Supplementary Material for
Beyond Image Super-Resolution for Image Recognition with Task-Driven Perceptual Loss

In this supplementary document, we show the additional results and ablation studies omitted from the main manuscript due to the lack of space and describe the details:

S1. Effectiveness of our SR results

In the main manuscript, we argue that our SR4IR can restore task-relevant high-frequency details that are beneficial for a subsequent image recognition task. To further demonstrate our claim, we compare the SR results trained by our SR4IR (referred to as $\boldsymbol{I}_{\text{SR(SR4IR)}}$) with two other sets of SR results that are trained by (1) using pixel-wise reconstruction loss (referred to as $\boldsymbol{I}_{\text{SR(pixel)}}$) and (2) employing a combination of pixel-wise loss and conventional perceptual loss (referred to as $\boldsymbol{I}_{\text{SR(pixel+percep)}}$).

Performance on $S \rightarrow T$ setting. We compare the performance of the task network trained by the three types of SR images following the same setting as $S \rightarrow T$ described in Section 4. Table S1 shows that using $\boldsymbol{I}_{\text{SR(SR4IR)}}$ achieves significantly superior performance (71.1%) compared to the cases using $\boldsymbol{I}_{\text{SR(pixel)}}$ (68.3%) and $\boldsymbol{I}_{\text{SR(pixel+percep)}}$ (69.7%). Furthermore, it is worth noting that the $S \rightarrow T$ setting with $\boldsymbol{I}_{\text{SR(SR4IR)}}$ achieves performance comparable to our final performance, which attains the accuracy of 71.4%, as presented in Table 3. These results demonstrate that the performance improvement in our SR4IR model is largely attributed to the SR results, in which the task-relevant high-frequency details are successfully restored.

Performance on task network trained on HR. To further investigate the similarity between HR and SR images in terms of task-relevant features, we evaluate the SR results $\boldsymbol{I}_{\text{SR(pixel)}}$, $\boldsymbol{I}_{\text{SR(pixel+percep)}}$, and $\boldsymbol{I}_{\text{SR(SR4IR)}}$ using a task network that has been exclusively trained on HR images (referred to as T_{HR}). Figure S1 shows the t-SNE [79] visualization results of each SR result within the feature space of T_{HR}. Our $\boldsymbol{I}_{\text{SR(SR4IR)}}$ exhibits the closest resemblance to HR images in the feature space of T_{HR} compared to $\boldsymbol{I}_{\text{SR(pixel)}}$ and $\boldsymbol{I}_{\text{SR(pixel+percep)}}$. In addition, it should be noted that $\boldsymbol{I}_{\text{SR(pixel)}}$ resembles poorly HR images even with the highest PSNR values, indicating that the PSNR is a less important factor in representing task-relevant features. Table S2 further shows the superior performance of our $\boldsymbol{I}_{\text{SR(SR4IR)}}$ when the SR results are evaluated by T_{HR}. These results demonstrate that $\boldsymbol{I}_{\text{SR(SR4IR)}}$ contains high-frequency contents closely related to tasks, validating the effectiveness of our SR4IR.

Training images	Top-1 Acc.$\uparrow$ (%)
$\boldsymbol{I}_{\text{SR(pixel)}}$	68.3
$\boldsymbol{I}_{\text{SR(pixel+percep)}}$	69.7
$\boldsymbol{I}_{\text{SR(SR4IR)}}$ **(Ours)**	**71.1**

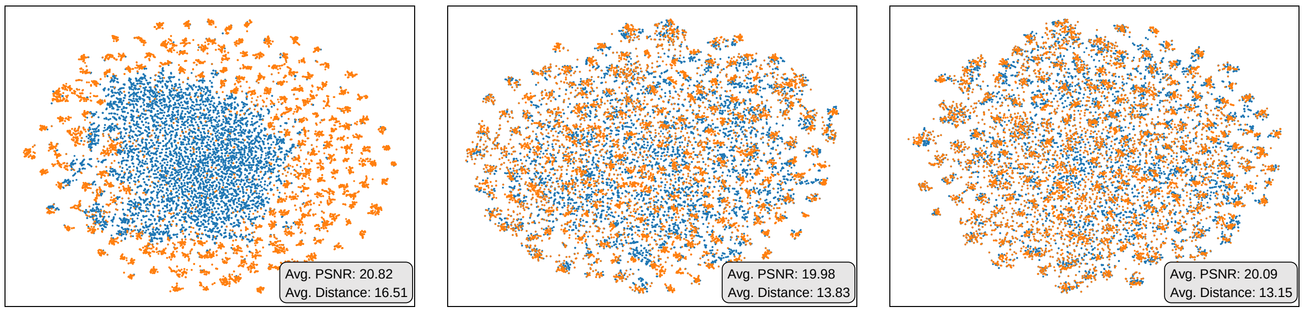
Table S1. **Performance on $S \rightarrow T$ setting.** We evaluate the image classification accuracy on the StandfordCars dataset with an SR scale of x8. We use the EDSR-baseline as an SR network.

Test images	Top-1 Acc.$\uparrow$ (%)
$\boldsymbol{I}_{\text{HR}}$ (Oracle)	86.4
$\boldsymbol{I}_{\text{SR(pixel)}}$	29.4
$\boldsymbol{I}_{\text{SR(pixel+percep)}}$	46.5
$\boldsymbol{I}_{\text{SR(SR4IR)}}$ **(Ours)**	**52.3**

Table S2. **Performance on the task network trained with HR.** We evaluate the image classification accuracy on the Standford-Cars dataset with an SR scale of x8.

S2. Analysis on CQMix

In Section 3.2, we propose the CQMix to prevent a task network from learning biased features that could undermine the effectiveness of TDP loss. To further analyze the effect of the CQMix, we visualize class-activation maps using Grad-CAM [78] obtained from a task network trained by our framework with/without the CQMix, which are pre-



(a) $\boldsymbol{I}_{\text{HR}} \leftrightarrow \boldsymbol{I}_{\text{SR(pixel)}}$ (b) $\boldsymbol{I}_{\text{HR}} \leftrightarrow \boldsymbol{I}_{\text{SR(pixel+percep)}}$ (c) $\boldsymbol{I}_{\text{HR}} \leftrightarrow \boldsymbol{I}_{\text{SR(SR4IR)}}$ **(Ours)**

Figure S1. **t-SNE visualization results.** We visualize the t-SNE plots based on the final feature of the feature extractor of a task network trained with HR images. The orange and blue points represent the HR and SR samples, respectively. We display the average PSNR in the RGB space and the average Euclidean distance in the feature space between the HR and SR pairs, in the bottom right corner of each figure. For the distance, the lower value indicates a higher resemblance between the SR results and the HR counterpart.

sented in Figure S2. When CQMix is not used (upper row), the task network tends to focus on specific image features, the *i.e*. shortcut feature [16], such as car wheels or headlights. In contrast, when CQMix is used (lower row), the task network tends to focus on wider regions and utilizes additional diverse image features, such as the car body and emblem. These visualization results demonstrate that our CQMix is effective in preventing the task network from learning shortcut features, leading to performance improvements when combined with the TDP loss in our framework.

S3. Efficiency of our SR4IR

In Table S3, we assess the efficiency of our SR4IR with the baseline method $\mathbb{I}_{\text{LR}} \rightarrow T$. As introduced in Section 4 of the main manuscript, the baseline method trains a task network using bilinear-upscaled LR images without the assistance of SR networks. Despite the superior task performance, our SR4IR incurs a certain degree of computational cost increase compared to $\mathbb{I}_{\text{LR}} \rightarrow T$ due to introducing the SR network. Nevertheless, we highlight that the performance gain from SR4IR does not simply come from the increased computational costs, demonstrated through a comparison with the baseline using a larger backbone model, $\mathbb{I}_{\text{LR}} \rightarrow T_{\text{Large}}$. In terms of training time, our SR4IR does require some additional training time compared to $\mathbb{I}_{\text{LR}} \rightarrow T$ due to the incorporation of the SR network. For the real-time capability, we observe that the throughput of our SR4IR experiences a modest decrease compared to $\mathbb{I}_{\text{LR}} \rightarrow T$ but still faster than $\mathbb{I}_{\text{LR}} \rightarrow T_{\text{Large}}$, indicating practical applicability of SR4IR.

S4. Effectiveness of the task-driven training

In Table S4, we show the effectiveness of task-driven training by evaluating the performance across three image recognition tasks with SR networks specially tailored for each task. Compared to the case of 'task-driven' SR, which is presented in the **diagonal** entries, using the SR network trained for different tasks largely degrades the performance.

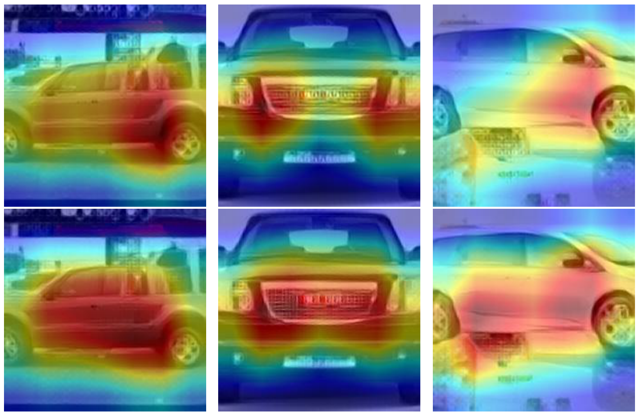


Figure S2. **Grad-CAM visualization results according to the CQMix.** The upper and lower rows represent the Grad-CAM [78] results from the task network trained without and with the proposed CQMix. We use EDSR-baseline [35] with a scale factor of x8 and StandfordCars [30] dataset.

	mIoU$_\uparrow$	# Params	GFLOPs	Memory Cache	Training time (GPU hours)	Throughput (img/s)
$\mathbb{I}_{\text{LR}} \rightarrow T$	49.3	11.0M	17.5	456MB	0.64	33.8
$\mathbb{I}_{\text{LR}} \rightarrow T_{\text{Large}}$	50.2	42.0M	305.5	1621MB	4.25	15.2
SR4IR **(Ours)**	**55.0**	12.7M	49.5	695MB	2.41	25.3

Table S3. **Efficiency of our SR4IR on x8 segmentation task.** GFLOPs are calculated based on 480x480 resolution images. Memory cache and throughput are evaluated using the PASCAL VOC [14] validation set and an NVIDIA RTX A6000 GPU. We use [35] as an SR network. The T and T_{Large} are DeepLabV3 [7] with MobileNetV3 [22] and ResNet50 [20] backbone.

Used SR network	Segmentation (mIoU$_\uparrow$)	Detection (mAP$_\uparrow$)	Classification (Top1 Acc.$_\uparrow$)
$S_{\text{Segmentation}}$	**55.0**	22.4	61.0
$S_{\text{Detection}}$	50.3	**25.5**	60.2
$S_{\text{Classification}}$	48.4	20.0	**71.4**

Table S4. **SR4IR performance according to the SR network.** S_{Task} represents the SR network specifically tailored for each *Task* through our SR4IR. We use EDSR-baseline [35], PASCAL VOC [14] and StanfordCars [30] with x8 downsampling.

These results demonstrate the importance of employing a task-driven SR network in improving task performance.

S5. Diverse degradation scenarios for SR4IR

Table S5 shows the performance of our SR4IR on diverse versions of degraded LR images, which have undergone bicubic downsampling with an additional Gaussian blur filter. Our SR4IR consistently improves the task performance in all degradation cases, indicating its general applicability. Moreover, the performance gain from SR4IR becomes more pronounced as the degradation severity increases. This observation is consistent with the discussions in Section 4.1 of our main manuscript, where we noted that the performance improvement of SR4IR is more significant as the SR scale changes from x4 to x8.

S6. Evaluation on the SR benchmark datasets

Table S6 shows the performance of our task-driven SR network on SR benchmark datasets. Compared to conventional SR methods trained on pixel-wise loss, the SR network trained by our SR4IR framework exhibits lower PSNR/SSIM values on SR benchmark datasets. However, as discussed in Section S1 and 4.1 of our main manuscript, we highlight that restoring task-relevant high-frequency details is crucial for the task performance rather than such distortion-oriented metrics.

S7. Training details

The number of training epochs is set to 100 for semantic segmentation, 30 for object detection, and 200 for image classification. The task loss is set to cross-entropy loss in semantic segmentation and image classification, and a combination of classification, roi box regression, objectness, and rpn box regression loss in object detection, following the official PyTorch implementation github [77]. In the case of applying the TDP loss, we exclude the TDP loss during the initial one-tenth of the training process to allow the task network to learn meaningful task-relevant features to some extent before introducing the TDP loss.

S8. Implementation details for previous works

TDSR [19]. In our re-implementation, we use TDSR-0.01, which does not initially employ the task loss in the first one-third of the training process and then introduces the task loss with a ratio of 0.01 in the remaining training iteration. Unlike the original paper, which used confidence loss and localization loss as task loss, we adopt the cross-entropy loss as task loss, as our ablation studies cover semantic segmentation and image classification.

	Bicubic		Gaussian blur + Bicubic					
			std = 5.0		std = 10.0		std = 15.0	
	mIoU$_\uparrow$	LPIPS$_\downarrow$	mIoU$_\uparrow$	LPIPS$_\downarrow$	mIoU$_\uparrow$	LPIPS$_\downarrow$	mIoU$_\uparrow$	LPIPS$_\downarrow$
$\mathbb{I}_{\text{LR}} \rightarrow T$	49.3	0.476	47.9	0.543	38.3	0.631	29.5	0.650
SR4IR (**Ours**)	**55.0**	**0.380**	**56.4**	**0.367**	**49.8**	**0.485**	**41.4**	**0.543**

Table S5. **Diverse degradation scenarios on x8 segmentation.** We use the EDSR-baseline [35] as an SR network. We evaluate the performance on the PASCAL VOC [14] dataset.

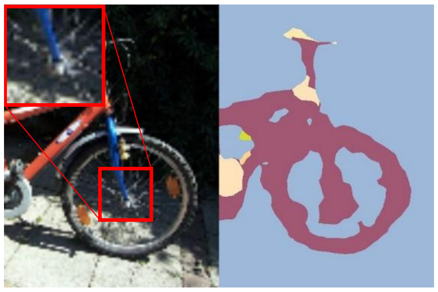
	Set5	Set14	B100	Urban100
EDSR-baseline	32.10 / 0.863	28.58 / 0.743	27.56 / 0.711	26.04 / 0.763
EDSR-baseline-SR4IR$_{\text{Segmentation}}$	29.32 / 0.800	26.76 / 0.688	26.39 / 0.660	23.77 / 0.667

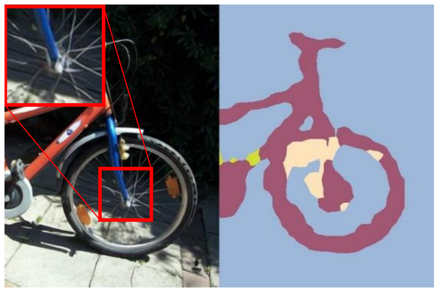
Table S6. **PSNR / SSIM on x4 SR benchmark datasets.** For the task-driven SR network, we use EDSR-baseline [35] trained by our SR4IR on the segmentation task.

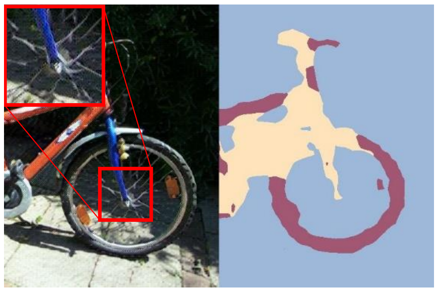
SOD-MTGAN [3]. Following the original paper, we introduce an additional fully connected layer in the task-specific head module $H_{\boldsymbol{\theta}_{\text{head}}}$, and utilize the output from that branch as the discriminator output. Similarly to the TDSR, we replace the detection losses used in the original paper with the cross-entropy loss to cover semantic segmentation and image classification. In the original paper, the authors set the loss weights for adversarial loss, task loss, and pixel-wise reconstruction loss as 0.001, 0.01, and 1.0, respectively. However, we found that this setting resulted in significantly lower performance in our experiments. Hence, we adjusted the task loss weight from 0.01 to 1.0.

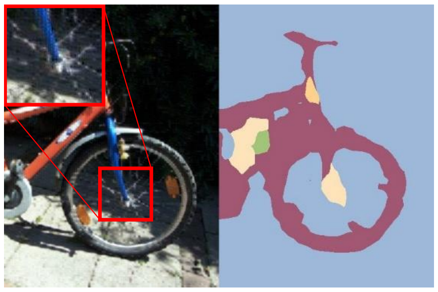
S9. More visualization comparison results

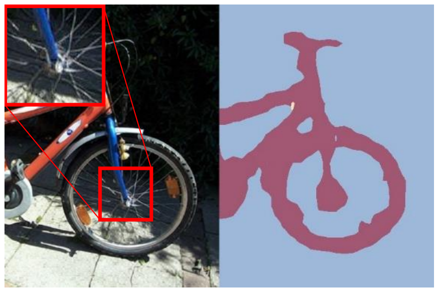
We present additional qualitative results for semantic segmentation (Figure S3), object detection (Figure S4), and image classification (Figure S5, S6). We compare our SR4IR with all baselines, $\mathbb{I}_{\text{LR}} \rightarrow T$, $S \rightarrow T$, $T \rightarrow S$, and $S + T$, as introduced in Section 4 of our main manuscript. These results demonstrate that our SR4IR framework achieves the most accurate predictions across all tasks while producing visually pleasing results.

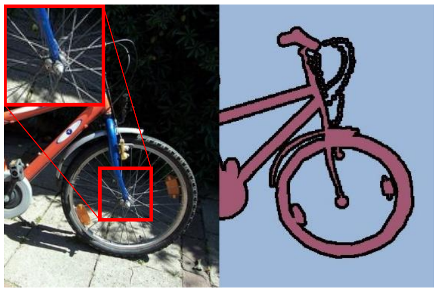

(a) $\mathbb{I}_{\text{LR}} \rightarrow T$

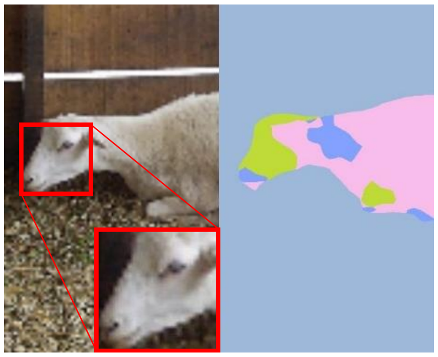

(b) $S \rightarrow T$

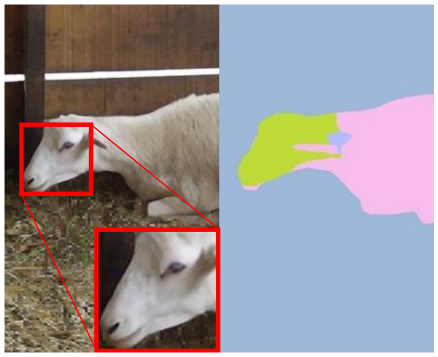

(c) $T \rightarrow S$

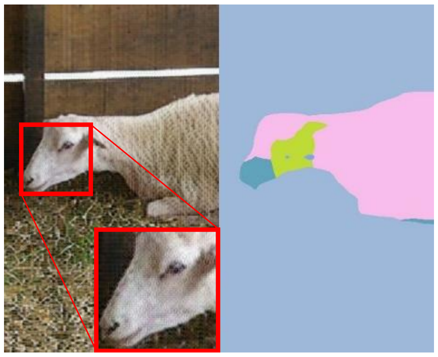

(d) $S + T$

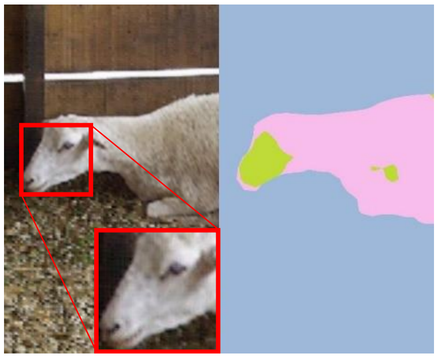

(e) **SR4IR (Ours)**

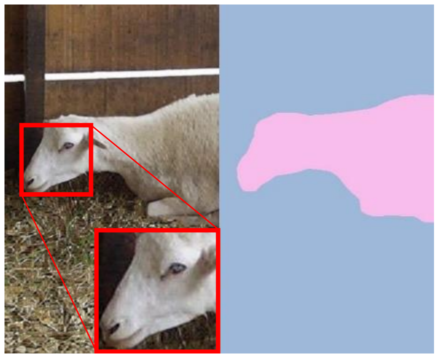

(f) Ground-truth


(a) $\mathbb{I}_{\text{LR}} \rightarrow T$


(b) $S \rightarrow T$


(c) $T \rightarrow S$


(d) $S + T$


(e) **SR4IR (Ours)**

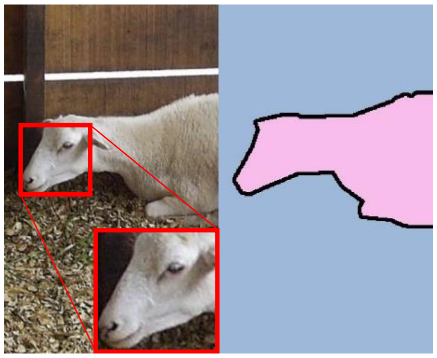
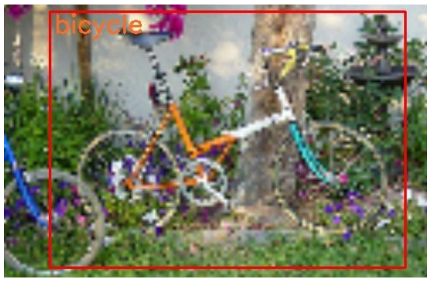
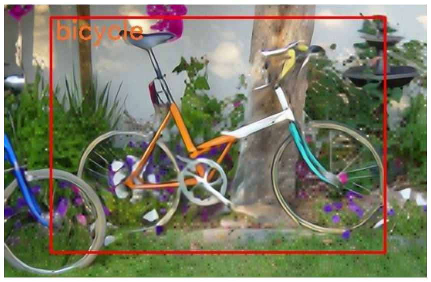

(f) Ground-truth

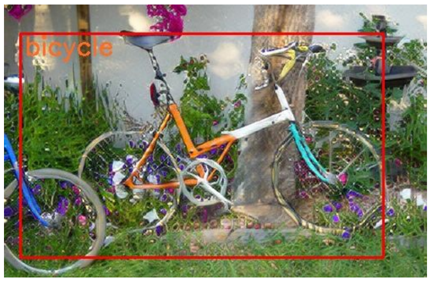
Figure S3. **Visualization of images and semantic segmentation results on PASCAL VOC dataset [14].** We present the restored images and the corresponding predicted segmentation maps. For (b), (c), (d), and (e), we use the SwinIR [33] model with an SR scale factor of x4.



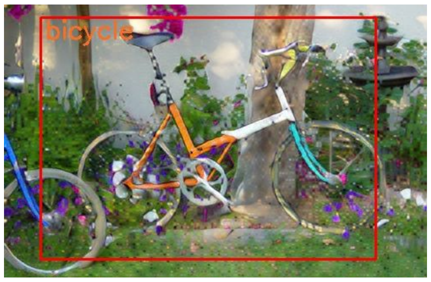
(a) $\mathbb{I}_{\text{LR}} \rightarrow T$



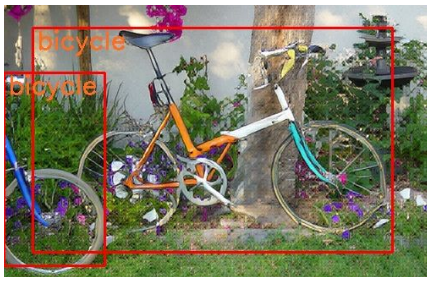
(b) $S \rightarrow T$



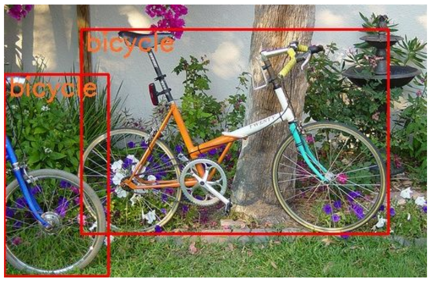
(c) $T \rightarrow S$



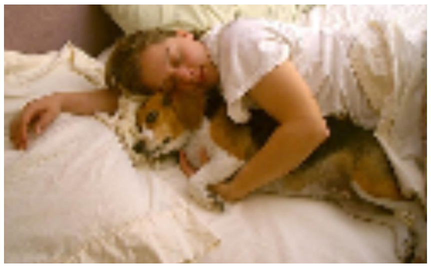
(d) $S + T$



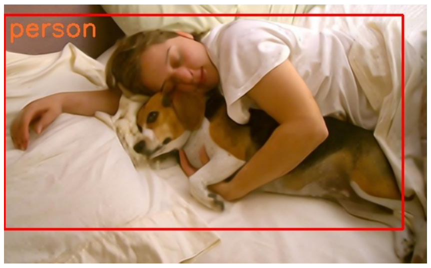
(e) **SR4IR (Ours)**



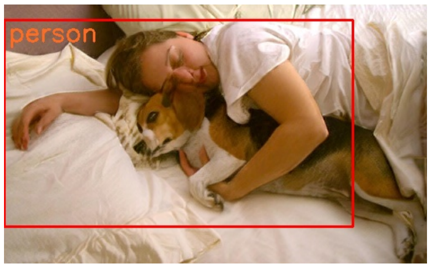
(f) Ground-truth



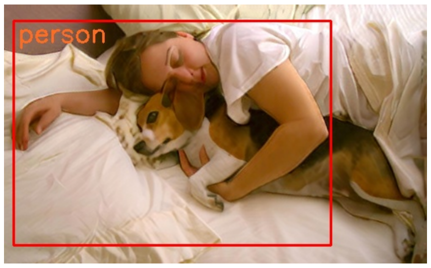
(a) $\mathbb{I}_{\text{LR}} \rightarrow T$



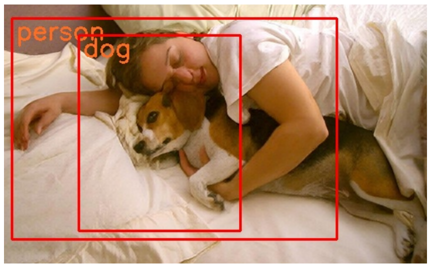
(b) $S \rightarrow T$



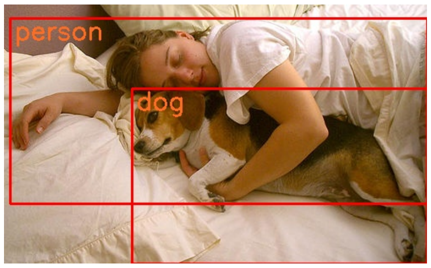
(c) $T \rightarrow S$



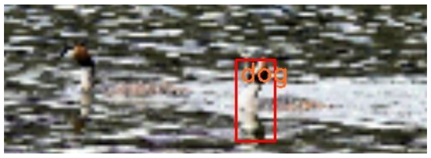
(d) $S + T$



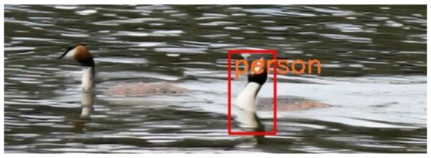
(e) **SR4IR (Ours)**



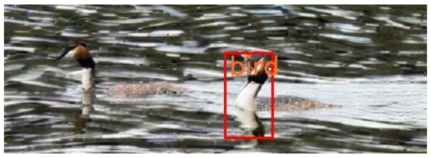
(f) Ground-truth



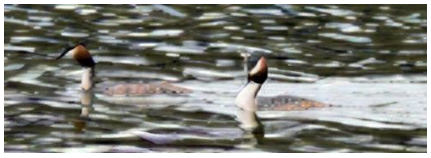
(a) $\mathbb{I}_{\text{LR}} \rightarrow T$



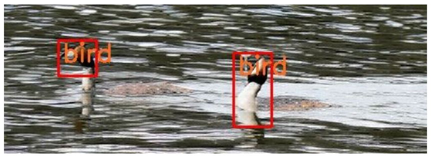
(b) $S \rightarrow T$



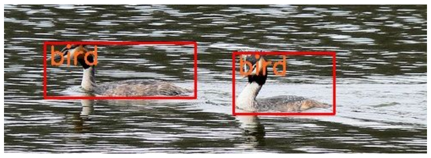
(c) $T \rightarrow S$



(d) $S + T$



(e) **SR4IR (Ours)**



(f) Ground-truth

Figure S4. **Visualization of object detection results on PASCAL VOC dataset.** The red box with orange annotation means the predicted object bounding box with the corresponding prediction. For (b), (c), (d), and (e), we use the SwinIR model with an SR scale factor of x4.

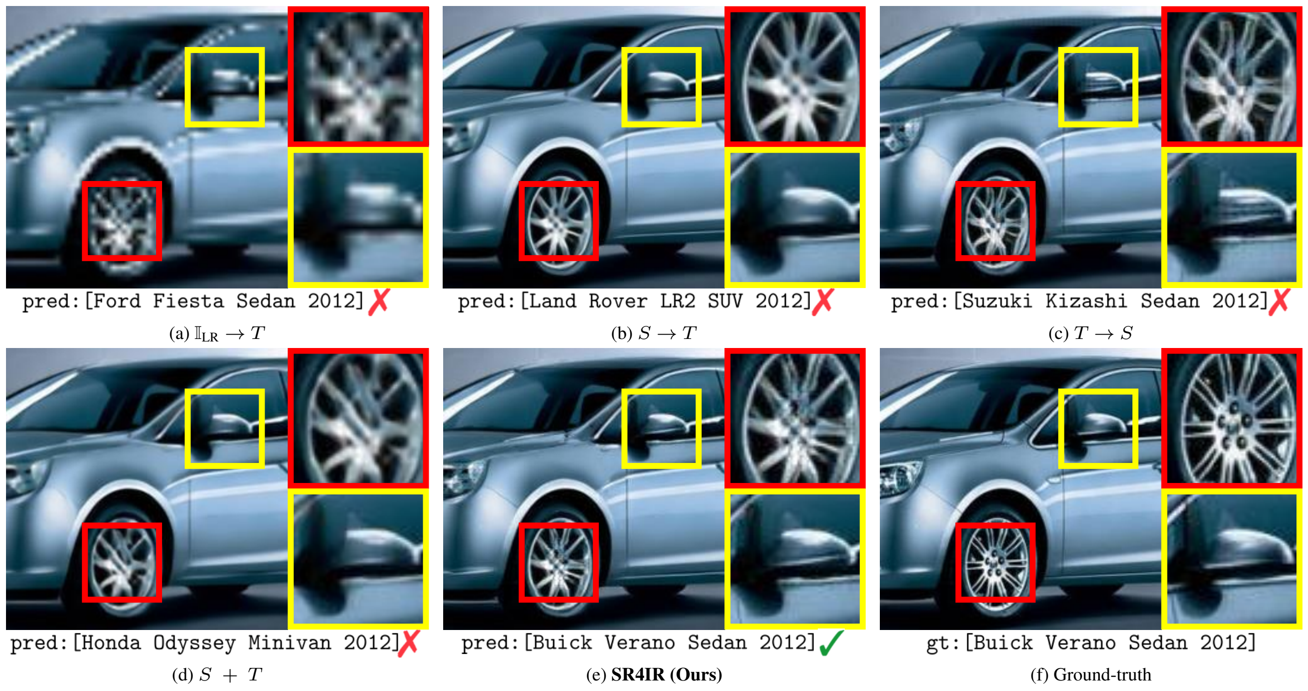

pred: [Ford Fiesta Sedan 2012] ✗
(a) $\mathbb{I}_{\text{LR}} \rightarrow T$
pred: [Land Rover LR2 SUV 2012] ✗
(b) $S \rightarrow T$
pred: [Suzuki Kizashi Sedan 2012] ✗
(c) $T \rightarrow S$
pred: [Honda Odyssey Minivan 2012] ✗
(d) $S + T$
pred: [Buick Verano Sedan 2012] ✓
(e) **SR4IR (Ours)**
gt: [Buick Verano Sedan 2012]
(f) Ground-truth

Figure S5. **Visualization of images and image classification results on StanfordCars [30] dataset.** We present the restored images and the corresponding caption. The caption below the image represents the predicted image classification results, and a checkmark indicates if the prediction is correct. For (b), (c), (d), and (e), we use the SwinIR model with an SR scale factor of x4.

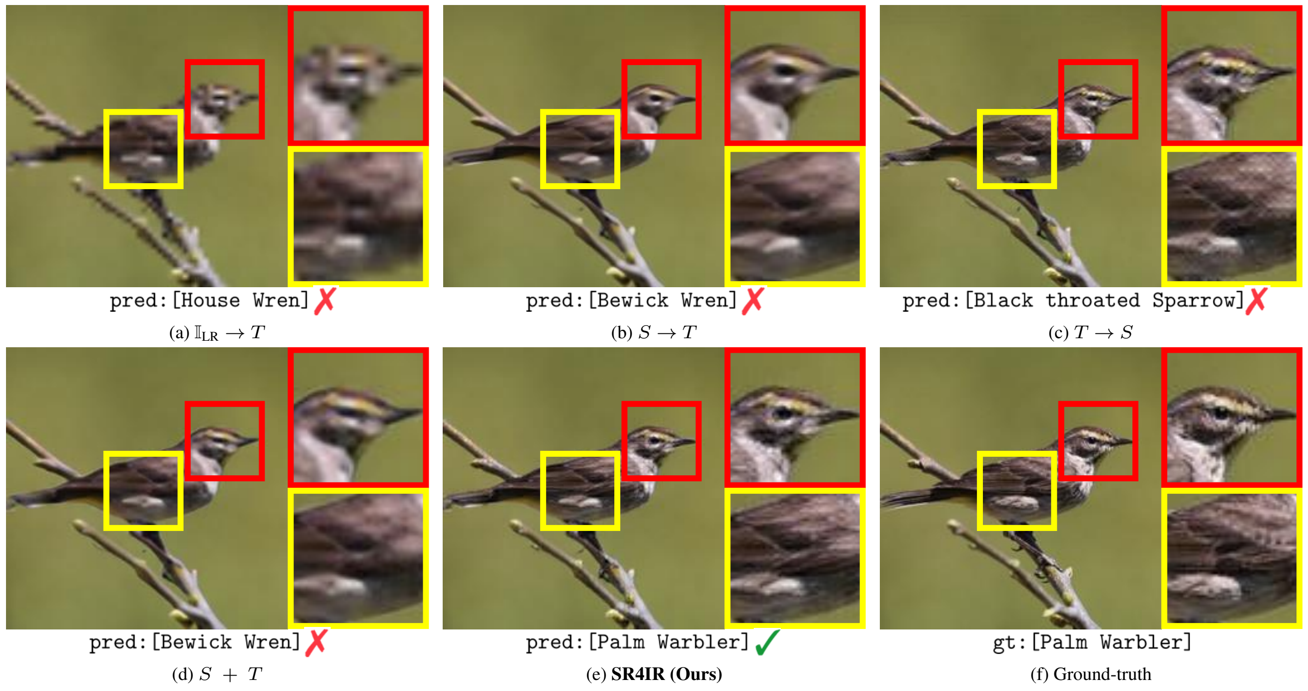


Figure S6. **Visualization of images and image classification results on CUB-200-2011 dataset [57].** We present the restored images and the corresponding caption. The caption below the image represents the predicted image classification results, and a checkmark indicates if the prediction is correct. For (b), (c), (d), and (e), we use the SwinIR model with an SR scale factor of x4.

References

[77] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. https://github.com/pytorch/vision, 2016.

[78] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

[79] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.