. Inspired by this, we propose a new DCT-domain reconstruction scheme by exploiting the 2D DCT with 3D convolutions. Our idea is to use the fact that the JPEG encoder processes the 8×8 pixels in a unit, where the 64 DCT coefficients are zigzag-ordered according to the energy compaction property of DCT. Specifically, we propose DCT and IDCT converters based on this fact, where the DCT converter generates X_{dct}, which is the 64 coefficients in the zigzag order. In addition to this, we employ a 3D convolution in order to preserve the coefficients order in the frequency dimension. Thus, 3D convolution can generate spatio-frequency feature maps in the DCT-domain. Precisely, we perform the 3D convolution by sliding the $3 \times 3 \times 3$ kernel into the frequency and spatial dimensions. It needs to be noted that the $3 \times 3 \times 3$ kernel in the DCT-domain has larger spatial receptive field than in the pixel-domain, because the receptive field of DCT-domain includes eight adjacent blocks. We suppose that generated spatio-frequency feature maps, which consider frequency-domain receptive field as well as spatial-domain, has the effect of providing distinct feature representations that are different from pixel-domain. The last output of the DCT-domain reconstruction network is $\hat{Y}_{dct} \in R^{H/8 \times W/8 \times 64}$, and $\hat{Y}_{dct}$ is converted to the pixel-domain as $\hat{Y}_{idct} \in R^{H \times W \times 1}$ using the IDCT converter.



(a) 3DG-ResBlock (b) Gate-ResBlock

FIGURE 3. (a) Proposed 3DG-ResBlock in the DCT-domain and (b) Gate-ResBlock at $1/s$ scale in the pixel-domain. The number of features and size of inputs ($\sigma(g^s_{pix})$ and x_s) are different depending on the scale factor (s) in (b). The symbol × denotes element-wise multiplication.

2) 3D GATE RESIDUAL BLOCK

We propose a 3DG-ResBlock that enables the reconstruction network to be adaptive to the variation of Q by gating the magnitude of feature map pixel-wisely. The illustration of 3DG-ResBlock is presented in Fig. 3(a), which indicates that the output feature is updated with input and newly represented feature. In this process, the gate-weight determines the update scale of newly represented feature. We compose the gate-weight as the combination of g_{dctQ} and g_{dctT} that are dependently generated from Q. Precisely, the input feature (x) is recalibrated frequency-wisely and feature-wisely from g_{dctT} and g_{dctQ} respectively. Before the recalibration, g_{dctQ} and g_{dctT} are applied with corresponding transposition and sigmoid function as $\sigma(g_{dctQ}) \in R^{64\times H\times W\times 1}$ and $\sigma(g_{dctT}) \in R^{1\times H\times W\times N}$ where N is the number of features that is set to 32. Then, the overall weight combination process of 3DG-ResBlock, which is illustrated in the left part in Fig. 3(a), can be written as

$$w = C_1(x \otimes \sigma(g_{dctQ})) \otimes C_1(x \otimes \sigma(g_{dctT})), \qquad (8)$$

where C_1 is the $1 \times 1 \times 1$ convolution that slides in the frequency and spatial dimensions, and $\otimes$ means the element-wise multiplication. $\sigma(g_{dctQ})$ and $\sigma(g_{dctT})$ are tiled to have the same dimension with x before the element-wise multiplication. We omit the sigmoid function $\sigma(\cdot)$ in the left part of Fig. 3(a) for simplicity.

Then, w adjusts the update amount of the newly represented feature that is generated from consecutive convolution. Formally, the update process of $z \in R^{64\times H\times W\times N}$, which is illustrated in the right part in Fig. 3(a), can be written as

$$z = x + \sigma(w) \otimes C_3(C_3(x)), \qquad (9)$$

where C_3 is a $3 \times 3 \times 3$ convolution and the ReLU is omitted here for simplicity.

The proposed 3DG-ResBlock makes the network adaptive to the variation of Q and also has the effect of feature dimension attention [27], [28] that recalibrates the feature-wise response to boost the representation power. Since 3DG-ResBlocks are frequency-wisely and feature-wisely recalibrated according to Q, the feature response can be boosted more abundantly.

D. PIXEL-DOMAIN RECONSTRUCTION NETWORK

The proposed pixel-domain reconstruction network takes X as an input and generates pixel-domain $\hat{Y}_{pix}$. We adopt an hourglass architecture as F_{pix}, because it can consider inter-block correlations efficiently with a large receptive field. For this, downsampling and upsampling are operated twice using the 2×2 average pooling and 2×2 bilinear interpolation respectively. Thus, overall F_{pix} consists of down/upsampling module, residual block [21] and gate-residual block (Gate-ResBlock) that is presented in Fig. 3(b). Before the second average pooling (encoder part), we employ a ResBlock, which is similar to Gate-ResBlock excluding the gate process, to represent integrated features regardless of Q as a backbone network. After the second average pooling (decoder part), Gate-ResBlocks are stacked to represent Q-adaptive feature maps. The update process of Gate-ResBlock is g^s_{pix} controlling the magnitude of scale $1/s$ feature maps, which can be written

$$z_s = x_s + \beta \tanh(C^*_3(C^*_3(x_s)) \otimes \sigma(g^s_{pix})), \qquad (10)$$

where β and C^*_3 are update parameter (1×10^{-1}) and 3×3 convolution sliding in spatial dimensions respectively, and ReLU operation is omitted for simplicity.

III. TRAINING DETAILS

The proposed network is trained with 128×128 compressed patches X_i and corresponding original patches Y_i extracted from DIV2K [29] images. We select Q_i, which generates a uniform Q_{map} from 10 to 80 with the steps of 10. The overall loss functions are IDCT reconstruction loss, pixel reconstruction loss, and Q-estimator loss, which can be written as

$$\begin{aligned}\mathbf{L}(\Theta) = \frac{1}{M}\sum_{i=1}^{M}&[\gamma\|Y_i - F_{dct}(X_i, Q_i)\|_1^1 \\ &+ (1-\gamma)\|Y_i - F(X_i, Q_i)\|_2^2] \\ &+ \frac{1}{M}\sum_{i=1}^{M}\|Q_{map} - Q_{EST}(X_i)\|_2^2, \end{aligned} \qquad (11)$$

where M, $F(\cdot)$, and $Q_{EST}(\cdot)$ are the number of patches, reconstruction networks including gate-weight generating networks, and estimator. γ is a reconstruction balance parameter, which is empirically decided to 0.05. We empirically decide the loss function of IDCT reconstruction as $L1$ loss, because it converges more stably than $L2$.

We pre-train Q_{EST} with estimator loss and freeze the weights when training the reconstruction network. The main reason is that the quantization table converter, which generates input of the reconstruction network, can cause unstable training. Specifically, the quantization table converter contains operations such as floor, clip, division, and condition that can hinder training stability even if we employ relaxations. Both estimator and reconstruction loss are minimized using ADAM [30] optimizer, and the learning rate is 1×10^{-4}.

IV. EXPERIMENTS

A. EXPERIMENTAL SETUP

The proposed AGARNet can employ Q_{EST} when Q is not provided or unreliable. Hence, we show the results of AGARNet/AGARNet-EST according to absence/presence of Q_{EST}. These proposed methods are compared with ARCNN [6], REDNet [34], CAS-CNN [14], DnCNN [5], OTM [13], MemNet [35], Galteri *et al.* [15], Yoo *et al.* [16], and DURR [31] with classic5 (baboon, Barbara, boats, Lena, and peppers), LIVE1 [32], and Urban100 [33] test sets. Specifically, we apply the published trained models for ARCNN, REDNet, DnCNN-3, and MemNet, and we directly refer to the results from CAS-CNN, Galteri *et al.* [15], Yoo *et al.* [16], and DURR [31].

OTM [13] that has shown to provide comparable performance with [36] is retrained by us, because the code is not available and the results of above test sets are not reported. We do not directly compare proposed methods with DMCNN [37], because we could not access the code and the model cannot be recalled (retrained) due to the lack of architecture details such as the depth of layers and the number of channels. Since there have been few works that experimented for a wide range of Q, we train DnCNN architecture named as DnCNN-B with Q from 10 to 80 in a single network using DIV2K [29], and we also train DnCNN-BW,

TABLE 2. Average PSNR / SSIM of the JPEG and restored images, where the inputs are compressed with Q from 10 to 80 with the steps of 10 (Red: the best result, Blue: the second best). The number of training parameters is also notated in right part. Since the Q_{EST} provides quite accurate estimates, AGARNet-EST can have the same results with AGARNet. * denotes that the corresponding Q is not included in the training phase. DURR [31] applies the network for the unseen quality factors by "modulation parameters," and it is fair to note that these quality factors are not in the training data.

Dataset	Method	Q								Params
		10	20	30	40	50	60	70	80	
Classic5	JPEG	27.82 / 0.780	30.12 / 0.854	31.48 / 0.884	32.43 / 0.901	33.20 / 0.913	33.96 / 0.923	34.98 / 0.934	36.44 / 0.948	-
	ARCNN	29.03 / 0.811	31.15 / 0.869	32.51 / 0.896	33.32 / 0.910	- / -	- / -	- / -	- / -	107 K
	REDNet	29.35 / 0.821	31.60 / 0.877	- / -	- / -	- / -	- / -	- / -	- / -	4.1 M
	DnCNN	29.40 / 0.820	31.63 / 0.878	32.91 / 0.901	33.77 / 0.914	- / -	- / -	- / -	- / -	664 K
	OTM	29.62 / 0.826	31.77 / 0.879	33.04 / 0.903	33.89 / 0.915	- / -	- / -	- / -	- / -	1.2 M
	MemNet	29.68 / 0.828	31.88 / 0.881	- / -	- / -	- / -	- / -	- / -	- / -	677 K
	DnCNN-B	29.42 / 0.819	31.76 / 0.879	33.07 / 0.903	33.93 / 0.916	34.62 / 0.925	35.26 / 0.933	36.13 / 0.942	37.32 / 0.953	664 K
	DnCNN-BW	29.78 / 0.829	31.89 / 0.880	33.14 / 0.903	33.98 / 0.915	34.63 / 0.924	35.26 / 0.932	36.10 / 0.941	37.27 / 0.951	10.1 M
	Baseline-S	30.00 / 0.836	32.21 / 0.886	33.39 / 0.907	34.22 / 0.919	34.90 / 0.928	35.55 / 0.935	36.41 / 0.944	37.63 / 0.954	9.6 M
	Baseline-B	29.95 / 0.835	32.16 / 0.886	33.37 / 0.907	34.20 / 0.919	34.85 / 0.928	35.46 / 0.935	36.29 / 0.943	37.47 / 0.954	9.6 M
	AGARNet	30.02 / 0.836	32.25 / 0.887	33.47 / 0.908	34.29 / 0.920	34.96 / 0.928	35.59 / 0.935	36.43 / 0.944	37.65 / 0.954	10.2 M
	AGARNet-EST	30.02 / 0.836	32.25 / 0.887	33.47 / 0.908	34.29 / 0.919	34.96 / 0.928	35.59 / 0.935	36.43 / 0.944	37.64 / 0.954	10.3 M
LIVE1 [32]	JPEG	27.77 / 0.791	30.07 / 0.868	31.41 / 0.900	32.35 / 0.917	33.16 / 0.930	33.98 / 0.940	35.13 / 0.951	36.87 / 0.964	-
	ARCNN	28.96 / 0.822	31.29 / 0.887	32.67 / 0.916	33.61 / 0.930	- / -	- / -	- / -	- / -	107 K
	REDNet	29.27 / 0.829	31.64 / 0.894	- / -	- / -	- / -	- / -	- / -	- / -	4.1 M
	CAS-CNN-S	29.44 / 0.833	31.70 / 0.895	- / -	34.10 / 0.937	- / -	35.78 / 0.954	- / -	38.55 / 0.973	5.1 M
	CAS-CNN-B	29.36 / 0.830	31.67 / 0.894	- / -	33.98 / 0.935	- / -	- / -	- / -	- / -	5.1 M
	OTM	29.36 / 0.830	31.68 / 0.895	33.09 / 0.921	34.09 / 0.936	- / -	- / -	- / -	- / -	1.2 M
	MemNet	29.47 / 0.834	31.83 / 0.897	- / -	- / -	- / -	- / -	- / -	- / -	677 K
	Galteri *et al.*	29.45 / 0.834	31.77 / 0.896	33.15 / 0.922	34.09 / 0.935	- / -	- / -	- / -	- / -	1.2 M
	Yoo *et al.*	29.40 / 0.833	31.68 / 0.895	- / -	- / -	- / -	- / -	- / -	- / -	N/A
	DnCNN-B	29.19 / 0.826	31.66 / 0.895	33.09 / 0.922	34.09 / 0.936	34.90 / 0.946	35.72 / 0.954	36.83 / 0.962	38.43 / 0.972	664 K
	DnCNN-BW	29.45 / 0.832	31.78 / 0.895	33.19 / 0.921	34.19 / 0.936	34.99 / 0.945	35.80 / 0.953	36.89 / 0.962	38.46 / 0.972	10.1 M
	DURR	29.23* / -	31.68 / -	33.05 / -	34.01* / -	- / -	- / -	- / -	- / -	233 K
	Baseline-S	29.65 / 0.838	32.00 / 0.900	33.38 / 0.925	34.38 / 0.939	35.19 / 0.948	36.03 / 0.956	37.15 / 0.964	38.75 / 0.974	9.6 M
	Baseline-B	29.55 / 0.837	31.92 / 0.900	33.32 / 0.925	34.30 / 0.939	35.09 / 0.948	35.90 / 0.956	36.99 / 0.964	38.58 / 0.973	9.6 M
	AGARNet	29.64 / 0.837	32.02 / 0.900	33.44 / 0.925	34.42 / 0.939	35.24 / 0.948	36.06 / 0.956	37.17 / 0.964	38.81 / 0.973	10.2 M
	AGARNet-EST	29.63 / 0.837	32.01 / 0.900	33.43 / 0.925	34.42 / 0.939	35.24 / 0.948	36.06 / 0.956	37.17 / 0.964	38.81 / 0.973	10.3 M
Urban100 [33]	JPEG	26.33 / 0.810	28.57 / 0.876	30.00 / 0.905	31.07 / 0.922	31.98 / 0.934	32.91 / 0.944	34.32 / 0.956	36.69 / 0.970	-
	ARCNN	28.06 / 0.852	30.29 / 0.902	31.93 / 0.928	32.80 / 0.939	- / -	- / -	- / -	- / -	107 K
	REDNet	28.58 / 0.864	31.06 / 0.914	- / -	- / -	- / -	- / -	- / -	- / -	4.1 M
	OTM	29.01 / 0.971	31.39 / 0.917	32.90 / 0.938	33.98 / 0.949	- / -	- / -	- / -	- / -	1.2 M
	MemNet	29.14 / 0.874	31.61 / 0.921	- / -	- / -	- / -	- / -	- / -	- / -	677 K
	DnCNN-B	28.77 / 0.864	31.43 / 0.918	33.00 / 0.938	34.09 / 0.950	34.93 / 0.957	35.74 / 0.963	36.92 / 0.970	38.54 / 0.978	664 K
	DnCNN-BW	29.37 / 0.876	31.86 / 0.921	33.40 / 0.940	34.44 / 0.951	35.25 / 0.958	36.01 / 0.963	37.10 / 0.970	38.58 / 0.978	10.1 M
	Baseline-S	29.85 / 0.886	32.37 / 0.928	33.76 / 0.945	34.80 / 0.954	35.63 / 0.961	36.43 / 0.966	37.56 / 0.973	39.13 / 0.980	9.6 M
	Baseline-B	29.66 / 0.883	32.15 / 0.927	33.64 / 0.944	34.66 / 0.954	35.45 / 0.960	36.20 / 0.965	37.31 / 0.972	38.90 / 0.980	9.6 M
	AGARNet	29.87 / 0.885	32.41 / 0.928	33.88 / 0.945	34.88 / 0.954	35.68 / 0.961	36.44 / 0.966	37.55 / 0.972	39.15 / 0.980	10.2 M
	AGARNet-EST	29.82 / 0.885	32.39 / 0.928	33.87 / 0.945	34.88 / 0.954	35.68 / 0.961	36.43 / 0.966	37.55 / 0.973	39.15 / 0.980	10.3 M
Overall	Q_{EST}	10.32 ± 0.991	20.24 ± 0.880	30.19 ± 0.798	39.88 ± 1.584	50.50 ± 1.045	60.09 ± 1.471	70.38 ± 1.628	80.74 ± 0.755	-

which amounts similar training parameters compared to proposed method, by building more convolutions for each depth and deeper architecture. Moreover, we show the results of some baselines, where 3DG-ResBlock and Gate-ResBlocks are replaced to plain resblocks, and the gate-weight generating networks are omitted, such as Baseline-S (Q specifically trained network) and Baseline-B (trained with all Q in a single network). Lastly, we also show the results using uniform $\hat{Q}_{map}$ (AGARNet-EST) and using spatially variant $\hat{Q}_{map}$ (AGARNet-ESP).

B. EXPERIMENTAL RESULTS

1) EXPERIMENTS ON SINGLE JPEG COMPRESSION

Table 2 presents the average PSNR/SSIM of proposed and compared methods. We first observe that the proposed DCT with 3D convolution and hourglass architecture is proper for training a wide range of Q by observing that the performance of Baseline-B surpasses DnCNN-BW and most of conventional specifically trained methods. Moreover, we observe that AGARNet always has better performance (especially at high Q) than Baseline-B, which has a similar training parameter. Therefore, AGARNet always shows the best performance among the compared methods, including Baselines-B which gives significant margin at high Q. We can see that AGARNet can also have better or comparable results than Baseline-S that needs multiple Q training models. Lastly, proposed estimator's results are presented in the last row of the table where the quantization step is 1. It can be seen the proposed method shows quite accurate estimation results, and thus the results of AGARNet and AGARNet-EST can have similar performance.

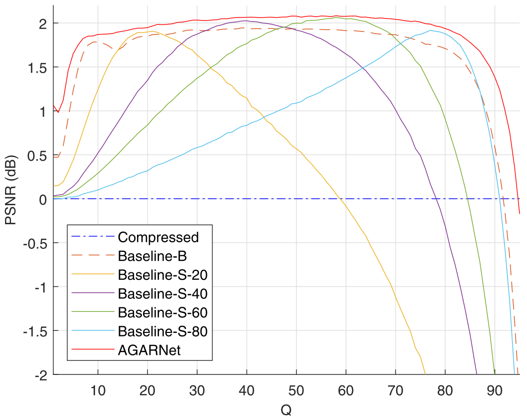
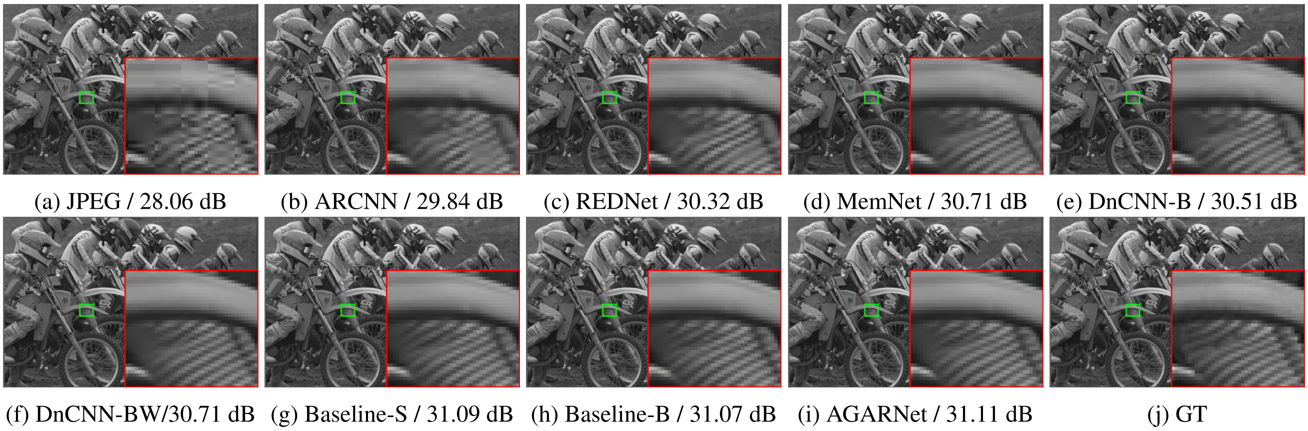


FIGURE 4. PSNR gain obtained by applying the artifacts reduction methods on LIVE1. The x-axis is the JPEG Q of input and y-axis is the PSNR gain where Baseline-S-K (K = 20, 40, 60, and 80) means specifically trained model with $Q = K$.

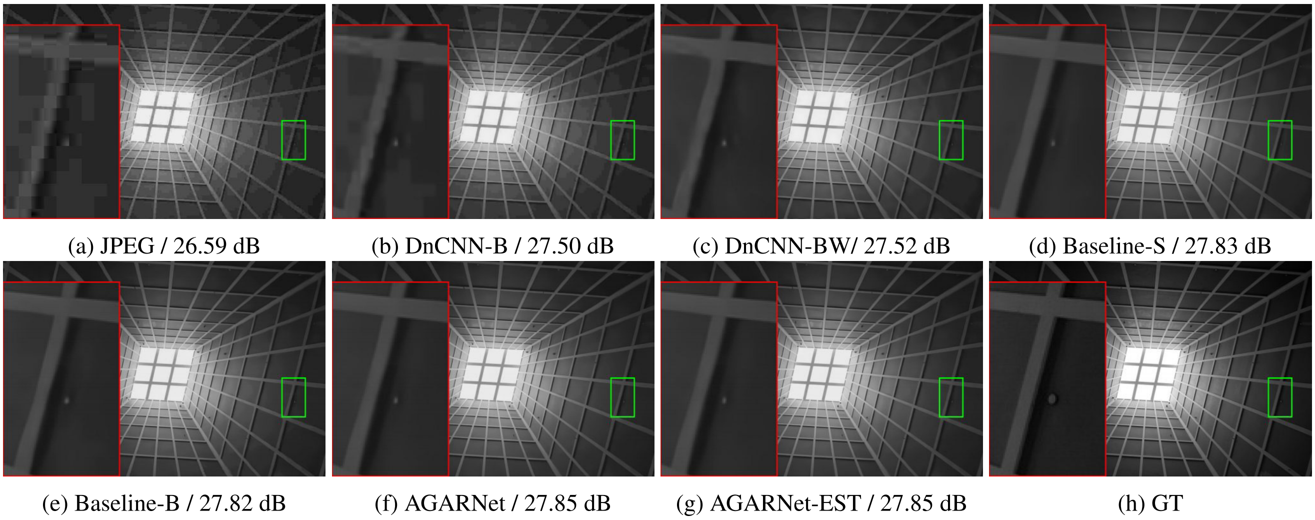
We also present the improvement curves of average PSNR for $1 < Q < 95$ with step size 1 which includes untrained regions ($Q < 10$, $Q > 80$, and all the Qs between the trained Qs) in Fig. 4. It is noticeable that proposed AGARNet



(a) JPEG / 28.06 dB (b) ARCNN / 29.84 dB (c) REDNet / 30.32 dB (d) MemNet / 30.71 dB (e) DnCNN-B / 30.51 dB

(f) DnCNN-BW/30.71 dB (g) Baseline-S / 31.09 dB (h) Baseline-B / 31.07 dB (i) AGARNet / 31.11 dB (j) GT

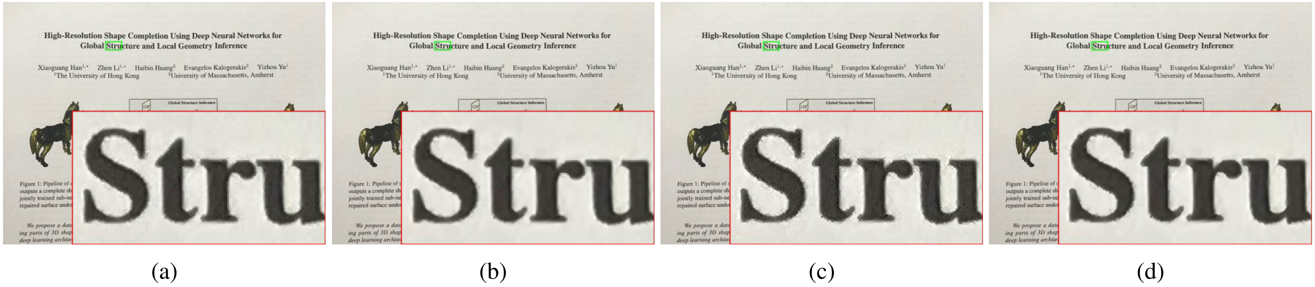
FIGURE 5. The 'bikes' image from the LIVE1 dataset compressed in *Q* 20, and the comparison of the results post-processed by various methods.



(a) JPEG / 26.59 dB (b) DnCNN-B / 27.50 dB (c) DnCNN-BW/ 27.52 dB (d) Baseline-S / 27.83 dB

(e) Baseline-B / 27.82 dB (f) AGARNet / 27.85 dB (g) AGARNet-EST / 27.85 dB (h) GT

FIGURE 6. The 90th image from the Urban100 dataset compressed in *Q* 10, and the comparison of the results post-processed by various methods.



High-Resolution Shape Completion Using Deep Neural Networks for Global Structure and Local Geometry Inference

Stru

(a) (b) (c) (d)

FIGURE 7. (a) A photo taken by *iPad 6th generation* camera. (b) The output of the proposed AGARNet. (c) The ×2 super-resolution output from (a). (d) The ×2 super-resolution output from (b).

shows better performance than Baseline-S for most of Q and also AGARNet can be generalized to unseen Qs more robustly at the $Q < 10$ and $Q > 80$. On the other hand, Baseline-S provides full potential improvement when the model is matched with corresponding Q, otherwise, it even degrades the input images. Thus, applying the non-blind approach, which requires lots of multiple networks to cover a wide range, is unrealistic. From these objective comparisons, we believe that proposed AGARNet can replace the bundle of Baseline-Ss, which saves lots of memory.

We provide the visualized comparison in Fig. 5 and 6. We observe that proposed methods (including Baselines) reconstruct patterns without pattern distortion. We also provide the result of AGARNet in Fig. 7(a) and (b) where the input is taken by *iPad 6th generation*. The photo is compressed in Q 93, but it still has some compression artifacts at the edges

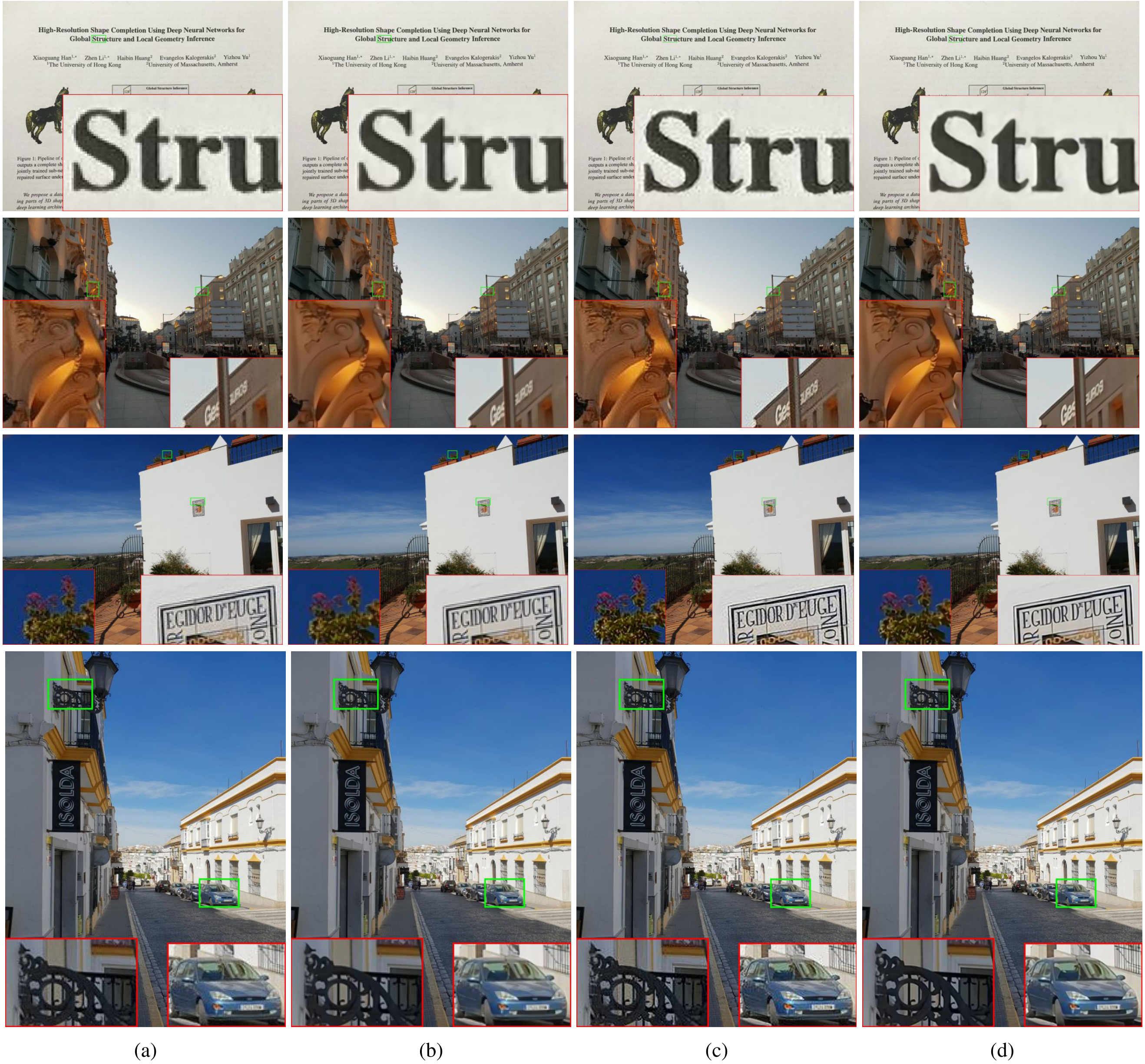


FIGURE 8. (a) Recompressed photos where the inputs are firstly taken (compressed) by *Galaxy 9th generation*, and then their outputs are recompressed in the *Facebook* site. (b) The outputs of the proposed AGARNet-EST where the input images are the recompressed images. (c) The ×2 super-resolution outputs from (a). (d) The ×2 super-resolution outputs from (b).

of characters. As shown in the figure, AGARNet can remove artifacts, although it is trained for Q from 10 to 80. We also visualize the super-resolved images using ×2 EDSR [21] in Fig. 7(c) and (d). It can be seen that the artifacts are boosted and become more noticeable in Fig. 7(c) and proposed method alleviates these artifacts in (d).

2) EXPERIMENTS ON JPEG RECOMPRESSION

As mentioned in the introduction, there are many recompressed images in the real-environments. Hence, we conduct experiments on the synthetic and real-world recompressed images. For experiments on synthetically recompressed images, we divide the Q range [10,80] into four parts as [10,30), [30,50), [50,70), [70,80] and name them as low, mid, high, ulthigh (ultra-high) image quality respectively. Then, we assume some practical recompression cases as in the first column of Table 3. We randomly select

TABLE 3. Comparison of average PSNR/SSIM on the recompressed images (Red: the best result). The evaluation is conducted on LIVE1.

Q-*Q*	JPEG	Baseline-S	Baseline-B	AGARNet-EST
low-high	29.70 / 0.859	30.50 / 0.874	31.60 / 0.893	31.70 / 0.894
low-ulthigh	29.44 / 0.846	29.76 / 0.853	31.33 / 0.883	31.41 / 0.884
mid-high	32.64 / 0.922	34.16 / 0.938	34.40 / 0.941	34.49 / 0.941
mid-ulthigh	32.22 / 0.915	33.38 / 0.929	34.16 / 0.937	34.28 / 0.937
high-mid	31.39 / 0.904	33.97 / 0.939	33.83 / 0.938	34.02 / 0.939
high-high	33.57 / 0.936	35.64 / 0.954	35.51 / 0.954	35.67 / 0.954
high-ulthigh	33.43 / 0.934	34.76 / 0.948	35.59 / 0.954	35.76 / 0.954
ulthigh-mid	31.89 / 0.909	33.73 / 0.930	33.67 / 0.930	33.76 / 0.930
ulthigh-high	33.33 / 0.932	35.56 / 0.952	35.39 / 0.952	35.55 / 0.952
ulthigh-ulthigh	36.13 / 0.959	38.16 / 0.971	38.00 / 0.971	38.22 / 0.971

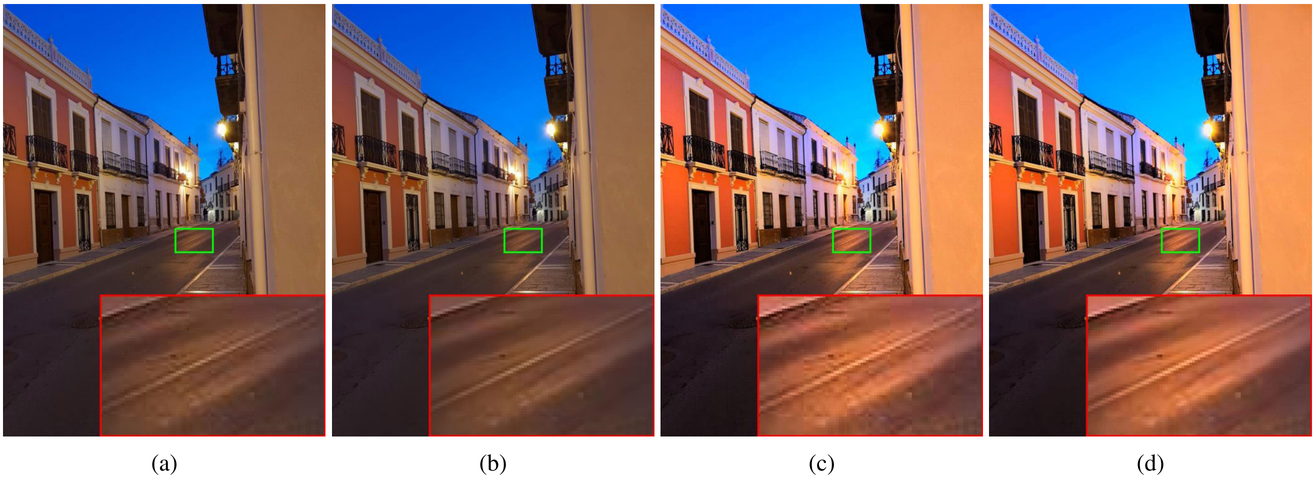
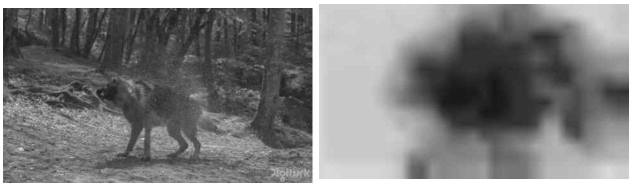


FIGURE 9. **(a) A recompressed photo where the input is firstly taken (compressed) by *Galaxy 9th generation*, and then its output is recompressed in the *Facebook* site. (b) The output of the proposed AGARNet-EST where the input image is the recompressed image. (c) The HDR output from (a). (d) The HDR output from (b).**

three pairs of the first and second Q because different pairs of Qs result in quite different images. The averaged results of three pairs for each case are presented in the table. We can see that proposed AGARNet-EST outperforms other methods in most of the cases. We believe that proposed Q_{EST} works robustly for the recompressed images, by estimating the appropriate Q that represents the actual degradation of consecutive compression. On the other hand, baseline-S does not present stable results because it relies only on the last Q and thus cannot reflect the actual degradation. Baseline-B also shows stable results, but it has inferior PSNR compared to the AGARNet-EST.

For experiments on real-world recompressed images, the input images are downloaded from *Facebook* [38][2] that mostly recompresses photos when uploading. Since the Qs are different from cameras and uploaded materials, the proposed AGARNet-EST, which has shown to provide actual degradation, is employed for reducing the recompression artifacts. Moreover, we present the visualized results of the image enhancement tasks such as super-resolution [21] and high dynamic range (HDR) [39] where the inputs of each task are recompressed images (downloaded from *Facebook*) and the preprocessed images using the proposed AGARNet-EST. It can be seen from Fig. 8 that the recompressed images contain unpleasant compression artifacts, which are boosted when the images are super-resolved with ×2 EDSR. On the other hand, AGARNet-EST can remove compression artifacts, and the results of super-resolution are more pleasant. We also present figures when the recompressed image is processed with HDR imaging [39] in Fig. 9. As shown in the figure, the artifacts are more salient when the artifacts regions are extended to HDR. The proposed AGARNet-EST can suppress the prominent artifacts in the recompressed image.



(a) Compressed frame (b) The spatially variant $\hat{Q}_{map}$

FIGURE 10. **(a) A 21st frame of *ShakeNDry* compressed with MPEG-2. (b) The output of corresponding frame using proposed estimator.**

TABLE 4. **Comparison of average PSNR/SSIM on the MPEG-2 to JPEG transcoded frames (Red: the best result). The evaluation is conducted on ShakNDry 30 frames. *CBR* and *Q* are constant bit rate of MPEG-2 and quality factor of JPEG respectively.**

CBR-Q	1Mbps-60	1Mbps-80	4Mbps-60	4Mbps-80
MPEG-2	35.34 / 0.8872	35.34 / 0.8872	35.55 / 0.8894	35.55 / 0.8894
MPEG-2 to JPEG	34.72 / 0.8798	35.10 / 0.8847	34.87 / 0.8820	35.27 / 0.8868
DNCNN-BW	35.30 / 0.8915	35.57 / 0.8942	35.45 / 0.8936	35.75 / 0.8963
Baseline-B	35.41 / 0.8947	35.52 / 0.8941	35.56 / 0.8968	35.70 / 0.8962
Baseline-S	35.25 / 0.8901	35.36 / 0.8899	35.40 / 0.8921	35.55 / 0.8921
AGARNet-EST	35.39 / 0.8932	35.61 / 0.8945	35.55 / 0.8952	35.81 / 0.8967
AGARNet-ESP	35.57 / 0.8959	35.80 / 0.8977	35.73 / 0.8980	35.99 / 0.8998

3) EXPERIMENTS ON VIDEO TO JPEG RECOMPRESSION

In this paragraph, we assume that compressed video frames are recompressed to JPEG images (transcoded), which is the case that we capture a frame from YouTube or other video streaming services. For generating the transcoded datasets, we first compress raw videos with MPEG-2 [40] and recompress video frames with JPEG. Precisely, we set two constant bitrates (*CBR*) for MPEG-2 as 1 Mbps and 4 Mbps, and two Qs for JPEG as 60 and 80. The test video is *ShakeNDry* where most of the frames show a dog shaking his/her body to get rid of water as in Fig. 10(a). The image quality of compressed *ShakeNDry* is quite different from region to region because the dog region contains severe artifacts due to the large motion. The average results of transcoded frames are listed in Table 4, where we can see that the proposed methods show robust results even though they are trained for the single compression

[2] Specifically, the images are the recompressed ones, as we had uploaded them (recompressed at this time) on a private account, and later downloaded them.

case. Moreover, the proposed AGARNet-ESP can reduce the MPEG-2 compression artifacts by comparing the result of AGARNet-ESP with MPEG-2 (not transcoded images). Since spatially variant $\hat{Q}_{map}$ provides the region-wise actual degradations, the AGARNet-ESP exceeds AGARNet-EST and other methods. Lastly, the estimated spatially variant $\hat{Q}_{map}$ is provided in Fig 10(b), which shows a plausible result that the dog regions have lower-qualities.

TABLE 5. Comparison of average PSNR for varying the architectures (Red: the best result). The evaluations are conducted on LIVE1 compressed in *Q* 20 without applying any gate process.

Method	pixel	DCT		PSNR
		Conv Type	IDCT Loss	
w/o Pixel	✗	3D Conv	✓	31.41
w/o DCT	✓	✗	✗	31.80
w/ Group Conv	✓	Group Conv	✓	31.84
w/ 1 × 1 Conv	✓	1 × 1 Conv	✓	31.98
w/o IDCT Loss	✓	3D Conv	✗	31.85
Proposed	✓	3D Conv	✓	32.00

4) ABLATION STUDY

In this paragraph, we conduct ablation study about the architecture of reconstruction network and gate process. First, we investigate the effect of dual-domain process and 3D convolution as in Table 5. The detail description of each method is listed as:

- w/o Pixel: the network does not include the F_{pix} network, *i.e.*, it processes only in the DCT-domain and allocates more training parameters to F_{dct}.
- w/o DCT: the network does not include the F_{dct} network, *i.e.*, it processes only in the pixel-domain and allocates more training parameters to F_{pix}.
- w/ Group Conv: the network is dual-domain network, but the 3D convolution is replaced to group convolution where the generated feature maps are $H/8 \times W/8 \times 64N$ tensors and the group is a set of 64 DCT coefficients.
- w/ 1 × 1 Conv: the network is dual-domain network, but the 3D convolution is replaced to 1 × 1 convolution where the generated feature maps are $H/8 \times W/8 \times 64N$ tensors.
- w/o IDCT Loss: the network is dual-domain network with 3D convolution, but it does not include IDCT loss.

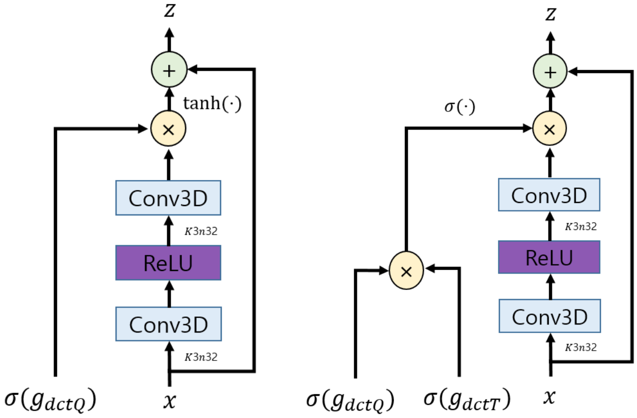
We find that processing in the dual-domain with the IDCT loss and 3D convolution provides the best architecture for compression artifacts removal. Although "w/ 1×1 Conv" has comparable performance to the proposed method, it requires lots of parameters about 150 times larger than 3D convolution. Specifically "w/ 1 × 1 Conv" requires $64N \times 64N$ training parameter for each convolution between feature maps where 3D convolution needs $N \times 3 \times 3 \times N$.

We also investigate the effects of the proposed gating scheme and its variations in Table 6. The detail explanation of each method is described as:

- Gate1: the Gate-ResBlock is only applied to the ×1/4 scaled feature maps in F_{pix}, and F_{dct} does not include any gate-process.

TABLE 6. Comparison of average PSNR for varying the method of providing conditional input (*Q*) (Red: the best result, Blue: the second best). The evaluations are conducted on LIVE1.

Q	10	20	30	40	50	60	70	80	Avg.
Baseline-S	29.65	32.00	33.38	34.37	35.19	36.03	37.14	38.75	34.56
Baseline-B	29.55	31.92	33.32	34.30	35.09	35.90	36.99	38.58	34.46
Concat	29.53	31.90	33.31	34.29	35.09	35.89	36.99	38.63	34.45
Gate1	29.62	31.99	33.40	34.38	35.19	36.01	37.11	38.71	34.55
Gate2	29.62	31.98	33.40	34.39	35.20	36.01	37.12	38.74	34.56
Gate3	29.65	32.01	33.42	34.41	35.21	36.03	37.13	38.74	34.57
Gate4	29.64	32.01	33.42	34.41	35.21	36.02	37.13	38.73	34.57
Gate5	29.64	32.02	33.43	34.42	35.23	36.04	37.15	38.75	34.58
Proposed	29.64	32.02	33.44	34.42	35.24	36.06	37.17	38.81	34.60



(a) 3DG-ResBlock of Gate3 (b) 3DG-ResBlock of Gate4

FIGURE 11. (a) and (b) are the proposed 3D Gate-ResBlocks for the ablation study with corresponding 3D kernel size (*K*), and number of features (*n*). The symbol × denotes element-wise multiplication.

- Gate2: the Gate-ResBlock is applied to all F_{pix} network, and F_{dct} does not include any gate-process.
- Gate3: the Gate-ResBlock is applied to all F_{pix} network and the 3DG-ResBlock is replaced to Fig. 11(a) in F_{dct}, which is an extended 3D gating scheme of Gate-ResBlock.
- Gate4: the Gate-ResBlock is applied to all F_{pix} network, and the 3DG-ResBlock is replaced to Fig. 11(b) in F_{dct}.
- Gate5: the sliding dimensions of the right C_1 in equation 8, which are spatial and frequency dimensions, change to the spatial and feature dimensions.

It can be seen that the conventional concatenation method [23], [24] cannot work for JPEG artifacts removal with the proposed scheme by comparing to the Baseline-B. Moreover, we observe that the proposed gate process applying to the part of the decoder ("Gate1") can provide certain improvement. It is notable that the proposed gating scheme achieves the best performance for most of Qs.

V. CONCLUSION

We have proposed a new adaptively gated compression artifacts removal network which robustly works for a wide range of quality factor. Unlike conventional methods, the proposed method trains a single network whose parameters are not changed according to the variation of compression qualities, but the learned features are adaptively scaled instead. Specifically, proposed (3D) Gate-ResBlock, which gates the feature map and acts as an attention module according to quality factor, makes the reconstruction network pixel-wise adaptive. We have tested the proposed method on single compressed and recompressed images, and the results show that our method yields the best performance among state-of-the-art methods. We believe that the proposed method is practical in that it works for a wide range of compression rates and also for the recompressed/transcoded images with unknown qualities. We will make our codes and datasets publicly available at https://github.com/terryoo/AGARNet for further research and comparisons.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

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

[4] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2392–2399.

[5] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian Denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[6] C. Dong, Y. Deng, C. C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 576–584.

[7] S. Lefkimmiatis, "Universal denoising networks: A novel CNN architecture for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3204–3213.

[8] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, "Burst denoising with kernel prediction networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018.

[9] J. W. Soh, J. Park, Y. Kim, B. Ahn, H.-S. Lee, Y.-S. Moon, and N. I. Cho, "Reduction of video compression artifacts based on deep temporal networks," *IEEE Access*, vol. 6, pp. 63094–63106, 2018.

[10] X. Liu, W. Lu, W. Liu, S. Luo, Y. Liang, and M. Li, "Image deblocking detection based on a convolutional neural network," *IEEE Access*, vol. 7, pp. 26432–26439, 2019.

[11] J. Guan, R. Lai, and A. Xiong, "Learning spatiotemporal features for single image stripe noise removal," *IEEE Access*, vol. 7, pp. 144489–144499, 2019.

[12] J. Chen, G. Zhang, S. Xu, and H. Yu, "A blind CNN denoising model for random-valued impulse noise," *IEEE Access*, vol. 7, pp. 124647–124661, 2019.

[13] J. Guo and H. Chao, "One-to-many network for visually pleasing compression artifacts reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4867–4876.

[14] L. Cavigelli, P. Hager, and L. Benini, "CAS-CNN: A deep convolutional neural network for image compression artifact suppression," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 752–759.

[15] L. Galteri, L. Seidenari, M. Bertini, and A. D. Bimbo, "Deep generative adversarial compression artifact removal," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4826–4835.

[16] N. Kwak, J. Yoo, and S.-H. Lee, "Image restoration by estimating frequency distribution of local patches," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6684–6692.

[17] R. Yang, M. Xu, Z. Wang, and T. Li, "Multi-frame quality enhancement for compressed video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6664–6673.

[18] G. Lu, W. Ouyang, D. Xu, X. Zhang, Z. Gao, and M.-T. Sun, "Deep Kalman filtering network for video compression artifact reduction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 568–584.

[19] P. Liu, H. Zhang, W. Lian, and W. Zuo, "Multi-level wavelet convolutional neural networks," *IEEE Access*, vol. 7, pp. 74973–74985, 2019.

[20] S. Yu and J. Jeong, "Local excitation network for restoring a JPEG-compressed image," *IEEE Access*, vol. 7, pp. 138032–138042, 2019.

[21] 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.

[22] Y. Kim, J. W. Soh, J. Park, B. Ahn, H.-S. Lee, Y.-S. Moon, and N. I. Cho, "A pseudo-blind convolutional neural network for the reduction of compression artifacts," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[23] 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.

[24] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, Sep. 2018.

[25] Y. Kim, J. W. Soh, and N. I. Cho, "Adaptively tuning a convolutional neural network by gate process for image denoising," *IEEE Access*, vol. 7, pp. 63447–63456, 2019.

[26] 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.

[27] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[28] 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. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 286–301.

[29] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, L. Zhang, B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Ntire 2017 challenge on single image super-resolution: Methods and results," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1110–1121.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–41.

[31] X. Zhang, Y. Lu, J. Liu, and B. Dong, "Dynamically unfolding recurrent restorer: A moving endpoint control method for image restoration," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

[32] H. Sheikh, M. Sabir, and A. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3440–3451, Nov. 2006.

[33] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5197–5206.

[34] X.-J. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," 2016, *arXiv:1603.09056*. [Online]. Available: https://arxiv.org/abs/1603.09056

[35] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: A persistent memory network for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4539–4547.

[36] J. Guo and H. Chao, "Building dual-domain representations for compression artifacts reduction," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 628–644.

[37] X. Zhang, W. Yang, Y. Hu, and J. Liu, "Dmcnn: Dual-domain multi-scale convolutional neural network for compression artifacts removal," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 390–394.

[38] *Facebook*. Accessed: Jan. 27, 2020. [Online]. Available: https://www.facebook.com

[39] R. P. Kovaleski and M. M. Oliveira, "High-quality reverse tone mapping for a wide range of exposures," in *Proc. 27th SIBGRAPI Conf. Graph., Patterns Images*, Aug. 2014, pp. 49–56.

[40] *Generic Coding of Moving Pictures and Associated Audio Information—Part 2: Video*, document International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) JTC 1, Rec. H. 262 and ISO/IEC 13 818-2 (MPEG-2 Video), Union-Telecommunication, International Telecommunication, 1994.