BOOTPLACE: Bootstrapped Object Placement with Detection Transformers

Supplementary Material

6. Network and parameter details

For the image encoder, We use ImageNet-pretrained ResNet-50 with frozen batchnorm layers and discard the last classification layer as the CNN backbone. The Transformer encoder contains 6 blocks and the Transformer decoder contains 6 block. Each attention layer has 8 attention heads. Additive dropout of 0.1 is applied after every multi-head attention and FFN before layer normalization. The weights are randomly initialized with Xavier initialization. The intermediate size of the feedforward layers in the transformer blocks is set 2048 and the size of the embeddings d in the transformer is set 256. The number of object (region) queries N is set to 100 and the maximum number of objects M is set to 120. For the bounding box encoder, we utilize 2-layer MLP which transforms the bounding box embedding into 256 dimensiton and multiple it with the embedding obtained from the Transformer decoder, and then use 3-layer MLP to map it to 4-dim embeddings.

7. Computational cost

To compare the computation costs of different methods, we show the number of Params, FLOPs and inference time of PlaceNet, SAC-GAN, TopNet and BOOTPLACE in Table 4. Our method requires slightly more parameters than the other methods. The theoretical computation cost (FLOPs) of our method is 7x larger, but is similar to DETR-based detection models. We also tested the inference time on the Cityscapes validation set using an Nvidia GeForce GTX TITAN X with a batch size of 1. The inference time per sample of our method is less than 1 second, significantly faster than TopNet (2.7 seconds) and similar to PlaceNet. Therefore, despite the higher FLOPs, the computational complexity of our method is manageable and affordable, with efficient real-world inference times.

8. Loss function analysis

In Figure 13, we compare the impact of various loss functions. The regression loss, with sparse annotation, poses challenges in training the model effectively. Gaussian assigned loss overlooks the impact of scaling and fails to accommodate multi-peak distributions for possible placements. Sparse contrastive loss supports the fluctuation of neighboring placements but lacks accurate constraints for complex scenes with location-varying placements. Our proposed loss function is derived from bounding box loss using Generalized IOU loss, offering more precise constraints on box scaling.

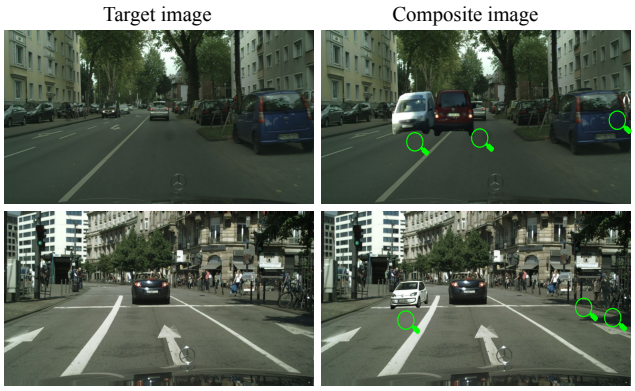

Figure 12. **Multi-object placement.** Two cars and one person (top), one car and two people (below) are composed into the scene.

9. Multi-object placement

Compared to single-object placement, multi-object placement is significantly more challenging as it necessitates an understanding of the prior state of composed objects, scene objects and background image. Though our network makes parallel bounding box predictions, it has learned a robust correspondence between objects and their associated regions. In Figure 12, we illustrate the potential for composing three objects into street scenes, showing the capability of our network to learn object orientation and the distribution of various object categories.

10. Dataset construction

In Figure 14, we illustrate the data construction process for the Cityscapes [11] dataset, which is applicable to other datasets. We start with the source image ① and employ a pretrained MaskFormer [9] model for panoptic semantic segmentation ②, jointly performing semantic and instance segmentation. Scene primitives are manually categorized into object classes, including `car`, `person`, `rider`, `train`, `bus`, `bicycle`, `truck`, `motorcycle`, resulting in binary object masks ③. To obtain object-free backgrounds, we dilate the binary object masks to address boundary inaccuracies and obtain dilated object masks ④. The next step involves using pretrained LaMa [41] inpainting model to remove objects, yielding inpainted images ⑦. As many segmentation models tend to classify shadows as background, we manually remove these shadows using an online PhotoKit tool, refining the background image to obtain corrected inpainted images ⑧. Simultaneously, we create an object pool ⑤ consisting of both intact objects

	PlaceNet (ECCV'20) [51]	SAC-GAN (IEEE TVCG'22) [53]	TopNet (CVPR'23) [55]	BOOTPLACE (ours)
# Params (M)	35.9	35.9	25.0	41.4
# FLOPs (G)	4.40	6.96	6.79	44.4
Inference time (sec)	0.68	0.1	2.7	0.27

Table 4. Comparison of computational cost and model parameters tested on Cityscapes dataset.

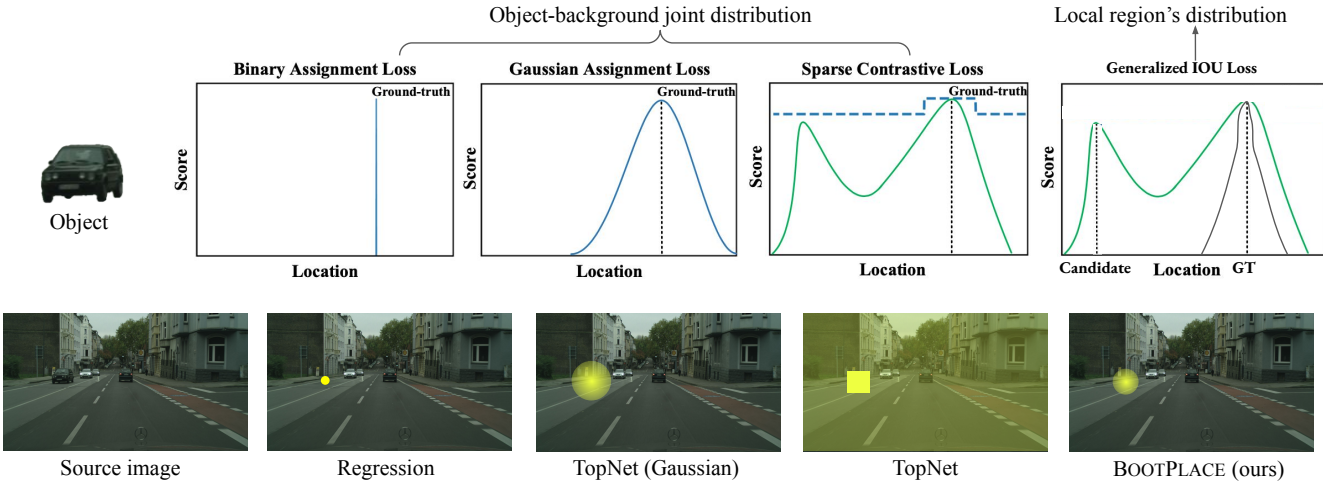

Figure 13. **Different losses** exemplified in 1D space. The yellow marks, depicted with varying intensities, represent the constraint intensity.

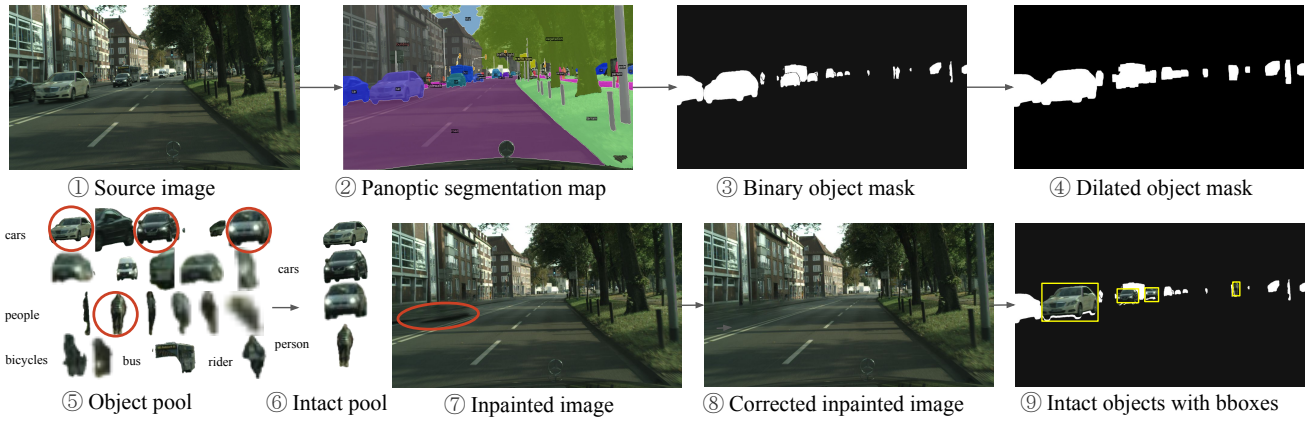

Figure 14. **Data preparation** of background inpainted images and corresponding scene objects processed from source images.

Figure 15. **Data preparation** for boundary harmonization. The boundary dilated regions are automatically segmented by dilation of object silhouette.

and those that are partially occluded, each with varying resolutions. After manual curation, we retain only the intact objects, resulting in an intact object pool ⑥ with their bounding box coordinates. After data cleaning, we construct a multi-object dataset including 2,953 training images with 22,270 objects and their corresponding ground-truth labels, as well as 372 testing images with 2,713 objects.

Figure 16. **Harmonization results** of composite Cityscapes samples. Zoom in to see visual details.

	TopNet	BOOTPLACE (Ours)
Encoder	ViT-small	CNN + MLP + Transformer encoders
Decoder	2D upsampling	Transformer decoder + MLPs
Output	3D heatmap	Bbox + class predictions
Losses	Sparse contrastive + range	Bbox regression + class prediction + association
Training	AdamW optimizer + lr=1e-5	AdamW optimizer + le= 4e-4

Table 5. TopNet *vs* BOOTPLACE (ours).

	Copy-paste		ObjectStitch		
	FID (↓)	LPIPS (↓)	FID (↓)	LPIPS (↓)	Scale
PlaceNet [51]	52.02	0.088	77.67	0.217	0.204
SAC-GAN [53]	42.89	0.066	63.44	0.194	0.156
TopNet [55]	38.21	0.043	49.74	0.189	0.079
BOOTPLACE (ours)	58.74	0.105	79.50	0.246	0.310

Table 6. Quantitative comparison using FID and LPIPS.

Figure 17. Comparison of w/o (row 1 and 3) *vs* w/ (row 2 and 4) object compositing for two examples on object replacement task.

11. Method comparison with TopNet

We provide comparisons between TopNet and BOOTPLACE in architecture and training strategy in Table 5.

12. Image blending

To address boundary artifacts arising from copy-paste object composition, we employ two distinct methods: (1) use a diffusion model initially designed for image inpainting to harmonize boundaries. We finetune the Stable Diffusion Inpainting model without prompt conditioning [2] on Cityscapes objects. This involves extracting object-centric patches and dilating their masks to create boundary masks, as depicted in Figure 15. Once a sufficient number of such patch-mask pairs are collected, we finetune the model to harmonize boundary region. Figure 16 shows the visual performance of image harmonization on Cityscapes samples. (2) combine placement learning with identity-preserving compositing methods such as ObjectStitch [40] for visual refinement, as shown in Figure 17. This process significantly reduces boundary artifacts while naturally generating shadows, resulting in an enhanced level of realism compared to compositions without harmonization.

[2] https://github.com/lorenzo-stacchio/Stable-Diffusion-Inpaint/tree/1b44f2f9e4f233f68d48c56b68b9c111c1538d4d

13. Evaluation on FID and LPIPS

In Table 6, we evaluate plausibility of composite images using FID and LPIPS on the Cityscapes dataset. We observe that both metrics are strongly correlated with bbox scale, where smaller bounding boxes result in fewer modifications to the image and correspondingly lower FID and FPIPS values. Therefore, they are not suitable metrics for evaluating placement quality, which are excluded from evaluation.

14. More qualitative results

We show qualitative results of object placement on Cityscapes dataset in Figures 18 and 19, object reposition on Cityscapes dataset in Figures 20 and 21, and object reposition on OPA dataset in Figure 22.

15. More decoder attention visualization

We provide additional visualization results showing the distribution of the detection decoder in Figure 23.

Figure 18. **Qualitative results of single object placement** on Cityscapes dataset. Objects are randomly chosen from its testing set.

Figure 19. **Qualitative results of object placement** on Cityscapes dataset. Objects are randomly chosen from its testing set.

Figure 20. **Qualitative results of object reposition** on Cityscapes dataset.

Figure 21. **Qualitative results of object reposition** on Cityscapes dataset.

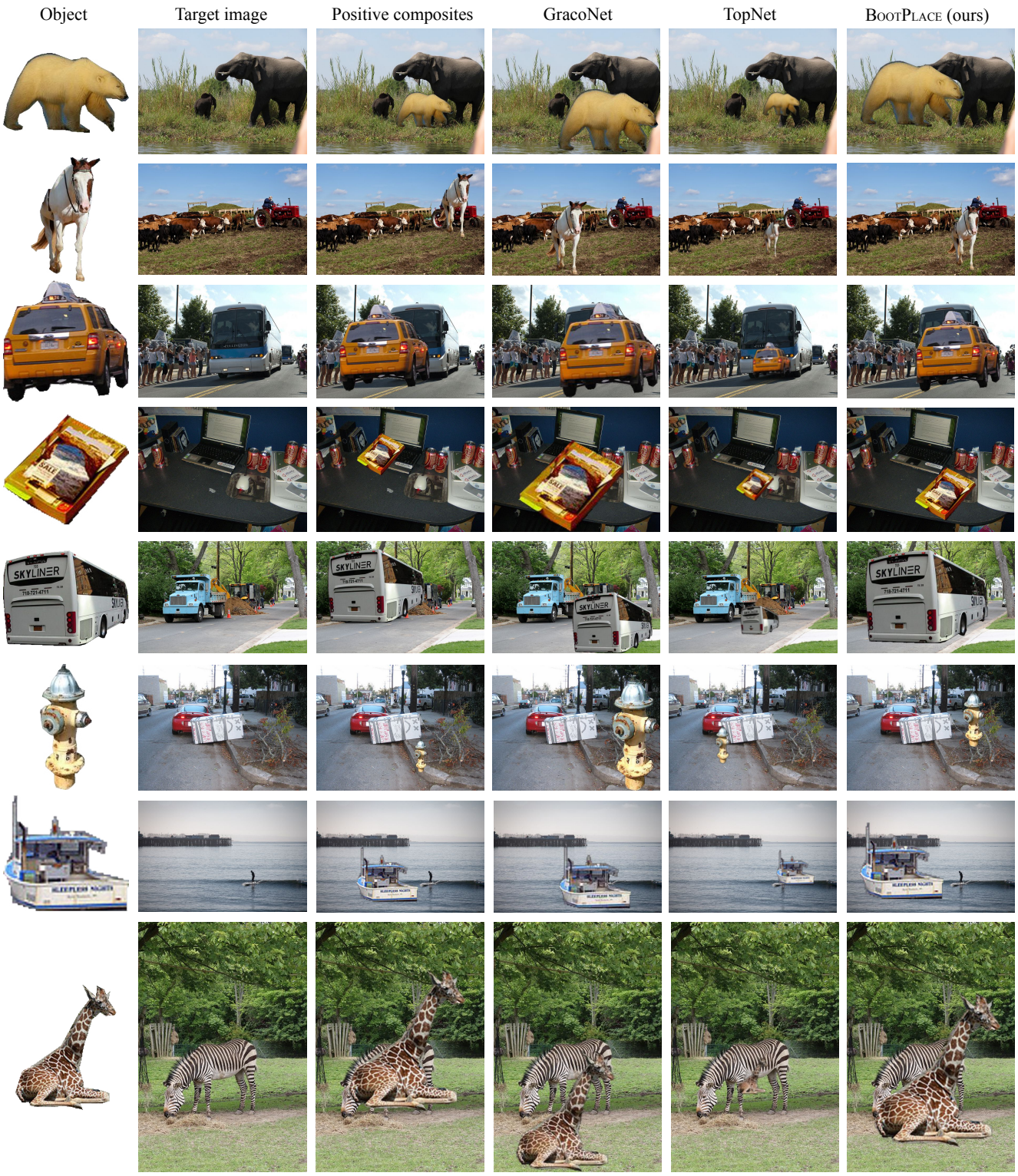

Figure 22. **Qualitative results of object reposition** on OPA dataset.

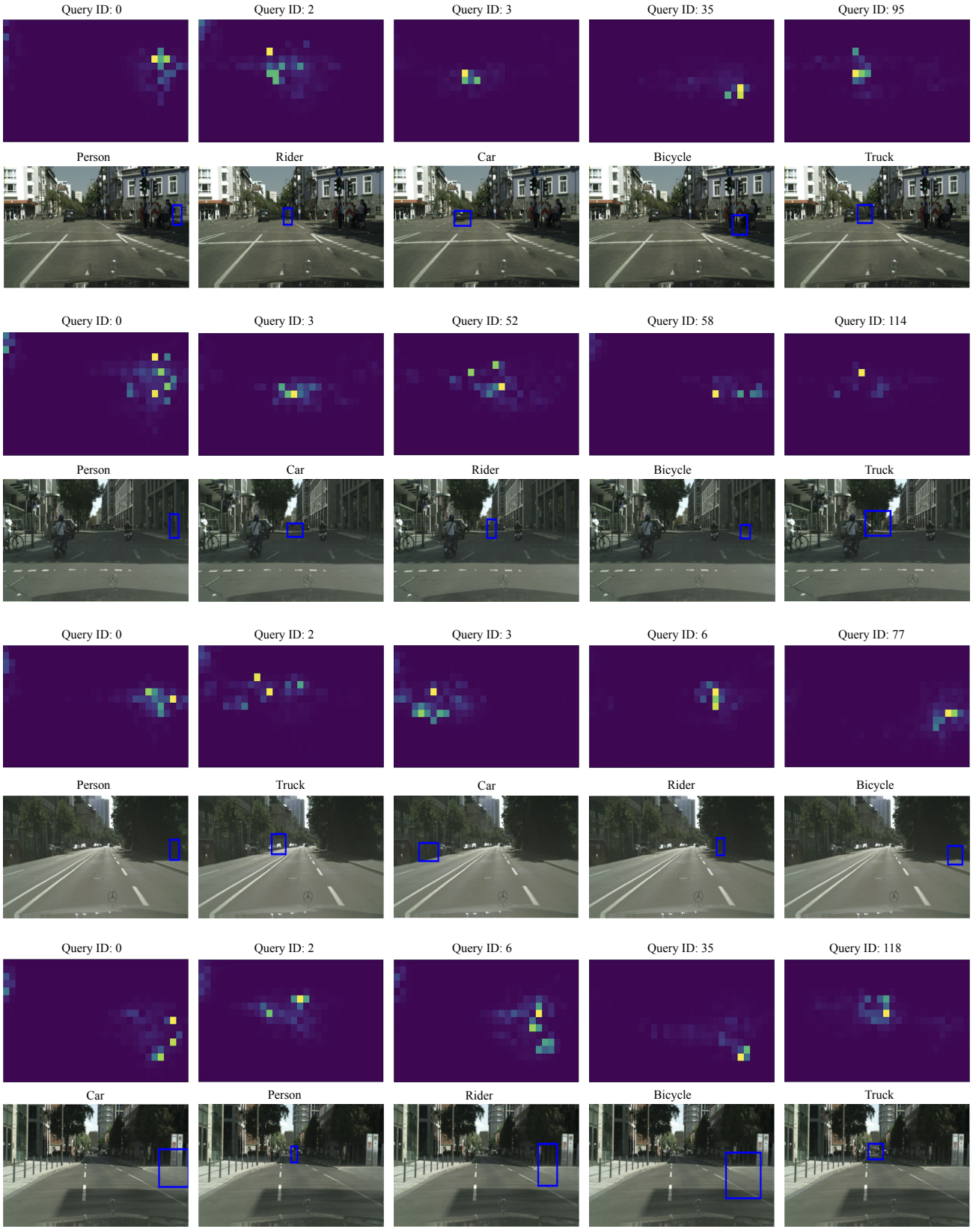

Figure 23. **Decoder attention visualization** of Cityscapes samples.