2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)

Rethinking Data Augmentation for Image Super-resolution: A Comprehensive Analysis and a New Strategy

Jaejun Yoo*
EPFL.
jaejun.yoo88@gmail.com

Namhyuk Ahn*
Ajou University
aa0dfg@ajou.ac.kr

Kyung-Ah Sohn†
Ajou University
kasohn@ajou.ac.kr

Abstract

*Data augmentation is an effective way to improve the performance of deep networks. Unfortunately, current methods are mostly developed for high-level vision tasks (*e.g.*, classification) and few are studied for low-level vision tasks (*e.g.*, image restoration). In this paper, we provide a comprehensive analysis of the existing augmentation methods applied to the super-resolution task. We find that the methods discarding or manipulating the pixels or features too much hamper the image restoration, where the spatial relationship is very important. Based on our analyses, we propose **CutBlur** that cuts a low-resolution patch and pastes it to the corresponding high-resolution image region and vice versa. The key intuition of CutBlur is to enable a model to learn not only "how" but also "where" to super-resolve an image. By doing so, the model can understand "how much", instead of blindly learning to apply super-resolution to every given pixel. Our method consistently and significantly improves the performance across various scenarios, especially when the model size is big and the data is collected under real-world environments. We also show that our method improves other low-level vision tasks, such as denoising and compression artifact removal.*

1. Introduction

Data augmentation (DA) is one of the most practical ways to enhance model performance without additional computation cost in the test phase. While various DA methods [7, 29, 30, 13] have been proposed in several high-level vision tasks, DA in low-level vision has been scarcely investigated. Instead, many image restoration studies, such as super-resolution (SR), have relied on the synthetic datasets [22], which we can easily increase the number of training samples by simulating the system degradation functions (*e.g.*, using the bicubic kernel for SR).

Because of the gap between a simulated and a real data distribution, however, models that are trained on simulated datasets do not exhibit optimal performance in the real environments [4]. Several recent studies have proposed to mitigate the problem by collecting real-world datasets [1, 4, 32]. However, in many cases, it is often very time-consuming and expensive to obtain a large number of such data. Although this is where DA can play an important role, only a handful of studies have been performed [9, 24].

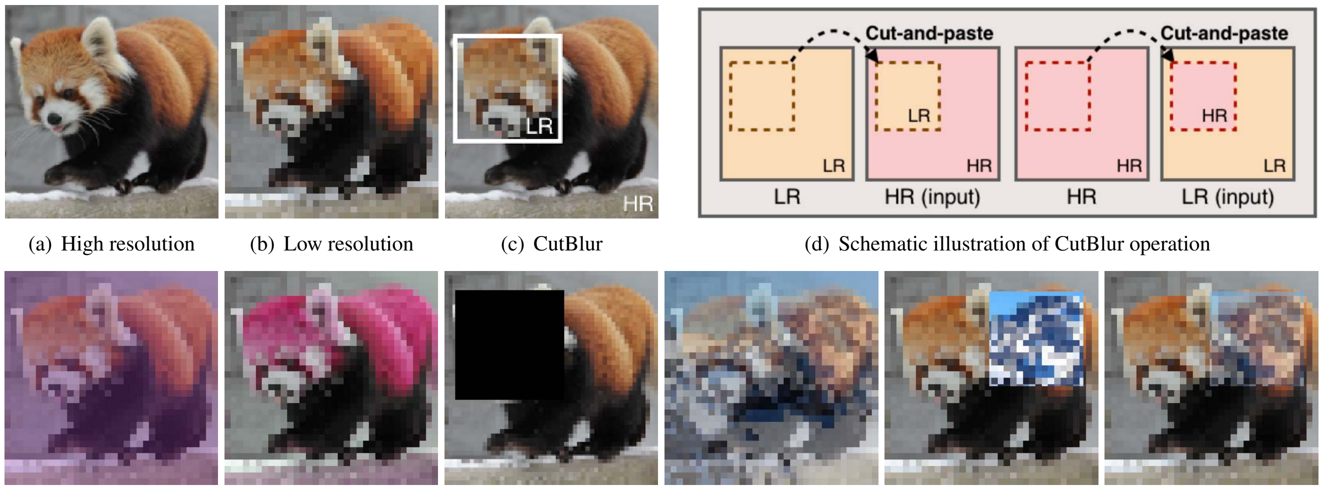
Radu *et al.* [24] was the first to study various techniques to improve the performance of example-based single-image super-resolution (SISR), one of which was data augmentation. Using rotation and flipping, they reported consistent improvements across models and datasets. Still, they only studied simple geometric manipulations with traditional SR models [12, 23] and a very shallow learning-based model, SRCNN [8]. To the best of our knowledge, Feng *et al.* [9] is the only work that analyzed a recent DA method (Mixup [30]) in the example-based SISR problem. However, the authors provided only a limited observation using a single U-Net-like architecture and tested the method with a single dataset (RealSR [4]).

To better understand DA methods in low-level vision, we provide a comprehensive analysis on the effect of various DA methods that are originally developed for high-level vision tasks (Section 2). We first categorize the existing augmentation techniques into two groups depending on where the method is applied; pixel-domain [7, 29, 30] and feature-domain [11, 10, 26, 27]. When directly applied to SISR, we find that some methods harm the image restoration results and even hampers the training, especially when a method largely induces the loss or confusion of spatial information between nearby pixels (*e.g.*, Cutout [7] and feature-domain methods). Interestingly, basic manipulations like RGB permutation that do not cause a severe spatial distortion provide better improvements than the ones which induce unrealistic patterns or a sharp transition of the structure (*e.g.*, Mixup [30] and CutMix [29]).

Based on our analyses, we propose **CutBlur**, a new augmentation method that is specifically designed for the low-level vision tasks. CutBlur cut and paste a low resolution

* indicates equal contribution. Most work was done in NAVER Corp.
† indicates corresponding author.

2575-7075/20/$31.00 ©2020 IEEE
DOI 10.1109/CVPR42600.2020.00840



(a) High resolution (b) Low resolution (c) CutBlur (d) Schematic illustration of CutBlur operation

(e) Blend (f) RGB permute (g) Cutout (25%) [7] (h) Mixup [30] (i) CutMix [29] (j) CutMixup

Figure 1. Data augmentation methods. **(Top)** An illustrative example of our proposed method, CutBlur. CutBlur generates an augmented image by cut-and-pasting the low resolution (LR) input image onto the ground-truth high resolution (HR) image region and vice versa (Section 3). **(Bottom)** Illustrative examples of the existing augmentation techniques and a new variation of CutMix and Mixup, CutMixup.

(LR) image patch into its corresponding ground-truth high resolution (HR) image patch (Figure 1). By having partially LR and partially HR pixel distributions with a random ratio in a single image, CutBlur enjoys the regularization effect by encouraging a model to learn both *"how"* and *"where"* to super-resolve the image. One nice side effect of this is that the model also learns *"how much"* it should apply super-resolution on every local part of a given image. While trying to find a mapping that can simultaneously maintain the input HR region and super-resolve the other LR region, the model adaptively learns to super-resolve an image.

Thanks to this unique property, CutBlur prevents over-sharpening of SR models, which can be commonly found in real-world applications (Section 4.3). In addition, we show that the performance can be further boosted by applying several curated DA methods together during the training phase, which we call **mixture of augmentations (MoA)** (Section 3). Our experiments demonstrate that the proposed strategy significantly and consistently improves the model performance over various models and datasets. Our contributions are summarized as follows:

1. To the best of our knowledge, we are the first to provide comprehensive analysis of recent data augmentation methods when directly applied to the SISR task.

2. We propose a new DA method, **CutBlur**, which can reduce unrealistic distortions by regularizing a model to learn not only "how" but also "where" to apply the super-resolution to a given image.

3. Our mixed strategy shows consistent and significant improvements in the SR task, **achieving state-of-the-art (SOTA) performance in RealSR** [4].

2. Data augmentation analysis

In this section, we analyze existing augmentation methods and compare their performances when applied to EDSR [15], which is our baseline super-resolution model. We train EDSR from scratch with DIV2K [2] dataset or RealSR [4] dataset. We used the authors' official code.

2.1. Prior arts

DA in pixel space. There have been many studies to augment images in high-level vision tasks [7, 29, 30] (Figure 1). Mixup [30] blends two images to generate an unseen training sample. Cutout and its variants [7, 34] drop a randomly selected region of an image. Addressing that Cutout cannot fully exploit the training data, CutMix [29] replaces the random region with another image. Recently, AutoAugment and its variant [6, 16] have been proposed to learn the best augmentation policy for a given task and dataset.

DA in feature space. DA methods manipulating CNN features have been proposed [5, 10, 11, 19, 26, 27] and can be categorized into three groups: 1) feature mixing, 2) shaking, and 3) dropping. Like Mixup, Manifold Mixup [26] mixes both input image and the latent features. Shake-shake [10] and ShakeDrop [27] perform a stochastic affine transformation to the features. Finally, following the spirit of Dropout [19], a lot of feature dropping strategies [5, 11, 25] have been proposed to boost the generalization of a model.

DA in super-resolution. A simple geometric manipulation, such as rotation and flipping, has been widely used in SR models [24]. Recently, Feng *et al.* [9] showed that Mixup can alleviate the overfitting problem of SR models [4].

2.2. Analysis of existing DA methods

The core idea of many augmentation methods is to partially block or confuse the training signal so that the model acquires more generalization power. However, unlike the high-level tasks, such as classification, where a model should learn to abstract an image, the local and global relationships among pixels are especially more important in the low-level vision tasks, such as denoising and super-resolution. Considering this characteristic, it is unsurprising that DA methods, which lose the spatial information, limit the model's ability to restore an image. Indeed, we observe that the methods dropping the information [5, 11, 25] are detrimental to the SR performance and especially harmful in the feature space, which has larger receptive fields. Every feature augmentation method significantly drops the performance. Here, we put off the results of every DA method that degrades the performance in the supplementary material.

On the other hand, DA methods in pixel space bring some improvements when applied carefully (Table 1)[1]. For example, Cutout [7] with default setting (dropping 25% of pixels in a rectangular shape) significantly degrades the original performance by 0.1 dB. However, we find that Cutout gives a positive effect (DIV2K: +0.01 dB and RealSR: +0.06 dB) when applied with 0.1% ratio and erasing random pixels instead of a rectangular region. Note that this drops only 2∼3 pixels when using a 48×48 input patch.

CutMix [29] shows a marginal improvement (Table 1), and we hypothesize that this happens because CutMix generates a drastically sharp transition of image context making boundaries. Mixup improves the performance but it mingles the context of two different images, which can confuse the model. To alleviate these issues, we create a variation of CutMix and Mixup, which we call CutMixup (below the dashed line, Figure 1). Interestingly, it gives a better improvement on our baseline. By getting the best of both methods, CutMixUp benefits from minimizing the boundary effect as well as the ratio of the mixed contexts.

Based on these observations, we further test a set of basic operations such as RGB permutation and Blend (adding a constant value) that do not incur any structural change in an image. (For more details, please see our supplementary material.) These simple methods show promising results in the synthetic DIV2K dataset and a big improvement in the RealSR dataset, which is more difficult. These results empirically prove our hypothesis, which naturally leads us to a new augmentation method, CutBlur. When applied, CutBlur not only improves the performance (Table 1) but provides some good properties and synergy (Section 3.2), which cannot be obtained by the other DA methods.

[1]For every experiment, we only used geometric DA methods, flip and rotation, which is the default setting of EDSR. Here, to solely analyze the effect of the DA methods, we did not use the ×2 pre-trained model.

Table 1. PSNR (dB) comparison of different data augmentation methods in super-resolution. We report the baseline model (EDSR [15]) performance that is trained on DIV2K (×4) [2] and RealSR (×4) [4]. The models are trained from scratch. δ denotes the performance gap between with and without augmentation.

Method	DIV2K (δ)	RealSR (δ)
EDSR	29.21 (+0.00)	28.89 (+0.00)
Cutout [7] (0.1%)	29.22 (+0.01)	28.95 (+0.06)
CutMix [29]	29.22 (+0.01)	28.89 (+0.00)
Mixup [30]	29.26 (+0.05)	28.98 (+0.09)
CutMixup	29.27 (+0.06)	29.03 (+0.14)
RGB perm.	29.30 (+0.09)	29.02 (+0.13)
Blend	29.23 (+0.02)	29.03 (+0.14)
CutBlur	**29.26 (+0.05)**	**29.12 (+0.23)**
All DA's (random)	**29.30 (+0.09)**	**29.16 (+0.27)**

3. CutBlur

In this section, we describe the CutBlur, a new augmentation method that is designed for the super-resolution task.

3.1. Algorithm

Let $x_{LR} \in \mathbb{R}^{W\times H\times C}$ and $x_{HR} \in \mathbb{R}^{sW\times sH\times C}$ are LR and HR image patches and s denotes a scale factor in the SR. As illustrated in Figure 1, because CutBlur requires to match the resolution of x_{LR} and x_{HR}, we first upsample x_{LR} by s times using a bicubic kernel, x^s_{LR}. The goal of CutBlur is to generate a pair of new training samples $(\hat{x}_{HR\to LR}, \hat{x}_{LR\to HR})$ by cut-and-pasting the random region of x_{HR} into the corresponding x^s_{LR} and vice versa:

$$\begin{aligned} \hat{x}_{HR\to LR} &= \mathbf{M} \odot x_{HR} + (\mathbf{1}-\mathbf{M}) \odot x^s_{LR} \\ \hat{x}_{LR\to HR} &= \mathbf{M} \odot x^s_{LR} + (\mathbf{1}-\mathbf{M}) \odot x_{HR} \end{aligned} \quad (1)$$

where $\mathbf{M} \in \{0,1\}^{sW\times sH}$ denotes a binary mask indicating where to replace, $\mathbf{1}$ is a binary mask filled with ones, and $\odot$ is element-wise multiplication. For sampling the mask and its coordinates, we follow the original CutMix [29].

3.2. Discussion

Why CutBlur works for SR? In the previous analysis (Section 2.2), we found that sharp transitions or mixed image contents within an image patch, or losing the relationships of pixels can degrade SR performance. Therefore, a good DA method for SR should not make unrealistic patterns or information loss while it has to serve as a good regularizer to SR models.

CutBlur satisfies these conditions because it performs cut-and-paste between the LR and HR image patches of the same content. By putting the LR (resp. HR) image region onto the corresponding HR (resp. LR) image region, it can minimize the boundary effect, which majorly comes from

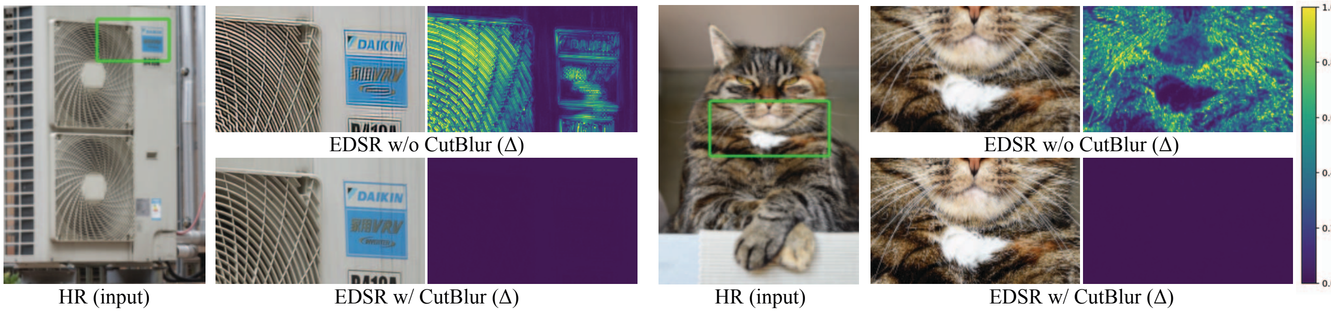


Figure 2. Qualitative comparison of the baseline with and without CutBlur when the network takes the HR image as an input during the inference time. Δ is the absolute residual intensity map between the network output and the ground-truth HR image. CutBlur successfully preserves the entire structure while the baseline generates unrealistic artifacts (**left**) or incorrect outputs (**right**).

a mismatch between the image contents (*e.g.*, Cutout and CutMix). Unlike Cutout, CutBlur can utilize the entire image information while it enjoys the regularization effect due to the varied samples of random HR ratios and locations.

What does the model learn with CutBlur? Similar to the other DA methods that prevent classification models from over-confidently making a decision (*e.g.*, label smoothing [21]), CutBlur prevents the SR model from over-sharpening an image and helps it to super-resolve only the necessary region. This can be demonstrated by performing the experiments with some artificial setups, where we provide the CutBlur-trained SR model with an HR image (Figure 2) or CutBlurred LR image (Figure 3) as input.

When the SR model takes HR images at the test phase, it commonly outputs over-sharpened predictions, especially where the edges are (Figure 2). CutBlur can resolve this issue by directly providing such examples to the model during the training phase. Not only does CutBlur mitigate the over-sharpening problem, but it enhances the SR performance on the other LR regions, thanks to the regularization effect (Figure 3). Note that the residual intensity has significantly decreased in the CutBlur model. We hypothesize that this enhancement comes from constraining the SR model to discriminatively apply super-resolution to the image. Now the model has to simultaneously learn both "how" and "where" to super-resolve an image, and this leads the model to learn "how much" it should apply super-resolution, which provides a beneficial regularization effect to the training.

Of course it is unfair to compare the models that have been trained with and without such images. However, we argue that these scenarios are not just the artificial experimental setups but indeed exist in the real-world (*e.g.*, out-of-focus images). We will discuss this more in detail with several real examples in Section 4.3.

CutBlur vs. Giving HR inputs during training. To make a model learn an identity mapping, instead of using CutBlur, one can easily think of providing HR images as an input of the network during the training phase. With the EDSR model, CutBlur trained model (29.04 dB) showed better

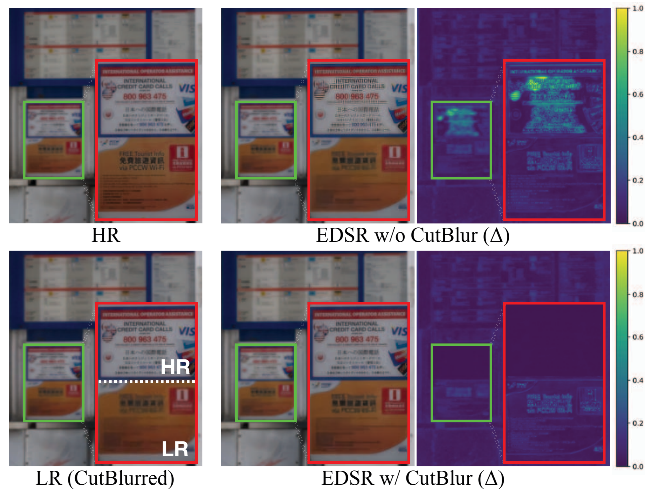
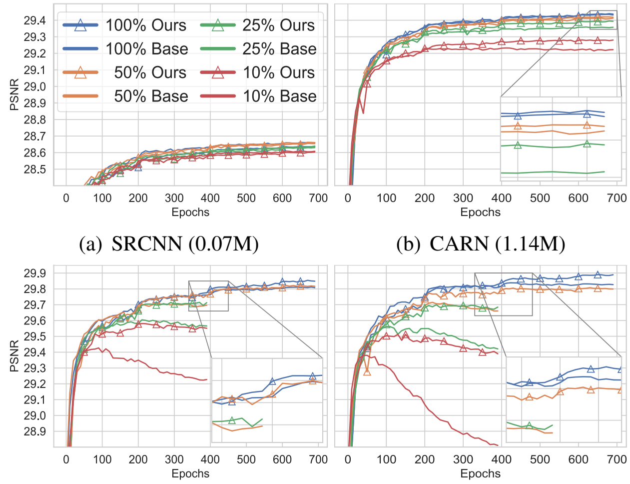


Figure 3. Qualitative comparison of the baseline and CutBlur model outputs when the input is augmented by CutBlur. Δ is the absolute residual intensity map between the network output and the ground-truth HR image. Unlike the baseline (**top right**), CutBlur model not only resolves the HR region but reduces Δ of the other LR input area as well (**bottom right**).

performance in PSNR than naïvely providing the HR images (28.87 dB) to the network. (The detailed setups can be found in the supplementary material.) This is because CutBlur is more general in that HR inputs are its special case ($\mathbf{M} = \mathbf{0}$ or $\mathbf{1}$). On the other hand, giving HR inputs can never simulate the mixed distribution of LR and HR pixels so that the network can only learn "how", not "where" to super-resolve an image.

Mixture of augmentation (MoA). To push the limits of performance gains, we integrate various DA methods into a single framework. For each training iteration, the model first decides with probability p whether to apply DA on inputs or not. If yes, it randomly selects a method among the DA pool. Based on our analysis, we use all the pixel-domain DA methods discussed in Table 1 while excluding all feature-domain DA methods. Here, we set $p = 1.0$ as a default. From now on, unless it is specified, we report all the experimental results using this MoA strategy.



(a) SRCNN (0.07M) (b) CARN (1.14M)

(c) RCAN (15.6M) (d) EDSR (43.2M)

Figure 4. PSNR (dB) comparison on ten DIV2K ($\times 4$) validation images during training for different data size (%). Ours are shown by triangular markers. Zoomed curves are displayed (inlets).

4. Experiments

In this section, we describe our experimental setups and compare the model performance with and without applying our method. We compare the super-resolution (SR) performance under various model sizes, dataset sizes (Section 4.1), and benchmark datasets (Section 4.2). Finally, we apply our method to the other low-level vision tasks, such as Gaussian denoising and JPEG artifact removal, to show the potential extensibility of our method (Section 4.4).[2,3]

Baselines. We use four SR models: SRCNN [8], CARN [3], RCAN [33], and EDSR [15]. These models have different numbers of parameters from 0.07M to 43.2M (million). For fair comparisons, every model is trained from scratch using the authors' official code unless mentioned otherwise.

Dataset and evaluation. We use the DIV2K [2] dataset or a recently proposed real-world SR dataset, RealSR [4] for training. For evaluation, we use Set14 [28], Urban100 [14], Manga109 [17], and test images of the RealSR dataset. Here, PSNR and SSIM are calculated on the Y channel only except the color image denoising task.

4.1. Study on different models and datasets

Various model sizes. It is generally known that a large model benefits more from augmentation than a small model does. To see whether this is true in SR, we investigate how the model size affects the maximum performance gain using our strategy. Here, we set the probability of applying augmentations differently depending on the model size, $p = 0.2$ for the small models (SRCNN and CARN) and $p = 1.0$ for the large models (RCAN and EDSR). With the small models, our proposed method provides no benefit or marginally increases the performance (Table 2). This demonstrates the severe underfitting of the small models, where the effect of DA is minimal due to the lacking capacity. On the other hand, it consistently improves the performance of RCAN and EDSR, which have enough capacity to exploit the augmented information.

[2]The overall experiments were conducted on NSML [20] platform.
[3]Our code is available at clovaai/cutblur

Table 2. PSNR (dB) comparison on DIV2K ($\times 4$) validation set by varying the model and the size of dataset for training. Note that the number of RealSR dataset, which is more difficult to collect, is around 15% of DIV2K dataset.

Model	Params.	Training Data Size				
		100%	50%	25%	**15%**	**10%**
SRCNN + proposed	0.07M	27.95 -0.02	27.95 -0.01	27.95 -0.02	27.93 -0.02	27.91 -0.01
CARN + proposed	1.14M	28.80 +0.00	28.77 **+0.01**	28.72 **+0.02**	28.67 **+0.03**	28.60 **+0.04**
RCAN + proposed	15.6M	29.22 **+0.08**	29.06 **+0.16**	29.01 **+0.11**	28.90 **+0.13**	28.82 **+0.14**
EDSR + proposed	43.2M	29.21 **+0.08**	29.10 **+0.08**	28.97 **+0.10**	28.87 **+0.10**	28.77 **+0.11**

Various dataset sizes. We further investigate the model performance while decreasing the data size for training (Table 2). Here, we use 100%, 50%, 25%, 15% and 10% of the DIV2K dataset. SRCNN and CARN show none or marginal improvements with our method. This can be also seen by the validation curves while training (Figure 4a and 4b). On the other hand, our method brings a huge benefit to the RCAN and EDSR in all the settings. The performance gap between the baseline and our method becomes profound as the dataset size diminishes. RCAN trained on half of the dataset shows the same performance as the 100% baseline when applied with our method ($29.06 + 0.16 = 29.22$ dB). Our method gives an improvement of up to 0.16 dB when the dataset size is less than 50%. This tendency is observed in EDSR as well. This is important because 15% of the DIV2K dataset is similar to the size of the RealSR dataset, which is more expensive taken under real environments.

Our method also significantly improves the overfitting problem (Figure 4c and 4d). For example, if we use 25% of the training data, the large models easily overfit and this can be dramatically reduced by using our method (denoted by curves with the triangular marker of the same color).

4.2. Comparison on diverse benchmark dataset

We test our method on various benchmark datasets. For the synthetic dataset, we train the models using the DIV2K dataset and test them on Set14, Urban100, and Manga109. Here, we first pre-train the network with $\times 2$ scale dataset, then fine-tune on $\times 4$ scale images. For the realistic case, we train the models using the training set of the RealSR dataset

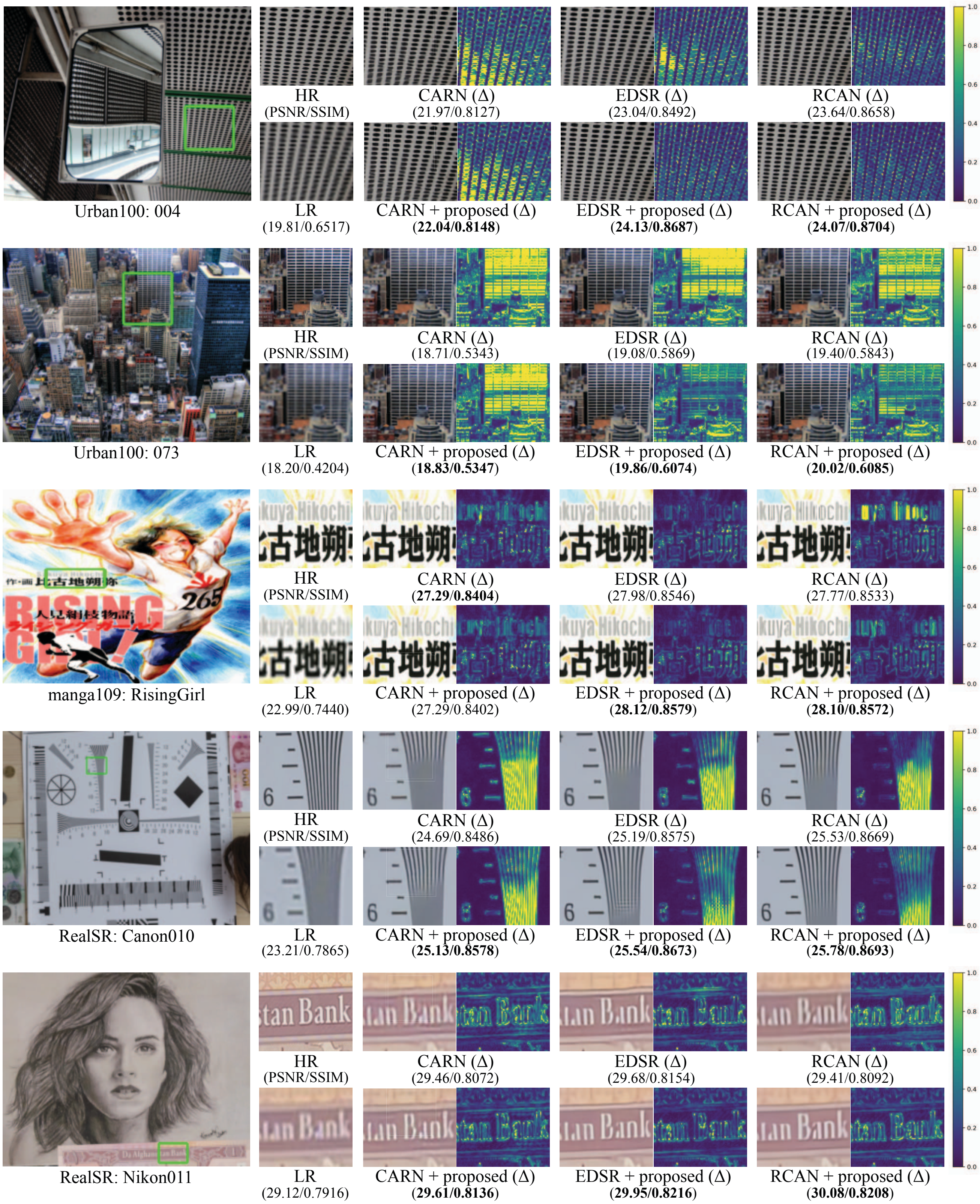


Figure 5. Qualitative comparison of using our proposed method on different datasets and tasks. Δ is the absolute residual intensity map between the network output and the ground-truth HR image.

Table 3. Quantitative comparison (PSNR / SSIM) on SR (scale $\times 4$) task in both synthetic and realistic settings. δ denotes the performance gap between with and without augmentation. For synthetic case, we perform the $\times 2$ scale pre-training.

Model	# Params.	Synthetic (DIV2K dataset)			Realistic (RealSR dataset)
		Set14 (δ)	Urban100 (δ)	Manga109 (δ)	RealSR (δ)
CARN	1.14M	28.48 (+0.00) / 0.7787	25.85 (+0.00) / 0.7779	30.17 (+0.00) / 0.9034	28.78 (+0.00) / 0.8134
+ proposed		28.48 (+0.00) / 0.7788	25.85 (+0.00) / 0.7780	30.16 (-0.01) / 0.9032	29.00 (+0.22) / 0.8204
RCAN	15.6M	28.86 (+0.00) / 0.7879	26.76 (+0.00) / 0.8062	31.24 (+0.00) / 0.9169	29.22 (+0.00) / 0.8254
+ proposed		**28.92 (+0.06) / 0.7895**	**26.93 (+0.17) /0.8106**	**31.46 (+0.22) / 0.9190**	**29.49 (+0.27) / 0.8307**
EDSR	43.2M	28.81 (+0.00) / 0.7871	26.66 (+0.00) / 0.8038	31.06 (+0.00) / 0.9151	28.89 (+0.00) / 0.8204
+ proposed		28.88 (+0.07) / 0.7886	26.80 (+0.14) / 0.8072	31.25 (+0.19) / 0.9163	29.16 (+0.27) / 0.8258

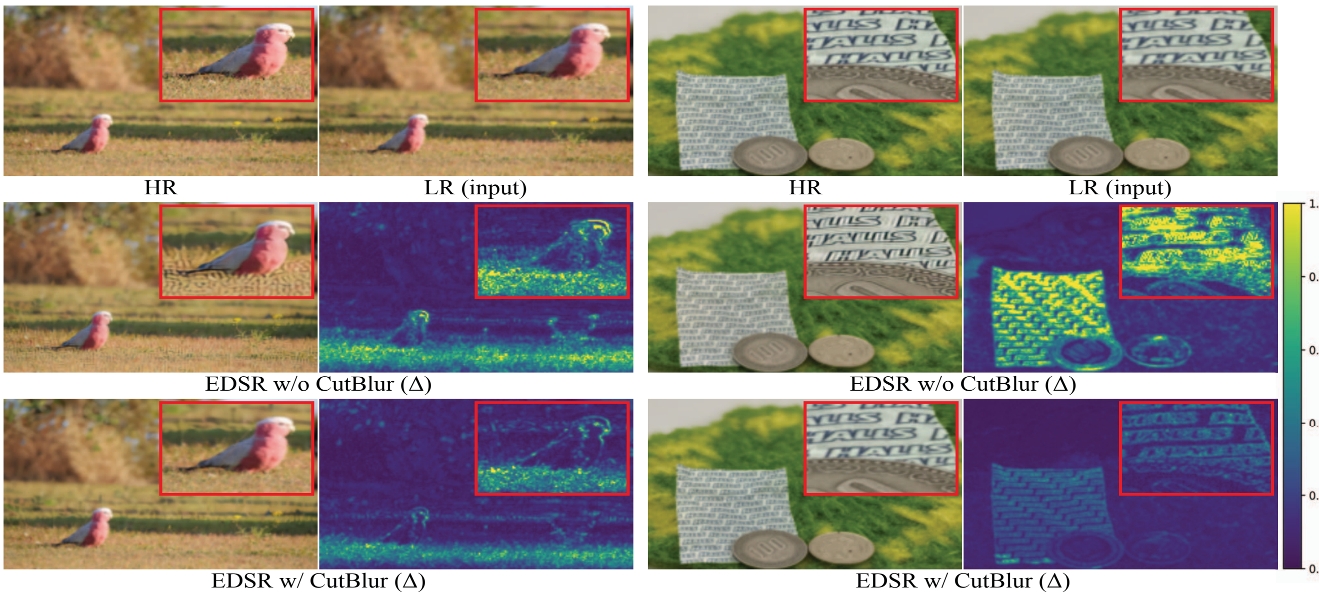


Figure 6. Qualitative comparison of the baseline and CutBlur model outputs. The inputs are the real-world out-of-focus photography ($\times 2$ bicubic downsampled), which are taken from a web (**left**) and captured by iPhone 11 Pro (**right**). Δ is the absolute residual intensity map between the network output and the ground-truth HR image. The baseline model over-sharpens the focused region resulting in unpleasant distortions while the CutBlur model effectively super-resolves the image without such a problem.

($\times 4$ scale) and test them on its unseen test images.

Our proposed method consistently gives a huge performance gain, especially when the models have large capacities (Table 3). In the RealSR dataset, which is a more realistic case, the performance gain of our method becomes larger, **increasing at least 0.22 dB for all models in PSNR**. We achieve the **SOTA performance (RCAN [33])** compared to the previous SOTA model (LP-KPN [4]: 28.92 dB / 0.8340). Note that our model **increase the PSNR by 0.57 dB** with a comparable SSIM score. Surprisingly, **the lightest model (CARN [3]: 1.14M) can already beat the LP-KPN (5.13M)** in PSNR with only 22% of the parameters.

Figure 5 shows the qualitative comparison between the models with and without applying our DA method. In the Urban100 examples (1st and 2nd rows in Figure 5), RCAN and EDSR benefit from the increased performance and successfully resolve the aliasing patterns. This can be seen more clearly in the residual between the model-prediction and the ground-truth HR image. Such a tendency is consistently observed across different benchmark images. In RealSR dataset images, even the performance of the small model is boosted, especially when there are fine structures (4th row in Figure 5).

4.3. CutBlur in the wild

With the recent developments of devices like iPhone 11 Pro, they offer a variety of features, such as portrait images. Due to the different resolutions of the focused foreground and the out-focused background of the image, the baseline SR model shows degraded performance, while the CutBlur model does not (Figure 6). These are the very real-world examples, which are simulated by CutBlur. The baseline model adds unrealistic textures in the grass (left, Figure 6) and generates ghost artifacts around the characters and coin patterns (right, Figure 6). In contrast, the CutBlur model does not add any unrealistic distortion while it adequately super-resolves the foreground and background of the image.

Table 4. Performance comparison on the color Gaussian denoising task evaluated on the Kodak24 dataset. We train the model with both mild ($\sigma = 30$) and severe noises ($\sigma = 70$) and test on the mild setting. LPIPS [31] (lower is better) indicates the perceptual distance between the network output and the ground-truth.

Model	Train σ	Test ($\sigma = 30$)		
		PSNR ↑	SSIM ↑	LPIPS ↓
EDSR	30	31.92	0.8716	0.136
+ proposed		**+0.02**	**+0.0006**	**-0.004**
EDSR	70	27.38	0.7295	0.375
+ proposed		-2.51	**+0.0696**	**-0.193**

Table 5. Performance comparison on the color JPEG artifact removal task evaluated on the LIVE1 [18] dataset. We train the model with both mild ($q = 30$) and severe compression ($q = 10$) and test on the mild setting.

Model	Train q	Test ($q = 30$)		
		PSNR ↑	SSIM ↑	LPIPS ↓
EDSR	30	33.95	0.9227	0.118
+ proposed		-0.01	-0.0002	+0.001
EDSR	10	32.45	0.8992	0.154
+ proposed		**+0.97**	**+0.0187**	**-0.023**

4.4. Other low-level vision tasks

Interestingly, we find that our method also gives similar benefits when applied to the other low-level vision tasks. We demonstrate the potential advantages of our method by applying it to the Gaussian denoising and JPEG artifact removal tasks. For each task, we use EDSR as our baseline and trained the model from scratch with the synthetic DIV2K dataset with the corresponding degradation functions. We evaluate the model performance on the Kodak24 and LIVE1 [18] datasets using PSNR (dB), SSIM, and LPIPS [31]. Please see the appendix for more details.

Gaussian denoising (color). We generate a synthetic dataset using Gaussian noise of different signal-to-noise ratios (SNR); $\sigma = 30$ and 70 (higher σ means stronger noise). Similar to the over-sharpening issue in SR, we simulate the over-smoothing problem (bottom row, Table 4). The proposed model has lower PSNR (dB) than the baseline but it shows higher SSIM and lower LPIPS [31], which is known to measure the perceptual distance between two images (lower LPIPS means smaller perceptual difference).

In fact, the higher PSNR of the baseline model is due to the over-smoothing (Figure 7). Because the baseline model has learned to remove the stronger noise, it provides the over-smoothed output losing the fine details of the image. Due to this over-smoothing, its SSIM score is significantly lower and LPIPS is significantly higher. In contrast, the proposed model trained with our strategy successfully denoises the image while preserving the fine structures, which demonstrates the good regularization effect of our method.

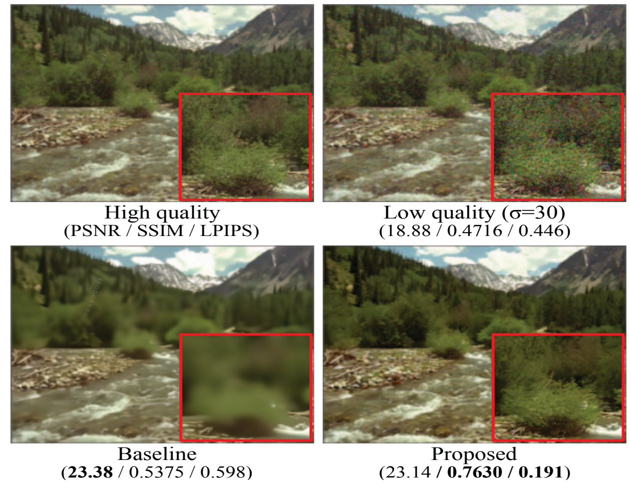


Figure 7. Comparison of the generalization ability in the denoising task. Both the baseline and the proposed method are trained using $\sigma = 70$ (severe) and tested with $\sigma = 30$ (mild). Our proposed method effectively recovers the details while the baseline over-smooths the input resulting in a blurry image.

JPEG artifact removal (color). We generate a synthetic dataset using different compression qualities, $q = 10$ and 30 (lower q means stronger artifact) on the color image. Similar to the denoising task, we simulate the over-removal issue. Compared to the baseline model, our proposed method shows significantly better performance in all metrics we used (bottom row, Table 5). The model generalizes better and gives 0.97 dB performance gain in PSNR.

5. Conclusion

We have introduced CutBlur and Mixture of Augmentations (MoA), a new DA method and a strategy for training a stronger SR model. By learning how and where to super-resolve an image, CutBlur encourages the model to understand how much it should apply the super-resolution to an image area. We have also analyzed which DA methods hurt SR performance and how to modify those to prevent such degradation. We showed that our proposed MoA strategy consistently and significantly improves the performance across various scenarios, especially when the model size is big and the dataset is collected from real-world environments. Last but not least, our method showed promising results in denoising and JPEG artifact removals, implying its potential extensibility to other low-level vision tasks.

Acknowledgement. We would like to thank Clova AI Research team, especially Yunjey Choi, Seong Joon Oh, Youngjung Uh, Sangdoo Yun, Dongyoon Han, Youngjoon Yoo, and Jung-Woo Ha for their valuable comments and feedback. This work was supported by NAVER Corp and also by the National Research Foundation of Korea grant funded by the Korea government (MSIT) (no.NRF-2019R1A2C1006608)

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1692–1700, 2018. 1

[2] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 2, 3, 5

[3] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–268, 2018. 5, 7

[4] Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang. Toward real-world single image super-resolution: A new benchmark and a new model. *arXiv preprint arXiv:1904.00523*, 2019. 1, 2, 3, 5, 7

[5] Junsuk Choe and Hyunjung Shim. Attention-based dropout layer for weakly supervised object localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2219–2228, 2019. 2, 3

[6] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 2

[7] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 1, 2, 3

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 1, 5

[9] Ruicheng Feng, Jinjin Gu, Yu Qiao, and Chao Dong. Suppressing model overfitting for image super-resolution networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 2

[10] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017. 1, 2

[11] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, pages 10727–10737, 2018. 1, 2, 3

[12] Shuhang Gu, Wangmeng Zuo, Qi Xie, Deyu Meng, Xiangchu Feng, and Lei Zhang. Convolutional sparse coding for image super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1823–1831, 2015. 1

[13] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 1

[14] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015. 5

[15] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 2, 3, 5

[16] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019. 2

[17] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838, 2017. 5

[18] HR Sheikh. Live image quality assessment database release 2. *http://live. ece. utexas. edu/research/quality*, 2005. 8

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 2

[20] Nako Sung, Minkyu Kim, Hyunwoo Jo, Youngil Yang, Jingwoong Kim, Leonard Lausen, Youngkwan Kim, Gayoung Lee, Donghyun Kwak, Jung-Woo Ha, et al. Nsml: A machine learning platform that enables you to focus on your models. *arXiv preprint arXiv:1712.05902*, 2017. 5

[21] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 4

[22] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 114–125, 2017. 1

[23] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Asian conference on computer vision*, pages 111–126. Springer, 2014. 1

[24] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1865–1873, 2016. 1, 2

[25] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 2, 3

[26] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. *arXiv preprint arXiv:1806.05236*, 2018. 1, 2

[27] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *arXiv preprint arXiv:1802.02375*, 2018. 1, 2

[28] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE*

transactions on image processing, 19(11):2861–2873, 2010. 5

[29] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *arXiv preprint arXiv:1905.04899*, 2019. 1, 2, 3

[30] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1, 2, 3

[31] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 8

[32] Xuaner Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4786–4794, 2018. 1

[33] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018. 5, 7

[34] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017. 2