UniRes: Universal Image Restoration for Complex Degradations

Mo Zhou[1,2*] Keren Ye[1] Mauricio Delbracio[1] Peyman Milanfar[1] Vishal M. Patel[2] Hossein Talebi[1]
[1]Google [2]Johns Hopkins University

A. More Experiments and Discussions

A.1. Why Specifically Four Degradation Types?

In this paper, we particularly focus on complex degradation, an arbitrary mixture of four fundamental degradation types: low resolution, motion blur, defocus blur, and real noise. Those degradations stem from capture condition, capture device and post-processing pipelines. This background is clarified in the manuscript, including the abstract and the first paragraph of the introduction section.

Apart from the four types of degradations, in the low-level vision literature, there are other types of degradations such as rain, haze, fog, and snow. These degradations are not caused by capture device or post-processing pipelines, and hence are not included in the scope of this paper. The effectiveness of our method on these types of degradations is left for future study.

A.2. Time Complexity and Search Space

As mentioned in the ”Implementation Details” in the paper, the default search space parameters for UniRes are $(\gamma, \delta) = (-0.2, 1.2)$, with an interval of 0.2. Namely, each weight w_i has $n = 7$ possible values (*i.e.*, $-0.2, 0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2$). Additionally, the weights should sum to one, *i.e.*, $\sum_{i=1}^{K} w_i = 1.0$, and only one negative value is allowed among w_i, $i = 1, \ldots, K$. While the complexity of the grid search algorithm is $O(n^K)$, the concrete size of the search space is not n^K due to the two constraints. The search space size in the default settings is 1512. We provide a Python snippet below for the search space and contraints.

```python
from typing import *
import numpy as np
import itertools as it

def search_grid(vmin: float = -0.2,
                vmax: float =  1.2,
                nvars: int = 6,
                interval: float = 0.2,
                ) -> List[List[float]]:
    """
    Find all valid possible combination weights.
    """
    values = np.arange(vmin, vmax + 1e-3, interval)
    allcombs = it.product(*([values] * nvars))
    allcombs = [np.array(x) for x in allcombs]
    # figure out valid combinations
    validcombs = []
    for x in allcombs:
        if not np.abs(1 - np.sum(x)) < 1e-5:
            # they must sum to one
            continue
        elif not np.count_nonzero(x < -1e-5) <= 1:
            # no more than one negative value
            continue
        else:
            # this one is valid
            validcombs.append(x.tolist())
    print(’Valid Combinations:’, len(validcombs))
    return validcombs

if __name__ == ’__main__’:
    validcombs = search_grid(-0.2, 1.2)
```

A.3. Detailed Results on DiversePhotos×1

The grid search algorithm for optimization has an exponential complexity. The search space size given the default setting of UniRes is 1512. The inference time of UniRes per image for a given set of combination weights is $2.332 \pm 0.005 s$ on JAX/TPUv5. MUSIQ takes $0.1 s$ per image on CPU. The full experimental details on DiversePhotos×1, including the total inference time per image is shown in Tab. 1. Two potential speed-up methods are discussed in the manuscript, and they are also included in this table. Some other related works, including AutoDIR [10] are not compared in the paper due to space limit, and their performance lagging behind the other methods such as DiffBIR [15] and SUPIR [21] by a margin. The PromptIR [18] is advertized as “all-in-one” image resotration, but the official model only support denoise, derain, and dehaze.

Potential future directions for accelerating the proposed method includes, but are not limited to (1) distillation for single-step inference, (2) caching mechanisms, (3) better degradation-aware image features and combination weight prediction. They are beyond the scope of this paper, so we leave them for future explorations.

[*] Work done during internship at Google LLC.

Method	Combination Weights	Platform	Inference Time per image (seconds)	ClipIQA	MUSIQ	ManIQA
SwinIR [14]	N/A	PyTorch/Nvidia A100	0.374 ± 0.063	0.3727	49.26	0.3008
Restormer [22]	N/A	PyTorch/Nvidia A100	0.132 ± 0.035	0.3407	41.80	0.2243
PromptIR [18]	N/A	PyTorch/Nvidia A100	0.136 ± 0.032	0.3069	36.20	0.1950
AirNet [13]	N/A	PyTorch/Nvidia A100	0.074 ± 0.033	0.3031	35.93	0.1893
AutoDIR [10]	N/A	PyTorch/Nvidia A100	10.633 ± 15.909	0.3260	40.32	0.2147
NAFNet [4]	N/A	PyTorch/Nvidia A100	0.023 ± 0.007	0.3372	43.62	0.2323
StableSR [20]	N/A	PyTorch/Nvidia A100	11.002 ± 0.171	0.6277	61.39	0.3992
DiffBIR [15]	N/A	PyTorch/Nvidia A100	6.522 ± 0.034	0.6453	59.97	0.4922
SUPIR [21]	N/A	PyTorch/Nvidia A100	15.601 ± 0.629	0.5060	51.68	0.3745
DACLIP-IR [16]	N/A	PyTorch/Nvidia A100	4.940 ± 0.064	0.3497	46.16	0.2567
RAM-SwinIR [19]	N/A	PyTorch	N/A	0.3367	40.25	0.2842
RAM-PromptIR [19]	N/A	PyTorch	N/A	0.3110	36.55	0.1920
UniRes	Grid search	JAX/TPUv5	$(2.332 + 0.1) \times 1512 \approx 3677$	0.6519	68.22	0.5021
UniRes	Most frequent 8 sets of combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 8 = 19.456$	0.6613	68.02	0.5101
UniRes	Most frequent 6 sets of combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 6 = 14.592$	0.6633	67.92	0.5096
UniRes	Most frequent 4 sets of combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 4 = 9.728$	0.6655	67.68	0.5095
UniRes	Most frequent 2 sets of combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 2 = 4.864$	0.6581	66.89	0.5052
UniRes	Most frequent 1 set of combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 1 = 2.432$	0.6590	66.44	0.5042
UniRes	Average optimal combination weights	JAX/TPUv5	$(2.332 + 0.1) \times 1 = 2.432$	0.5941	62.10	0.4266
UniRes	Random Forest (skip search)	JAX/TPUv5	$0.035 + 2.332 = 2.367$	0.5873	61.91	0.4257

Table 1. Full Quantitative Experimental Details on DiversePhotos×1.

A.4. More Visualizations and Failure Cases

In this section, we provide additional visualization results on DiversePhotos×1, as shown in Fig. 1, and Fig. 2. Some failure cases are shown in Fig. 3. The failures include hallucination, color change, artifacts, and failure to restore some degradations. See the caption of Fig. 3 for details.

A.5. Task Weight Sensitivity

We provide examples to demonstrate how the changes in combination weights in Eq. (2) could impact the results. Fig. 4 shows the trade-off effect between super resolution and denoise, where a smooth trade-off between two different effects can be observed by adjusting the combination weights. Fig. 5 shows different results on the same input LQ image when applying different combination weights. Fig. 6 shows the MUSIQ score curves when trading-off every pair of restoration tasks.

The average optimal weight over the DiversePhotos×1 dataset is (BR=0.07, SR=0.12, MD=0.07, DD=0.06, DN=-0.15, DownLQ=0.83). The denoing task has a negative weight in average largely because the MUSIQ metric prefers sharp images, while the denoiser (the DN=1 case, *i.e.*, the weight for denoising is set to 1, while the rest are set to zero) does not sharpen the given image. So the denoiser is not preferred by MUSIQ in most cases, and the algorithm leans towards using it as a negative classifier-free guidance term [8] to push the latent diffusion prediction to be closer to other high-quality directions. Nevertheless, the denoiser is qualitatively effective as demonstrated by the example in Fig. 4. If the average optimal weight is used for all images from DiversePhotos×1, the results are 0.5941, 62.10, 0.4266 for ClipIQA, MUSIQ, and ManIQA, respectively. The most popular optimal weight on the dataset is (DN=-0.2, DownLQ=1.20), where 66 out of 160 images (41.25%) reach the peak MUSIQ value. If this most popular optimal weight is used for all images, the results are 0.6590, 66.44, 0.5042 for ClipIQA, MUSIQ, and ManIQA, respectively.

Model	Task Weights	PSNR	SSIM	LPIPS↓	FID↓	ClipIQA	MUSIQ	ManIQA
DIV2K (3000 Crops from StableSR, size 512 × 512)								
StableSR	-	21.94	0.5343	0.3113	24.44	0.6771	65.92	0.4201
DiffBIR	-	21.82	0.5050	0.3670	32.72	0.7300	69.87	0.5667
SUPIR	-	20.85	0.4945	0.3904	31.60	0.7134	63.69	0.5477
DACLIP-IR	-	21.93	0.4864	0.4881	71.93	0.3295	48.68	0.2654
SR-Only	-	22.12	0.5352	0.3028	21.17	0.6083	66.45	0.4366
UniRes	SR=1	21.41	0.5127	0.3325	25.99	0.6308	67.29	0.4567
GoPro (1111 Images)								
UniRes	MD=1	25.04	0.7629	0.1604	11.83	0.3073	58.65	0.2630
DPDD (74 Images)								
UniRes	DD=1	24.03	0.6980	0.1678	-	0.5088	63.42	0.4198
SIDD (1280 Crops, size 256 × 256)								
UniRes	DN=1	26.94	0.8120	0.1821	-	0.2945	22.19	0.2603

Table 2. Evaluation on Well-Isolated Restoration Tasks. In this paper, we focus on complex degradations, instead of these well-isolated degradations.

A.6. Evaluation on Well-Isolated Degradations

The focus of this paper is real-world complex degradations, instead of well-isolated degradations. The quantitative evaluation for those well-isolated tasks, such as super-resolution, motion deblur, defocus deblur, and denoise are carried out for sanity testing purpose. We evaluate our model on the validation sets of DIV2K [3][1], GoPro [17], DPDD [2], and SIDD [1]. The quantitative metrics can be found in Tab. 2.

[1] huggingface.co/datasets/Iceclear/StableSR-TestSets

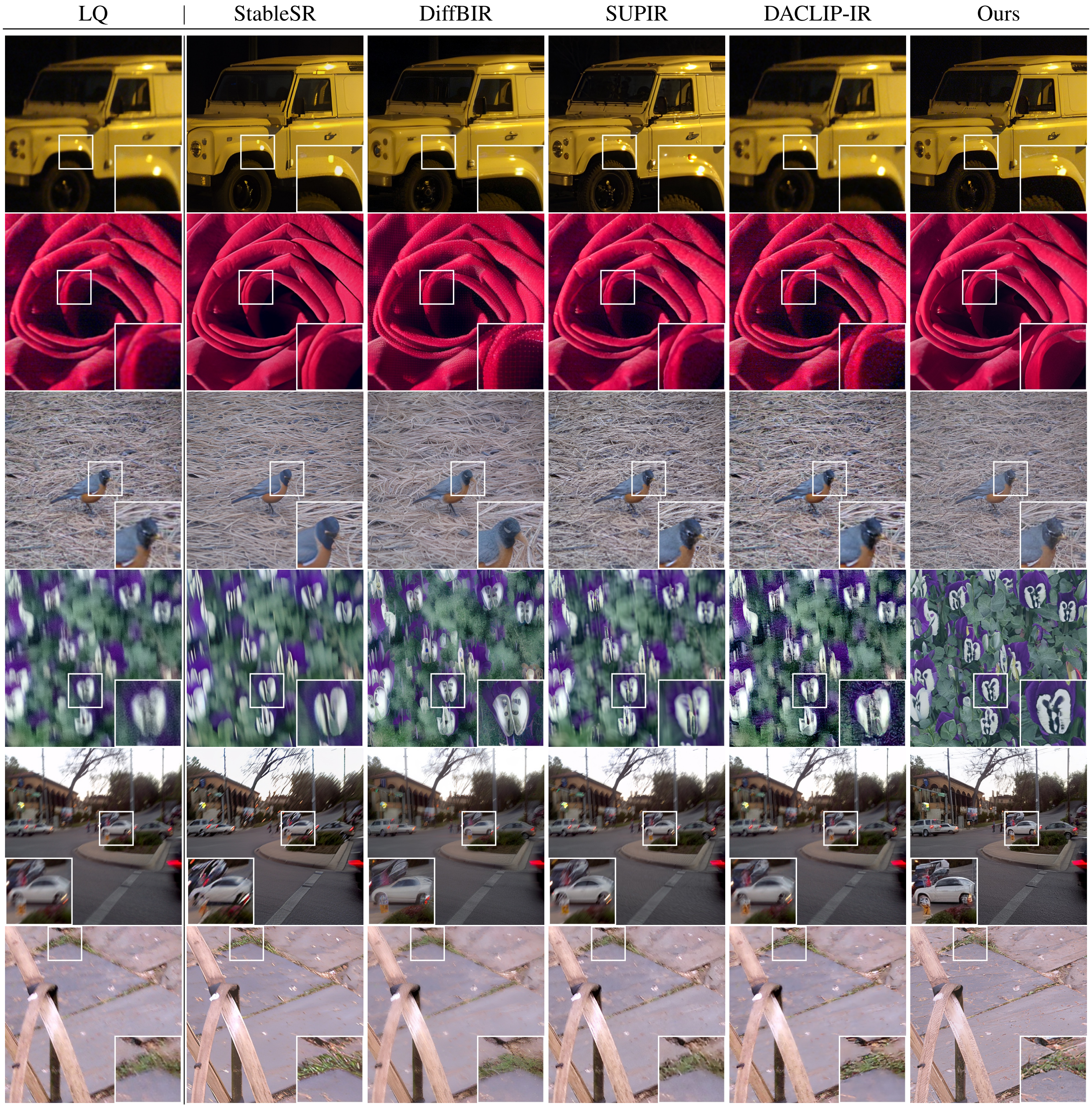

Figure 1. More visualizations about real-world image restoration on the DiversePhotos×1 dataset.

A.7. Positive and Negative Prompts

Recent works [15, 21] demonstrate the effectiveness of positive and negative prompts (*e.g.*, "blur", "low-quality", *etc*.). To make the model correctly understand the negative-quality concepts, [21] explicitly introduce negative-quality images to the training samples. Similarly, to extend our proposed method with positive and negative prompt words, we need to modify the training data pipeline.

In particular, after sampling each training tuple with (LQ image, text prompt, HQ image), there is (1) 1% probability that the text prompt will be replaced with positive-quality words: *"photorealistic, clean, high-resolution, ultra-high definition, 4k detail, 8k resolution, masterpiece, cinematic, highly detailed."*; (2) 1% probability that the text prompt will be replaced with the negative-quality words: *"oil painting, cartoon, blur, dirty, messy, low quality, deformation, low res-*

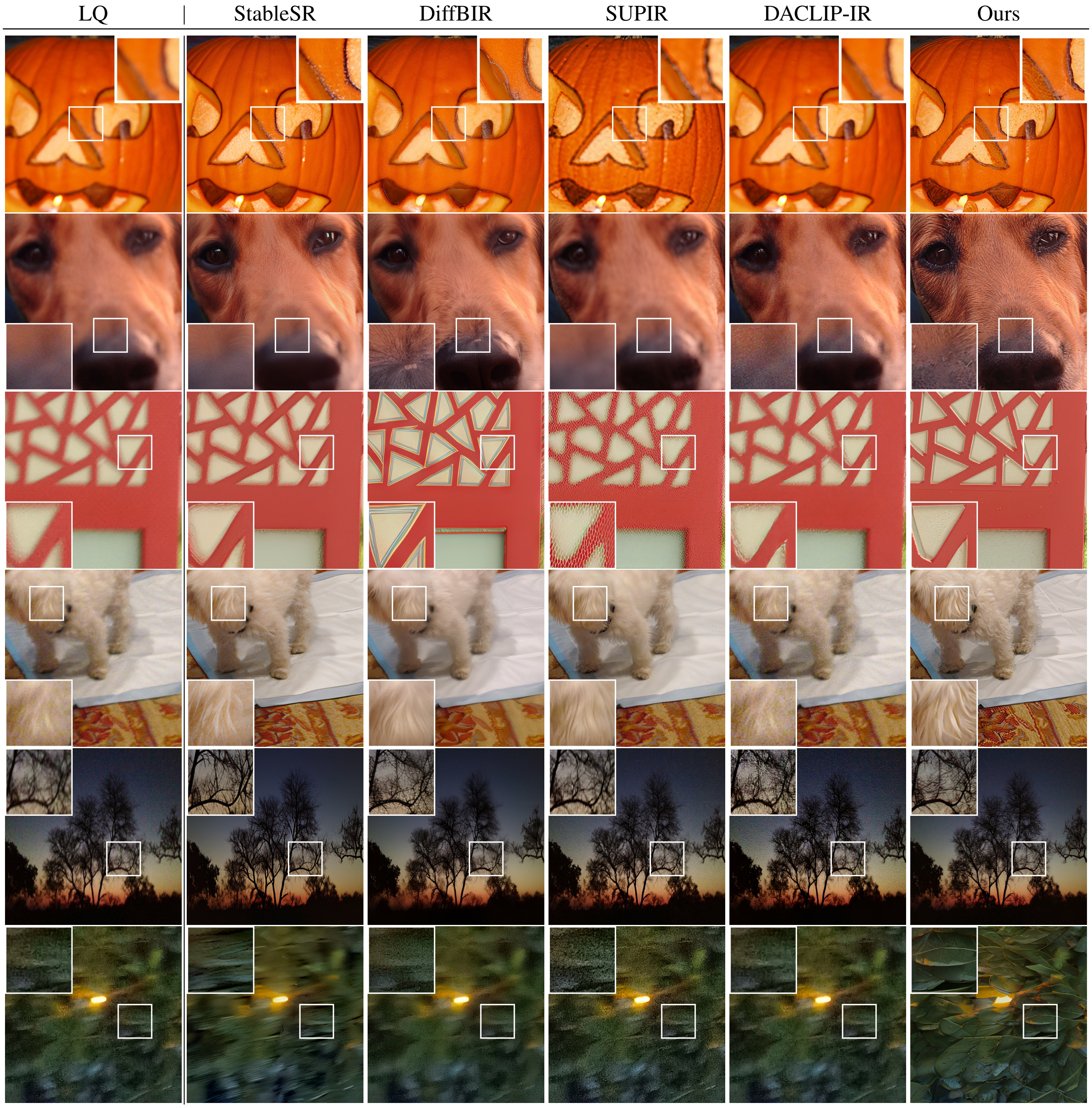

Figure 2. More visualizations about real-world image restoration on the DiversePhotos×1 dataset.

olution, over-smooth.", and meanwhile swap the LQ image and HQ image; (3) 98% probability that the training tuple is left intact. This modification allows the model to properly understand the concept of "positive quality" and "negative quality", which is similar to the observation in [21].

Then we validate the impact of those positive and negative words on the DiversePhotos×1 dataset. In particular, based on the optimal weights obtained by grid search, if we add the diffusion latent prediction for the positive words $\epsilon_\theta(z_t, z_{\mathrm{LQ}}, s_{\text{positive}})$ with weight $+1.0$, and that for the negative words $\epsilon_\theta(z_t, z_{\mathrm{LQ}}, s_{\text{negative}})$ with weight -1.0, the results will be 0.6748, 69.70, 0.5354 for ClipIQA, MUSIQ, and ManIQA, respectively. Comparing to the UniRes results under the default setting (*i.e.*, 0.6519, 68.22, 0.5021), the positive and negative words leads to a slight performance gain. Further increasing the absolute values for their weights

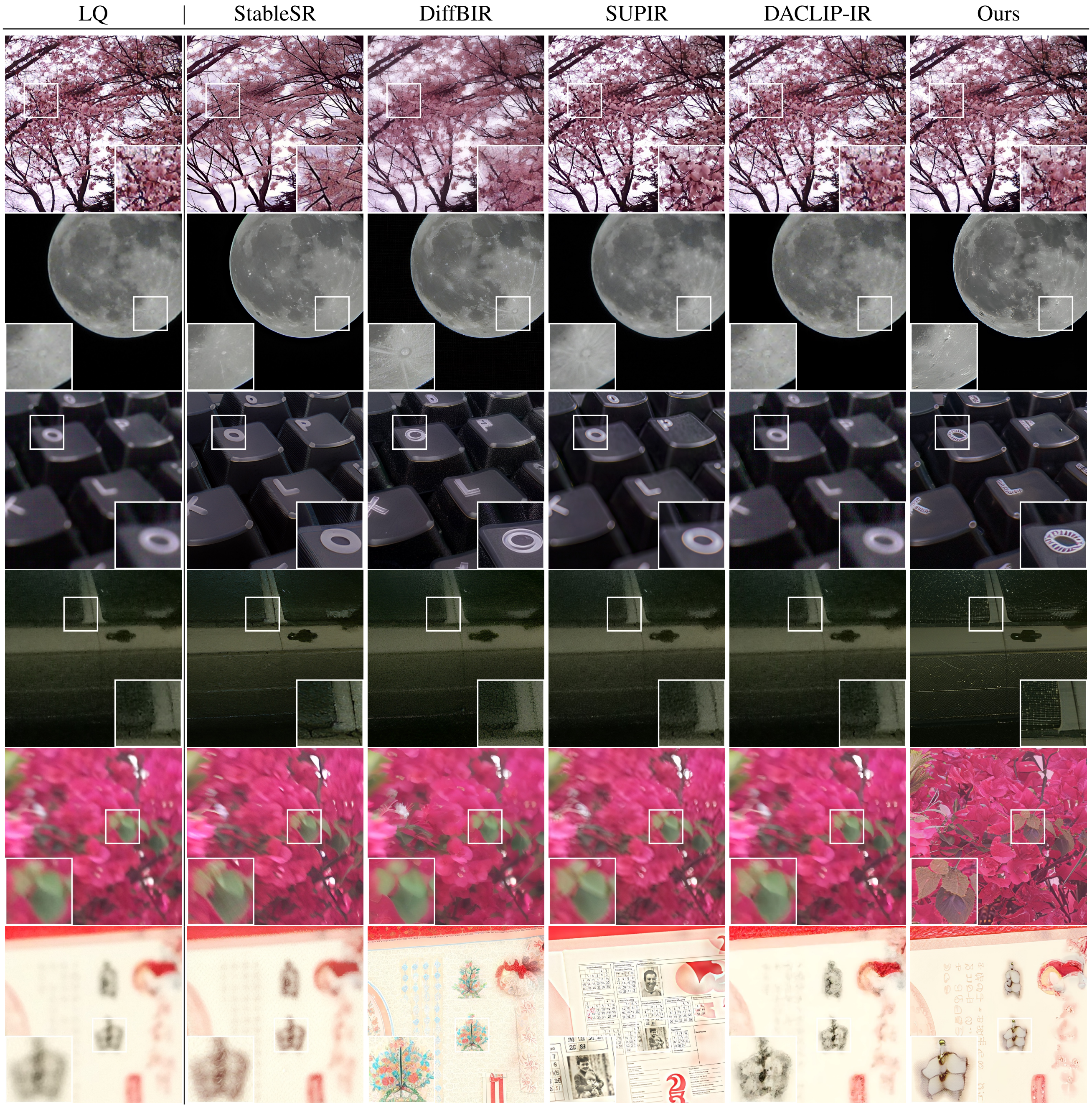

Figure 3. Failure cases on the DiversePhotos×1 dataset. *1st row*: our model does not make improvement in image details; *2nd row*: our model (occasionally) fails to keep fidelity while improving resolution; *3rd row*: our model removes noise but fails to remove defocus blur; *4th row*: our model removes noise but introduces non-existing mesh texture; *5th row*: our model removes low resolution and motion blur, but changes the color of the leaves; *6th row*: a hard example on which all models failed to restore.

may occasionally lead to artifacts according to our observation. Extending our proposed method with positive words and negative words is effective.

A.8. Limitation of Non-Reference Metrics

Our method employs MUSIQ [11] as an approximation to human perceptual preference for grid search. However, MUSIQ is not fully aligned with human, and can lead to some discrepancies where the grid search result is not visually the

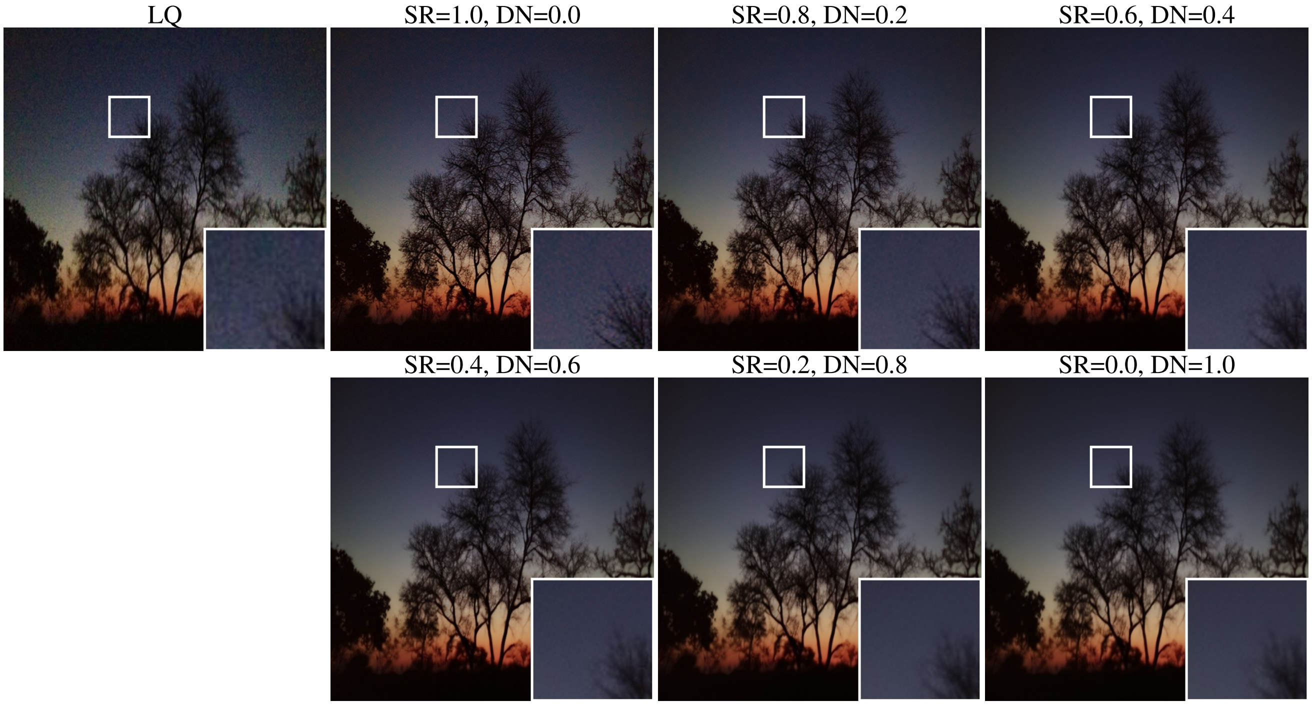

Figure 4. Qualitative demonstration on combination weight sensitivity. In this example, we adjust the weights for super resolution (SR) and denoise (DN), and keep the rest weights to zero. As shown from the images, the SR=1 case can improve the details to the tree, but does not remove all noise. In contrast, the DN=1 case can remove the noise, but not improve the details of the tree. By trading off the two weights, we can observe a smooth trade-off between the two effects. Zoom in for image details.

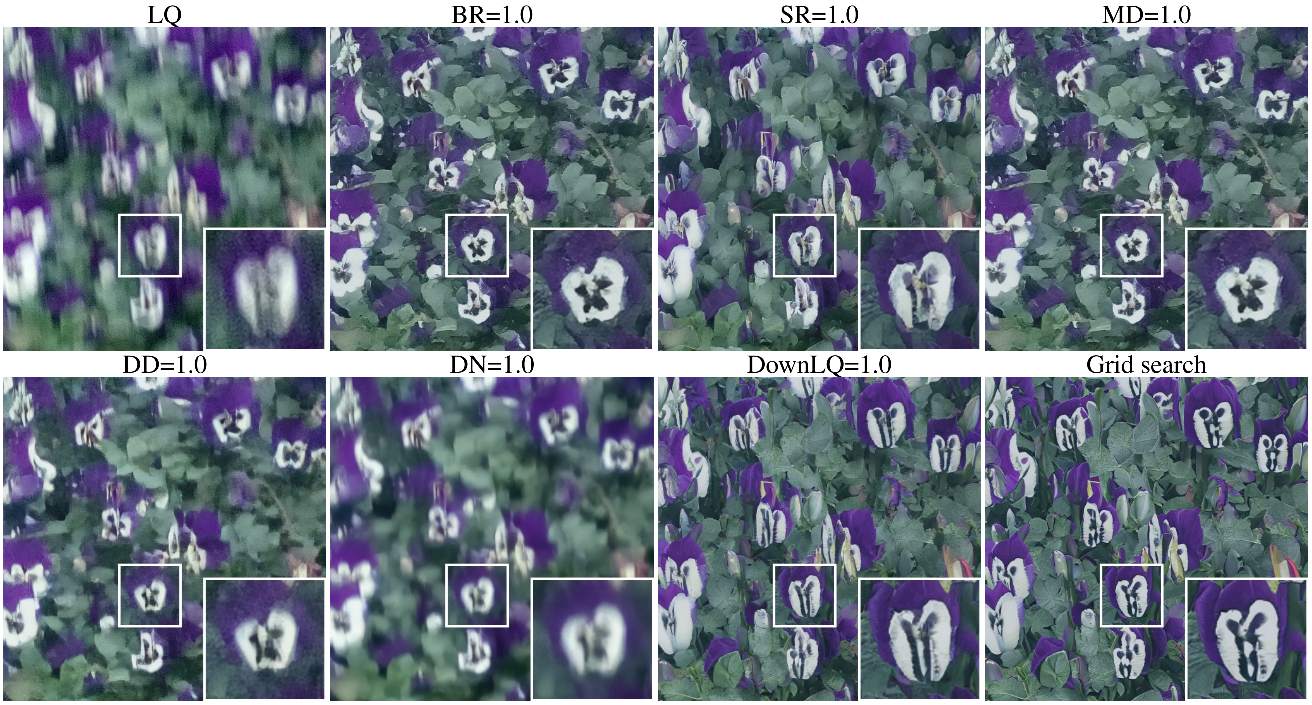

Figure 5. Qualitative demonstration on the same LQ input with different weights. In particular, "SR=1.0" means the weight for super resolution is 1.0, while the rest weights are set to 0.0. Zoom in for image details.

best. An example for such discrepancy is shown in Fig. 7. Potential future work may involve incorporating better image quality metrics.

A.9. Discussion on Some Technical Details

How to deal with any-resolution input. This paper only evaluate 512 × 512 resolution images following StableSR [20] and DiffBIR [15]. Tiling methods (*e.g.*, the one in Fig. 4 in the StableSR paper) can deal with any-resolution images. But tiling method is beyond the scope of paper.

Why probability (0.32,0.28,0.18,0.22) in Section 4. This is calculated using the log number of 512 × 512 crops in the training data for the four tasks. We empirically find this slightly better than an equal probability across tasks.

Why range [-0.2, 1.2] and only one negative weight. We empirically observe that either a larger range ($<$ 0.2 or $>$ 1.2), or allowing more than one negative weight, has a much higher chance for the model to generate visual artifacts.

How are the frequent combinations choisen in Table 5. We obtain the (MT-A feature, optimal combination weight) pairs from the 120 additional images (mentioned in the discussion section in the paper). These pairs are then used to train a Random Forest Regressor (from sklearn.ensemble). Finally we use the regressor to predict combination weights for DiversePhotos.

B. Dataset Details

B.1. DiversePhotos

	Low Resolution	Motion Blur	Defocus Blur	Noise	*sum*
SPAQ [6]	20	17	6	21	64
KONIQ [9]	14	4	14	8	40
LIVE [7]	6	19	20	11	56
sum	40	40	40	40	160

Table 3. DiversePhotos×1 Dataset Statistics. It contains 160 images in total, dedicating 40 images for each of the dominating degradation types: low resolution, motion blur, defocus blur, and noise. The table shows the number of images we curated from each public dataset for each degradation.

The "DiversePhotos" dataset is our curation of test images, curated from SPAQ [6], KONIQ [9], and LIVE [7]. The images in DiversePhotos collectively cover multiple mobile devices and DLSR cameras, as well as a wide range of degradations.

DiversePhotos×1. The DiversePhotos×1 image set involves 160 images, with 40 images for each dominating degradation: low-resolution, motion blur, defocus blur, and noise. Each image is in 512 × 512 resolution. See Tab. 3 for the statistics.

DiversePhotos×4. This set of test images are the 128 × 128 center crops of the DiversePhotos×1 images.

Steps for reproducing "DiversePhotos×1":

1. Download SPAQ [6], KONIQ [9], and LIVE [7] datasets.
2. Gather images whose file names are mentioned in the following 12 listings.
3. Center-crop all images from SPAQ and KONIQ datasets to 512 × 512 resolution.
4. Resize (bicubic) all images from LIVE dataset (from 500 × 500) to 512 × 512 resolution.

(SPAQ, low resolution as dominating degradation, with other degradations): 00019, 00025, 00033, 00109, 00192, 00226, 00251, 00381, 00414, 00559, 00561, 00585, 00743, 03973, 04085, 04136, 04270, 04317, 04334, 06682.

(SPAQ, motion blur as dominating degradation, with other degradations): 00043, 00075, 00121, 00161, 00175, 00178, 00236, 01868, 03513, 04089, 04272, 04380, 06341, 06863, 10388, 10391, 10495.

(SPAQ, defocus blur as dominating degradation, with other degradations): 00125, 00212, 00282, 04379, 06727, 09121.

(SPAQ, noise as dominating degradation, with other degradations): 00077, 00086, 00096, 00143, 00187, 00199, 00292, 00365, 00450, 04337, 04345, 06485, 06703, 07121, 07162, 07394, 07494, 07866, 07903, 08108, 09682.

(KONIQ, low resolution as dominating degradation, with other degradations): 1755366250, 187640892, 2096424103, 2443117568, 2 6393826, 2704811, 2836089223, 2956548148, 3015139450, 3435545140, 3551648026, 4378419360, 527633229, 86243803.

(KONIQ, motion blur as dominating degradation, with other degradations): 2367261033, 3147416579, 331406867, 62480371.

(KONIQ, defocus blur as dominating degradation, with other degradations): 1306193020, 315889745, 55711788, 1807195948, 206294085, 2166503846, 2214729676, 23371433, 2360058082, 2950983139, 3149433848, 324339500, 427196028, 518080817.

(KONIQ, noise as dominating degradation, with other degradations): 1317678723, 1987196687, 218457399, 2593384818, 2837843986, 2867718050, 3727572481, 4410900135,

(LIVE, low resolution as dominating degradation, with other degradations): 110, 723, 760, 805, 819, 875.

(LIVE, motion blur as dominating degradation, with other degradations): 1017, 104, 1156, 12, 154, 239, 270, 283, 29, 429, 458, 460, 468, 659, 663, 700, 732, 810, 856.

(LIVE, defocus blur as dominating degradation, with other degradations): 337, 550, 592, 698, 713, 714, 717, 731, 737, 750, 751, 787, 788, 855, 862, 873, 874, 876, 884, 887.

(LIVE, noise as dominating degradation, with other degradations): 1001, 1011, 1024, 1037, 1055, 1079, 1098, 1149, 370, 443, 5.

We will provide public download links to the resulting images in the future.

B.2. OID-Motion

To create a diverse dataset of degraded images, we simulated camera shake blur as described in [5]. This involves generating random blur kernels with a range of intensities and sizes, which were then applied to high-quality images from the Open Image Dataset [12] to simulate per-object motion blur.

We further degraded these images by introducing lens blur (using Gaussian blur kernels), shot noise, read-out noise, and JPEG compression. By randomly sampling the parameters for each degradation, we created a dataset that encompasses a wide spectrum of image quality, from heavily degraded to almost no degradation. Some OID-Motion sample images are shown in Fig. 8.

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[2] Abdullah Abuolaim and Michael S Brown. Defocus deblurring using dual-pixel data. In *European Conference on Computer Vision*, pages 111–126. Springer, 2020. 2

[3] 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*, 2017. 2

[4] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. *arXiv preprint arXiv:2204.04676*, 2022. 2

[5] Mauricio Delbracio and Guillermo Sapiro. Burst deblurring: Removing camera shake through fourier burst accumulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2385–2393, 2015. 7, 10

[6] Yuming Fang, Hanwei Zhu, Yan Zeng, Kede Ma, and Zhou Wang. Perceptual quality assessment of smartphone photography. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3677–3686, 2020. 7

[7] Deepti Ghadiyaram and Alan C. Bovik. Massive online crowdsourced study of subjective and objective picture quality. *IEEE Transactions on Image Processing*, 25(1):372–387, 2016. 7

[8] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. 2

[9] V. Hosu, H. Lin, T. Sziranyi, and D. Saupe. Koniq-10k: An ecologically valid database for deep learning of blind image quality assessment. *IEEE Transactions on Image Processing*, 29:4041–4056, 2020. 7

[10] Yitong Jiang, Zhaoyang Zhang, Tianfan Xue, and Jinwei Gu. Autodir: Automatic all-in-one image restoration with latent diffusion, 2024. 1, 2

[11] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. Musiq: Multi-scale image quality transformer, 2021. 5

[12] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128 (7):1956–1981, 2020. 7, 10

[13] Boyun Li, Xiao Liu, Peng Hu, Zhongqin Wu, Jiancheng Lv, and Xi Peng. All-In-One Image Restoration for Unknown Corruption. In *IEEE Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, 2022. 2

[14] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer, 2021. 2

[15] Xinqi Lin, Jingwen He, Ziyan Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Wanli Ouyang, Yu Qiao, and Chao Dong. Diffbir: Towards blind image restoration with generative diffusion prior, 2024. 1, 2, 3, 7

[16] Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B Schön. Photo-realistic image restoration in the wild with controlled vision-language models. *arXiv preprint arXiv:2404.09732*, 2024. 2

[17] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2017. 2

[18] Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Khan. Promptir: Prompting for all-in-one image restoration. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 2

[19] Chu-Jie Qin, Rui-Qi Wu, Zikun Liu, Xin Lin, Chun-Le Guo, Hyun Hee Park, and Chongyi Li. Restore anything with masks: Leveraging mask image modeling for blind all-in-one image restoration, 2024. 2

[20] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin C.K. Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. *International Journal of Computer Vision*, 2024. 2, 7

[21] Fanghua Yu, Jinjin Gu, Zheyuan Li, Jinfan Hu, Xiangtao Kong, Xintao Wang, Jingwen He, Yu Qiao, and Chao Dong. Scaling up to excellence: Practicing model scaling for photo-realistic image restoration in the wild, 2024. 1, 2, 3, 4

[22] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration, 2022. 2

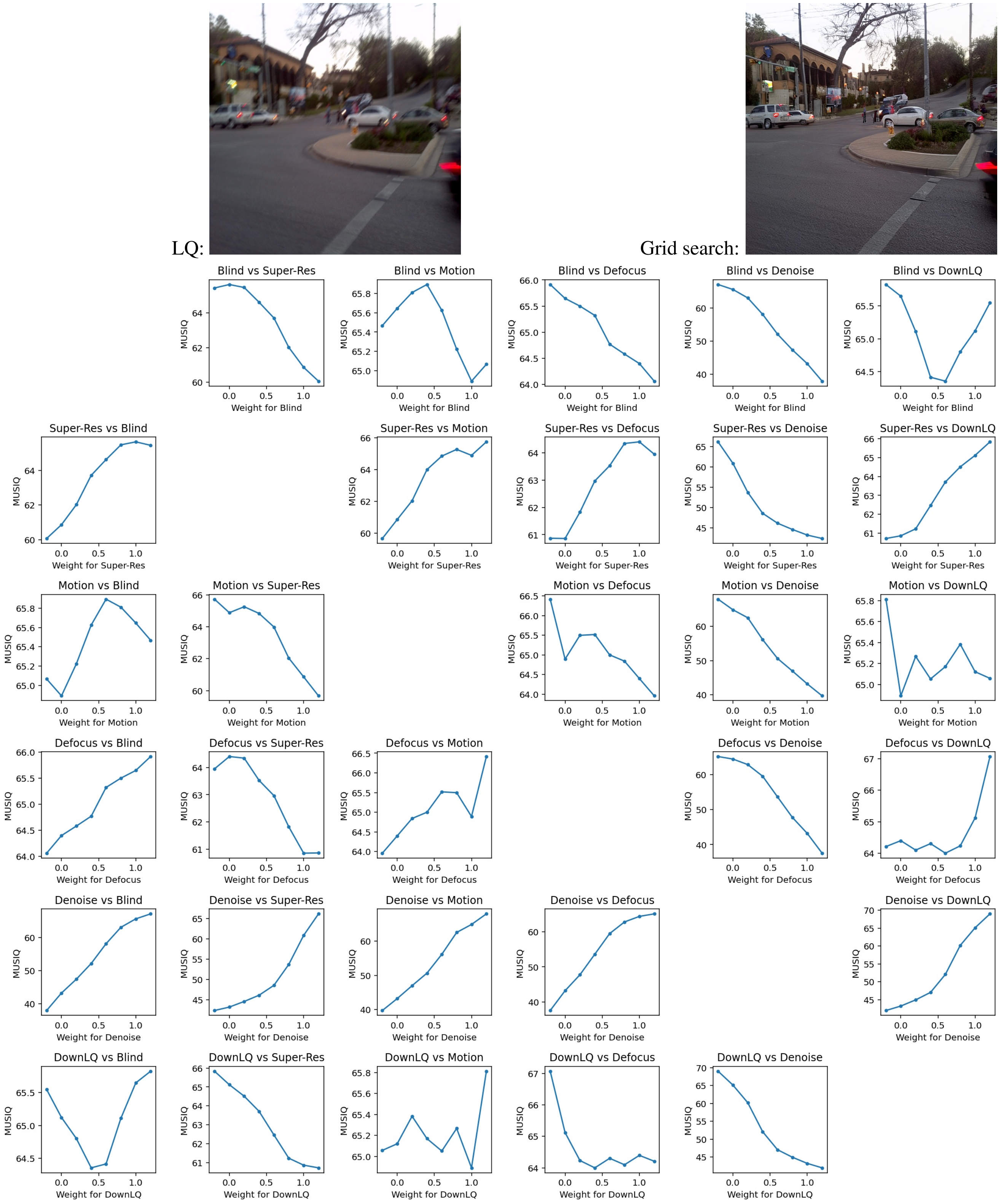

Figure 6. The trade-off curves for the example LQ image between each pair of restoration tasks: blind restoration (Blind), super resolution (Supre-Res), motion deblur (Motion), defocus deblur (Defocus), denoise (Denoise), and DownLQ. Here we only control the two weights for each pair of tasks, while keeping the rest weights as zero. For reference, the optimal weights (through grid search) for this LQ example are (MD=0.6,DN=-0.2,DownLQ=0.6).

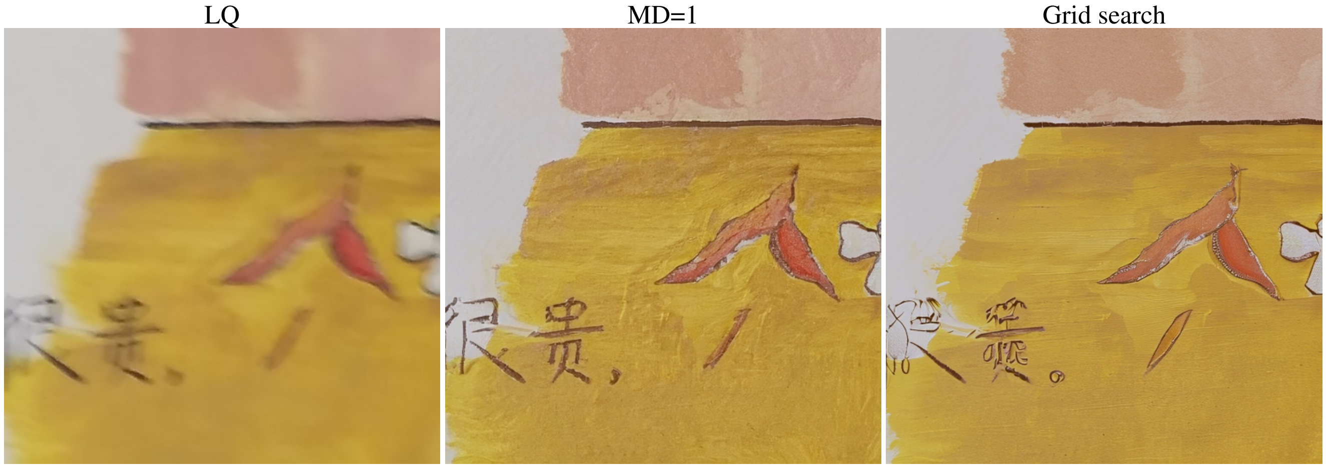

Figure 7. Demonstration of the occasional discrepancy between human preference and non-reference metric. The grid search result (right) removes motion blur from the LQ (left), but also impacts fidelity. However, by manually setting the weight for motion deblur (MD) to 1 and the rest to zero, a visually better result can be obtained (middle). This is an example where non-reference metric is not fully aligned with human preference.

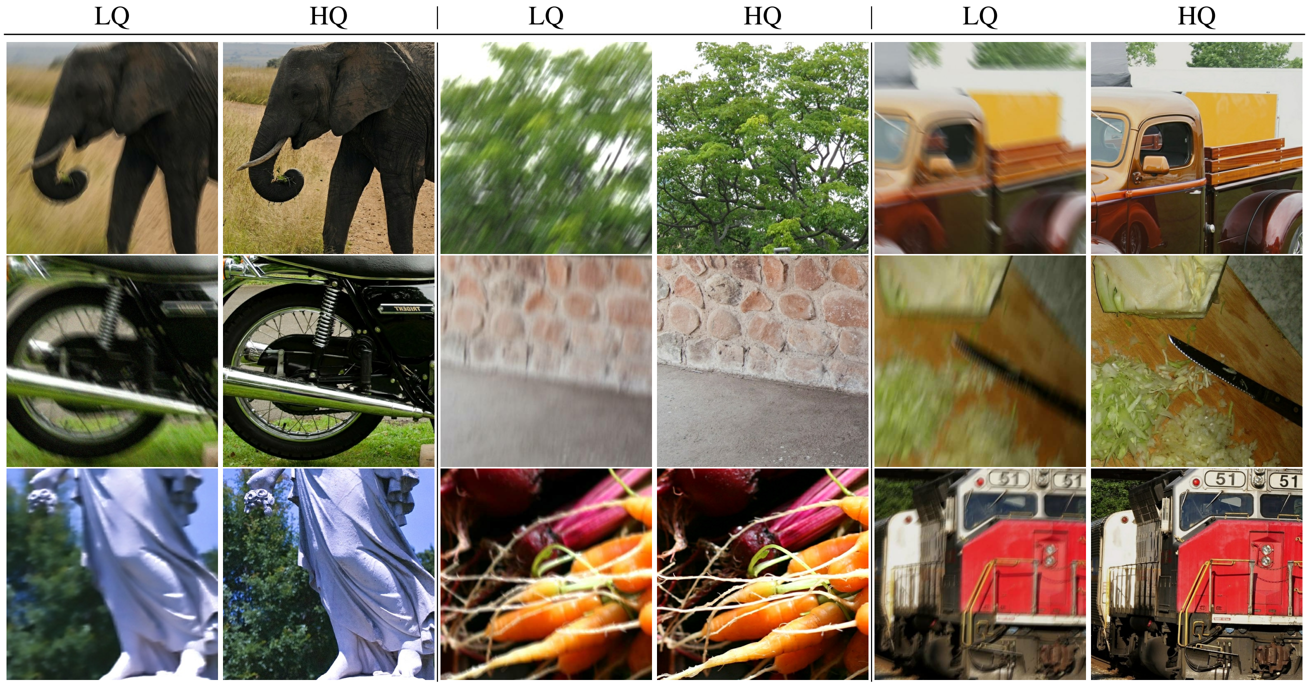

Figure 8. Samples from the OID-Motion training dataset. It is simulated with the camera shake blur [5] on the Open Image Dataset [12].