Learning to generate training datasets for robust semantic segmentation Supplementary material

Marwane Hariat,[1] **Olivier Laurent**,[1, 2] **Rémi Kazmierczak**,[1] **Shihao Zhang**,[3]
Andrei Bursuc,[4] **Angela Yao**[3] & **Gianni Franchi**[1]
U2IS, ENSTA Paris, Institut Polytechnique de Paris,[1] SATIE, Université Paris-Saclay,[2]
National University of Singapore,[3] valeo.ai[4]

Contents

A. From to pix2pixHD to Robusta

A.1. Background on label-to-image translation methods

Conditional Generative Adversarial Networks are are one of the most popular tools for semantic image synthesis in the literature [26]. These models consist of generators with conditioning designed to control the models and generate specific contents. The standard loss function for cGAN training is defined as follows:

$$\mathcal{L}_{cGAN}(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D) = \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x} \mid \mathbf{y})] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z} \mid \mathbf{y})))], \quad \text{(A1)}$$

where $\mathbf{y}$ is the conditional information and $\boldsymbol{\theta}_G$ and $\boldsymbol{\theta}_D$ represent the weights of the generator and the discriminator, respectively. In label-to-image translation, $\mathbf{y}$ is the input label, $\mathbf{x}$ is the target RGB image, and $\mathbf{z}$ is a sampled latent variable.

Pix2pix [18] is a type of cGAN specialized in image-to-image translation tasks. It uses a U-Net [33] as backbone and a PatchGAN [18] as discriminator. In addition to the adversarial loss defined in Eq.(A1), an L1 loss, defined in Eq.(A2), is added to the cost function of cGANs to reduce blur:

$$\mathcal{L}_{L1}(\boldsymbol{\theta}_G) = \mathbb{E}_{(\mathbf{x},\mathbf{y},\mathbf{z})}[\|\mathbf{x} - G(\mathbf{z} \mid \mathbf{y})\|_1]. \quad \text{(A2)}$$

The total cost function of pix2pix is a linear combination of $\mathcal{L}_{L1}$ and $\mathcal{L}_{cGAN}$, with λ as a regularization hyperparameter that balances the cGAN and reconstruction losses:

$$\mathcal{L}_{\text{pix2pix}}(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D) = \mathcal{L}_{cGAN}(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D) + \lambda \mathcal{L}_{L1}(\boldsymbol{\theta}_G). \quad \text{(A3)}$$

Pix2pixHD [44] improves the quality of the generated images thanks to enhanced multi-scale generators and discriminators. Pix2pixHD splits the generator and discriminator into two subGANs: G_1 and D_1, and G_2 and D_2. The loss function consists of the pix2pix loss function completed by a *feature-matching loss* and a *perceptual loss*. The feature-matching loss $\mathcal{L}_{\text{FM}}(\boldsymbol{\theta}_G, \boldsymbol{\theta} D_k)$ quantifies the distances between the feature maps of the real image $\mathbf{x}$ and the predicted image. The feature maps are extracted from discriminator layers denoted by i. The loss is defined as follows:

$$\mathcal{L}_{\text{FM}}\left(\boldsymbol{\theta}_G, \boldsymbol{\theta}_{D_k}\right) = \mathbb{E}_{(\mathbf{x},\mathbf{y},\mathbf{z})} \sum_{i=1}^{I} \frac{1}{N_i} \left\| D_k^{(i)}(\mathbf{x}) - D_k^{(i)}(G(\mathbf{z} \mid \mathbf{y})) \right\|_1, \quad \text{(A4)}$$

where $D_k^{(i)}$ stands for the ith-layer feature extractor of discriminator D_k, I is the total number of layers, and N_i represents the number of elements in each layer. The perceptual loss is designed to measure the similarity between the high-level features of the generated and real images. The feature maps are extracted from the i-th layer of a VGG network [36], F_{VGG}, pretrained on ImageNet [8]. The perceptual loss is defined as:

$$\mathcal{L}_{\text{VGG}}(\boldsymbol{\theta}_G) = \mathbb{E}_{(\mathbf{x},\mathbf{y},\mathbf{z})} \sum_{i=1}^{N} \frac{1}{M_i} \left\| F_{\text{VGG}}^{(i)}(\mathbf{x}) - F_{\text{VGG}}^{(i)}(G(\mathbf{z} \mid \mathbf{y})) \right\|_1, \quad \text{(A5)}$$

where M_i is the number of elements of the VGG network, and I is the number of layers on which we extract the feature maps.

SPADE [28] introduces a novel conditional normalization layer called SPatially-Adaptive (DE)normalization. In the context of image synthesis, Park et al. [28] highlight that deeper layers of DNNs can easily lose semantic information due to the relative sparsity of semantic input. Since semantic inputs exhibit low local variance, common normalization methods like Batch Normalization [17] can inadvertently remove semantic information. To avoid this issue, Instance Normalization [42] can be used. Notably, these normalization layers involve two steps: normalization and denormalization. In SPADE, Park et al. [28] propose learning denormalization factors at the pixel level, which are dependent on the label image $\mathbf{y}$. Therefore, the output at site (n, c, h, w) of the i-th SPADE layer is defined as:

$$\gamma_{c,h,w}^{(i)}(\mathbf{y}) \frac{a_{n,c,h,w}^{(i)} - \mu_c^{(i)}}{\sigma_c^{(i)}} + \beta_{c,h,w}^{(i)}(\mathbf{y}), \quad \text{(A6)}$$

where $\mu_c^{(i)}$ and $\sigma_c^{(i)}$ represent the mean and variance at the instance or batch level. The activation a is normalized by subtracting the mean and dividing by the standard deviation, followed by a modulation using the learned parameters γ and β. Finally, n, c, h, and w represent the number of the image in the batch, the channel, height, and width of the image, respectively.

A.2. Robusta's loss functions

To train our model, we divide the training procedure into two parts. We begin by training the first generator G_{coarse} using the original label maps to produce synthesized images. Next, we use these synthesized images to feed and train the second generator G_{fine}.

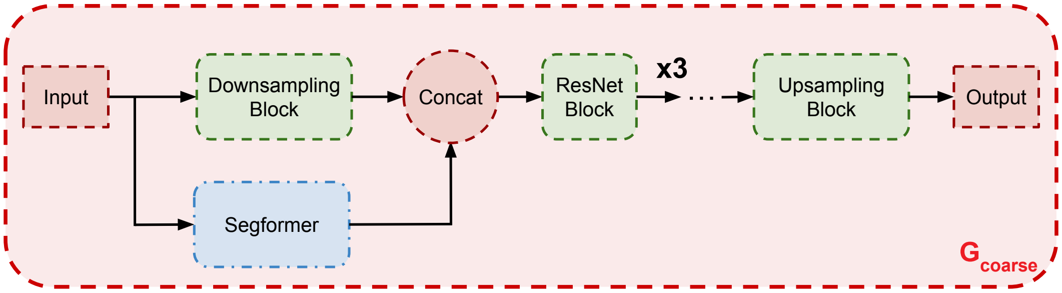


Figure A1. **Illustration of the first generator of Robusta, $G_{\textbf{coarse}}$.** The green blocks are detailed in Table A1.

$G_{\textbf{coarse}}$ - loss. We use the same loss functions as in Wang et al. [44], namely $\mathcal{L}_{\text{pix2pix}}$, $\mathcal{L}_{\text{FM}}$ and $\mathcal{L}_{\text{VGG}}$:

$$\mathcal{L}_{\text{coarse}} = \mathcal{L}_{\text{pix2pix}} + \lambda_{\text{FM}}\mathcal{L}_{\text{FM}} + \lambda_{\text{VGG}}\mathcal{L}_{\text{VGG}} \tag{A7}$$

$G_{\textbf{fine}}$ - loss. After training G_{fine}, we generate low-resolution images I_{LR} to be used in the training of G_{fine}. We use the Least Squares Generative Adversarial Network approach, as suggested in SRGAN [22], which yields the following loss function:

$$\begin{aligned}\mathcal{L}^{\text{MSE}}_{\text{cGAN}}(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D) = {} & \mathbb{E}_{I_{HR}\sim p_{I_{HR}}}[(1 - D_{\boldsymbol{\theta}_D}(I_{HR}))^2] \\ & + \mathbb{E}_{I_{LR}\sim p_{I_{LR}}}[D_{\boldsymbol{\theta}_D}(G_{\boldsymbol{\theta}_G}(I_{LR}))^2].\end{aligned} \tag{A8}$$

The final loss for training G_{fine} reads:

$$\mathcal{L}_{\text{fine}} = \mathcal{L}^{\text{MSE}}_{\text{cGAN}} + \lambda_{FM}\mathcal{L}_{FM} + \lambda_{VGG}\mathcal{L}_{VGG} \tag{A9}$$

The hyper-parameters used to train both G_{coarse} and G_{fine} are reported in Appendix B.1.

A.3. Details on Robusta's architecture

Figure A1 and Table A1 detail the architecture and technical specifications on the first generator of Robusta, G_{coarse}.

Why splitting $G_{\textbf{coarse}}$ into two modules? Robusta involves changing one of the submodules from G_{coarse} with a Segformer [46]. While we keep the global structure of pix2pixHD [44] with a GAN generating outputs to be concatenated in the latent space of the main generative network, the philosophy is completely different. The objective of pix2pixHD was to be able to work at different resolutions, possibly with a greater number of submodules. In our case, the objective of the Segformer submodule is to leverage two different architectures, which are known to focus on different aspects of the data [27], and fuse their learned information by concatenation. For instance, transformers [9] tend to learn more about the context and low frequencies in the images. With this in mind, we use the same input resolution for the transformer and its convolutional counterpart, as opposed to pix2pixHD.

Downsampling Block
Conv2D(in_ch=35, out_ch=64, kernel_size=7)
Batch_Norm(64)
ReLU
Conv2D(in_ch=64, out_ch=128, kernel_size=3, stride=2)
Batch_Norm(128)
ReLU
ResNet Block
Conv2D(in_ch=256, out_ch=256, kernel_size=3)
SPADE(256)
ReLU
Conv2D(in_ch=256, out_ch=256, kernel_size=3)
SPADE(256)
Upsampling Block
ConvTranspose2D(in_ch=256, out_ch=128, kernel_size=3, stride=2)
SPADE(128)
ReLU
Conv2D(in_ch=128, out_ch=3, kernel_size=3)
TanH

Table A1. **Technical details on $G_{\textbf{coarse}}$.**

The interface between the two modules We use a dense layer to project the output of the Segformer submodule to the size of the latent space of its convolutional counterpart to enable the concatenation of the activations. We also adapt the number of channels to those of the latent activation of the convolutional network using the `fuse-conv` proposed by Xie et al. [46].

The usefulness of SPADE layers To reinject spatial information, we incorporate SPADE in all the batch normalization layers of the ResNet and Upsampling Block of G_{coarse}. Specifically, we use SPADE after the convolutions that come after the concatenation of the Segformer output and the latent space of the convolutional network.

B. Implementation details

B.1. Training hyperparameters

This section includes the hyper-parameters utilized in the label-to-image translation, semantic segmentation, and anomaly detection experiments. These hyper-parameters are presented in Tables A2, A3, and A4. Our implementation is based on PyTorch [29], and we plan to release our code to the public.

C. Quality of the images generated by Robusta

C.1. Datasets, metrics & baselines

C.1.1 Datasets.

We performed experiments on well-established datasets in label-to-image translation: ADE20K [48], COCO-stuff [3], Cityscapes [6], KAIST [19], and ADE20K-outdoors [28], a subset of ADE20k containing only outdoor scenes. In all our label-to-image translation experiments, the images are resized at the resolution of 256×256, except for Cityscapes, for which the resolution is set to 256×512.

C.1.2 Performance metrics.

Following common protocol from prior label-to-image translation works [28, 37], we assess the quality of the generated images with two primary metrics: the Fréchet Inception Distance (FID) and the mean intersection over union (mIoU). FID measures the similarity of two sets of images based on their visual features, extracted by a pre-trained Inceptionv3 model [38]. Lower scores correspond to better visual similarity between the sets of images. The mean Intersection over Union is used to evaluate the accuracy of semantic segmentation DNN's prediction and therefore provides hints on the quality of the rendering. In addition, we evaluate the quality of OOD detection on Outlier-Cityscapes using the Areas Under the Precision/Recall curve (AUPR) and Under the operating Curve (AUROC), following Hendrycks & Gimpel [14].

To evaluate mIoU, we require a pre-trained DNN capable of performing semantic segmentation. As in [28, 37], we use multi-scale DRN-D-105 [47] for Cityscapes, DeepLabV2 [5] for COCO-Stuff, and UperNet101 [45] for both ADE20K and ADE20K-outdoors datasets. For the KAIST dataset, which only contains bounding box annotations of persons, we evaluate the mean Average Precision (mAP) specifically for the person class, using a faster R-CNN model [31] with ResNet-101 backbone trained on the COCO dataset [23].

C.1.3 Baselines.

In this study, we compare Robusta with Pix2PixHD [44] and six other state-of-the-art baselines: SPADE [28], CRN [21], SIMS [30], CC-FPSE [24], LGGAN [39], and OASIS [37]. For each approach, we either reproduce the results if the checkpoint is provided or retrain the models. In order to obtain the most accurate comparisons, all models are trained at the same resolutions.

C.2. Results of the experiments

C.2.1 Image quality results

We adopt the same evaluation protocol used in previous studies on label-to-image translation [28, 37] to assess the quality of our images. Specifically, we convert label maps into RGB synthetic images and measure the FID and mIoU. Table A5 shows that we achieve equivalent or superior performance compared to state-of-the-art methods on most of the datasets. Towards a more comprehensive evaluation, we also include qualitative assessments of the synthesized images in Appendix G with multiple visual examples for each dataset.

C.2.2 Domain adaptation results

The last column of Table A5 presents the results for domain adaptation. To evaluate the effectiveness of our approach, we use an object detection DNN trained on RGB images and test its performance on infrared images using both Pix2PixHD [44] and Robusta. Our results show that the mean average precision (mAP) on Pix2PixHD-generated images is only 1.40%, but it significantly improves to 4.57% with Robusta. These findings demonstrate the efficacy of our GAN cascade, which not only improves label-to-image translation but also enhances object detection performance across domains. For more details on the experimental protocol and instructions on how to use Robusta for this task, please refer to Table A3.

D. Background on morphology & editing

The field of mathematical morphology [35] is a well-established nonlinear image processing field that applies complete lattice theory to spatial structures.

Let $\mathbf{x}$ be a grey-scale image with the intensity at position p denoted as $\mathbf{x}(p)$. Morphology involves two fundamental operations, grey-level dilation, and grey-level erosion, which are defined as follows:

$$\varepsilon_b(\mathbf{x})(p) = \min_{h \in B}(\mathbf{x}(p-h) - b(h)), \tag{A10}$$

$$\delta_b(\mathbf{x})(p) = \max_{h \in B}(\mathbf{x}(p-h) + b(h)). \tag{A11}$$

In these equations, $\varepsilon_b(\mathbf{x})$ and $\delta_b(\mathbf{x})$ represent morphological erosion and dilation, respectively, using a given structuring element b of support B [35]. The structuring element's geometry determines the operators' effect, and for simplicity, we consider uniform structuring functions formalized by

	ADE20K	AO	Cityscapes	COCO-stuff	KAIST
Segformer	B5	B5	B5	B0	B5
Learning rate	0.0002	0.0002	0.0002	0.0002	0.0001
Batch size	32	16	8	8	32
Resolution	286×286	256×256	512×256	286×286	256×256
Weight decay	0	0	0.00001	0	0
Epochs	200				
Random crop	256×256	$\varnothing$	$\varnothing$	256×256	$\varnothing$
λ_{VGG}	5				
λ_{FM}	5				

Table A2. **Hyperparameter configuration used in the training of G_{coarse}.** AO is ADE20K-outdoors.

	ADE20K	AO	Cityscapes	COCO-stuff	KAIST
Learning rate	0.0002				
Batch size	12	12	8	10	10
Resize resolution	256×256	256×256	512×256	286×286	256×256
Weight decay	0				
Epochs	100				
Random crop	$\varnothing$	$\varnothing$	$\varnothing$	256×256	$\varnothing$
λ_{VGG}	1	10	10	1	10
λ_{FM}	0.1	1	1	1	1

Table A3. **Hyperparameter configuration used in the training of G_{fine}.** AO is ADE20K-outdoors.

their support shape. More complex operators like opening and closing can be obtained by applying these two basic operators repeatedly.

Extending morphological operators to multivariate images, particularly color images or hyperspectral images, requires appropriate vector-ordering strategies. Researchers have developed various orders for multivariate images [1, 4, 10, 43]. In this discussion, we focus on simplex ordering, like [10], but propose a lexicographic ordering based on two components. First, we consider all non-instance pixels to be in the foreground, representing the smallest element of the order. Second, it is essential to order classes based on the editing task at hand. We recommend that classes representing small objects be assigned the highest values. Thus, for example, on Cityscapes, we would have the following ordering:

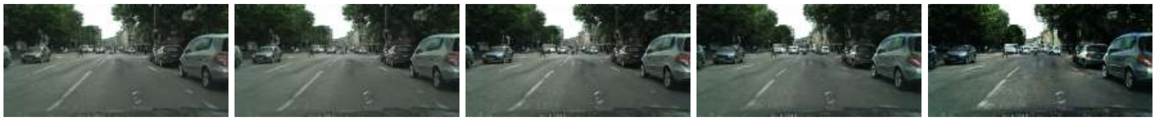
$$\text{traffic sign} > \text{traffic light} > \text{person} > \text{car} > \text{cycle} > \text{truck} > \text{train}.$$

	Perturbation robustness			**Outlier detection**	
	StreetHazards	BDD Anomaly	Cityscapes	StreetHazards	BDD Anomaly
Architecture	Deeplab v3+			Deeplab v3+	
Backbone	ResNet 50		ResNet 101	ResNet 50	
Output stride	8			8	
Learning rate	0.02		0.1	0.002	
Batch size	4		8	6	
Epochs	25		80	15	
Weight decay	0.0001			0	
Random Crop	∅		(768, 768)	(80, 150)	

Table A4. **Hyper-parameter configuration used in semantic segmentation experiments.**

Method	ADE20K [48]		AO [28]		Cityscapes [6]		COCO-stuff [3]		KAIST [19]	
	FID↓	mIoU↑	FID↓	mIoU↑	FID↓	mIoU↑	FID↓	mIoU↑	FID↓	mAP↑
CRN	73.3	22.4	99.0	16.5	104.7	52.4	70.4	23.7	n/a	n/a
SIMS	n/a	n/a	67.7	13.1	49.7	47.2	n/a	n/a	n/a	n/a
pix2pixHD	81.8	20.3	97.8	17.4	95.0	58.3	111.5	14.6	55.4	1.4
LGGAN	31.6	41.6	n/a	n/a	57.7	68.4	n/a	n/a	n/a	n/a
CC-FPSE	31.7	43.7	n/a	n/a	54.3	65.5	19.2	41.6	n/a	n/a
SPADE	33.9	38.5	63.3	30.8	71.8	62.3	22.6	37.4	n/a	n/a
OASIS	28.3	48.8	48.6	40.4	47.7	69.3	**17.0**	**44.1**	n/a	n/a
Robusta (G_{coarse})	29.4	46.7	49.7	40.6	56.3	68.8	30.4	42.1	53.7	**4.5**
Robusta ($G_{\text{coarse}}, G_{\text{fine}}$)	**28.1**	**49.0**	**48.4**	**41.8**	**47.1**	**70.8**	30.2	42.8	**52.1**	**4.5**

Table A5. **Comparison across datasets.** Our method outperforms the current leading methods in semantic segmentation (mIoU) and FID scores on most benchmark datasets. AO is ADE20K-outdoors.



(a) $\mathbf{x}$ (b) $\delta_{b_1}(\mathbf{x})$ (c) $\delta_{b_2}(\mathbf{x})$ (d) $\delta_{b_3}(\mathbf{x})$ (e) $\delta_{b_4}(\mathbf{x})$

Figure A2. **Illustration of the results of image morphological editing.** Figure (a) represents the results of label-to-image translation, and Figures (b)-(e) represent the results of label-to-image translation applied to the dilated label maps with increasing structuring elements.

E. Ablation studies

E.1. Architectural choices

E.1.1 Ablation study on the architecture

Table A6 shows the results of an ablation study for the different variations and improvements proposed by Robusta on two datasets - ADE20K-outdoors [28] (AO) and Cityscapes [6]. Specifically, we measure the impact of replacing one of the submodules of G_{coarse} with a Segformer [46], as well as the importance of using SPADE [28] layers in G_{coarse}, and G_{fine}, and the interaction with the new generator G_{fine}. The table reports FID and mIoU metrics, which indicate the quality of the generated images and the accuracy of the segmentation, respectively. The best-performing method is highlighted in the table and corresponds to Robusta with all the proposed architectural contributions. It achieves the lowest FID and highest mIoU scores for both datasets, specifically, it improves the FID by 19.4% and the mIoU by 19.6% on AO, and 30.3% and 13.4% on CS for the FID and the mIoU respectively.

Comparing the results, we can first observe that replacing the backbone of the submodule of G_{coarse} significantly improves the performance of the model by 17% of FID and 13.5% of mIoU on AO and 11.1% FID and 11.4% mIoU on CS. Second, adding G_{fine} overall improves results and yields improvements, for instance, 3.2% of mIoU on AO and 6% of mIoU on CS, despite an increase of 3.5% of the FID on AO and a decrease of 1% of the mIoU on CS. Using SPADE layers also improves results overall.

In conclusion, the ablation study results show that the addition of architectural contributions such as the Segformer submodule, SPADE layers, and G_{fine} can significantly improve the performance of the generating model for semantic segmentation tasks. Furthermore, the best results are obtained by using all the architectural contributions together in the Robusta model, which achieves the lowest FID and highest mIoU scores on both datasets. These results highlight the effectiveness of the proposed Robusta model for label-to-image tasks and demonstrate the importance of our architectural contributions to improve model performance.

E.1.2 Size of the Segformer

In this section, we investigate the influence of the size of the G_{coarse} network on the performance of (G_{coarse}, G_{fine}) on Cityscapes and ADE20K. To do so, we experiment with different sizes of G_{coarse} based on Segformer [46], which offers five variants ranging from the lightest B0 to the heaviest B5. We report the results for all Segformer models in Table A7. Our findings show that increasing the complexity of the Segformer model leads to a higher mIoU score; however, this may not hold true for the FID. Additionally, we observe that the lightweight B0 variant performs well enough with lower complexity, which can be a good option for when resources are limited.

SegFormer	SPADE 1	SPADE 2	G_{fine}	ADE20K-outdoors		Cityscapes	
				FID ↓	mIoU ↑	FID ↓	mIoU ↑
✗	✗	✗	✗	67.8	22.2	77.4	57.4
✓	✗	✗	✗	50.8	35.7	56.3	68.8
✗	✓	✗	✗	55.1	33.6	63.9	64.4
✗	✗	✗	✓	52.3	30.5	67.5	60.9
✗	✓	✓	✓	52.4	37.7	53.6	67.5
✓	✗	✗	✓	54.3	38.9	50.0	67.8
✓	✓	✗	✓	49.6	39.2	47.9	66.1
✓	✗	✓	✓	56.5	41.0	48.0	68.3
✓	✓	✓	✓	**48.4**	**41.8**	**47.1**	**70.8**

Table A6. **Ablation study of each architectural contribution**. Last row corresponds to Robusta. SPADE 1 is for SPADE in G_{coarse}. SPADE 2 is for SPADE in G_{fine}.

Segformer	ADE20K		AO		Cityscapes	
	FID ↓	mIoU ↑	FID ↓	mIoU ↑	FID ↓	mIoU ↑
B0	34.4	37.62	53.05	33.79	49.60	66.53
B1	32.29	38.94	52.73	35.56	49.5	65.74
B2	32.04	39.51	50.09	37.15	50.01	67.84
B3	30.98	40.84	51.57	36.52	49.58	67.92
B4	31.71	41.37	50.80	36.89	**48.56**	68.14
B5	**30.30**	**43.93**	**49,65**	**40,58**	56.34	**68.76**

Table A7. **Performance of Robusta with different SegFormers [46]**. The largest Segformer, B5, yields the best results. AO is ADE20K-outdoors.

E.1.3 Usefulness of G_{fine}

To evaluate the effectiveness of G_{fine}, we perform an analysis in the Fourier domain on 500 images from the Cityscapes validation set [6]. For each image, we compute the fast Fourier transform of the ground truth, the output of G_{coarse}, and of G_{fine}. We then analyze the distance between the spectra of the generated images and the ground truth. Figure A3 presents the mean distance among all the Fourier domains for three scenarios: one considering all the complete spectra and two considering only high frequencies.

As shown in Figure A3, G_{fine} yields a closer representation of the original image than G_{coarse} for all cases. In particular, we see that this analysis holds for high frequencies (with a `filter_rate` of 2).

E.2. Training choices

E.2.1 Ablation study on the perceptual losses

In this discussion, we explore the importance of perceptual losses in our experiments. Table A3 illustrates that for Cityscapes and ADE20-Outdoors, the best results are achieved when the values of λ_{VGG} and λ_{FM} are 1 and 0.1, respectively. However, for ADE20K and COCO-stuff, larger values of λ_{VGG} and λ_{FM} (10 and 1, respectively) lead to the best performance. Additionally, we present an ablation study

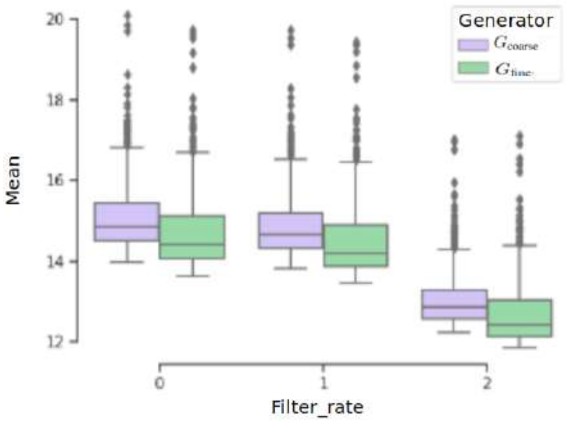


Figure A3. **Frequency analysis of the usefulness of G_{fine}.** Filter rate represents the cutoff frequency of the high-pass filter. `Filter_rate` $= 0$, No filter. `Filter_rate` $= 1$, Low cutoff frequency. `Filter_rate` $= 2$, High cutoff frequency.

λ_{VGG}	Cityscapes		ADE20K	
	FID ↓	mIoU ↑	FID ↓	mIoU ↑
10.0	**47.10**	**70.8**	34.6	45.2
1.0	55.4	68.6	**28.1**	**49.0**
0.1	59.8	65.9	35.8	42.7

Table A8. **Performance of the semantic image synthesis of Pix2PixHD for different values of λ_{VGG}.** Experiments done on the second-stage training part of G_{fine} only.

on the impact of λ_{VGG} in Table A8. This study highlights the different responses of Cityscapes and ADE20K to changes in λ_{VGG}. We believe that small and specific datasets like Cityscapes and ADE20-Outdoors pose a particularly challenging problem, where the key features required to generate photo-realistic images may not be easily captured by the network. In such cases, perceptual losses are vital to guide the learning process and achieve good results. Conversely, larger datasets like COCO-stuff and ADE20K provide enough samples for the network to learn by itself, and perceptual constraints can be relaxed to promote diversity in the image generation process.

E.2.2 Ablation study on the data augmentation

Data augmentation	FID ↓	mIoU ↑
Pix2PixHD w/ SegFormer (B5)	56.34	**68.76**
Robusta (B5) + CutMix	51.02	63.02
Robusta (B5) + LabelMix	**44.39**	67.25
Robusta (B5) + SuperPixelMix	45.38	67.46

Table A9. **Performance of Pix2PixHD and Robusta on Cityscapes** with different data augmentations.

In this section, we investigate the impact of data augmentation for label-to-image translation on Cityscapes. We note that OASIS employs LabelMix [25] data augmentation. Several mixing techniques have been proposed for tasks like semantic segmentation and semi-supervised learning, including CutMix [12] and Superpixel-mix [11], which is designed to improve the robustness of semantic segmentation. We present the results of applying data augmentation during the training of Robusta, composed of G_{coarse} and G_{fine} only, in Table A9. We find that mixing data augmentation improves the FID but reduces the mIoU. LabelMix greatly enhances the FID, at the cost of a significant drop in mIoU. Superpixel-mix appears to strike the best compromise. However, due to their poor performance on mIoU, we decide not to employ these strategies.

Dataset	Model	mIoU
Cityscapes		76.5
Cityscapes + *Corrupted-CS*	SPADE	76.0
Cityscapes + *Corrupted-CS*	OASIS	77.1
Corrupted-CS	Robusta	41.7
Cityscapes + *Corrupted-CS*	Robusta	**78.4**

Table A10. **Aleatoric uncertainty study** on Cityscapes-C, Foggy Cityscapes [15] and Rainy Cityscapes [34].

F. Discussions

F.1. On the importance of high-quality image generation for semantic segmentation

To evaluate the quality of images required for training a DNN, we train a Deeplab v3+ network using images generated by various cGANs. Our training protocol involves two steps and two DNNs, namely a student DNN optimized by backpropagation and a teacher DNN optimized by exponential moving average (EMA) [40]. The two training steps were conducted in parallel and involved classical supervised training on all data as well as training on images through pseudo-annotation with the teacher's DNN. This training protocol allowed the DNN to consider issues related to image synthesis.

Table A10 displays our results using different image generation techniques. Our findings indicate that when combined with real images, the Robusta-generated dataset offers the best performance. However, it is worth noting that using solely the images generated by Robusta did not result in ideal performance. Nevertheless, these results should be considered in relation to the performance of a model trained on GTA [32], which yields around 31% of mIoU.

F.2. On semantic segmentation and uncertainties

To achieve the goal of improving safety, it is necessary to not only improve the generalization properties of our algorithms, which translates, for instance, into an increase of the accuracy on corrupted datasets but also enhance their anomaly detection capabilities. To do so, it is crucial to address the issue of uncertainties in semantic segmentation [7, 20]. In other words, to improve both generalization and anomaly detection, our algorithms must be able to handle and be robust to uncertainties in semantic segmentation.

Let us begin by introducing the uncertainties that occur in semantic segmentation [7, 20]. As learning algorithms, they are confronted with uncertainties, which can be categorized into two types: aleatoric and epistemic uncertainties [13].

Aleatoric uncertainties [16] are related to the data and are caused by natural variation or lack of information. In semantic segmentation, they would mostly correspond to unclear object borders or to corruptions in the test data.

Architecture	MA ↓	Params ↓	VRAM ↓
Pix2PixHD	161.4	188.1	24
SPADE	272.5	98.6	256
OASIS	296.8	93.4	128
Robusta w/o G_{fine}	393.6	98.6	34
G_{fine}	297.3	50.5	24

Table A11. **MA: Mult-Adds** corresponds to the number of Giga multiply-add operations of a forward pass through both the generators and discriminators with a single input. **Params: total number of parameters (in million)** of the generators and discriminators combined. **VRAM:** memory (in GB) required for a forward/backward through both the generators and discriminators. **MA** and **Params** are estimated with Torchinfo [41].

Epistemic uncertainties [16] arise when the model is ill-constrained, that is that the outputs are not properly defined by the minimization of the loss on the training set. This would correspond to unknown shapes or textures on the inputs involving mispecified outputs.

F.3. Training speed

Our aim is for the algorithm to not only produce realistic images and perform well in terms of robustness but also to be scalable and require a reasonable amount of computational resources that match the complexity of the task at hand. To further explain this point, we analyze the balance between the training time and the FID score of the latest and most competitive algorithms for image-to-label translation. The results, as depicted in Figure A4, show that Robusta is positioned in a favorable spot. Compared to Pix2PixHD, it has fewer parameters, faster training time than OASIS, and a better FID score than other algorithms. By incorporating G_{fine}, the training time is slightly affected, but there is a significant performance improvement. It's important to note that the reason for OASIS's low training time is due to the per-pixel discriminator that predicts full-resolution segmentation maps, which requires a large amount of VRAM, as shown in Table A11. Additionally, the LabelMix data-augmentation strategy utilized by the algorithm is not as efficient when it comes to GPU usage.

Table A11 provides additional information on other relevant metrics about scalability. It should be noted that the number of parameters and VRAM is more closely related to space complexity. Since our training process involves two stages, the effective number of parameters/VRAM experienced during training corresponds to the values of the first stage, which is a bottleneck due to the lighter weight of G_{fine} compared to G_{coarse}. Overall, Robusta is comparable to SPADE and OASIS in terms of the number of parameters and Mult-Adds. However, our algorithm utilizes only half as much VRAM as OASIS.

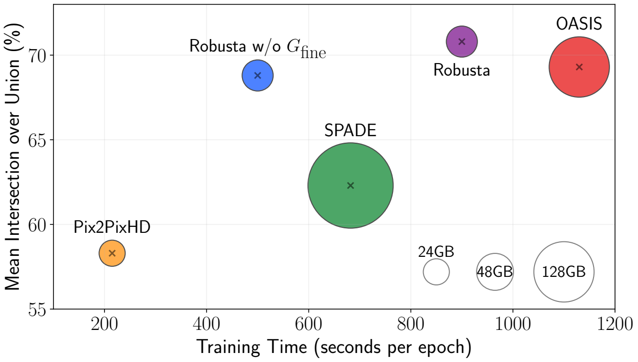


Figure A4. **Computation cost and performance trade-offs for several image-to-image translation techniques on Cityscapes.** The y-axis shows the FID and the x-axis shows the training time for one epoch with a batch size of 4. Training time is averaged over 5 epochs. The circle area is proportional to the required VRAM in GB to train the model. In the case of Robusta, G_{coarse} and G_{fine} are trained sequentially and the required memory corresponds to the heaviest generator, G_{coarse}. The best approaches are closer to the upper-left corner.

F.4. Evaluation protocol

Comparing the results of generative networks is often challenging, and thus either the FID or mIoU are commonly used. While human assessment is also used for qualitative validation, it can be time-consuming, and evaluators may introduce biases.

We note that the mIoU could also have a bias. Indeed, we see that most of the techniques outperform the network mIoU trained on the training set and tested on the real validation set. More specifically, the mIoU of DRN [47] on Cityscapes equals 66.35%, the one of Deeplabv2 [5] on COCO-stuff is 39.03%, and the one of UperNet101 [45] on ADE20K is 42.74%. Hence, the approaches that beat these values can transform the validation set into a training set.

In their work on domain adaptation, Ben David et al. [2] mention the divergence between the distribution of the target domain and that of the source domain, which would limit the empirical risk for the target domain. It seems that modern approaches are finding ways to facilitate the transfer between the source and target domains. One small concern is that access to the label is necessary, which is still a problem. However, it should be possible to replace it with a depth map and generate RGB images from that depth map.

G. Qualitative results

In this section, we illustrate the comparison between OASIS and Robusta with some visual examples. Experiments are done on three datasets: Cityscapes [6] in Figure A5, COCO-stuff [23] in A6, ADE20K [48] in A7, *Outlier-Cityscapes* in A8, *Corrupted-Cityscapes* in A9, and KAIST [19] in Figure A10.

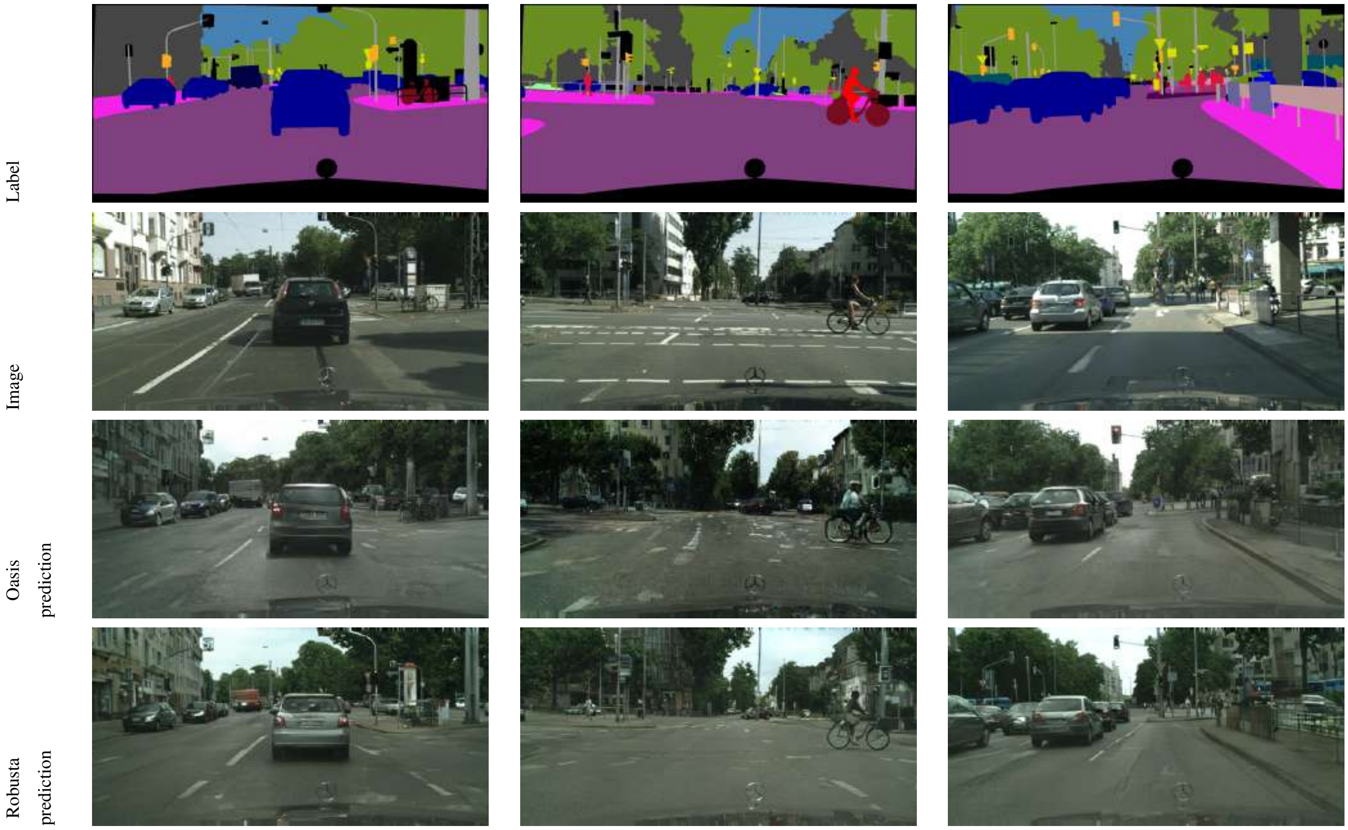


Figure A5. Qualitative comparison of Robusta with other methods on Cityscapes.

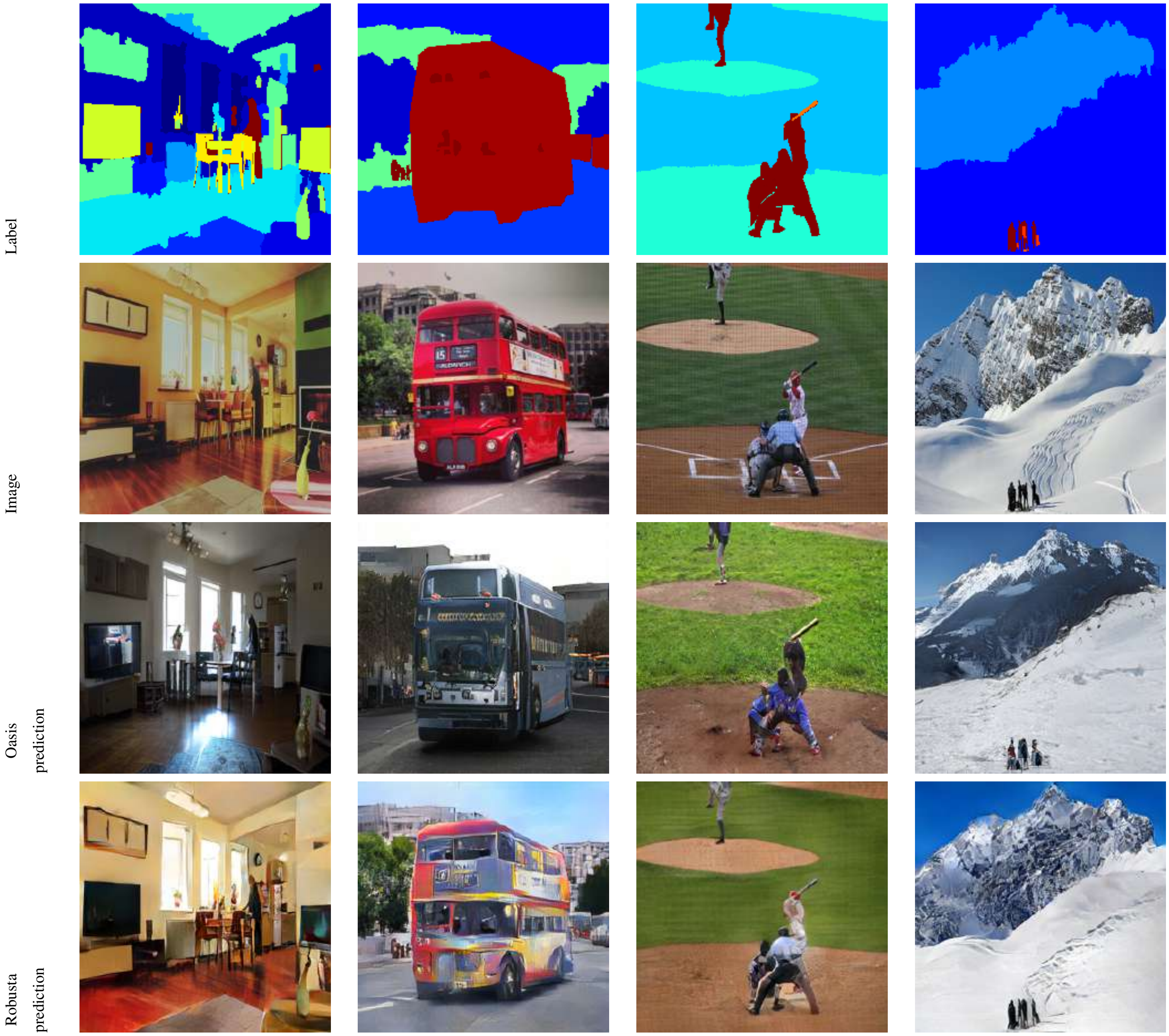


Figure A6. Qualitative comparison of Robusta with other methods on COCO-stuff.

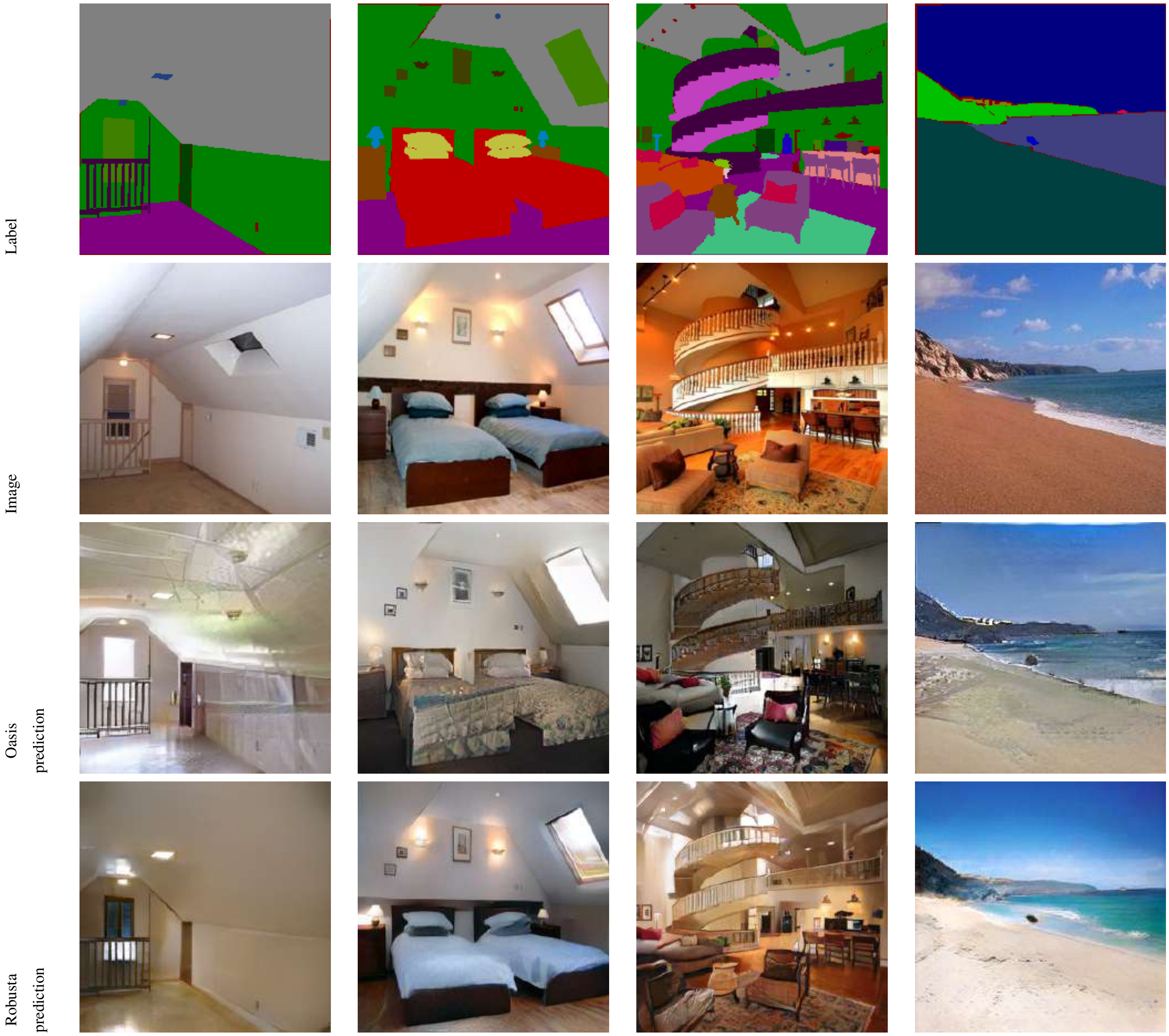


Figure A7. Qualitative comparison of Robusta with other methods on ADE20K.

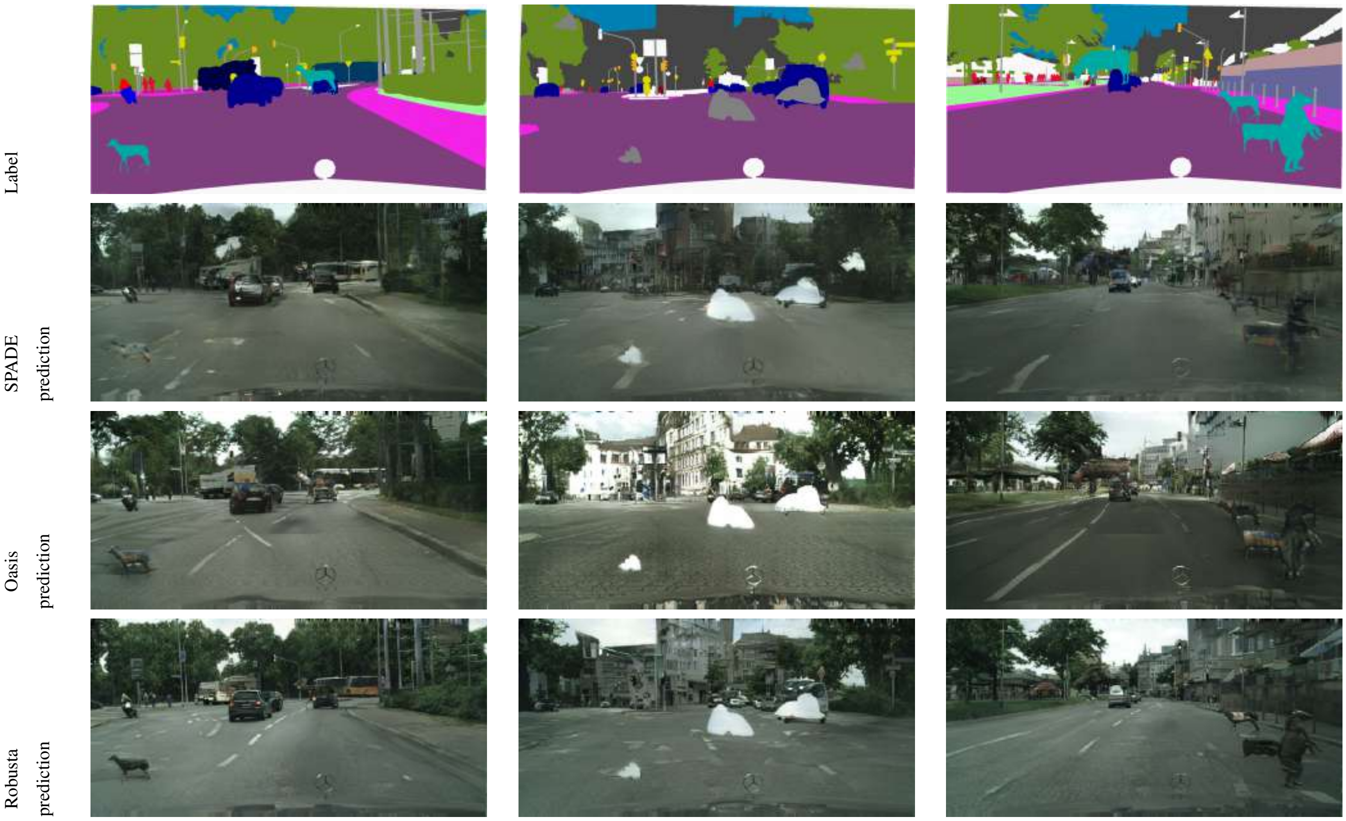


Figure A8. Qualitative comparison of Robusta with other methods on *Outlier-Cityscapes*.

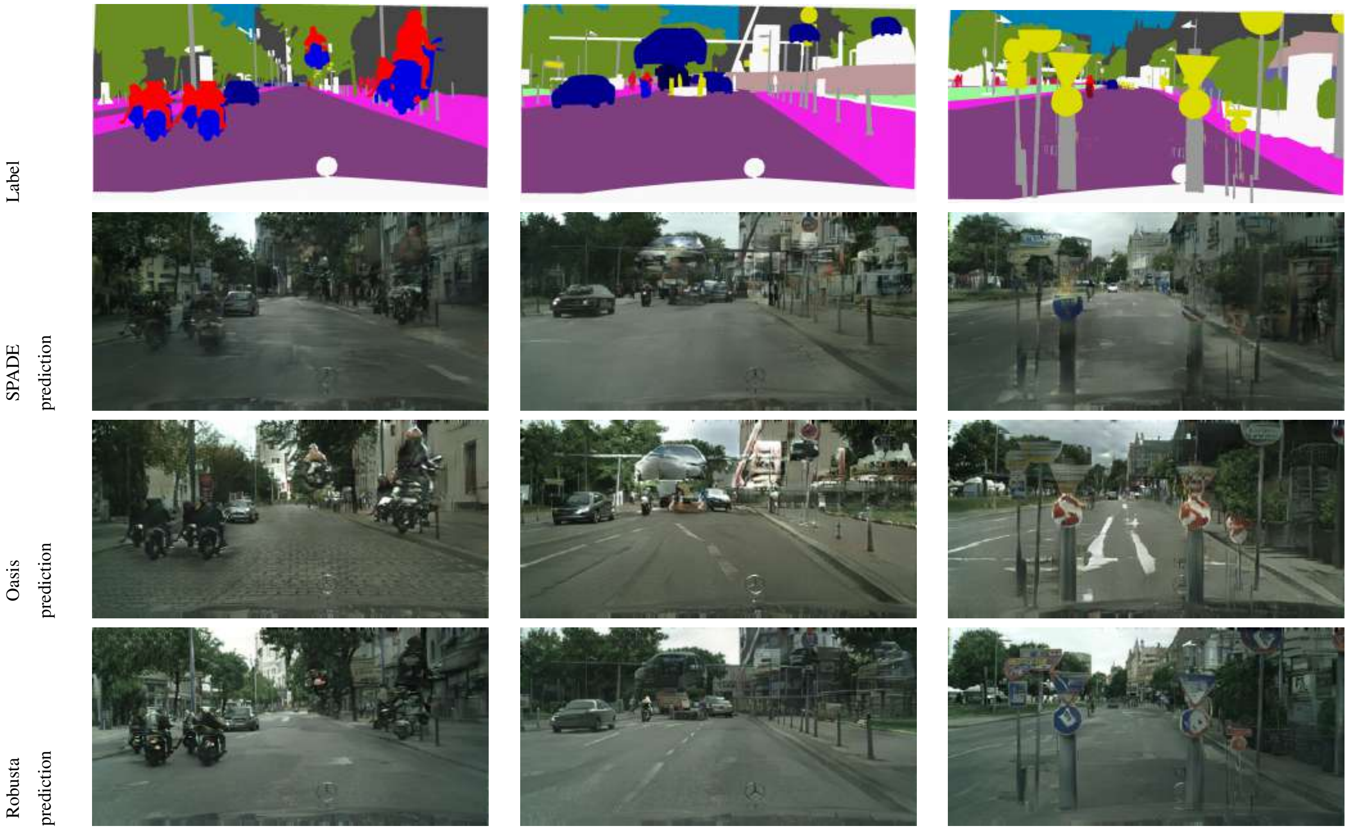


Figure A9. Qualitative comparison of Robusta with other methods on *Corrupted-Cityscapes*.

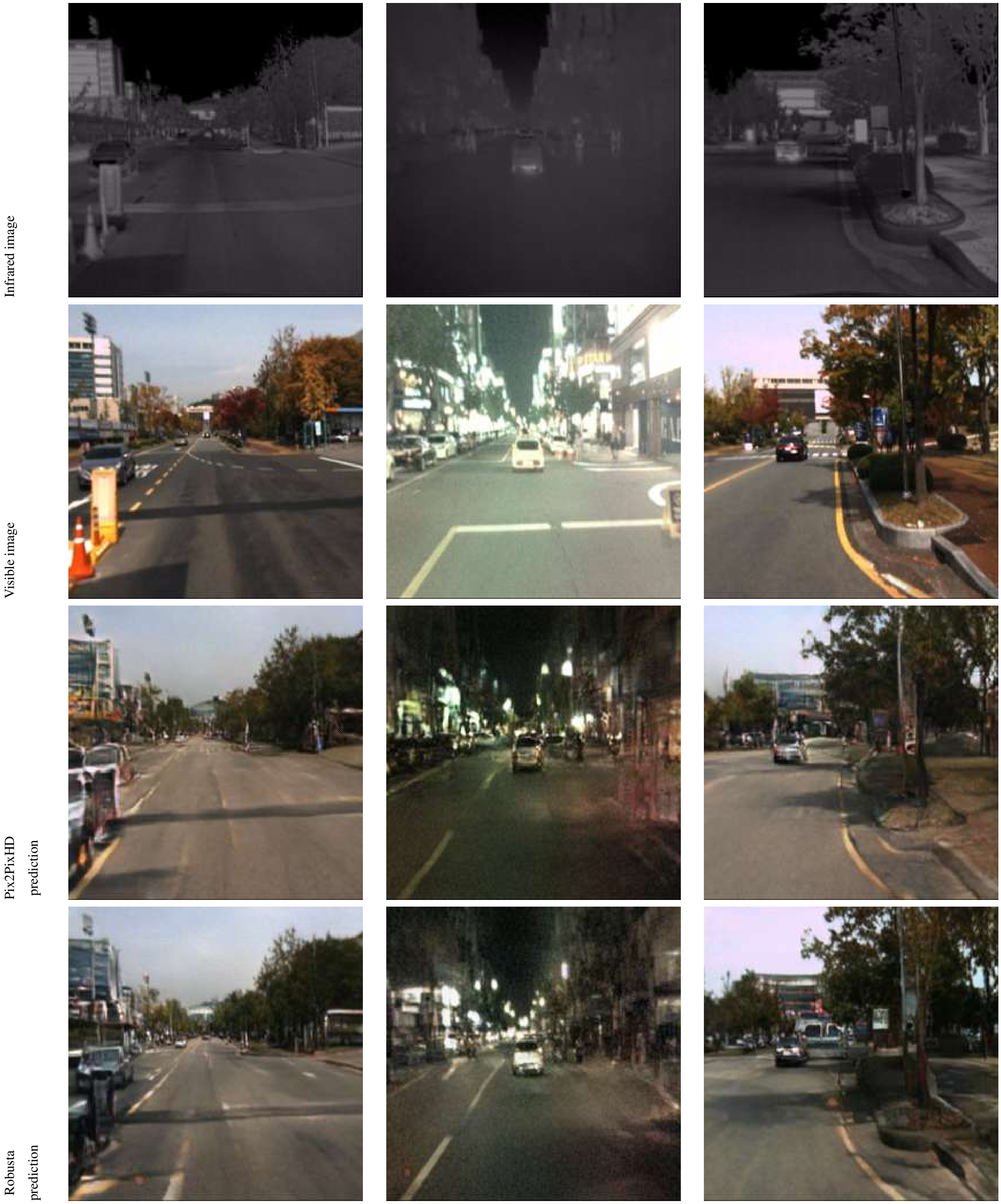


Figure A10. Qualitative comparison of Robusta with other methods on KAIST.

References

[1] Jesús Angulo. Morphological colour operators in totally ordered lattices based on distances: Application to image filtering, enhancement and analysis. *Computer vision and image understanding*, 2007. 5

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 2010. 10

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018. 4, 6

[4] Alexandru Căliman, Mihai Ivanovici, and Noël Richard. Probabilistic pseudo-morphology for grayscale and color images. *Pattern Recognition*, 2014. 5

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017. 4, 10

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 4, 6, 7, 10

[7] Sebastian Cygert, Bartłomiej Wróblewski, Karol Woźniak, Radosław Słowiński, and Andrzej Czyżewski. Closer look at the uncertainty estimation in semantic segmentation under distributional shift. In *IJCNN*, 2021. 9

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[10] Gianni Franchi and Jesus Angulo. Ordering on the probability simplex of endmembers for hyperspectral morphological image processing. In *ISMM*, 2015. 5

[11] Gianni Franchi, Nacim Belkhir, Mai Lan Ha, Yufei Hu, Andrei Bursuc, Volker Blanz, and Angela Yao. Robust semantic segmentation with superpixel-mix. In *BMVC*, 2021. 8

[12] Geoff French, Samuli Laine, Timo Aila, Michal Mackiewicz, and Graham Finlayson. Semi-supervised semantic segmentation needs strong, varied perturbations. In *BMVC*, 2020. 8

[13] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021. 9

[14] Dan Hendrycks, Steven Basart, Mantas Mazeika, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling out-of-distribution detection for real-world settings. In *ICML*, 2019. 4

[15] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, and Pheng-Ann Heng. Depth-attentional features for single-image rain removal. In *CVPR*, 2019. 9

[16] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine-Learning*, 2021. 9

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 2

[18] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2

[19] Jinyong Jeong, Younggun Cho, Young-Sik Shin, Hyunchul Roh, and Ayoung Kim. Complex urban dataset with multi-level sensors from highly diverse urban environments. *IJRR*, 2019. 4, 6, 10

[20] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NeurIPS*, 2017. 9

[21] Thao Minh Le, Vuong Le, Svetha Venkatesh, and Truyen Tran. Hierarchical conditional relation networks for video question answering. In *CVPR*, 2020. 4

[22] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 4, 10

[24] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, et al. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. *NeurIPS*, 2019. 4

[25] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning. In *WACV*, 2021. 8

[26] Yingxue Pang, Jianxin Lin, Tao Qin, and Zhibo Chen. Image-to-image translation: Methods and applications. *TMultimedia*, 2021. 2

[27] Namuk Park and Songkuk Kim. How do vision transformers work? In *ICLR*, 2022. 3

[28] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 2, 4, 6, 7

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4

[30] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. In *CVPR*, 2018. 4

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015. 4

[32] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 9

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2

[34] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 2018. 9

[35] Jean Serra. Image analysis and mathematical morphology. *(No Title)*, 1982. 4

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2

[37] Vadim Sushko, Edgar Schönfeld, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. In *ICLR*, 2021. 4

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 4

[39] Hao Tang, Dan Xu, Yan Yan, Philip HS Torr, and Nicu Sebe. Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation. In *CVPR*, 2020. 4

[40] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *NeurIPS*, 2017. 9

[41] Torchinfo. Torchinfo. `https://github.com/TylerYep/torchinfo`. Version: 1.7.1. 9

[42] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv:1607.08022*, 2016. 2

[43] Santiago Velasco-Forero and Jesus Angulo. Supervised ordering in $\mathbb{r}^{p}$: Application to morphological processing of hyperspectral images. *IEEE Transactions on Image Processing*, 2011. 5

[44] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 2, 3, 4

[45] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 4, 10

[46] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, 2021. 3, 7

[47] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *CVPR*, 2017. 4, 10

[48] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 4, 6, 10