Semantic Palette: Guiding Scene Generation with Class Proportions

Guillaume Le Moing* Inria†
Tuan-Hung Vu Valeo.ai
Himalaya Jain Valeo.ai
Patrick Pérez Valeo.ai
Matthieu Cord Valeo.ai

Abstract

Despite the recent progress of generative adversarial networks (GANs) at synthesizing photo-realistic images, producing complex urban scenes remains a challenging problem. Previous works break down scene generation into two consecutive phases: unconditional semantic layout synthesis and image synthesis conditioned on layouts. In this work, we propose to condition layout generation as well for higher semantic control: given a vector of class proportions, we generate layouts with matching composition. To this end, we introduce a conditional framework with novel architecture designs and learning objectives, which effectively accommodates class proportions to guide the scene generation process. The proposed architecture also allows partial layout editing with interesting applications. Thanks to the semantic control, we can produce layouts close to the real distribution, helping enhance the whole scene generation process. On different metrics and urban scene benchmarks, our models outperform existing baselines. Moreover, we demonstrate the merit of our approach for data augmentation: semantic segmenters trained on real layout-image pairs along with additional ones generated by our approach outperform models only trained on real pairs.

1. Introduction

Generative Adversarial Networks (GANs) [1, 8, 13, 21] have become powerful tools to generate photo-realistic images based on a collection of examples. When trained on real photo portraits in particular, they can produce stunning results [14, 15]. However, for complex structured images like urban scenes, they still struggle to produce satisfactory results: not only do generated scenes exhibit various types of artifacts, but they are also difficult to use for downstream tasks. In automotive applications for instance, generating a wide range of synthetic driving scenes to replace or complement limited amounts of real annotated data is expected to help training better models in the future, for critical-safety tasks such as object detection and segmentation. This is not yet the case, and our work is motivated by this goal.

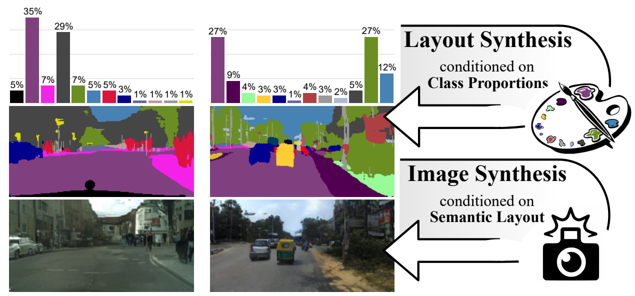

Figure 1: **Scene generation guided by semantic proportions.** The proposed approach, "Semantic Palette", allows a tight control of class proportions when generating semantic layouts and, conditioned on the latter, photo-realistic scenes such as urban scenes.

Our target application here is image semantic segmentation, the task of predicting semantic layouts, that is, a class label (or a set of class probabilities) for each pixel of a picture. State-of-the-art models being fully supervised, their training requires scene examples with corresponding semantic layouts. Hence, generating this type of data with a GAN amounts to producing matching image-layout pairs. To this end, recent works advocate decoupling the synthesis process into two consecutive phases: first generating semantic layouts with plausible object arrangements [2, 11], then translating these layouts into realistic images [24, 33].

To improve the usability of such a pipeline, we mostly focus here on the first layout generation step. Existing works [2, 11] cast it as a standard generative process that turns a random input code into a semantic map. While simple to use, this approach offers no real control on the modes of the output distribution [22], which is a limitation for complex scenes. In contrast, we propose to control the generation of layouts with a target distribution of semantic classes in the scene. Depending on applications, this class histogram can be manually defined, automatically derived from a true one, or sampled from a suitable distribution. To this end, we introduce a conditional layout GAN that takes a class histogram (the semantic code or *palette*) as input beside the standard random noise. As a result, our full image-layout generation pipeline (Figure 1) offers a simple yet powerful control over the scene composition. This ability brings benefits in various applications, ranging from

*Main part of this work was done during an internship at Valeo.ai
†Inria, École normale supérieure, CNRS, PSL Research University

real image editing to data augmentation for improved model training of a downstream task.

Using a progressive GAN [13] as base architecture to generate semantic layouts, we propose novel architecture designs and learning objectives to achieve our goal. First, we inject the semantic code throughout the progressive pipeline, *i.e.*, at multiple intermediate scales. To explicitly enforce the targeted class distribution while avoiding degenerate soft class assignments, we propose: a semantically-assisted activation (SAA) module along with two new learning objectives, as well as a novel residual conditional fusion module to ease the progressive propagation of the semantic target through the scales. Lastly, we introduce a variant of the proposed framework that allows partial editing of sub-regions in existing semantic layouts.

Our main experiments are conducted on different urban scene datasets. Using suitable direct metrics, we first assess the quality of the generated layouts and of the images derived from them. We also assess thoroughly the merit of our approach in the light of semantic segmentation downstream task. To this end, we train segmentation models on synthesized (resp. real) data and measure their performance on real (resp. synthesized) data, as a way to compare our method with the baselines. We finally assess the ability of several approaches to improve model training through augmentation of a real-data training set. In all experiments, Semantic Palette outperforms baselines and produces scenes that follow better the distribution of real ones. More importantly, for real-world applications, using it to extend real datasets boosts performance in semantic segmentation.

In summary, our main contributions are:

- A novel layout generative model that allows control of the distribution of semantic classes. This benefits both the quality of the images that are subsequently generated and the practical use of these images. To further enhance the quality of the generated scene-layout pairs, our method allows end-to-end training of both layout and image generators.
- A variant of our framework for partial editing of semantic layouts. This further benefits downstream task training and opens up interesting applications like semantic editing of real images.
- Extensive evaluations on three different driving benchmarks. The proposed framework significantly outperforms several baseline approaches.

2. Related work

Scene layout generation. SB-GAN [2] and PGAN-CGAN [11] were the first GAN-based approaches proposed for this task. The SB-GAN pipeline combines an unconditional model based on ProGAN [13] for generating the semantic masks and GauGAN [24] for transforming these masks into photo-realistic images. PGAN-CGAN [11] is similar but uses Pix2PixHD [33] instead as the image generator. In the same spirit, [31] uses DCGAN [25] as the layout generator and Pix2Pix [12] as the image generator.

Conditional generative adversarial network. Vanilla GANs offer little control over the generation process. In contrast, conditional GANs [22] (cGANs) are designed to guide the generation with conditioning input features, that is, target attributes of the generated data. Gradually deviating from the traditional GAN framework, alternative learning setups [23] and methods to better fuse the conditioning features with the generation pipeline [6, 7, 19, 24] have been proposed. An emerging trend is to infer some generator's parameters from the conditioning input, *e.g.*, normalization parameters [6, 7, 24], convolutional kernels [19], either uniformly [6, 7] or on class-specific regions [19, 24], to better take into account the scene structure.

Image generation conditioned on semantic layout. Pix2pix, a general-purpose image-to-image translation network [12], was among the first to produce compelling results for this conditional generation task. Later works [19, 24, 33] have produced more realistic images at higher resolution by using a multi-scale generator, adding a feature-matching loss within the discriminator, or using instance boundary map information. Recently, EdgeGAN [28] proposed to generate structure and texture in parallel and blend the two together thanks to an edge transfer module.

3. Proposed approach

The central goal of this work is to learn to generate plausible semantic layouts (*e.g.*, of urban scenes), conditioned on given class proportions. We describe our network architecture and objective functions for conditional layout generation in Sections 3.1 and 3.2. Section 3.3 discusses the image generation phase and how the complete pipeline with both layout and image generation can be trained end-to-end.

3.1. Conditional layout generative network

We wish to build a generative model that produces new layouts while controlling their semantic composition. The proposed architecture builds upon ProGAN [13], with new architecture designs and learning objectives to achieve our goal. It is thus a cascade of convolutional sub-networks handling information at multiple scales, and trained in a progressive way (Figure 2). At each scale, intermediate features are mapped into soft semantic maps (one per class) of the corresponding resolution via the TO MASK block. Only at the active resolution, the soft semantic layout is transformed through the FROM MASK block into input features used for adversarial training. The largest-scale output serves as the final generated outcome.

The network accepts as inputs not just a random noise

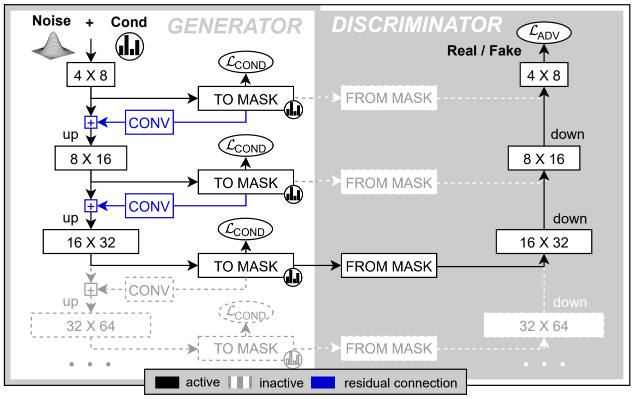

Figure 2: **Conditional synthesis of semantic layouts**. Snapshot at a certain resolution of the progressive generation (16×32 here). $\boxed{\text{H} \times \text{W}}$: two 3×3 convolutional layers with ReLU activations applied to feature maps of size $H\times W$; $\boxed{\text{TO MASK}}$: turns intermediate features into soft semantic maps; $\boxed{\text{FROM MASK}}$: turns soft semantic maps into input features for the discriminator; $\boxed{\text{CONV}}$: 1×1 convolutional layers with ReLU activation; " up" and "down": up- and down-sampling by a factor 2.

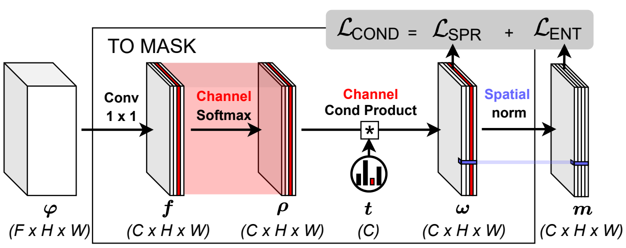

Figure 3: **Semantically-assisted activation**. In $\boxed{\text{TO MASK}}$ modules (Figure 2), which produce soft semantic maps $\boldsymbol{m}$ from features $\boldsymbol{\varphi}$, channel softmax combined with semantic modulation forces $\boldsymbol{\omega}$ to comply with the target proportions. Spread and entropy losses encourage the output mask $\boldsymbol{m}$ to retain these proportions from $\boldsymbol{\omega}$.

vector but also a conditioning code specifying the target class distribution, *i.e.*, the proportion of image surface that each class should occupy, for instance 50% of "sky", 30% of "road" and 20% of "pedestrian". At intermediate scales, we explicitly enforce conditioning constraints via a new *semantically-assisted activation* (SAA) module operating inside $\boxed{\text{TO MASK}}$ blocks. To propagate the conditioning information from previous scales onto the next, we propose to insert a *residual conditional fusion* between adjacent subnetworks. We explain the technical details next.

Conditioning input. We provide our model with the necessary information for conditional generation by concatenating an input noise vector $\boldsymbol{z}$ in $\mathbb{R}^Z$ (Z samples from standard Gaussian distribution) with a target normalized class histogram $\boldsymbol{t} \in \mathbb{R}_+^C$, with $\sum_{c=1}^{C} \boldsymbol{t}_c = 1$ and C the number of semantic classes. We also use this target explicitly throughout the generation scales, as shown in the following.

Semantically-assisted activation. We focus here on the design of the $\boxed{\text{TO MASK}}$ module, which converts deep features into soft semantic maps while respecting the prescribed class distribution. A common choice is to use a 1×1 convolutional layer to map the number of channels of the deep features to the number of semantic classes, followed by a spatial softmax activation. We stand out from this approach by introducing SAA, which makes again explicit use of the semantic code on top of the generation process. This way, we expect to enforce the respect of the class proportions in the generated maps. SAA (Figure 3) acts in three steps upon the C-channel feature maps $\boldsymbol{f} \in \mathbb{R}^{C\times H\times W}$ produced by the last convolutional layer, where $H \times W$ is the output resolution. First, a channel-wise spatial softmax is applied to $\boldsymbol{f}$ to obtain a density map $\boldsymbol{\rho}$ in $[0,1]^{C\times H\times W}$ with:

$$\boldsymbol{\rho}_{c,i,j} = \frac{\exp(\boldsymbol{f}_{c,i,j})}{\sum_{(k,\ell)\in\Omega} \exp(\boldsymbol{f}_{c,k,\ell})}, \tag{1}$$

for each class $c \in [\![1, C]\!]$ and each pixel location $(i,j) \in \Omega = [\![1,H]\!] \times [\![1,W]\!]$. Its slice $\boldsymbol{\rho}_{c,:,:}$ is a normalized spatial map for class c.

The next step is to use the semantic code to guide the output layout toward the target class distribution. To this end, the channels of the density map are weighted by their corresponding target proportions to define a new map $\boldsymbol{\omega}_{c,i,j} = \boldsymbol{t}_c \cdot \boldsymbol{\rho}_{c,i,j}$. This new weighted density verifies for each class c, $\sum_{(i,j)\in\Omega} \boldsymbol{\omega}_{c,i,j} = \boldsymbol{t}_c$. Thus, each class receives a "budget" amounting to its contribution in the semantic palette, *e.g.*, a class with the target proportion set to zero will not be represented in the final scene.

Finally, the semantic soft map output $\boldsymbol{m}$ in $[0,1]^{C\times H\times W}$ is obtained by L_1 normalization of $\boldsymbol{\omega}$, applied independently at every spatial location,

$$\boldsymbol{m}_{c,i,j} = \frac{\boldsymbol{\omega}_{c,i,j}}{\sum_{k\in[\![1,C]\!]} \boldsymbol{\omega}_{k,i,j}}, \tag{2}$$

which can be interpreted as defining at each pixel a probability distribution over all classes. As done classically, the final semantic map is obtained at each pixel by selecting the label with maximum score, *i.e.*, $\arg\max_c \boldsymbol{m}_{i,j,c} = \arg\max_c \boldsymbol{\omega}_{i,j,c}$. There is no guarantee that this hard labeling complies exactly with the target distribution, but it is tightly guided by it, as experiments will confirm. SAA can be seen as a mechanism that transports a well-proportioned but spatially-uniform semantic map (with slice c set to $\boldsymbol{t}_c$ everywhere) into a plausible spatial arrangement of the semantic content $\boldsymbol{m}$.[1]

In Section 3.2, we will detail the two losses attached to SAA at train time, which make use of this formalization to help $\boldsymbol{m}$ better follow the target semantic code.

Residual conditional fusion. With the SAA design above, the output $\boldsymbol{m}$ at an intermediate scale is already con-

[1] More formally, SAA can be interpreted as the first iteration of a Sinkhorn-type algorithm [27] in optimal transport, see Supp. Material.

ditioned by the semantic code. Such conditioned masks, though at lower resolutions, follow the target semantic code with realistic scene layout. It thus makes sense to pass those signals through the generation process of higher resolutions. This way, the $\boxed{\text{TO MASK}}$ layers are still of use when moving to higher resolutions, and having access to intermediate masks produced with semantic assistance can help the network better comply with the target proportions.

To that end, we propose to include a *residual conditional fusion* block before each upsampling layer (in blue in Figure 2). Via a 1×1 convolutional layer, the soft mask output by the current SAA module is mapped back to features of the same size as original features $\boldsymbol{\varphi}$ and added to them.

3.2. Learning objectives for layout generation

We train the conditional layout synthesis network with two objectives in mind: (1) a conditional objective to help generated layouts respect the target semantic proportions and (2) an adversarial objective to ensure realism. Both could be handled simultaneously by a conditional adversarial loss [22]. Here, we advocate a method that decouples the two, improving the layout quality by letting the discriminator focus on realism, as experiments will confirm.

Conditional objective. The conditional layout generator G maps noise and target proportions pairs $(\boldsymbol{z},\, \boldsymbol{t})$ to semantic probability masks $\boldsymbol{m}$. To make $\boldsymbol{m}$ follow as well as possible the target distribution, a matching loss could be used, *e.g.*, KL-divergence between the targeted and generated class distributions. But such a direct objective requires the non-differentiable counting of final max-score labels. While spatial aggregation of soft class maps, $\frac{1}{HW}\sum_{i,j}\boldsymbol{m}_{c,i,j}$, is a natural proxy for these label frequencies (coinciding with them in case of one-hot maps), its use can lead to undesirable solutions. For instance, a class could well be completely absent from the final layout (never being max-scoring at any location), while its soft map average matches exactly a non-zero target probability.[2] To this end, we introduce an alternative method which makes use of the proposed semantically-assisted activation.

Our novel conditional training loss has two parts: one to favor a peaky class distribution at each pixel (soft maps $\boldsymbol{m}$ close to one-hot), and one to favor an even spatial semantic coverage (uniform spread of activations in $\boldsymbol{\omega}$ over pixels). Intuitively, let the semantic palette be paint of different colors in various quantities. A sufficient condition to respect proportions from the palette in the final painting is to not mix colors too much in one spot (only dominant colors will be seen) while having all the paint from the palette evenly covering the frame (no accumulation or empty spot). The first part is translated into a loss that penalizes pixel-wise entropy of the soft masks $\boldsymbol{m}$ generated from $(\boldsymbol{z},\, \boldsymbol{t})$ pairs:

$$\mathcal{L}_{\mathrm{ENT}} = \frac{1}{HW}\mathbb{E}_{(\boldsymbol{z},\boldsymbol{t})}\Big[\sum_{(i,j)\in\Omega} e_{i,j}\Big], \tag{3}$$

where $e_{i,j} = -\sum_{c\in[\![1,C]\!]}\boldsymbol{m}_{c,i,j}\ln(\boldsymbol{m}_{c,i,j})$. The second part is a spread loss on the weighted density map $\boldsymbol{\omega}$. It encourages its activations to spread evenly across the image (hence to be close to $\frac{1}{HW}$ at each pixel since $\sum_{c,i,j}\boldsymbol{\omega}_{c,i,j} = 1$):

$$\mathcal{L}_{\mathrm{SPR}} = \frac{1}{HW}\mathbb{E}_{(\boldsymbol{z},\boldsymbol{t})}\Big[\sum_{(i,j)\in\Omega} s_{i,j}\Big], \tag{4}$$

where $s_{i,j} = (1 - HW\sum_{c\in[\![1,C]\!]}\boldsymbol{\omega}_{c,i,j})^2$. Intuitively, $\boldsymbol{\omega}$ defines a joint distribution over channels and pixel locations whose marginal over channels is defined by $\boldsymbol{t}$. The spread loss encourages the marginal over pixel locations to become uniform. This way, pixels should contribute evenly to the output semantic proportions.

The proposed conditional loss finally reads $\mathcal{L}_{\mathrm{COND}} = \mathcal{L}_{\mathrm{ENT}} + \mathcal{L}_{\mathrm{SPR}}$. Taking advantage of the progressive structure of the generator, supervision via $\mathcal{L}_{\mathrm{COND}}$ is possible at each resolution. The simultaneous action of SAA and of the two losses encourages the generated layouts to respect the target semantic code. Conditional to a palette $\boldsymbol{t}$, if $\mathcal{L}_{\mathrm{SPR}}$ is low, in average (over $\boldsymbol{z}$), each spatial location receives an overall $\boldsymbol{\omega}$ contribution close to $(HW)^{-1}$, hence $\boldsymbol{m} \approx HW\boldsymbol{\omega}$. We get: $\mathbb{E}_{\boldsymbol{z}}[\sum_{i,j}\boldsymbol{m}_{c,i,j}|\boldsymbol{t}] \approx HW\mathbb{E}_{\boldsymbol{z}}[\sum_{i,j}\boldsymbol{\omega}_{c,i,j}|\boldsymbol{t}] = HW\boldsymbol{t}_c$ since $\sum_{i,j}\boldsymbol{\omega}_{c,i,j} = \boldsymbol{t}_c$ by construction. Hence, the proportions of the generated soft maps are close to the target palette in average. If $\mathcal{L}_{\mathrm{ENT}}$ is low as well, these generated soft maps will be, in addition, close to one-hot distributions at each pixel, and this average compliance with the target palette extends from the soft maps to the final layouts.

Adversarial objective. We train our generator to produce realistic layouts by trying to make it fool a discriminator which is jointly trained to distinguish real and generated layouts. We use the Improved WGAN loss [9] as the adversarial objective. Note however that real layouts are *hard* masks (one-hot) while generated ones are *soft* (even if $\mathcal{L}_{\mathrm{ENT}}$ promotes, to some extent, layouts to be close to one-hot). This discrepancy may harm the training as the generator will put lots of efforts trying to output discrete masks as well. The solution adopted by SB-GAN [2] is to apply a Gumbel-softmax [17] function to the generated soft masks. At each spatial location during forward pass, it samples one semantic class out of the multinomial distribution given by the semantic probabilities. During the backward pass, it behaves as a differentiable approximator of this operator.

In practice, we observed that this solution results in noisy sampled masks and using an approximator is not ideal for efficiency of training. We propose instead to tackle this issue the other way around, *i.e.*, by softening the real layouts.

[2]Details on the direct matching loss objective can be found in the Supplementary Material.

We apply a Gaussian filter to them, with a variance adapted to the image resolution. To ensure that the prevailing class at each pixel location remains the true one, we use *soft* semantic masks which are a weighted sum of blurred masks and original ones. We will show the merit of our soft-layout approach compared to Gumbel-softmax in the experiments.

3.3. Image generation and end-to-end training

We want to take advantage of our controllable layout generator in a complete pipeline where photo-realistic images are generated from produced layouts. To this end, we use GauGAN [24], a state-of-the-art layout-to-image translation network.[3] The layout generator and the image generator are first trained individually using the default training procedure presented respectively in ProGAN [13] and GauGAN [24]. Both can then be fine-tuned in an end-to-end fashion. By doing this, the layout generator benefits from additional supervision, while the image generator grows accustomed to being fed synthetic layouts which, in turn, improves the overall image quality. We use the end-to-end setup from SB-GAN [2] where an additional discriminator is trained to tell apart real images from synthetic ones generated from synthetic layouts.

4. Semantic Palette in action

4.1. Generating semantic codes

SB-GAN [2] directly maps noise vectors to layout-image pairs, whereas our conditional model also requires target semantic codes in inputs. To use it for systematic data generation, we thus need a means to sample suitable semantic codes: To this end, we propose a *palette generator* in the form of a Gaussian mixture model (GMM) fitted to a set of true semantic layouts, from which the GMM can capture multiple meaningful modes. Vectors sampled from this GMM are then projected onto the probability $C-$simplex so as to amount to proper semantic codes. Because the exact projection is slow to compute, in practice, we simply resort to the clipping of the sampled vectors to $[0,1]^C$ followed by L_1 normalization.

4.2. Partial editing of semantic layouts

We extend our conditional layout generation method to layout editing, with the aim to plausibly modify an input "real" layout by simply manipulating its semantic palette. To this end, the generator is now conditioned on both an input layout and the target proportions. Its output is a *partially edited* version of the input layout guided by the target proportions. To condition the generation on this additional input, we replace the ProGAN [13] convolutional blocks with the SPADE residual blocks from GauGAN [24].

[3] The choice of the base generative framework is orthogonal to our contributions and any improvement on it should increase the performance of our approach and the considered baselines.

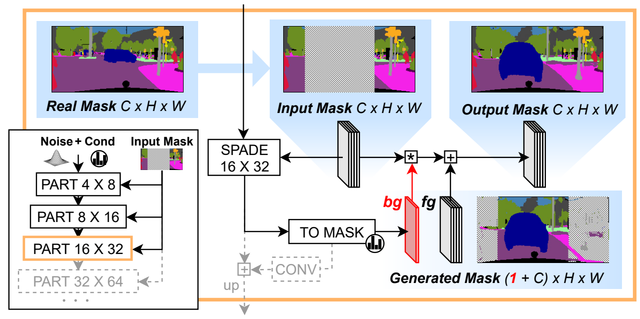

Figure 4: **Partial layout editing**. $\boxed{\text{SPADE } H \times W}$ replaces $\boxed{H \times W}$ from Figure 2. It is a SPADE [24] residual fusion block made of two convolutional layers with conditional batchnorm and ReLU activations. The generated partial layout is merged with the input one thanks to the extra *background* class ('bg').

Partial editing combined with conditional generation is a powerful tool as it facilitates data augmentation with higher fidelity to real data. It also provides controlled image editing capabilities as we will see in the Experiments section.

More specifically, we consider the task of replacing an arbitrary area of the input layout. During training, we randomly choose a rectangular patch to be replaced. The chosen patch is marked by setting all the class probabilities to $1/C$ at each pixel, while keeping the rest of the input layout as it is (C is the number of semantic classes). The "cropped" input mask is then passed to the generator through the SPADE [24] residual blocks, see Figure 4. The generator synthesizes an edited version of the input mask with the "cropped" part filled following the given target proportions for the crop. To this end, the generated mask has an additional *background* class whose proportion is also set by the semantic code so as to fill the "uncropped" part. We produce a coherent output mask by relying on this extra class to merge smoothly the generated mask to the input one. Specifically, the final output is computed as the sum of the generated mask (without background) and the input mask weighted by the background probabilities. The conditional loss $\mathcal{L}_{\mathrm{COND}}$ is applied to the generated mask while the adversarial loss is used on the output mask.

5. Experiments

Datasets. Evaluation is done on three urban datasets:
– Cityscapes [5] is composed of 2,975 training and 500 validation scenes taken in German suburbs. All images are annotated with 33 semantic classes.
– Cityscapes-25k [29] extends Cityscapes with 19,998 extra training scenes annotated by a pretrained state-of-the-art model. Note that only 19 classes out of the 35 original ones are effectively annotated in these additional 20K scenes.
– Indian Driving Dataset (IDD) [30] contains 6,993 training

Method	Layout KL ↓	Image FID ↓	GAN-test mIoU*	GAN-test mIoU	GAN-train mIoU*	GAN-train mIoU
Baseline 1	1.17	69.2	33.7	42.8	29.6	38.5
Baseline 2	0.32	69.0	35.3	46.9	30.2	39.4
Sem. Palette	**0.07**	60.7	34.6	45.7	30.6	40.1
Sem. Palette $_{\text{e2e}}$	0.08	**51.0**	**36.8**	**48.6**	**33.3**	**44.5**
Oracle	-	28.2	-	-	36.9	48.1

Table 1: **Conditional layout synthesis on Cityscapes**. "Oracle": real data for FID and for training segmentator in GAN-train metric; "e2e": end-to-end fine-tuning; "↓": smaller is better.

and 981 validation scenes, with 35 semantic classes.

Metrics. We use the following metrics:
– Kullback–Leibler (KL) divergence between target class proportions and generated ones: It measures how well the generator respects the semantic codes.
– Fréchet Segmentation Distance (FSD) [3]: It assesses how the overall statistics of real and synthetic layouts differ; We use real layouts from the training set.
– Fréchet Inception Distance (FID) [10]: It is an approximate measure of generated image quality; We compute FID w.r.t. real images from the validation set.
– GAN-test [26]: We use a segmenter pretrained on real data to yield predictions for generated images. We then report the mean Intersection-over-Union both on standard official classes (mIoU) and on all classes (mIoU*).
– GAN-train [26], the opposite of GAN-test: the segmenter is trained on generated data and tested on the real validation set. Though all metrics are interesting, GAN-train better assesses the overall utility of the generated data.

Implementation details. Generators are trained using ADAM [16]. For segmentation, we train a DeeplabV3 [4] model with Stochastic Gradient Descent, 0.01 initial lr, 0.9 momentum, 5×10^{-4} weight decay, in 300 epochs with batch-size 16. In all experiments, we generate layouts and images up to resolution 128×256. Please see the Supplementary Material for details of the palette generator.

5.1. Conditional layout generation

Comparison to conditional baselines. We compare Semantic Palette with two straightforward conditional layout generation baselines, *baseline 1* and *baseline 2*. Both accept semantic code as input. To get layout predictions respecting target class proportions, *baseline 1* directly penalizes unsatisfying outputs via a matching loss, and *baseline 2* leverages a conditional discriminator similar to cGAN [22]. Note that the same fixed pretrained image synthesizer is used for all.

Results are reported in Table 1 on the Cityscapes dataset using the 4 metrics previously introduced. The direct matching method, *baseline 1*, fails to reconstruct the semantic code: the KL value of 1.17 is nearly as bad as of random guesses made on the ground-truth semantic distribution (1.25). Though better on KL, *baseline 2* produces images with FID comparable to *baseline 1*. Semantic Palette

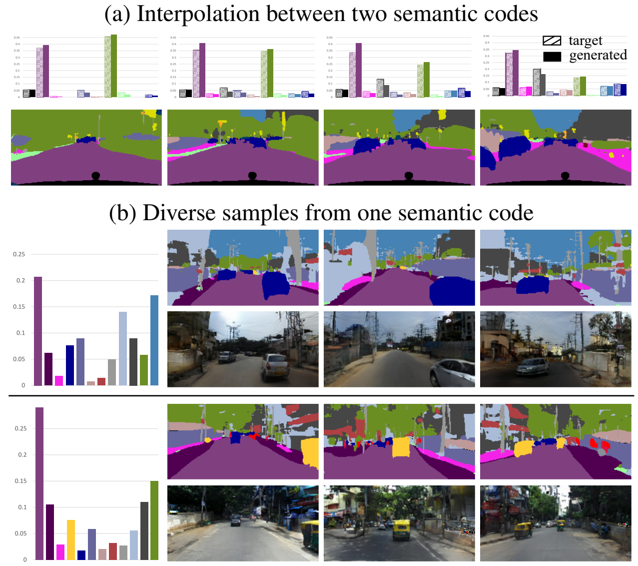

Figure 5: **Conditional layout-and-scene generation.** (*a*) For two semantic codes (left-/right-most) and interpolations between them; Class histograms in generated scenes (solid) closely follow target ones (dashed). (*b*) Two examples (top/bottom) of various layout-scene pairs sampled from the same semantic code (left).

improves on all metrics. Especially, we observe significant drops in KL and FID values, meaning that our conditional framework not only better respects the input semantic code, but also produces more realistic layouts.[4] While we see a slight drop compared to *baseline 2* on GAN-test, more importantly, the GAN-train performance improves along with KL and FID. The best performance is reached for Semantic Palette$_{e2e}$ after fine-tuning the layout generator and the image generator end-to-end. By doing this, the image generator grows accustomed to synthetic layouts while the layout generator benefits from additional supervision. Figure 5-(a) illustrates some qualitative results. Our layout generator clearly follows input semantic codes. We provide additional ablation studies on architecture choices in Section 5.3

Comparison to unconditional baselines. We follow [2] and report performance on both Cityscapes and Cityscapes-25k datasets. We additionally evaluate our method on the IDD dataset, to account for a very different urban landscape. On Cityscapes-25k, missing labels in the 20K extra images deteriorate performance of the 16 missing classes, resulting in lower mIoU* as compared to Cityscapes-trained models.

We compare Semantic Palette to unconditional baselines on image-layout pairs generation (Table 2). We note that SB-GAN [2] does better than PGAN-CGAN [11] thanks to the improved image generator. We train these base-

[4]We note that, because the same fixed image synthesizer is used in all experiments, a low FID score is a proxy indicator of layout quality. Indeed, the image synthesizer is pretrained on real layout-scene pairs; the model is thus used to real layout inputs. The closer generated layouts are to the real distribution, the better the synthesized images are.

Method	(a) Cityscapes Layout FSD ↓	Image FID ↓	GAN-test mIoU*	GAN-test mIoU	GAN-train mIoU*	GAN-train mIoU	(b) Cityscapes-25k Layout FSD ↓	Image FID ↓	GAN-test mIoU*	GAN-test mIoU	GAN-train mIoU*	GAN-train mIoU	(c) IDD Layout FSD ↓	Image FID ↓	GAN-test mIoU*	GAN-test mIoU	GAN-train mIoU*	GAN-train mIoU
PCGAN [11]	63.8	85.7	30.4	39.0	28.2	35.7	161.7	62.6	20.3	34.9	16.7	31.7	104.5	53.7	30.8	39.7	25.0	32.4
SB-GAN [2]	63.8	71.0	31.8	41.2	28.8	37.2	161.7	59.9	20.7	36.8	17.6	34.1	104.5	46.7	**32.1**	**41.5**	26.0	33.7
Sem. Palette	**25.3**	**60.7**	**34.6**	**45.7**	**30.6**	**40.1**	**37.8**	**56.3**	**26.8**	**46.3**	**22.4**	**43.7**	**60.0**	**43.5**	31.1	40.2	**27.0**	**35.0**
SB-GAN$_{e2e}$ [2]	20.4	61.8	34.5	44.7	29.6	37.0	148.5	55.1	**28.1**	42.9	24.3	41.8	116.0	44.8	31.7	41.0	27.4	35.6
Sem. Palette$_{e2e}$	**11.8**	**51.0**	**36.8**	**48.6**	**33.3**	**44.5**	**61.3**	**52.8**	27.1	**43.9**	**24.7**	**45.1**	**40.2**	**43.2**	**32.3**	**41.7**	**27.7**	**35.9**
Oracle	-	28.2	-	-	36.9	48.1	-	30.9	-	-	36.5	53.0	-	26.3	-	-	33.8	43.8

Table 2: **Comparison to unconditional GANs**. Same notations as in Table 1.

Data	Method	(a) Cityscapes mIoU*	(a) Cityscapes mIoU	(b) Cityscapes-25k mIoU*	(b) Cityscapes-25k mIoU	(c) IDD mIoU*	(c) IDD mIoU
Real	Baseline	36.9	48.1	36.5	53.0	33.8	43.8
Real + Semi-Syn	GauGAN [24]	37.2 $_{\uparrow 0.3}$	48.2 $_{\uparrow 0.1}$	**43.0** $_{\uparrow 6.5}$	58.0 $_{\uparrow 5.0}$	33.6 $_{\downarrow 0.2}$	43.5 $_{\downarrow 0.3}$
Real + Syn	SB-GAN [2]	34.6 $_{\downarrow 2.3}$	45.5 $_{\downarrow 2.6}$	35.5 $_{\downarrow 1.0}$	51.4 $_{\downarrow 1.6}$	33.5 $_{\downarrow 0.3}$	43.4 $_{\downarrow 0.4}$
	Sem. Palette	38.0 $_{\uparrow 1.1}$	49.4 $_{\uparrow 1.3}$	36.9 $_{\uparrow 0.4}$	54.4 $_{\uparrow 1.4}$	33.8 $_{-}$	43.8 $_{-}$
	Sem. Palette (DA)	38.6 $_{\uparrow 1.7}$	51.6 $_{\uparrow 3.5}$	38.6 $_{\uparrow 2.1}$	57.3 $_{\uparrow 4.3}$	34.5 $_{\uparrow 0.7}$	44.7 $_{\uparrow 0.9}$
	Sem. Palette (Part.)	**40.7** $_{\uparrow 3.8}$	51.9 $_{\uparrow 3.8}$	42.4 $_{\uparrow 5.9}$	59.1 $_{\uparrow 6.1}$	**35.6** $_{\uparrow 1.8}$	**46.1** $_{\uparrow 2.3}$
	Sem. Palette (Part. + DA)	**40.7** $_{\uparrow 3.8}$	**52.6** $_{\uparrow 4.5}$	42.5 $_{\uparrow 6.0}$	**60.5** $_{\uparrow 7.5}$	35.3 $_{\uparrow 1.5}$	45.8 $_{\uparrow 2.0}$

Table 3: **Data-augmentation** grouped by training data regime, tested on real data. "DA": domain adaptation; "Part.": partial editing.

lines from scratch, nonetheless, results for SB-GAN [2] on Cityscapes are in line with the ones from the original paper.

Our method outperforms the baselines in both diversity of semantic content and image quality with a clear gap in FSD, FID and GAN-train. We facilitate the layout synthesis task by guiding explicitly the generation with semantic proportions, partially lifting the burden of figuring out the scene composition. We show pairs synthesized from a single target histogram on IDD in Figure 5-(b). Though sharing the same semantic palette, diverse scenes are produced.

5.2. Data augmentation

Once trained, the image and layout generators can be used to sample new pairs, hence *augmenting* the real training data. Different from standard data augmentation techniques, which only modify existing data points, synthetic models create new data points, which allows not only altering the visual appearance in the image space, but also applying structural changes in the layout space. We consider two different data augmentation setups: (i) "Semi-Syn", which only relies on the pretrained image generator to synthesize images from ground-truth layouts, and (ii) "Syn", which uses both generators to synthesize new data pairs. Table 3 shows test performance of segmenters trained only on "Real", "Real + Semi-Syn", or "'Real + Syn" data.

Real + Semi-Syn. A pretrained GauGAN [24] is used as image generator. On Cityscapes and IDD datasets, we only observe marginal changes in performance compared to the baseline. However, when having more layouts to feed the image generator in Cityscapes-25k, the segmenter trained on augmented data significantly outperforms the baseline. We conjecture that there is a trade-off between the quality of synthesized images and the diversity of semantic layouts: if

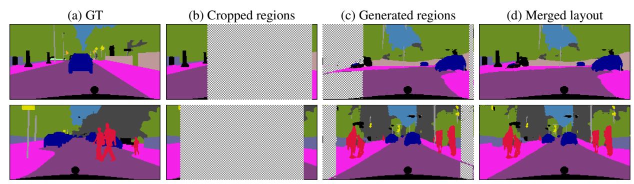

Figure 6: **Partial editing of layouts.** The procedure consists in cropping ground-truth layouts and then synthesizing new objects within the cropped area, guided by the initial semantic proportions.

layouts cannot provide enough diversity to counter-balance the loss of image quality, this may harm the performance.

Real + Syn. To further highlight the benefit of layout generators, we do not use the end-to-end models. The same pretrained GauGAN is used as in the "Real + Semi-Syn" setup. On the three benchmarks, with SB-GAN [2] as layout generator, we observe drops in mIoU as compared to the baseline. The unconditional model shows its limitations in the data augmentation context where it fails to complement real data with more diverse samples, resulting in negative results. In contrast, Semantic Palette consistently improves upon baselines, except for IDD dataset where the performance is unchanged. These results demonstrate the merits of our pipeline for data augmentation.

We propose several variants of our method to further push its performance. First, to alleviate distribution gaps between synthetic and real data, we adopt AdvEnt [32], a domain adaptation technique for semantic segmentation.[5] This strategy is used to ensure synthetic and real supervisions are consistent. This domain adaptation (DA) technique boosts further the performance of our approach (Table 3). Second, we test a variant using the partial layout editing method presented in Section 4.2 and illustrated in Fig-

[5] See the Supplementary Material for implementation details.

Residual Fusion	Multi Scale	Soft GT Masks	Palette Gen.	Layout KL ↓	Layout FSD ↓	Image FID ↓
				0.32	33.9	70.6
✓				0.13	**23.4**	65.5
✓	✓			0.11	37.1	63.3
✓	✓	✓		**0.03**	24.1	64.3
✓	✓	✓	✓	0.07	25.3	**60.7**

Table 4: **Semantic Palette ablation on Cityscapes.** First row: model with only SAA; In models with "Soft GT Masks" unmarked, Gumbel-softmax is used; If "Palette Gen." is unmarked, ground-truth codes are used instead of generated ones.

ure 6. The ensuing performance (Table 3) demonstrates the clear benefit of leveraging extra real information in the generation process, *i.e.*, partial areas and semantic proportions. A straightforward combination of the two proposed strategies achieves the best performance on two benchmarks.

In the Supplementary Material, we provide further results and discuss the additional use of standard data augmentation during training.

5.3. Ablation studies

We report the results of an ablation study in Table 4. Using residual fusion significantly decreases KL, FSD, and FID values, highlighting the benefit of leveraging lower-scale information. We then achieve further improvements in KL and FID with multi-scale training. Using a strategy based on soft ground-truth masks, as detailed in Section 3.2, instead of Gumbel-softmax improves KL score by a large margin while preserving comparable FID scores. Our final model using the palette generator presented in Section 4.1 achieves the best FID, with a slightly worse KL score compared to the model using ground-truth semantic codes.

5.4. Face editing

We showcase the new editing capabilities offered by the combination of conditional and partial layout generation on face images, using the CelebAMask-HQ dataset [13, 18, 20]. For the image synthesis, we use a pretrained SEAN [34] model, an upgrade of GauGAN [24] where one can fix independently the style of individual semantic classes. We use it to maintain the content while editing the semantic structure. Our method, illustrated in Figure 7, allows one to adjust semantic attributes by a chosen amount with realistic details. This is achieved by simply modifying class proportions, avoiding the tedious task of direct manual editing of the original face layout. For this task, we do not crop the original layouts but allocate some budget for semantic additions. When target content is already present in the original layout, the generator will be inclined to replicate the original content as it fully satisfies both conditional and adversarial objectives; *e.g.*, to increase the amount of hair, it will copy the existing hair as long as the proportion matches since it is the definition of realism for the discriminator. To counter this undesired behaviour, we introduce a

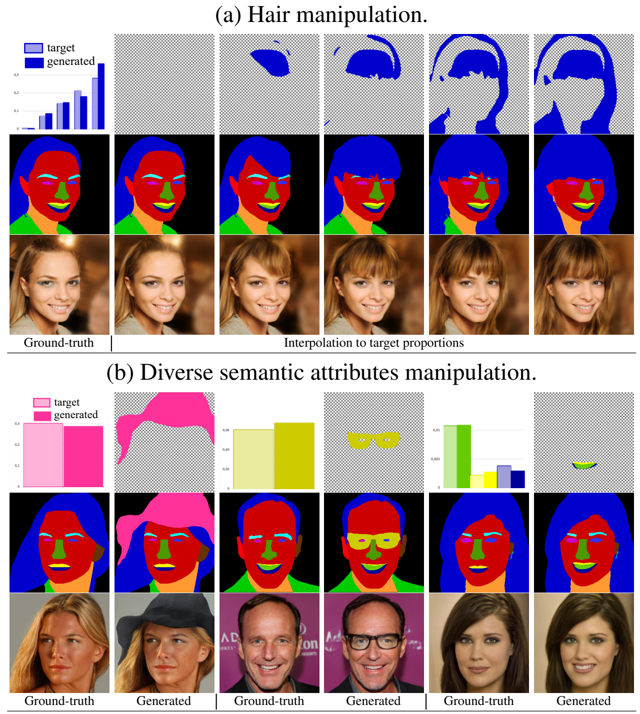

Figure 7: **Application of Semantic Palette to face editing** at resolution 256×256. In (a), we illustrate the fine-controlled editing of layouts by gradually increasing the budget for the hair. Edits are convincing both in the layout and image spaces. Thanks to the novelty loss, there is little overlap between original and additional hair. In (b), we show the editing of diverse semantic attributes. Although we have a unique layout generator, we can perform very different edits. Moreover, one can play with latent codes to generate various edits for the same proportion of semantic attributes.

novelty loss that encourages edits to be different from original semantic classes (details in Supplementary Material).

6. Conclusion

We have proposed the Semantic Palette, a new framework for scene generation, and editing, guided by semantic proportions. Using novel architecture designs and learning objectives – semantically assisted activation and residual conditional fusion coupled with novel conditional losses –, it generates plausible scene layouts with class proportions close to target ones, which then translate into realistic images. Experiments assess the superior quality of the generated layout-image pairs as well as their utility for downstream-task training: used in particular to augment original real-data set, they deliver performance gain in semantic segmentation.

Acknowledgements. Part of this work was done using HPC resources from GENCI–IDRIS (Grant 2020-AD011012227). We would like to thank Jean Ponce for useful comments.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. In *ICML*, 2017. 1

[2] Samaneh Azadi, Michael Tschannen, Eric Tzeng, Sylvain Gelly, Trevor Darrell, and Mario Lucic. Semantic bottleneck scene generation. *arXiv preprint arXiv:1911.11357*, 2019. 1, 2, 4, 5, 6, 7

[3] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Seeing what a gan cannot generate. In *ICCV*, 2019. 6

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 6

[5] 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. 5

[6] Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *NeurIPS*, 2017. 2

[7] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017. 2

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 1

[9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. In *NeurIPS*, 2017. 4

[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NeurIPS*, 2017. 6

[11] Jonathan Howe, Kyle Pula, and Aaron A Reite. Conditional generative adversarial networks for data augmentation and adaptation in remotely sensed imagery. In *Applications of Machine Learning*, 2019. 1, 2, 6, 7

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

[13] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1, 2, 5, 8

[14] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1

[15] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of styleGAN. In *CVPR*, 2020. 1

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[17] Matt J Kusner and José Miguel Hernández-Lobato. GANs for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016. 4

[18] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. MaskGan: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020. 8

[19] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, and Hongsheng Li. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. In *NeurIPS*, 2019. 2

[20] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 8

[21] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 1

[22] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1, 2, 4, 6

[23] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017. 2

[24] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 1, 2, 5, 7, 8

[25] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 2

[26] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my GAN? In *ECCV*, 2018. 6

[27] Richard Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 1967. 3

[28] Hao Tang, Xiaojuan Qi, Dan Xu, Philip HS Torr, and Nicu Sebe. Edge guided GANs with semantic preserving for semantic image synthesis. *arXiv preprint arXiv:2003.13898*, 2020. 2

[29] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*, 2020. 5

[30] Girish Varma, Anbumani Subramanian, Anoop Namboodiri, Manmohan Chandraker, and CV Jawahar. IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments. In *WACV*, 2019. 5

[31] Anna Volokitin, Ender Konukoglu, and Luc Van Gool. Decomposing image generation into layout prediction and conditional synthesis. In *CVPRw*, 2020. 2

[32] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. AdvEnt: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *CVPR*, 2019. 7

[33] 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. 1, 2

[34] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. SEAN: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020. 8