Supplementary Materials: Reference-based Image Super-Resolution with Deformable Attention Transformer

Jiezhang Cao[1], Jingyun Liang[1], Kai Zhang[1], Yawei Li[1], Yulun Zhang[1]⋆,
Wenguan Wang[1], and Luc Van Gool[1,2]

[1]Computer Vision Lab, ETH Zürich, Switzerland [2]KU Leuven, Belgium
{jiezhang.cao, jingyun.liang, kai.zhang, yawei.li, yulun.zhang,
wenguan.wang, vangool}@vision.ee.ethz.ch
https://github.com/caojiezhang/DATSR

In the supplementary materials, we first introduce more details of STL in Section A. In Section B, we provide more experiment details and network architectures. In Section C, we provide the comparisons of model size and the performance. In Section D, we provide more ablation studies of our proposed method. In Section E, we provide more disccusions for our method. In Section F, we provide more visual comparisons with state-of-the-art methods.

A More Details of STL

Recall that we use Swin Transformer layers STL$(\cdot)$ and a residual connection to extract deeper features of the LR and transferred features,

$$\boldsymbol{F}'_{l+1} = \mathrm{STL}(\boldsymbol{F}'_{l+1}) + \boldsymbol{F}_l, \tag{S1}$$

where $\boldsymbol{F}'_{l+1}$ is the output feature of the Conv$_3$ and $\boldsymbol{F}_l$ is the LR feature at the l-th scale. Based on the standard multi-head self-attention of the original Transformer layer [7], Swin Transformer [4] uses a shifted window mechanism to improve the performance. Specifically, given a feature $\boldsymbol{F}'_{l+1} \in \mathbb{R}^{H\times W\times C}$, we first reshape it to local window features $\hat{\boldsymbol{F}} \in \mathbb{R}^{\frac{HW}{M^2}\times M^2\times C}$ by partitioning the input into non-overlapping $M\times M$ local windows, where $\frac{HW}{M^2}$ is the total number of windows. Then, for a local window feature $\hat{\boldsymbol{F}} \in \mathbb{R}^{M^2\times C}$, the Query, Key and Value matrices $\hat{\boldsymbol{Q}}$, $\hat{\boldsymbol{K}}$ and $\hat{\boldsymbol{V}}$ can be computed as

$$\begin{cases} \hat{\boldsymbol{Q}} = \hat{\boldsymbol{F}}\boldsymbol{W}_q \in \mathbb{R}^{M^2\times d}, \\ \hat{\boldsymbol{K}} = \hat{\boldsymbol{F}}\boldsymbol{W}_k \in \mathbb{R}^{M^2\times d}, \\ \hat{\boldsymbol{V}} = \hat{\boldsymbol{F}}\boldsymbol{W}_v \in \mathbb{R}^{M^2\times d}, \end{cases} \tag{S2}$$

where $\boldsymbol{W}_q$, $\boldsymbol{W}_k$ and $\boldsymbol{W}_v$ are weights shared across different windows. Then, in a local window, the attention matrix can be computed with softmax function,

$$\mathrm{Attention}(\hat{\boldsymbol{Q}}, \hat{\boldsymbol{K}}, \hat{\boldsymbol{V}}) = \mathrm{Softmax}\left(\hat{\boldsymbol{Q}}\hat{\boldsymbol{K}}^\top/\sqrt{d} + \boldsymbol{B}\right)\hat{\boldsymbol{V}}, \tag{S3}$$

⋆ Corresponding author.

where $\boldsymbol{B}$ is the learnable relative positional encoding. Next, STL uses a regular multi-head self-attention (W-MSA), a shifted windowing multi-head self-attention (SW-MSA) and a multi-layer perceptron (MLP), followed by a LayerNorm (LN) layer and a residual connection. Here, MLP has two fully-connected layers with GELU non-linearity. Then, the above can be formulated as

$$\begin{cases} \hat{\boldsymbol{F}} = \text{W-MSA}(\text{LN}(\hat{\boldsymbol{F}})) + \hat{\boldsymbol{F}}, \\ \hat{\boldsymbol{F}} = \text{MLP}(\text{LN}(\hat{\boldsymbol{F}})) + \hat{\boldsymbol{F}}, \\ \hat{\boldsymbol{F}} = \text{SW-MSA}(\text{LN}(\hat{\boldsymbol{F}})) + \hat{\boldsymbol{F}}, \\ \hat{\boldsymbol{F}} = \text{MLP}(\text{LN}(\hat{\boldsymbol{F}})) + \hat{\boldsymbol{F}}. \end{cases} \tag{S4}$$

With the help of W-MSA and SW-MSA, STL can enable cross-window connections, which can relieve the issues of traditional vision Transformer.

B More Experiment Details

More implementation details. Besides, we also augment the training data by randomly changing the brightness, contrast and hue of an image by using ColorJitter in pytorch. Specifically, the factors of brightness, contrast and saturation are chosen uniformly from $[0.8, 1.2]$, and the hue factor is chosen uniformly from $[-0.05, 0.05]$. Following [1], we set the hype-parameters λ_1 and λ_2 as $1e{-}4$ and $1e{-}6$, respectively. We set the learning rate of the SR model and discriminator as $1e{-}4$. For the Adam optimizer, we set β_1=0.9 and β_2=0.999 in the training. The SR and LR images have the same aspect ratio. LR image is upsampled to the HR image size (the same as Ref image size) in training. In testing, when their sizes mismatch, we pad the smaller one with zeros to match the larger one. User study contains 20 users, and each user is asked to choose the image with better visual quality (e.g., ours v.s. TTSR) from SR image pairs for the WR-SR dataset. The final percentage is the average user preference of all images.

B.1 Network Architectures

Feature encoder. In our Transformer, we adopt two feature extractors because of the resolution gap between the LR and Ref images. In Table S1, we show the detailed architecture of the encoders. Following [1], the encoders E_l^q and E_l^k have the same architecture and share the same parameters, and the extractor E_l^v uses pretrained VGG-19 at each scale. Specifically, Conv is a convolution layer, and its the kernel size is 3×3. We use MaxPool to denote a Max pooling operation, and its kernel size is 2×2. Besides, ReLU is an activation function.

Table S1: The architecture of the feature encoder.

l-th layer	Layer information
0	Conv(3, 64), ReLU
1	Conv(64, 64), ReLU
2	MaxPool(2 × 2)
3	Conv(64, 128), ReLU
4	Conv(128, 128), ReLU
5	MaxPool(2 × 2)
6	Conv(128, 256)

RefSR network. The RefSR network consists of reference-based deformable attention (RDA) modules and residual feature aggregation (RFA) modules in both downsacling and upscaling, and the architecture is illustrated in Table. S2. Given an LR image, we first use bilinear to upsample it to the size of 160×160. For each image, we extract Query, Key and Value using pretrained VGG at each scale. For example, we use its relu1_1 features [6]) to denote *i.e.*, Query1, Key1 and Value1, and the size is 160×160. Similarly, we use its relu2_1 features to denote *i.e.*, Query2, Key2 and Value2, and the size is 80×80. Besides, we use relu3_1 features to denote Query3, Key3 and Value3, and the size is 40×40.

Then, we perform the RDA and RFA modules according to Figure 2 in the paper. For the RDA module, we take the Query, Key, Value and LR features as inputs. For the RFA module, it consists of two convolution layers and 8 Swin Transformer layers, and it takes the attention features as an input. For the convolution layer, the kernel size of convolution layers is 3×3, and S2 indicates the stride of 2. For the upsampling, we use PixelShuffle layers with the scale of 2×. The negative slope of LeakyReLU is 0.1. In SwinT, the window size is 8, the depth is [4, 4], and the number of heads is [4, 4]. In the training, the learning rate is set as 10^{-4}. For the training of the network with adversarial loss and perceptual loss, we adopt the same training setting as [12], and we use the RefSR model pre-trained on the reconstruction loss.

Table S2: The architecture of the SR network.

l**-th layer**	**Layer information**
0	Conv(3, 128), LeakyReLU(0.1)
1	RDA(Query1, Key1, Value1, #0)
2	Concat(#0, #1)
3	RFA(Conv(192, 128), SwinT×8, Conv(128, 128, S2), #2)
4	RDA(Query2, Key2, Value2, #3)
5	Concat(#3, #4)
6	RFA(Conv(256, 128), SwinT×8, Conv(128, 128, S2), #5)
7	RDA(Query3, Key3, Value3, #6)
8	Concat(#6, #7)
9	RFA(Conv(384, 128), SwinT×8, Conv(128, 512), #8)
10	PixelShuffle, LeakyReLU(0.1)
11	RDA(Query2, Key2, Value2, #10)
12	ElementwiseAdd(#3, #10)
13	Concat(#11, #12)
14	RFA(Conv(256, 128), SwinT×8, Conv(128, 512), #13)
15	PixelShuffle, LeakyReLU(0.1)
16	RDA(Query1, Key1, Value1, #15)
17	ElementwiseAdd(#0, #15)
18	Concat(#16, #17)
19	RFA(Conv(384, 128), SwinT×8, Conv(128, 64), #18)
20	LeakyReLU(0.1)
21	Conv(64, 3)

C More Comparisons of Model Size

As shown in Table S3, we show the comparison of model size (*i.e.*, the number of trainable parameters) of different models. Our proposed model has a total number of 18.0M parameters and achieves the best PSNR and SSIM of 28.72dB and 0.856, respectively. Our PSNR and SSIM are better than C^2-Matching [1] with a large margin, although our model size is higher than it. The light version has fewer parameters than C^2-Matching but has significantly better performance.

Table S3: Performance of different methods in terms of model sizes.

Methods	Params	PSNR	SSIM
RCAN [11]	16M	26.06	0.769
SwinIR [3]	11.9M	26.62	0.790
RankSRGAN [10]	1.5M	22.31	0.635
CrossNet [13]	33.6M	25.48	0.764
SRNTT [12]	4.2M	26.24	0.784
TTSR [9]	6.4M	27.09	0.804
C^2-Matching [1]	8.9M	28.24	0.841
Ours	18.0M	28.72	0.856

D More Ablation Studies

Effect on components in RDA. We conduct ablation studies on deformable convolution (DCN) in Table S4. Our model with DCN has higher PSNR/SSIM results as it can transfer better texture from reference images.

Table S4: Ablation study on the CUFED5 testing set.

Methods	w/o DCN	**Ours**
PSNR/SSIM	28.34/0.844	**28.72/0.856**

Effect of λ_1 and λ_2. We conduct an experiment to investigate the impact of the hyper-parameters λ_1 and λ_2. Following the settings of [12,1], we set $\lambda_1{=}1e{-}4$ and $\lambda_2{=}1e{-}6$. Based on this, we also select λ_1 from $\{1e{-}3, 1e{-}4, 1e{-}5\}$, and select λ_2 from $\{1e{-}5, 1e{-}6, 1e{-}7\}$. From Tables S5 and S6, when we set λ_1 as $1e{-}4$ or $1e{-}5$, we have the comparable PSNR but has the better LPIPS at $1e{-}4$. Similarly, we have the same conclusion for λ_2. To obtain a good trade-off among the regression loss, perceptual loss and adversarial loss, we set $\lambda_1 = 1e{-}4$ and $\lambda_2 = 1e{-}6$ in practice, which is the same as [12,1].

Table S5: Performance in terms of λ_1.

λ_1	1e−3	1e−4	1e−5
PSNR	27.89	27.95	28.09
LPIPS	0.150	**0.140**	0.184

Table S6: Performance in terms of λ_2.

λ_2	1e−5	1e−6	1e−7
PSNR	27.95	27.95	28.10
LPIPS	0.152	**0.140**	0.196

E More Disccusions

Contribution on using multiple patches. MuCAN [2] and IGNN [14] aggregate similar patches across frames and in local regions, respectively, but their performance may be limited when patches are from different distributions. In contrast, our model is robust to this case (see Fig. 7 in the paper) due to our deformable convolution.

Confidence intervals on performance results. We compare our model with C^2-Matching, and calculate the improvement gains of our method for each sample in the CUFED5 testing set. The histogram of improved PSNR values is shown in Fig. S1, from which we can see over 96% samples outperform C^2-Matching-rec and C^2-Matching in [0.002dB, 3dB].

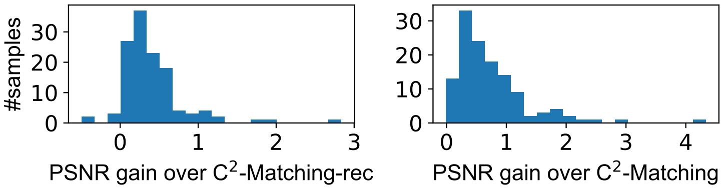

Fig. S1: The number of samples for performance gain.

Social impact concerns. In this paper, we propose a new reference-based image super-resolution Transformer, which is end-to-end trainable and achieves state-of-the-art performance. However, there may bring some societal impacts, such as subject identity change. RefSR is an ill-posed problem. Its input is severely degraded and may correspond to multiple HR images by nature. The model may reconstruct SR images with different details especially with GAN training, which may change the object identity.

F More Visual Comparisons

In Fig. S2, we provide more visual comparisons with RCAN [11], SwinIR [3], ESRGAN [8], RankSRGAN [10], SRNTT [12], TTSR [9], MASA [5] and C^2-Matching [1]. Our model achieves the best performance on visual quality. Thus, our method is able to transfer accurate textures from the Ref images to generate SR images. Moreover, our method can search and transfer meaningful texture in a local regions even if the Ref image is not globally relevant to the input image.

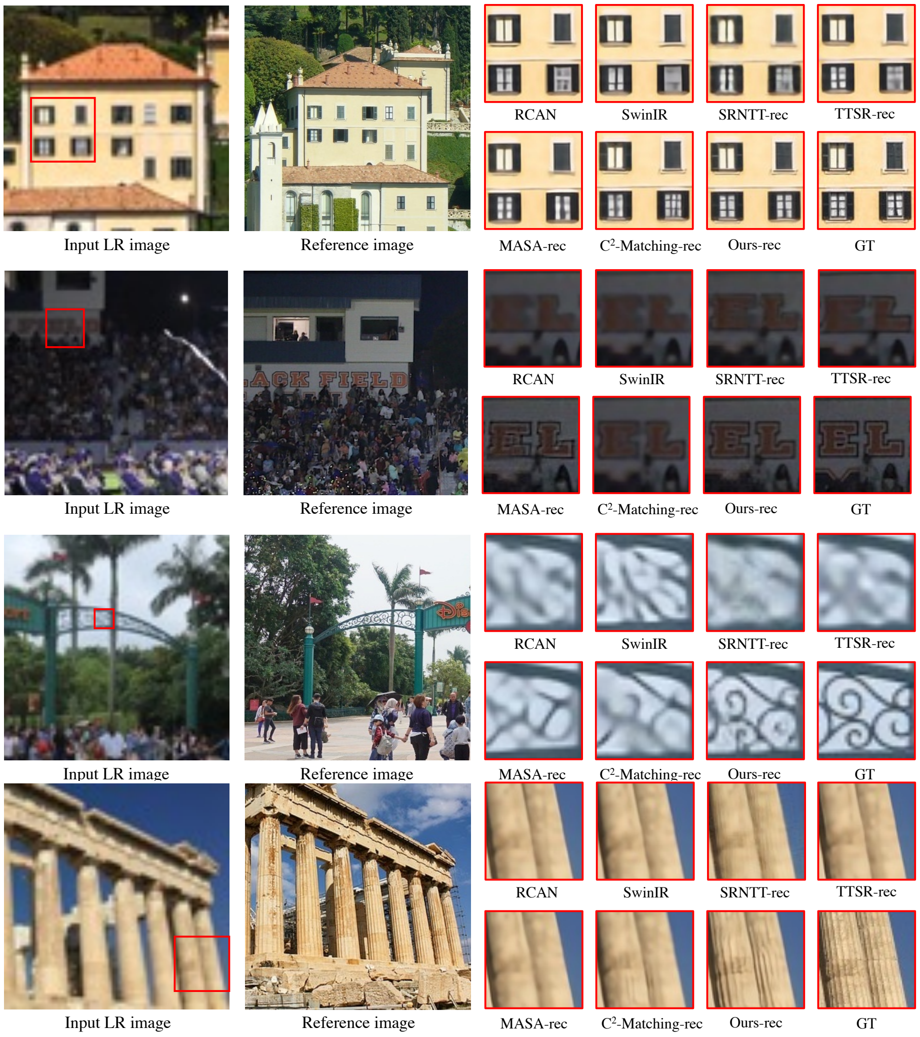

Fig. S2: Qualitative comparisons of SISR and RefSR models trained with the reconstruction loss.

In Fig. S3, when trained with adversarial loss, our model is able to restore the realistic details in the output images which are very close to the HR ground-truth images with the help of the given Ref images. In contrast, it is hard for ESRGAN and RankSRGAN to generate realistic images without the Ref images since the degradation is severely destroyed and high frequency details are lost.

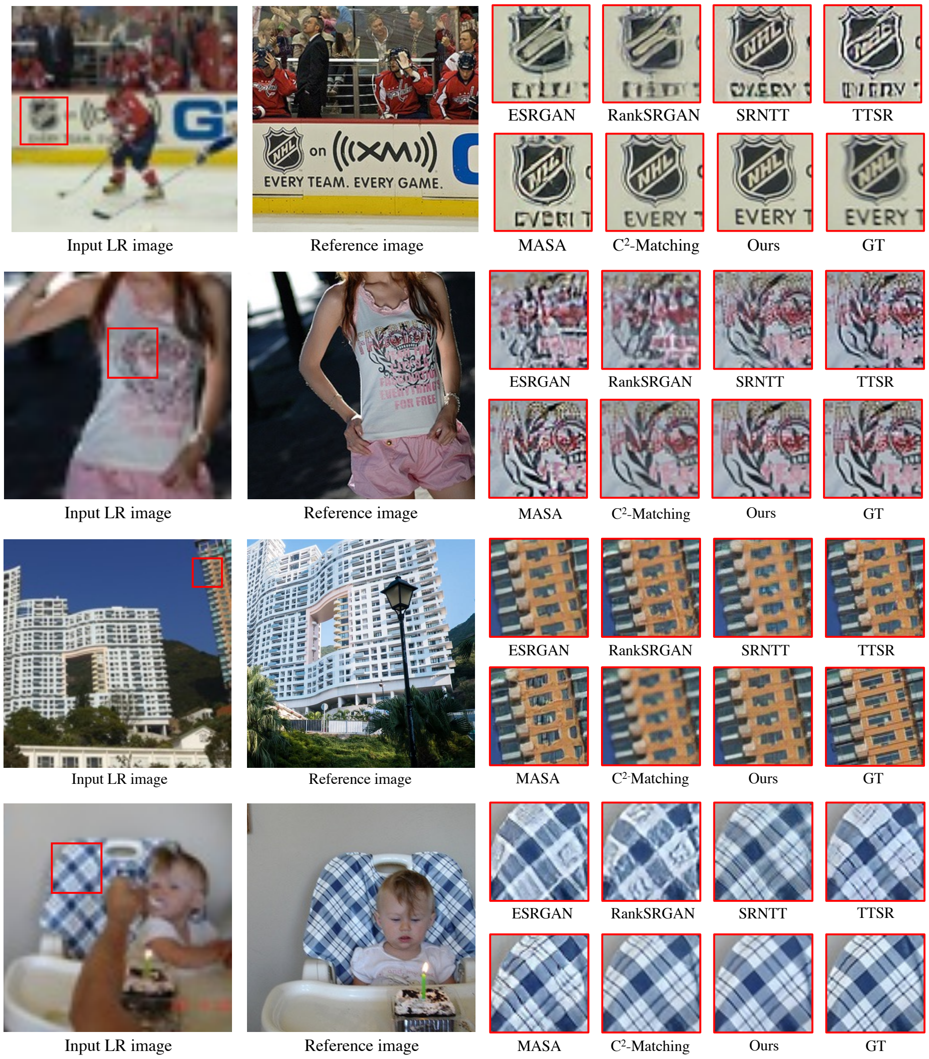

Fig. S3: Qualitative comparisons of SISR and RefSR models trained with all loss.

References

1. Jiang, Y., Chan, K.C., Wang, X., Loy, C.C., Liu, Z.: Robust reference-based super-resolution via c2-matching. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2103–2112 (2021)
2. Li, W., Tao, X., Guo, T., Qi, L., Lu, J., Jia, J.: Mucan: Multi-correspondence aggregation network for video super-resolution. In: European Conference on Computer Vision. pp. 335–351 (2020)
3. Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., Timofte, R.: Swinir: Image restoration using swin transformer. In: IEEE International Conference on Computer Vision. pp. 1833–1844 (2021)
4. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: IEEE International Conference on Computer Vision. pp. 10012–10022 (2021)
5. Lu, L., Li, W., Tao, X., Lu, J., Jia, J.: Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 6368–6377 (2021)
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015)
7. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. vol. 30 (2017)
8. Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., Change Loy, C.: Esrgan: Enhanced super-resolution generative adversarial networks. In: European Conference on Computer Vision Workshops. pp. 0–0 (2018)
9. Yang, F., Yang, H., Fu, J., Lu, H., Guo, B.: Learning texture transformer network for image super-resolution. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 5791–5800 (2020)
10. Zhang, W., Liu, Y., Dong, C., Qiao, Y.: Ranksrgan: Generative adversarial networks with ranker for image super-resolution. In: IEEE International Conference on Computer Vision. pp. 3096–3105 (2019)
11. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: European Conference on Computer Vision. pp. 286–301 (2018)
12. Zhang, Z., Wang, Z., Lin, Z., Qi, H.: Image super-resolution by neural texture transfer. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 7982–7991 (2019)
13. Zheng, H., Ji, M., Wang, H., Liu, Y., Fang, L.: Crossnet: An end-to-end reference-based super resolution network using cross-scale warping. In: European Conference on Computer Vision. pp. 88–104 (2018)
14. Zhou, S., Zhang, J., Zuo, W., Loy, C.C.: Cross-scale internal graph neural network for image super-resolution. In: Advances in Neural Information Processing Systems. pp. 3499–3509 (2020)