# Recovering Realistic Texture in Image Super-resolution by Deep Spatial Feature Transform Supplementary Material

Xintao Wang [1] Ke Yu[1] Chao Dong[2] Chen Change Loy[1]
[1]CUHK - SenseTime Joint Lab, The Chinese University of Hong Kong, [2]SenseTime Research
{wx016, yk017, ccloy}@ie.cuhk.edu.hk dongchao@sensetime.com
http://mmlab.ie.cuhk.edu.hk/projects/SFTGAN/

## Abstract

*In this supplementary material, we first present the network architecture details of the SR network and the discriminator in Sec. 1. We then introduce the datasets used in our training and testing in Sec. 2. Training details of segmentation and the results are illustrated in Sec. 3. Additional analysis including the quantitative results, impact of different categorical priors and repetitive conditioning are presented in Sec. 4. Finally, we provide more visualization examples in outdoor scenes and out-of-category scenes, together with failure case analysis.*

## 1. Network Architectures

In this section, we provide the details of the network architectures - SR network with SFT module and the discriminator.

### 1.1. SR network with SFT module

The detailed SR network structure with SFT module is shown in Fig. 1. Convolution layer is denoted as $Conv(in, out, k, s)$, where $in$ is the number of input channels, $out$ is the number of output channels, $k$ is the kernel size and $s$ is the stride. To avoid interference of different categorical regions in one image, we restrict the receptive field of the condition network by using $1 \times 1$ kernels for all the convolution layers except the first one. Since we perform segmentation on bicubic-ed images, the condition network first uses a convolutional layer to downsample the input with a $4 \times 4$ kernel and $stride = 4$.

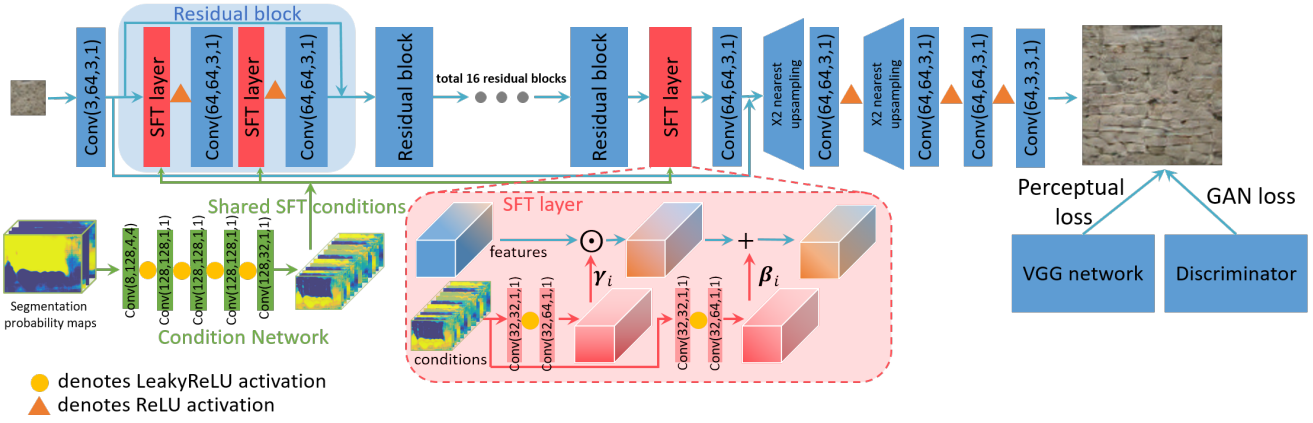


Figure 1. The details of SR network architecture with SFT module.

### 1.2. Discriminator

Fig. 2 shows the architecture of our discriminator. We apply a VGG-style [14] network of strided convolutions to gradually decrease the spatial dimensions. Convolution layer is denoted as $Conv(in, out, k, s)$. The network uses leaky ReLU activations with parameter 0.01.

Inspired by [12], our discriminator not only distinguishes whether the input is real or fake, but also predicts which category the input belongs to. All convolutional layers are shared and a fully connected layer is used to output the extra result for the auxiliary classifier. This is possible since our training images are cropped to contain only one category. For 'background' category, we do not give the penalty of auxiliary classification to the discriminator. We find this strategy facilitates the generation of images with more realistic textures. The discriminator is removed in test phase.

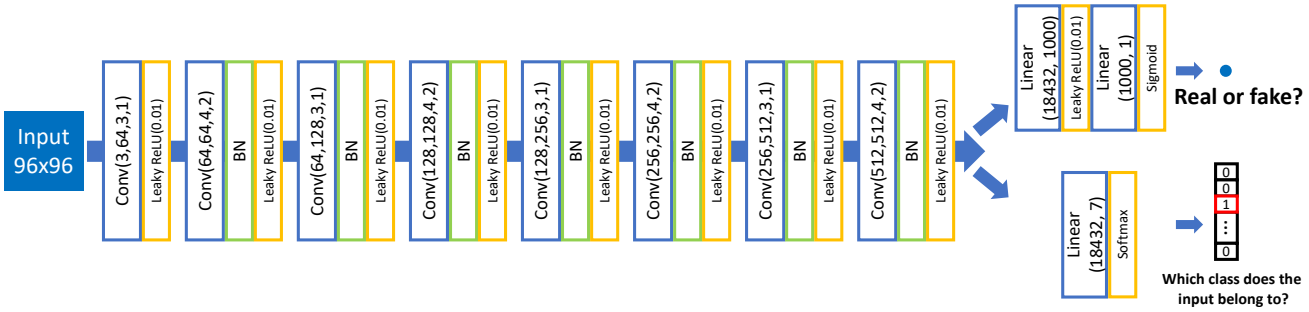


Figure 2. The network structure of the discriminator.

## 2. Dataset

We use two datasets for our SR task: a subset of ImageNet dataset and an OutdoorScene dataset. ImageNet dataset is used to pre-train our network with perceptual loss and GAN loss to avoid undesired local optima. We use the validation set of ImageNet after removing low resolution images of size below 30kB, resulting in roughly 450k training images.

After initialization, we fine-tune our full network on OutdoorScene conditionally on the input segmentation probability maps. The OutdoorScene dataset is collected by querying images from search engines using the defined categories as keywords. The dataset is divided into training and test partitions (OutdoorSceneTrain and OutdoorSceneTest). For OutdoorSceneTrain, we crop each image so that only one category exists. The image number of each category is listed in Table 1. Background images are randomly sampled from ImageNet. The OutdoorSceneTest partition consists of 300 images and we use OST300 for abbreviation (some OST300 images are from the ADE validation set and they are exclusive from the training dataset).

Table 1. The image number of each category in OutdoorSceneTrain.

| Category | sky | water | building | grass | plant | animal | mountain | Total |
|---|---|---|---|---|---|---|---|---|
| Image Number | 1,727 | 857 | 2,285 | 1,140 | 1,036 | 2,187 | 1,092 | 10,324 |

## 3. Segmentation

The segmentation model aims at parsing the input scenes into 7 categories, *i.e.*, sky, water, building, animal, plant, grass, mountain and an extra 'background' category. Segmentation probability maps are then generated for the condition network. We provide the training details in this section.

**Training and testing dataset**. We use a newly merged training dataset (called OutdoorSeg) for outdoor scene segmentation, which is composed of 8,447 images from ADE dataset [19], 899 mountain images from Flickr website and 554 animal images from COCO dataset [8], as listed in Table 2. We collect animal and mountain images from COCO and Fickr for balancing the training samples of each category. We test the performance of the segmentation network on OST300.

Table 2. Training dataset for segmentation - OutdoorSeg.

| Sources | Image Number |
|---|---|
| images from ADE [19] | 8,447 |
| animal images from COCO [8] | 554 |
| mountain images from Flickr | 899 |
| Total | 9,900 |

**Network architecture and training details**. The segmentation network is modified from ResNet101. To suit for segmentation task, we set the dilation parameter of some convolutional layers to 2 and 4, and remove the final fully connected

layers. An extra upsampling layer is added to output a dense prediction. The modification is standard and could be found in segmentation literature [10, 9, 2].

The network is pre-trained on the COCO dataset [8] and then fine-tuned on OutdoorSeg dataset. To better adapt to the bicubic-ed LR input in testing, we fed the network with bicubic-ed training samples during fine-tuning.

We train the network separately from the main SR network using Caffe [4] with stochastic gradient descent optimizer. The learning rate is set to 0.001 and is decreased under the 'poly' strategy. Both of the pre-training and fine-tuning have 30000 training iteration with batch size 8.

**Quantitative results**. From Table 3, we can see that satisfactory segmentation results can still be obtained even on LR images given a modern CNN-based segmentation model.

Table 3. Quantitative results of segmentation on OST300.

| Input | Pixel Accuracy | IoU | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | sky | water | grass | mountain | building | plant | animal | background | mean |
| Bicubic | 90.82 | 95.80 | 85.53 | 78.28 | 57.18 | 90.94 | 73.47 | 87.25 | 70.13 | 79.82 |
| GT | 91.72 | 96.17 | 89.16 | 79.26 | 57.84 | 92.49 | 76.30 | 91.20 | 70.54 | 81.62 |

# 4. Additional Analysis

## 4.1. Quantitative results

We provide quantitative results on standard benchmarks such as Set5 [1], Set14 [18] and BSD100 [11], and also on OutdoorSceneTest300 (OST300) dataset. We compare our method with Bicubic, SRCNN [3], VDSR [5], LapSRN [6], DRRN [15], MemNet [16], SRGAN [7] and EnhanceNet [13]. For Set5, Set14 and BSD100, our model treats all the regions as 'background', degrading itself to SRGAN. Table 4 and Table 5 summarize the PSNR and SSIM values performed on Y-channel and RGB channels respectively. For evaluation on RGB channels, we use the similar process to [17], where PSNR is evaluated over all RGB channels while SSIM is first computed for each R, G, B color channel and then averaged.

GAN-based methods, such as SRGAN, EnhanceNet, and our SFT-GAN yield lower PSNR and SSIM values, which is also observed in [7, 13]. MemNet [16] yields the highest PSNR values since its objective is to minimize the per-pixel distance to the ground truth. However, as can be seen in qualitative comparison, GAN-based methods clearly outperform PSNR-oriented approaches in terms of perceptual quality. From current results, there is no obvious relationship between the perceptual quality and PSNR/SSIM values on SRGAN, EnhanceNet, and SFT-GAN.

Table 4. Quantitative evaluation of state-of-the-art SR algorithms: average PSNR/SSIM on Y channel.

| | Set5 | Set14 | BSD100 | OST300 |
|---|---|---|---|---|
| algorithm | PSNR / SSIM | PSNR / SSIM | PSNR / SSIM | PSNR / SSIM |
| Bicubic | 28.41 / 0.810 | 26.08 / 0.704 | 25.96 / 0.667 | 25.74 / 0.663 |
| SRCNN [3] | 30.47 / 0.862 | 27.60 / 0.753 | 26.90 / 0.710 | 26.49 / 0.703 |
| VDSR [5] | 31.34 / 0.884 | 28.11 / 0.770 | 27.29 / 0.725 | 26.74 / 0.716 |
| LapSRN [6] | 31.51 / 0.885 | 28.17 / 0.770 | 27.31 / 0.726 | 26.75 / 0.716 |
| DRRN [15] | 31.67 / 0.889 | 28.31 / 0.774 | 27.38 / 0.728 | 26.79 / 0.719 |
| MemNet [16] | 31.81 / 0.891 | 28.40 / 0.776 | 27.44 / 0.730 | 26.84 / 0.721 |
| EnhanceNet [13] | 28.56 / 0.810 | 25.76 / 0.678 | 24.93 / 0.626 | 24.36 / 0.616 |
| SRGAN [7] | 29.91 / 0.847 | 26.36 / 0.707 | 24.49 / 0.661 | 24.77 / 0.637 |
| SFT-GAN (ours) | 29.82 / 0.840 | 26.13 / 0.694 | 25.33 / 0.651 | 24.71 / 0.634 |

## 4.2. Impact of different categorical priors

To analyze the impact of different categorical priors, we manually change the probability maps to each category for a certain input. As shown in Fig. 3, the given input LR image can be restored to various textures under different categorical priors. Each restored texture presents the characteristics of its corresponding prior. For example, no matter which category the input belongs to, the restored textures under building categorical prior always have regular geometric shapes. This phenomenon also shows the importance of categorical priors in restoring realistic and natural textures in SR. Without the categorical priors, the model struggles in distinguishing these LR patches and restoring proper textures thereon.

### 4.3. Repetitive conditioning

We investigate the contribution of repetitive conditioning of SFT. In our full SFT-GAN model, all the 16 residual blocks (RB) are equipped with SFT layers. In this experiment, we train SFT-GAN with successively fewer SFT layers and compare models with 8 RB, 4 RB and 1 RB equipped with SFT layers, respectively. As shown in Fig. 4, models with fewer SFT layers

Table 5. Quantitative evaluation of state-of-the-art SR algorithms: average PSNR/SSIM on RGB color channels.

| algorithm | Set5 PSNR / SSIM | Set14 PSNR / SSIM | BSD100 PSNR / SSIM | OST300 PSNR / SSIM |
|---|---|---|---|---|
| Bicubic | 26.68 / 0.773 | 24.46 / 0.664 | 24.63 / 0.640 | 24.38 / 0.639 |
| SRCNN [3] | 28.51 / 0.826 | 25.77 / 0.711 | 25.55 / 0.685 | 25.11 / 0.681 |
| VDSR [5] | 29.23 / 0.847 | 26.18 / 0.725 | 25.91 / 0.700 | 25.34 / 0.693 |
| LapSRN [6] | 29.40 / 0.850 | 26.26 / 0.728 | 25.95 / 0.701 | 25.36 / 0.695 |
| DRRN [15] | 29.52 / 0.853 | 26.37 / 0.731 | 26.01 / 0.704 | 25.40 / 0.698 |
| MemNet [16] | 29.62 / 0.854 | 26.44 / 0.732 | 26.06 / 0.704 | 25.44 / 0.699 |
| EnhanceNet [13] | 26.71 / 0.768 | 23.98 / 0.631 | 23.51 / 0.595 | 22.93 / 0.588 |
| SRGAN [7] | 27.75 / 0.791 | 24.35 / 0.652 | 24.01 / 0.625 | 23.35 / 0.610 |
| SFT-GAN (ours) | 27.86 / 0.797 | 24.27 / 0.645 | 23.90 / 0.619 | 23.26 / 0.605 |

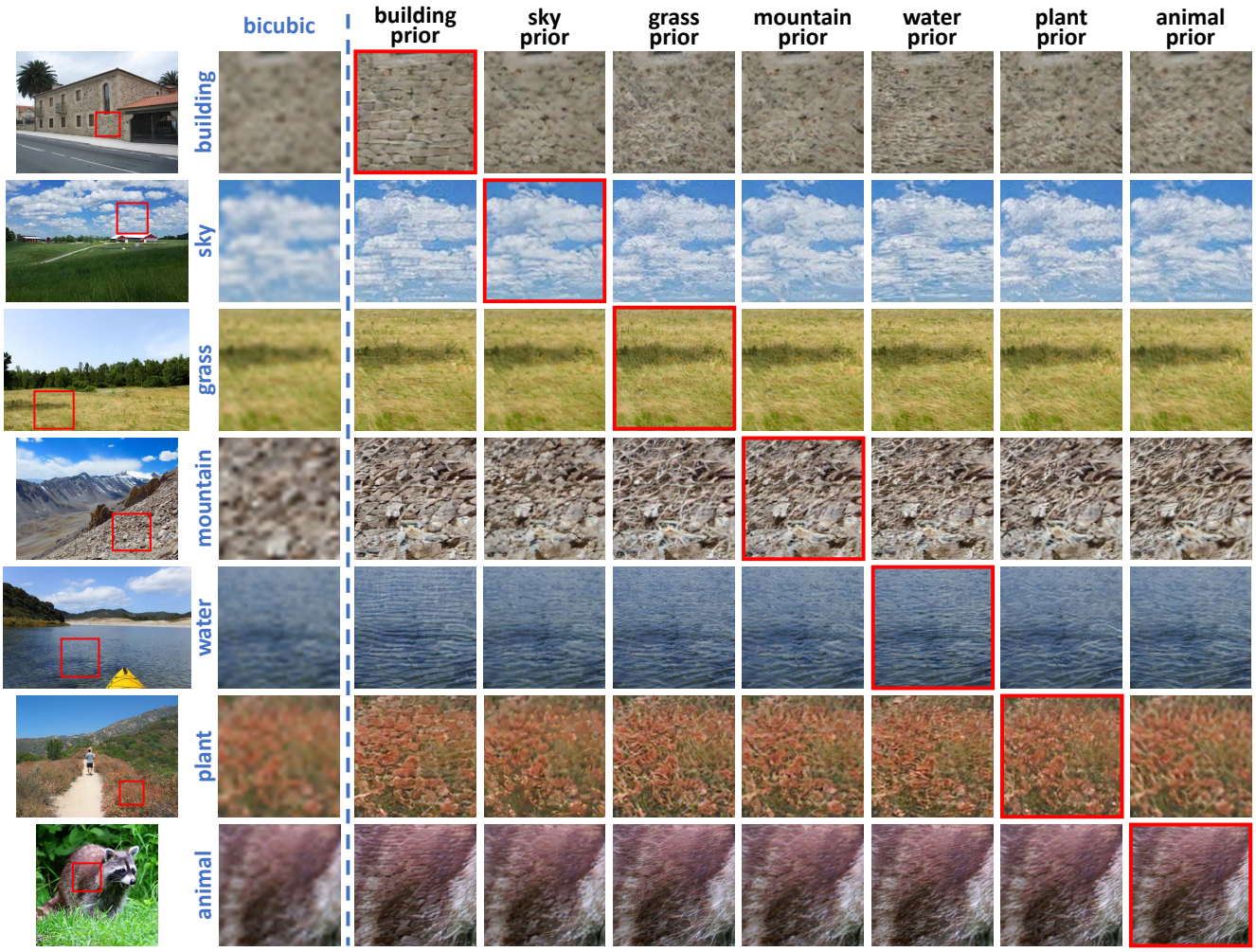


Figure 3. Super-resolved patches under different categorical priors. Restored textures under right priors are shown in red boxes. (**Zoom in for best view**).

are still able to generate results conditioned on the input segmentation probability maps. With more repetitive conditioning of SFT, we observe a slight improvement on the generated results. The observation indicates the capacity of SFT layer by modulating the feature maps. In future work, we will investigate more on the contribution of repetitive conditions.

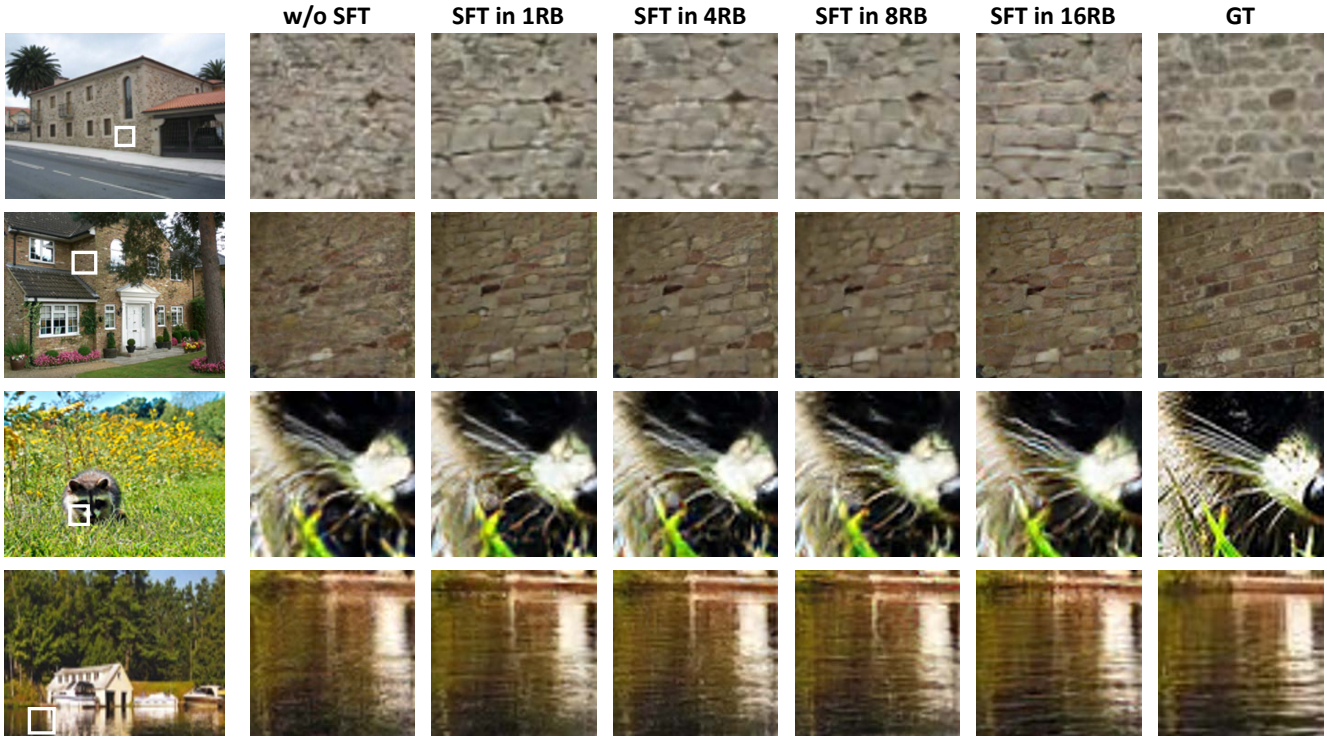


Figure 4. Models with fewer SFT layers are still able to generate results conditioned on the input segmentation probability maps. With more repetitive conditioning of SFT, we observe a slight improvement on the generated results. (**Zoom in for best view**).

## 5. Qualitative Comparisons

In this section, we provide more visual comparisons with state-of-the-art methods on OST300 and on out-of-category examples. Some failure cases are analyzed at the end of the supplemental material.

### 5.1. Visual comparisons on OST300

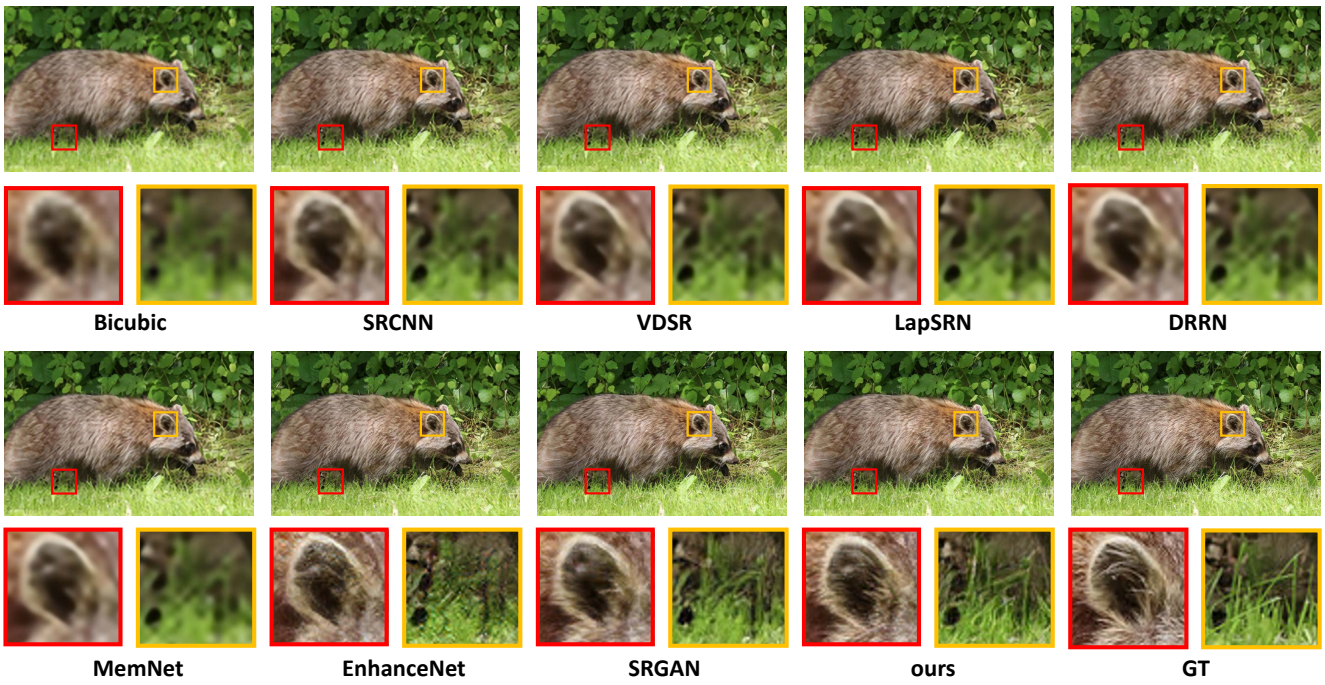


Figure 5. Qualitative comparison on OST_003 image.

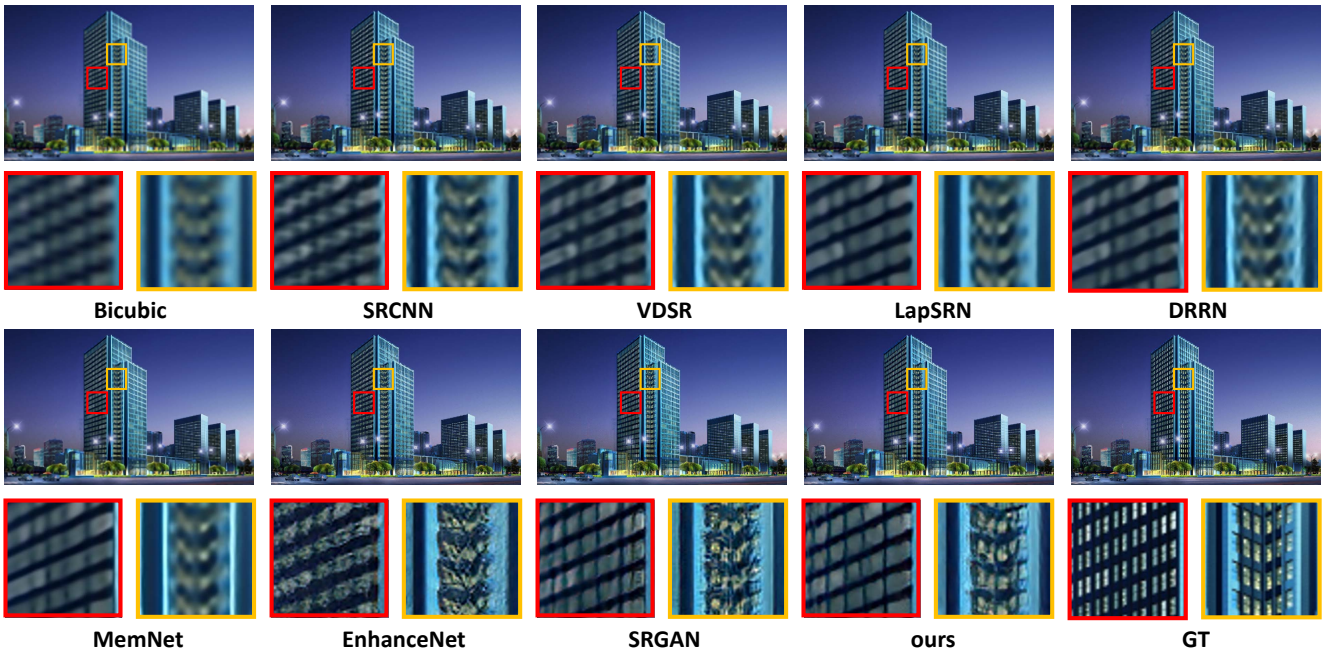


Figure 6. Qualitative comparison on OST_039 image.

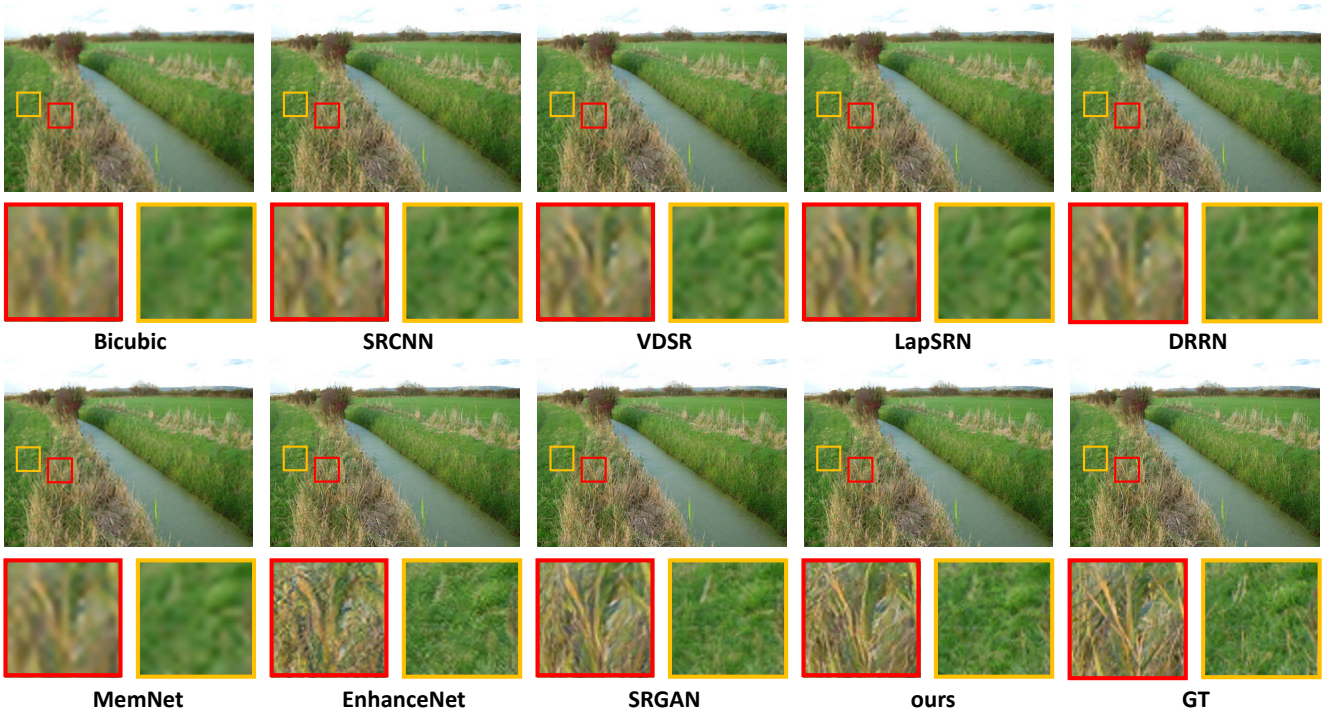


Figure 7. Qualitative comparison on OST_032 image.

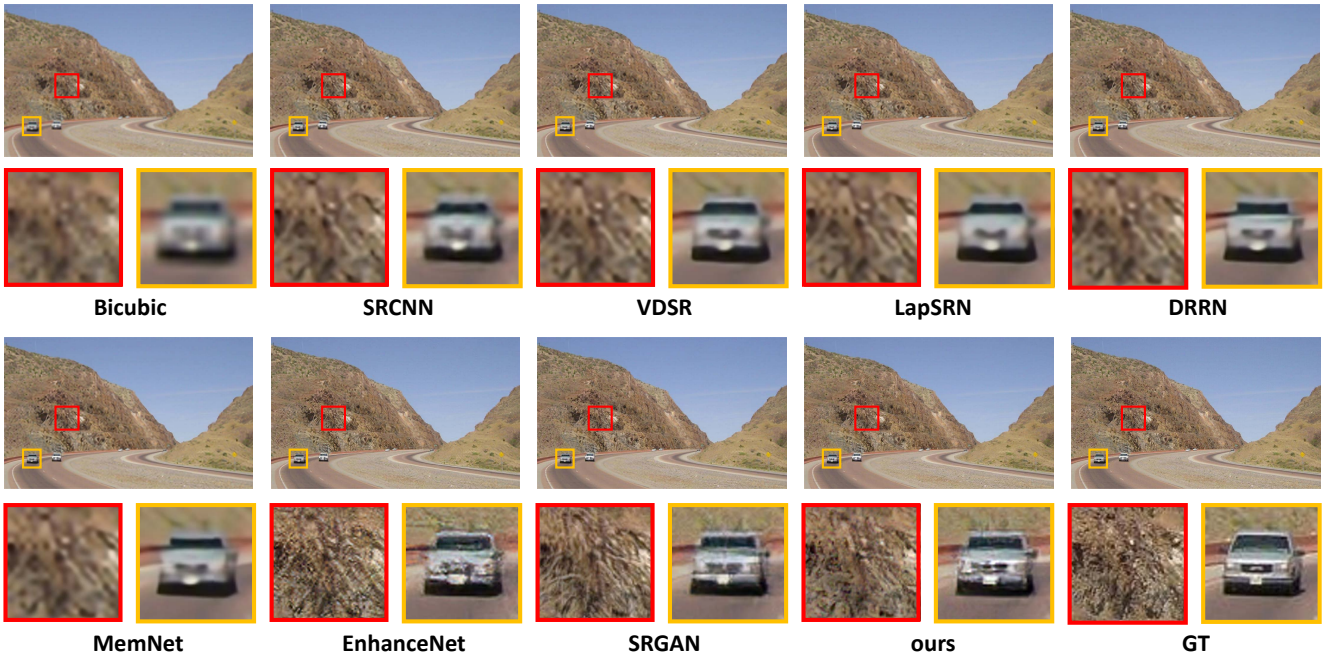


Figure 8. Qualitative comparison on OST_273 image.


Figure 9. Qualitative comparison on OST_158 image.

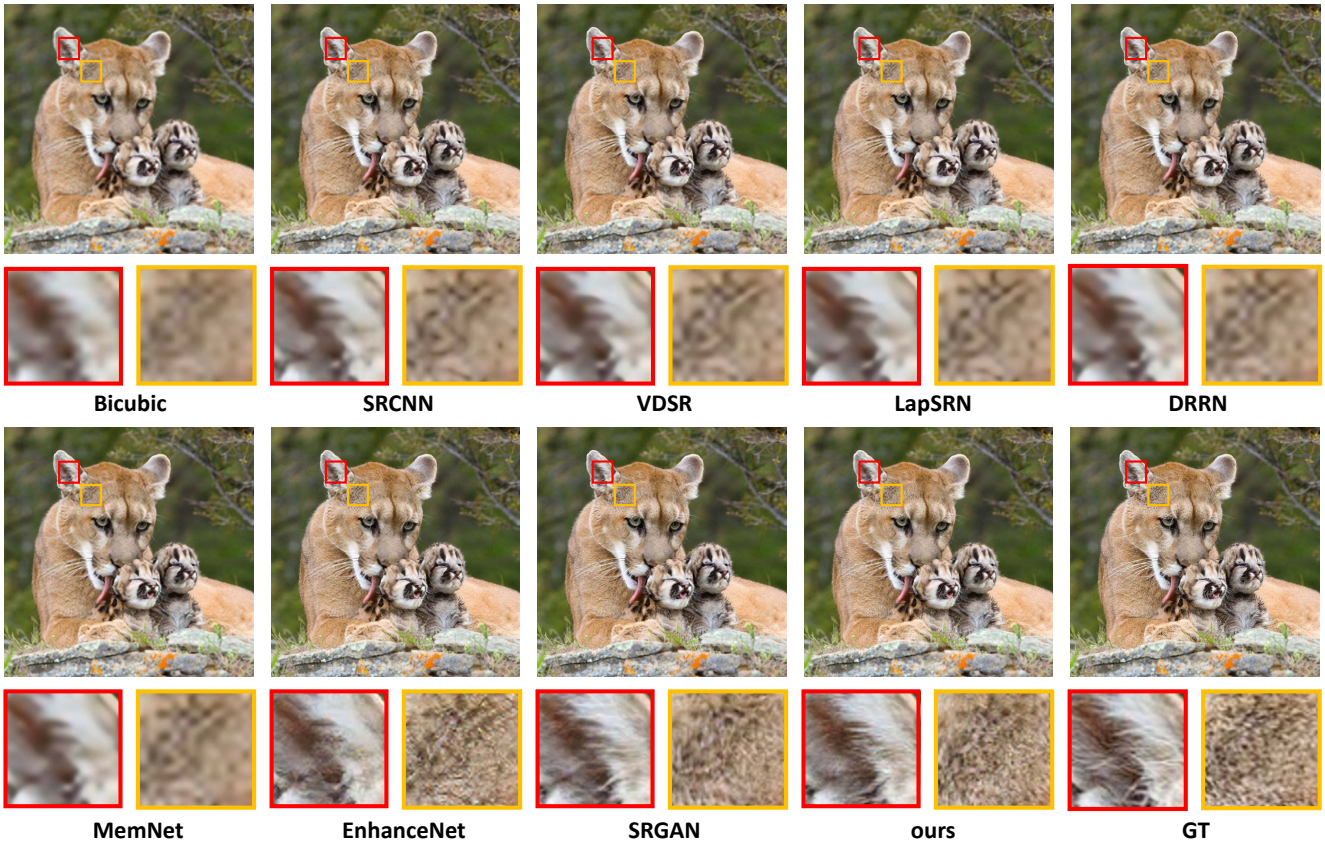


Figure 10. Qualitative comparison on OST_005 image.

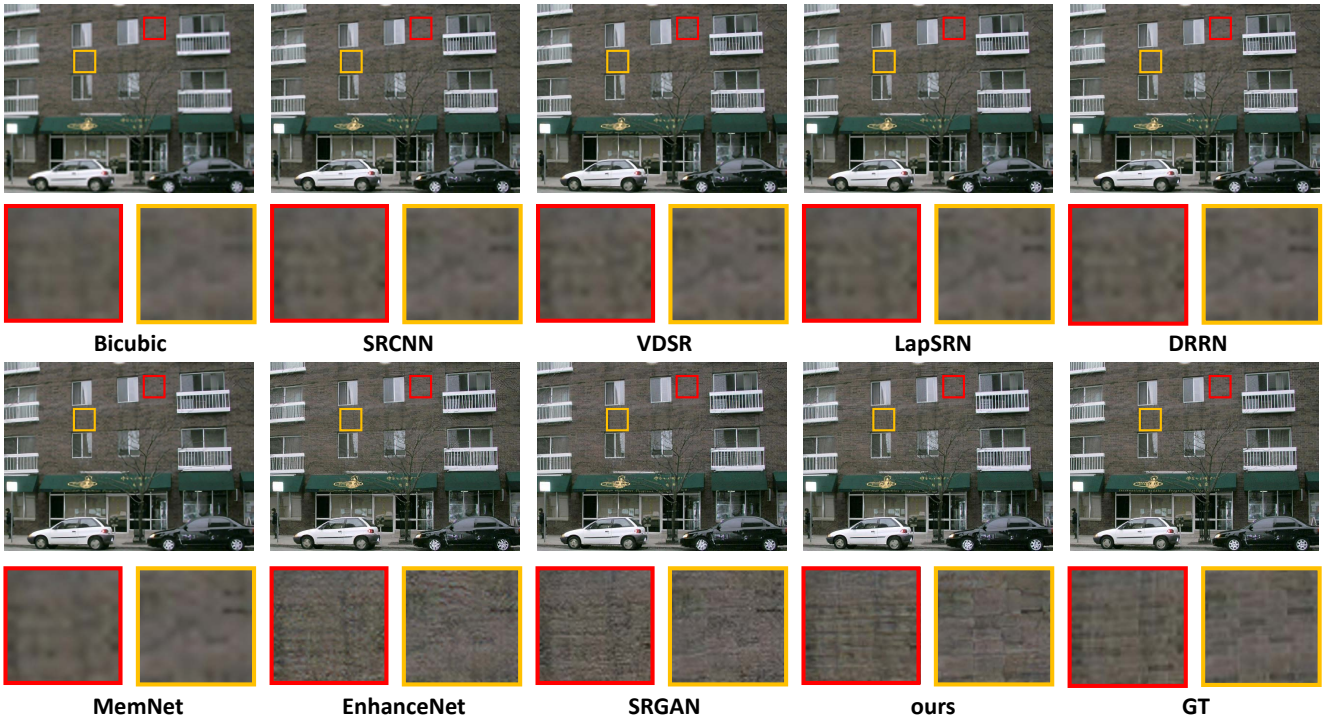


Figure 11. Qualitative comparison on OST_040 image.

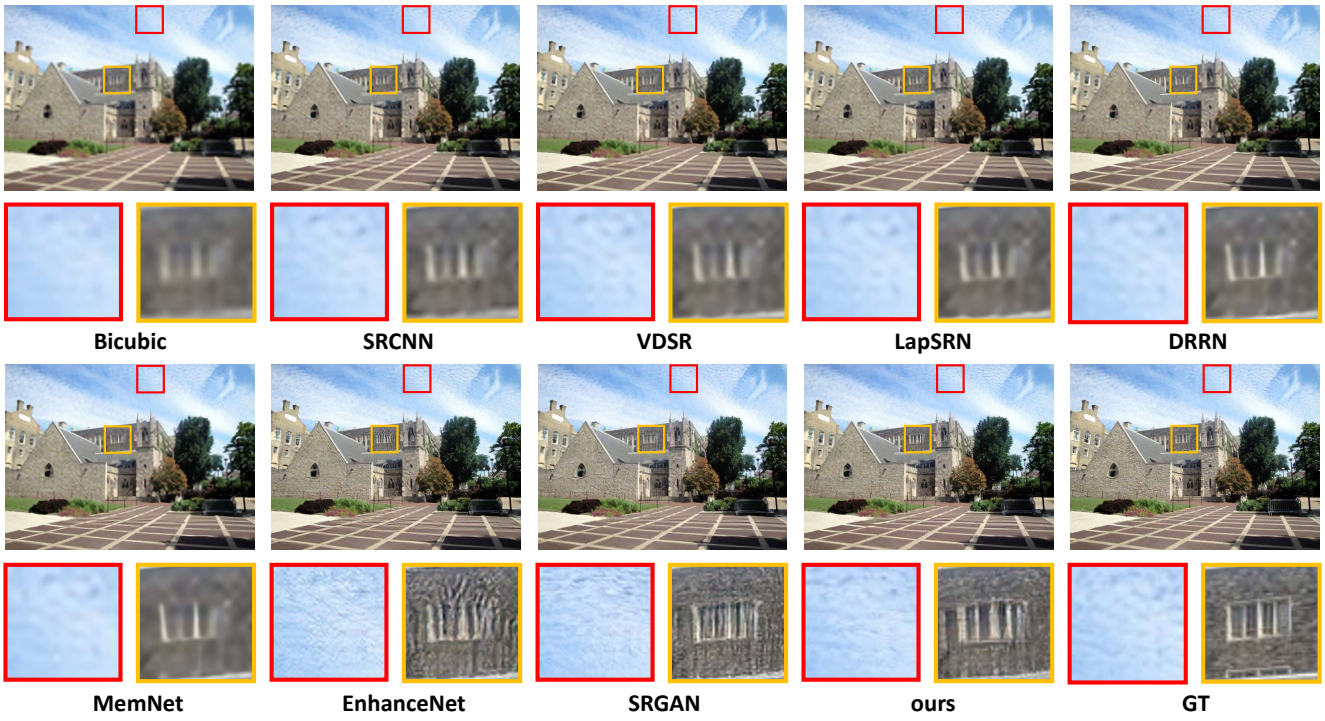


Figure 12. Qualitative comparison on OST_070 image.

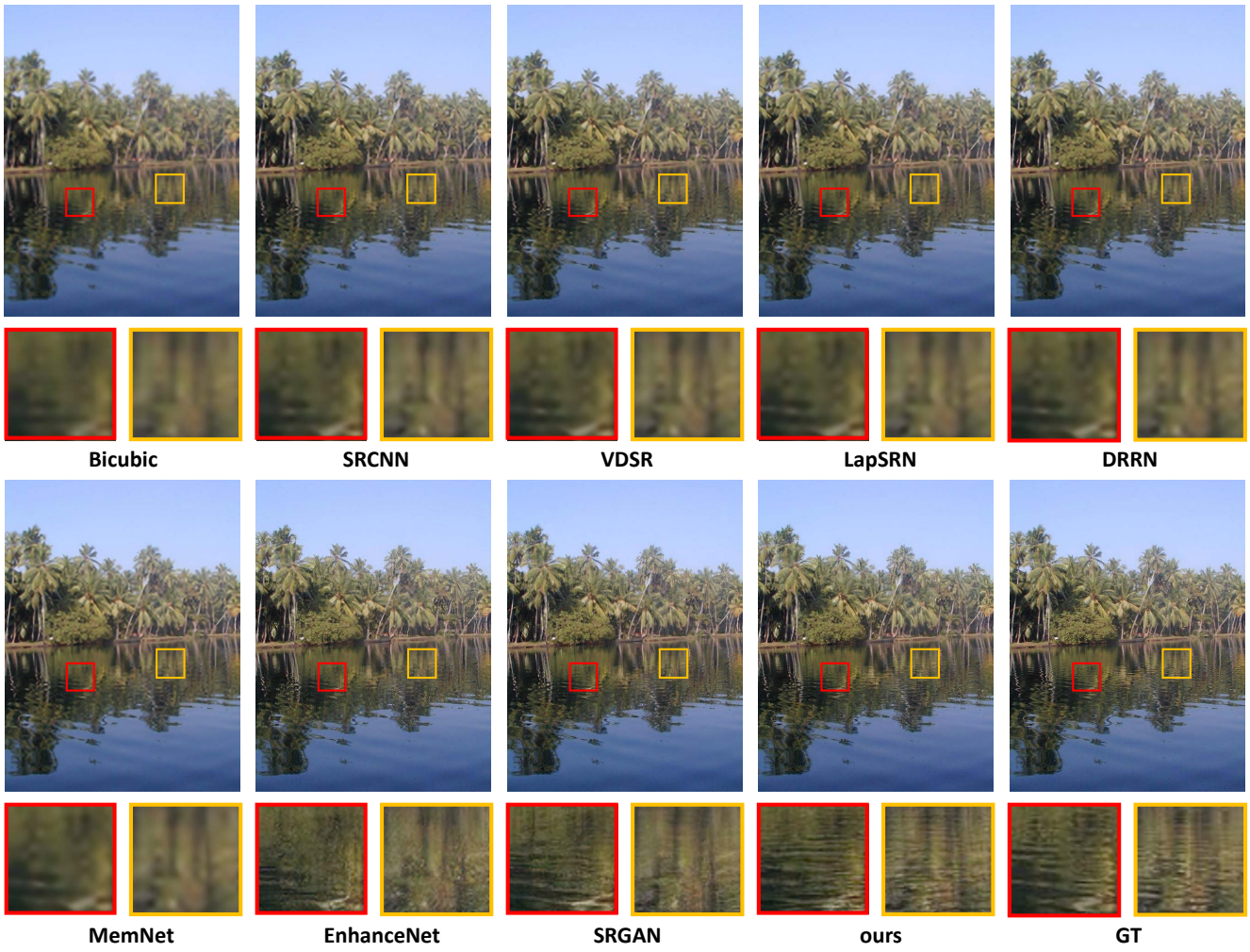


Figure 13. Qualitative comparison on OST_035 image.

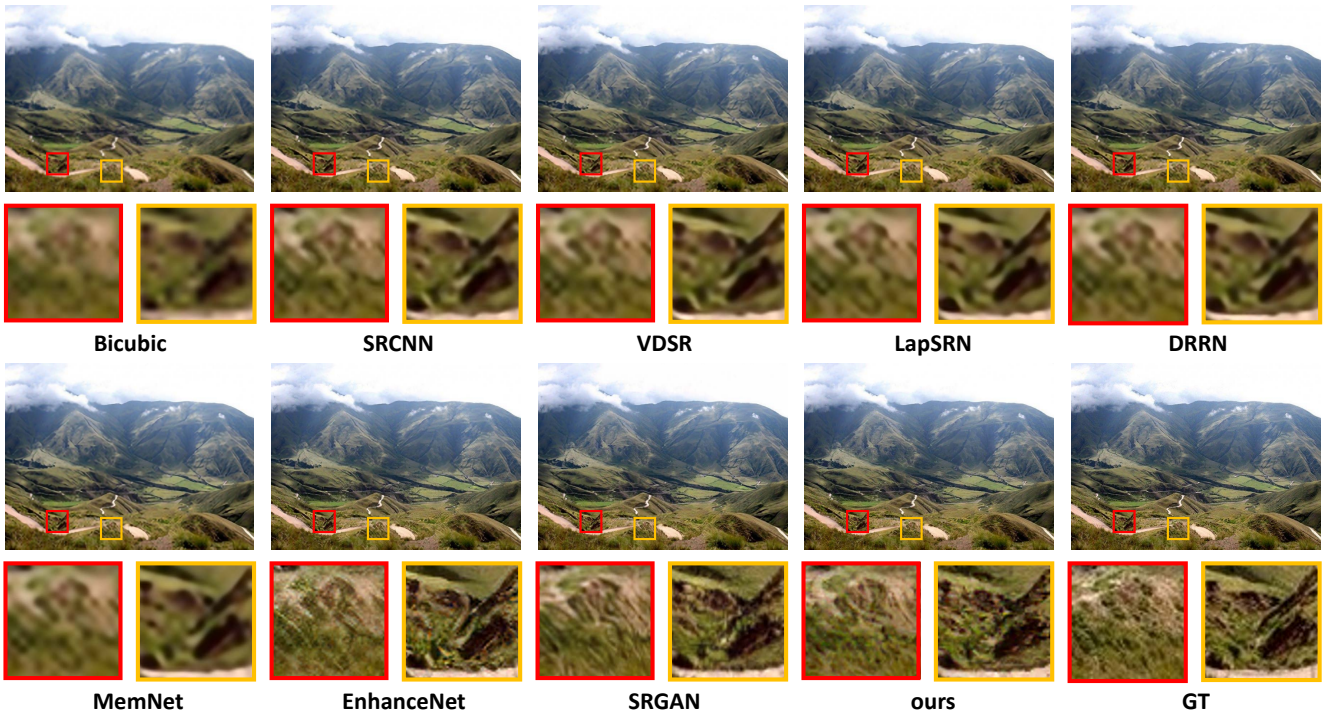


Figure 14. Qualitative comparison on OST_141 image.

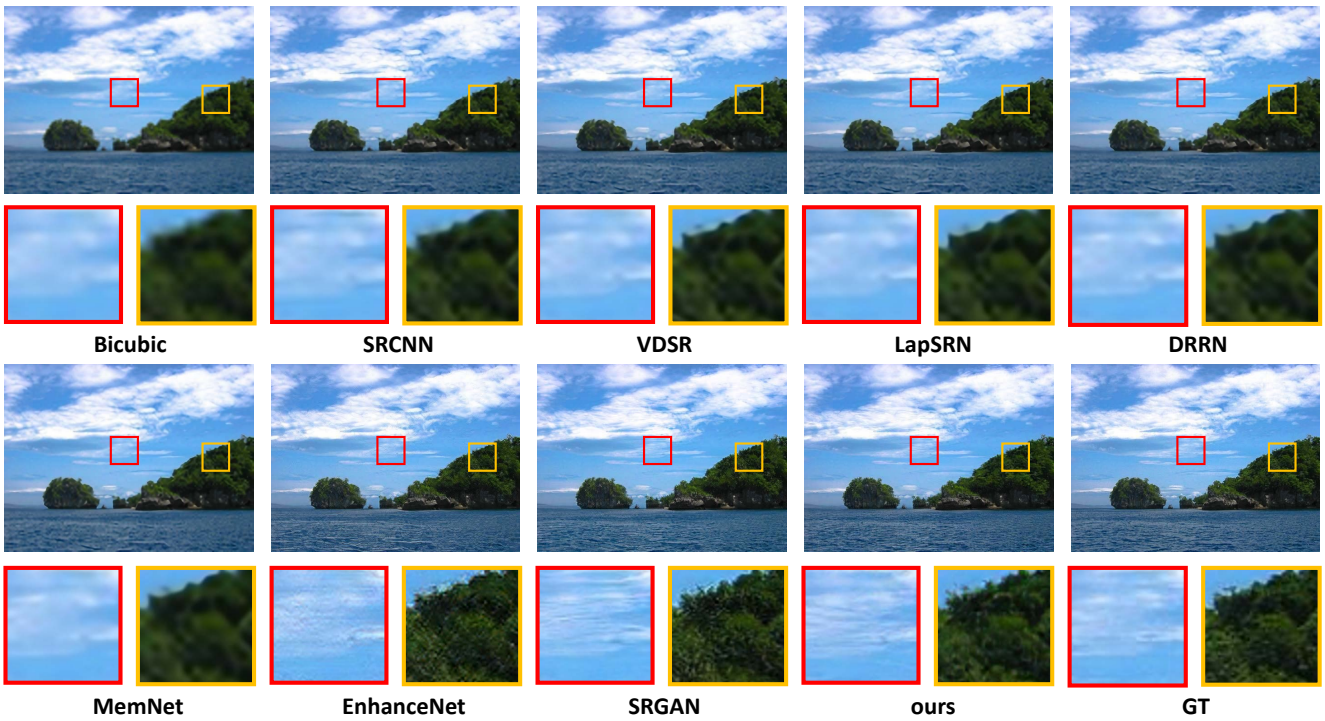


Figure 15. Qualitative comparison on OST_144 image.

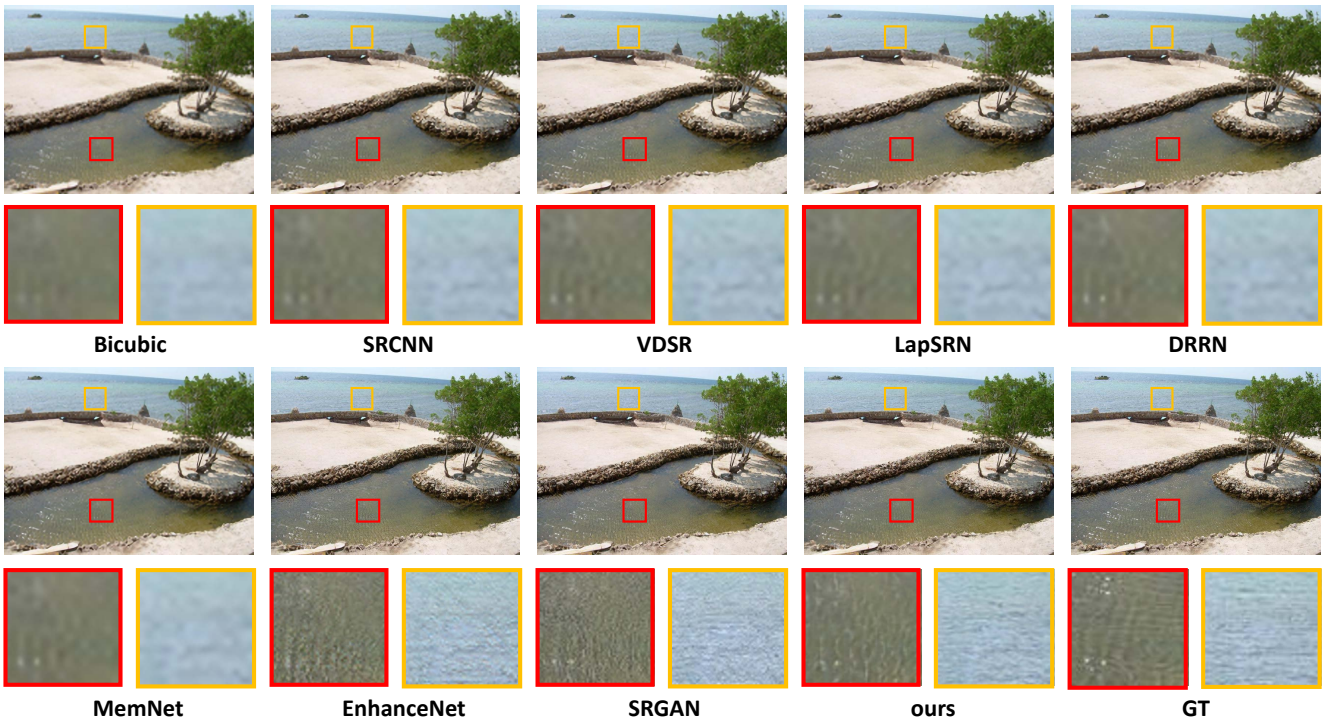


Figure 16. Qualitative comparison on OST_050 image.


Figure 17. Qualitative comparison on OST_225 image.

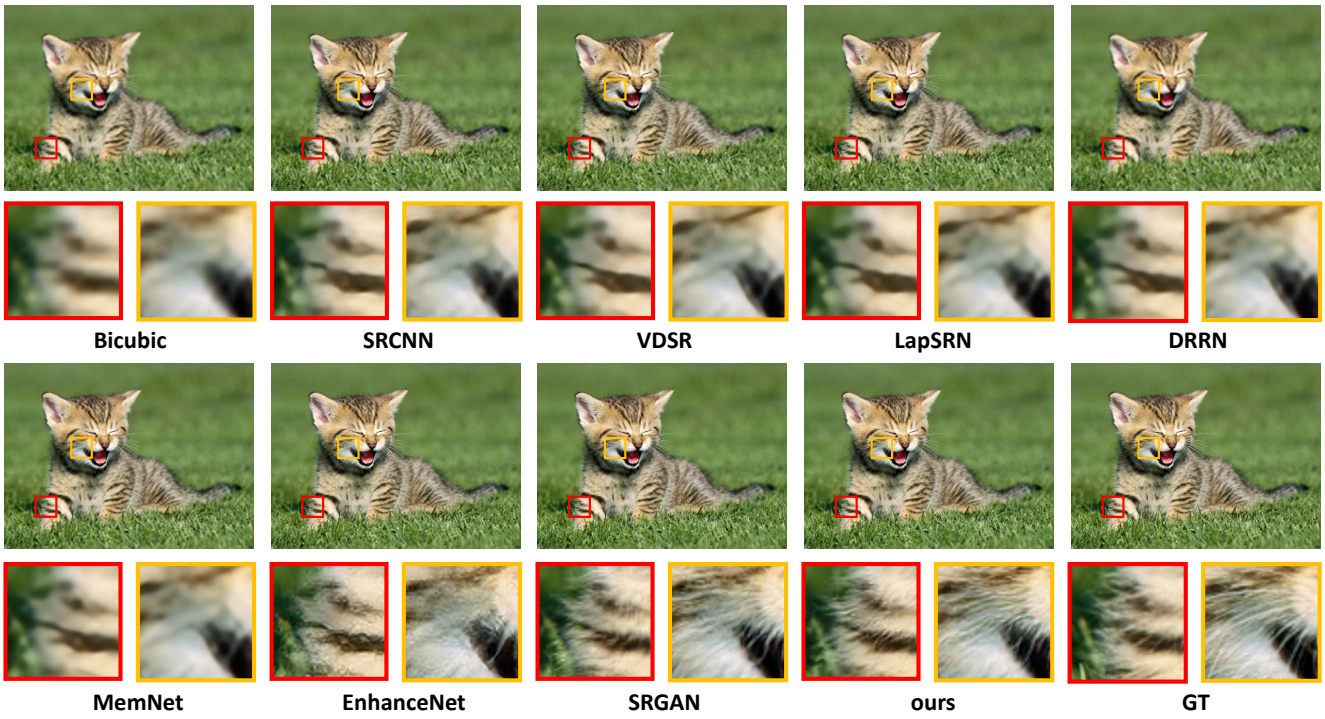


Figure 18. Qualitative comparison on OST_049 image.

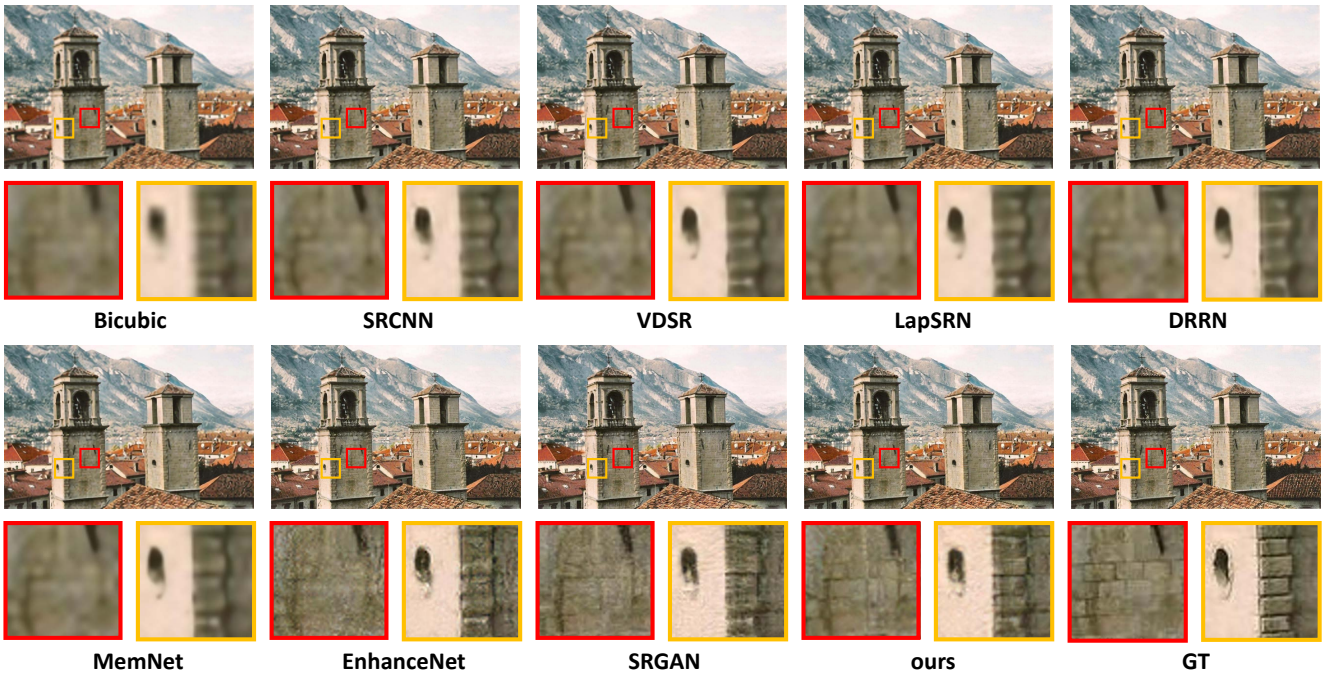


Figure 19. Qualitative comparison on OST_053 image.

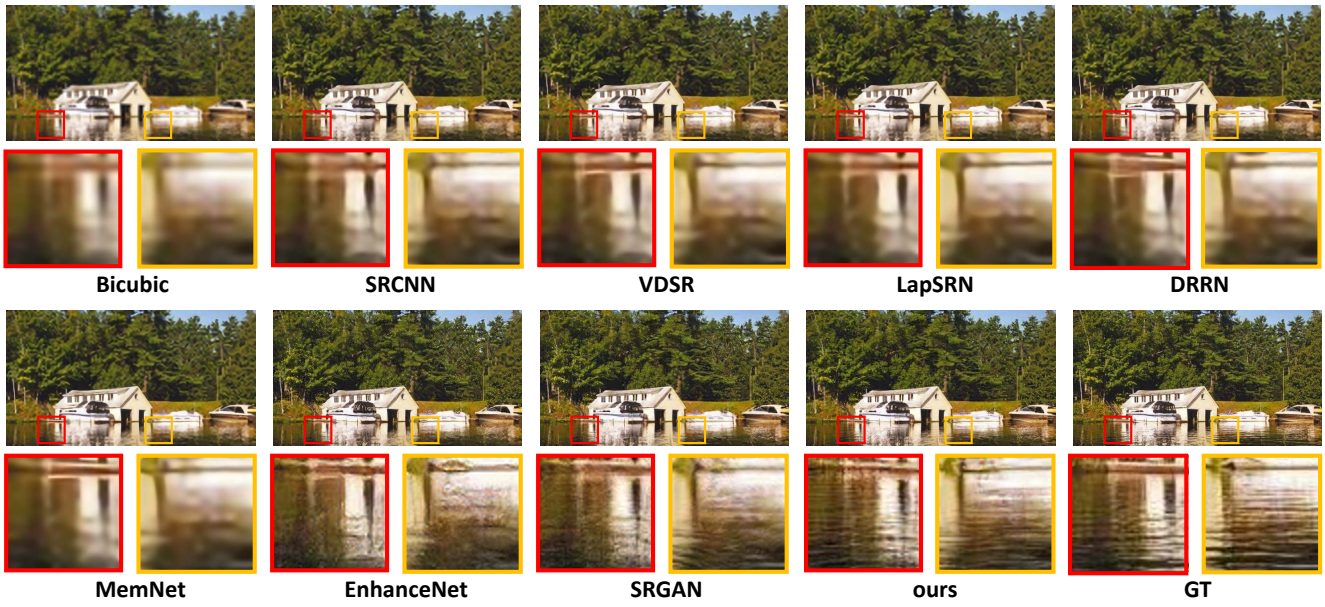


Figure 20. Qualitative comparison on OST_083 image.

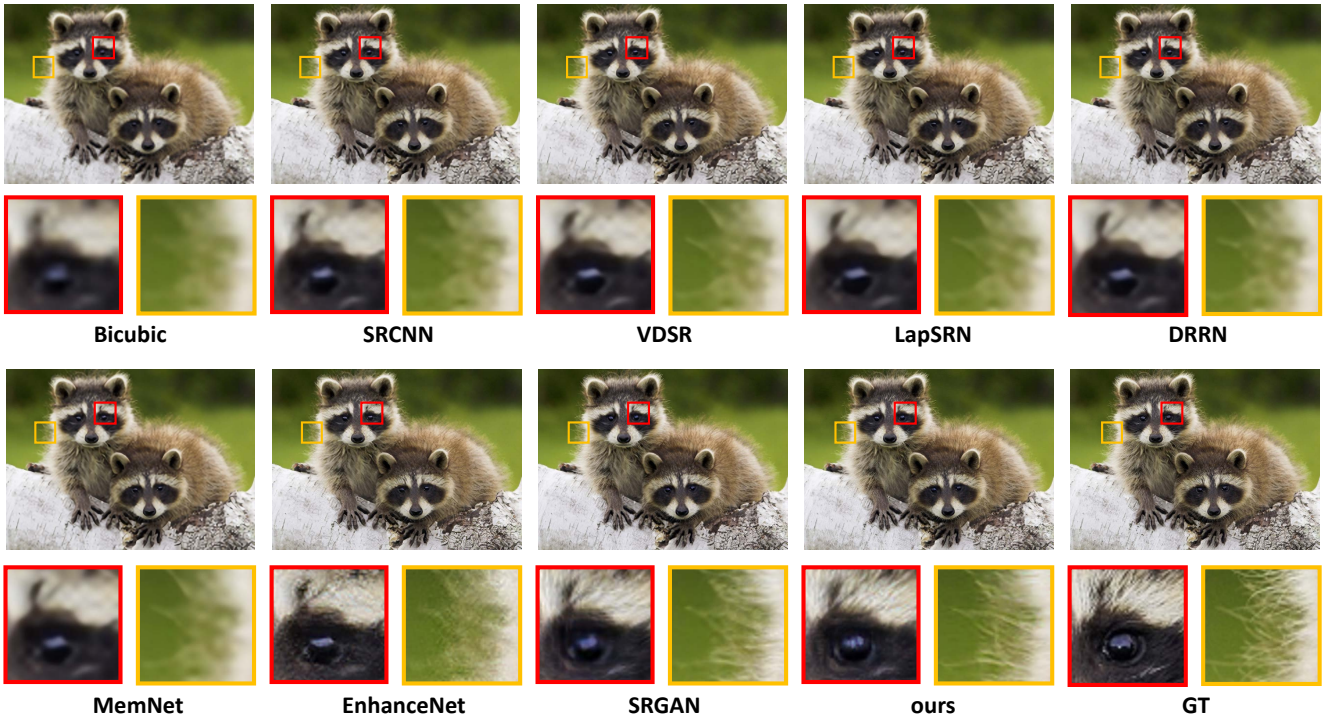


Figure 21. Qualitative comparison on OST_176 image.

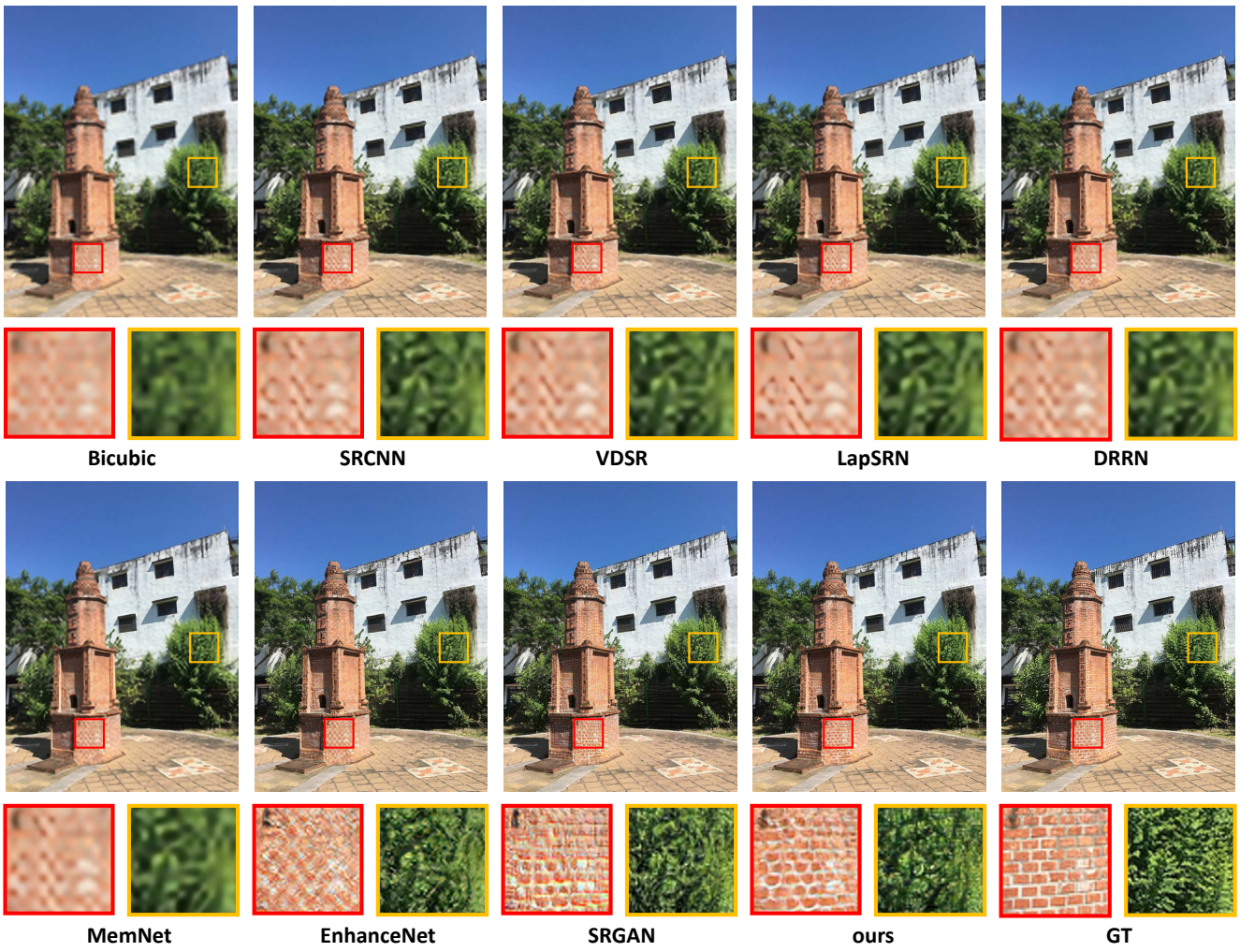


Figure 22. Qualitative comparison on OST_118 image.

### 5.2. Visual comparisons on out-of-category examples

When facing with other scenes or the absence of segmentation probability maps, our model degenerates itself as SRGAN and produces comparative results with SRGAN when all the regions are deemed as 'background'. Here, we shown some examples from Set14 and BSD100.

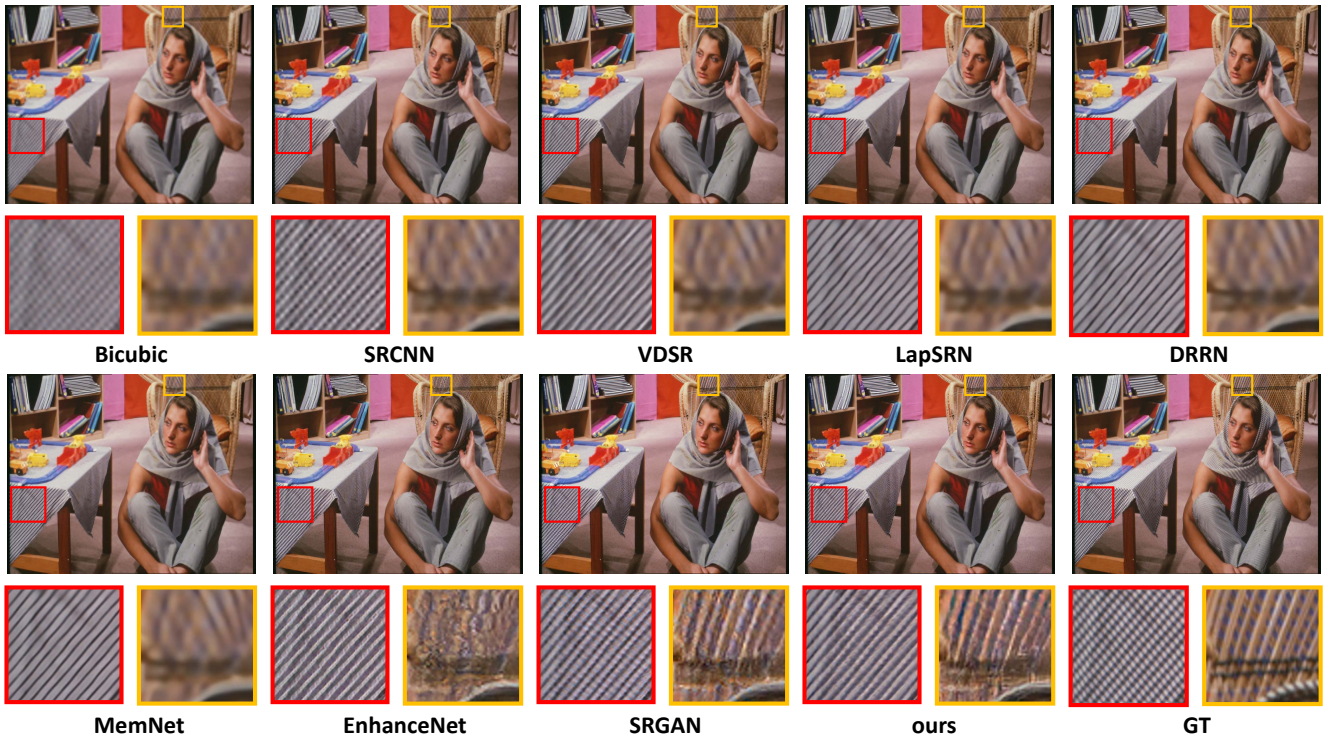


Figure 23. Qualitative comparison on Set14 - image *barbara*.

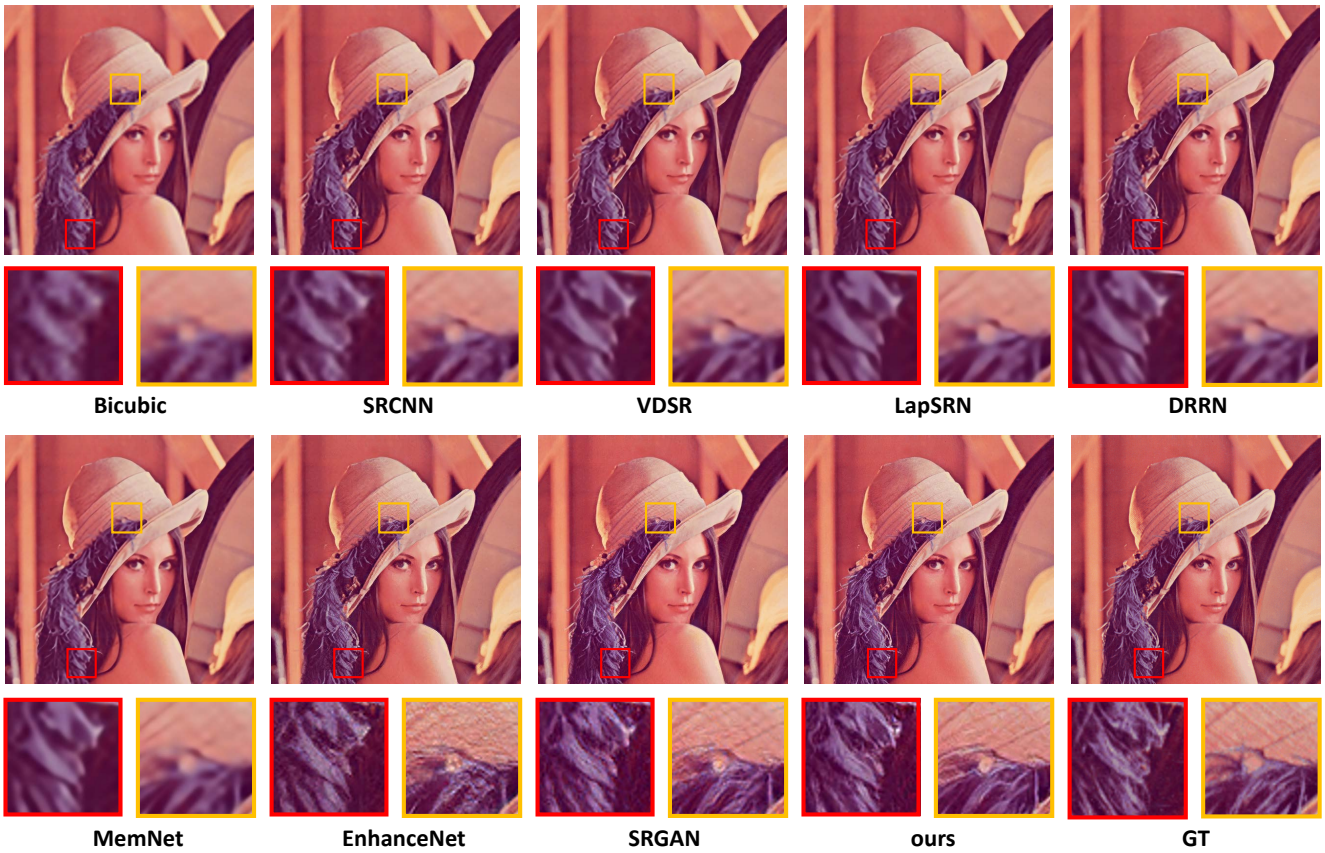


Figure 24. Qualitative comparison on Set14 - image *lenna*.


Figure 25. Qualitative comparison on Set14 - image *man*.

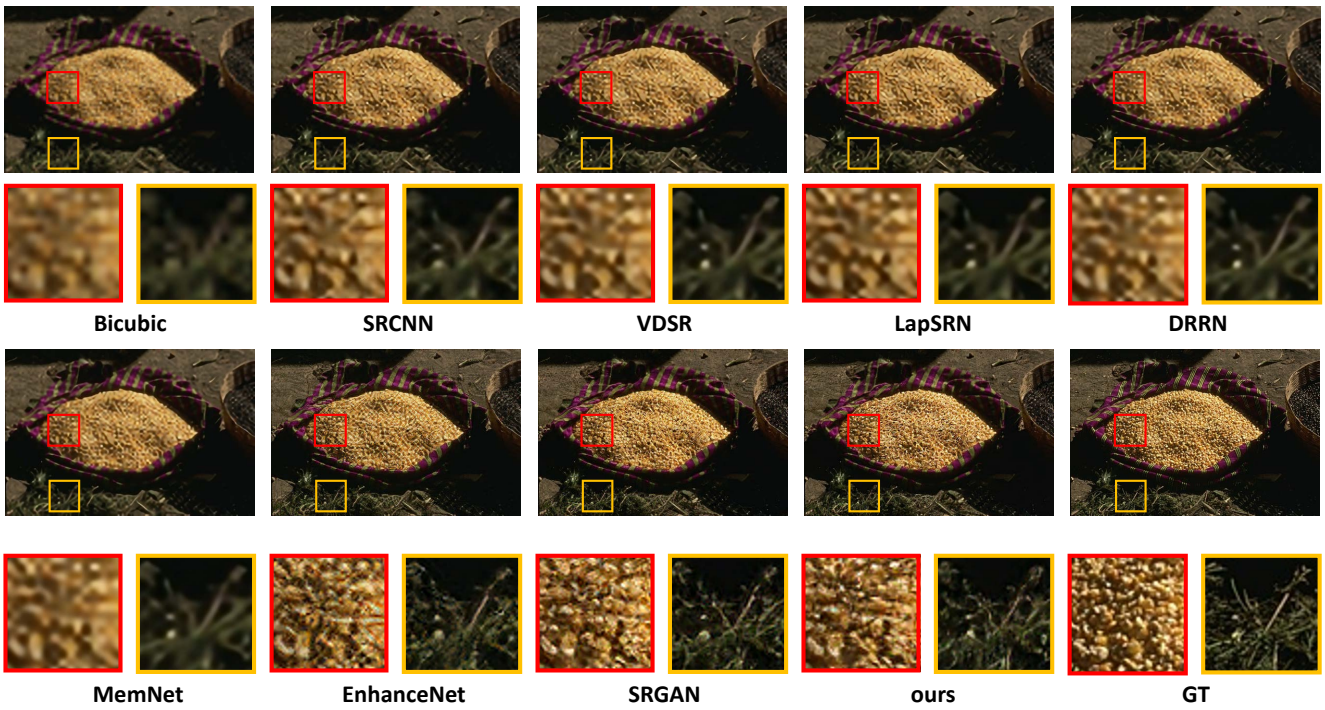


Figure 26. Qualitative comparison on BSD100 - image 58060.


Figure 27. Qualitative comparison on BSD100 - image 189080.

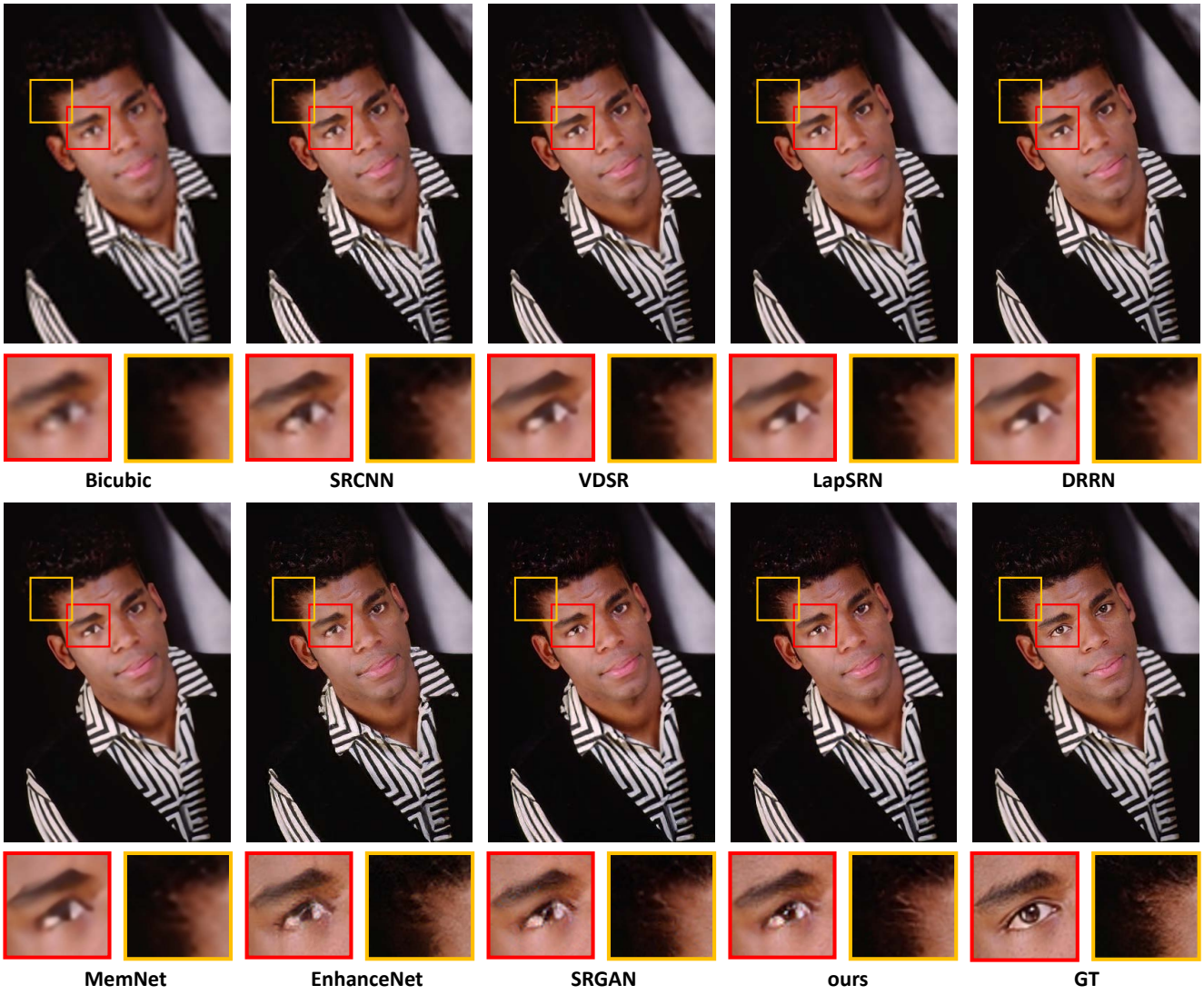


Figure 28. Qualitative comparison on BSD100 - image 302008.

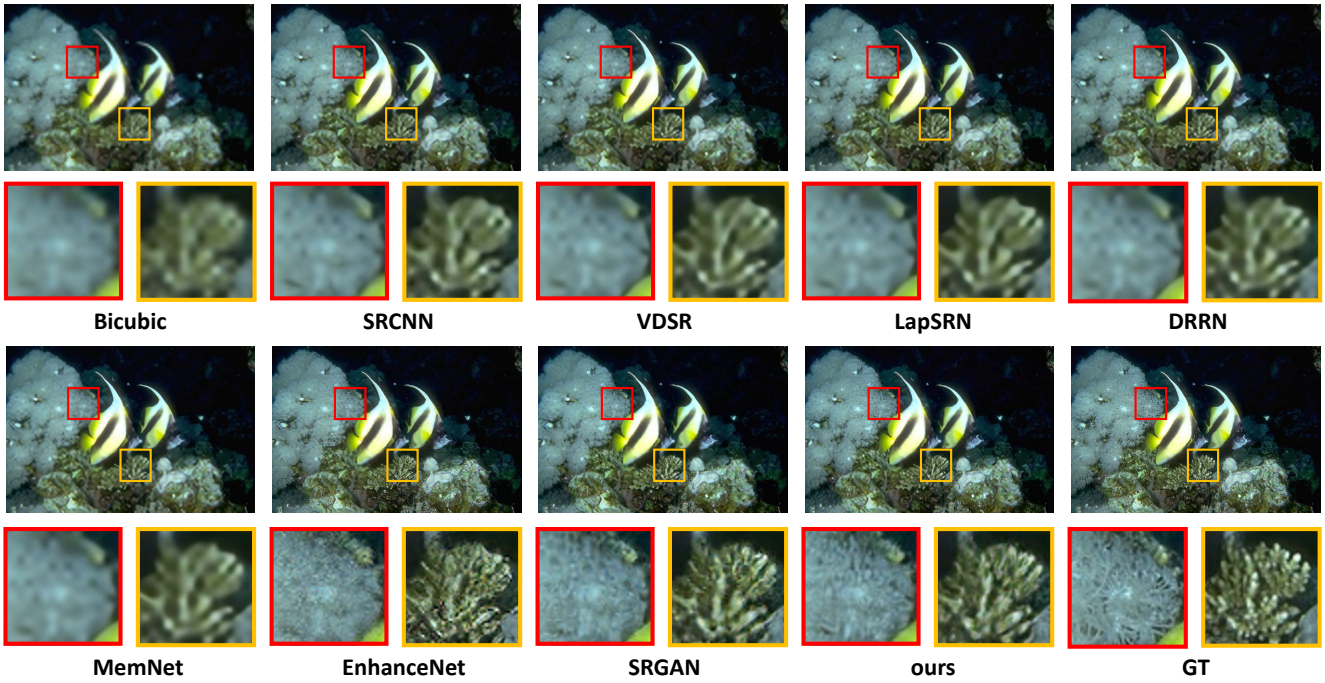


Figure 29. Qualitative comparison on BSD100 - image 306005.

### 5.3. Failure case analysis

We show two kinds of failure cases here. As shown in Fig. 30, wrong segmentation will result in improper texture restoration. In the example, the plant is on the surface of the building, increasing the difficulties for segmentation network. Unfortunately, part of plant region is misclassified as building region, leading to restoring plant texture under the building categorical prior. Therefore, the result is unsatisfactory.


Figure 30. Failure case arises in wrong segmentation.

Another failure case is caused by aliasing, as depicted in Fig. 31. The restored bricks have an unmatched direction as ground-truth since aliasing problem. Future work aims at addressing these shortcomings.

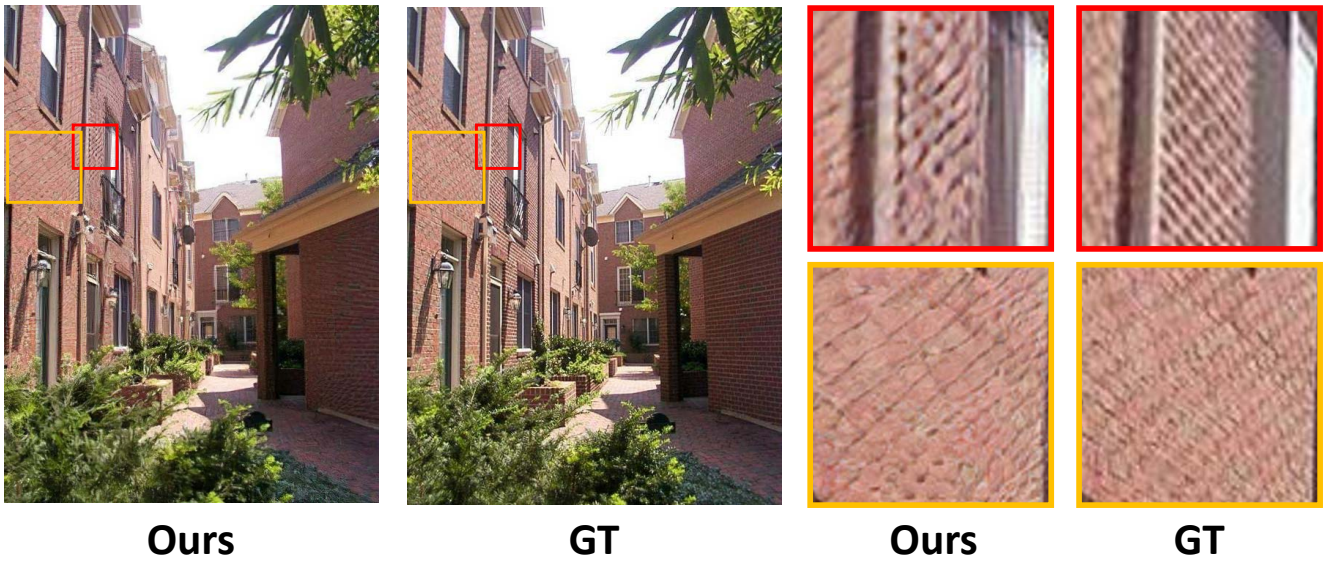


Figure 31. Failure case caused by aliasing.

## References


[1] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012. 3

[2] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 3

[3] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, 2014. 3, 4

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014. 3

[5] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016. 3, 4

[6] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *CVPR*, 2017. 3, 4

[7] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3, 4

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 2, 3

[9] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *ICCV*, 2015. 3

[10] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 3

[11] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001. 3

[12] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. *arXiv preprint arXiv:1610.09585*, 2016. 2

[13] M. S. Sajjadi, B. Schölkopf, and M. Hirsch. EnhanceNet: Single image super-resolution through automated texture synthesis. In *ICCV*, 2017. 3, 4

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[15] Y. Tai, J. Yang, and X. Liu. Image super-resolution via deep recursive residual network. In *CVPR*, 2017. 3, 4

[16] Y. Tai, J. Yang, X. Liu, and C. Xu. MemNet: A persistent memory network for image restoration. In *ICCV*, 2017. 3, 4

[17] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, L. Zhang, B. Lim, S. Son, H. Kim, S. Nah, K. M. Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *CVPR Workshop*, 2017. 3

[18] R. Zeyde, M. Elad, and M. Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, 2010. 3

[19] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017. 2