# Semantic Palette: Guiding Scene Generation with Class Proportions
## *Supplementary Material*

This document provides additional insights into the Semantic Palette, details on its use and on the baselines, and qualitative results. Please note that, in the following, we rely heavily on notations that are introduced in the main body of the paper.

## A. Connection to Sinkhorn algorithm

To carry on the discussion initiated in Section 3.1 of the paper, we here elaborate on the connection between our SAA module and the Sinkhorn algorithm [11], viewing SAA through the lens of optimal transport [3].

Given an initial blank "canvas" having $N = HW$ pixels, we define a uniform source histogram $\boldsymbol{r} = N^{-1}\mathbf{1}_N$, standing for the equal chance of each pixel to be "drawn" or occupied by one of the classes. The target histogram, or semantic palette, $\boldsymbol{t} \in \mathbb{R}_+^C$, defines the prescribed "budget" for the $C$ classes. One can defined the set of admissible *transport plans* from one distribution to the other one:

$$U(\boldsymbol{r}, \boldsymbol{t}) := \{\boldsymbol{P} \in \mathbb{R}_+^{C\times N} | \boldsymbol{P}\mathbf{1}_N = \boldsymbol{t}, \boldsymbol{P}^\top \mathbf{1}_C = \boldsymbol{r}\}. \quad (1)$$

A connection of the soft mask $\boldsymbol{m}$ with these transport plans is established as follows. Flattening spatial dimensions ($H \times W \rightarrow N$), the soft mask $\boldsymbol{m}$ is now in $[0,1]^{C\times N}$, and it is expected to simultaneously verify:

$$N^{-1}\boldsymbol{m}\mathbf{1}_N = \boldsymbol{t}\,, \quad (2)$$

$$N^{-1}\boldsymbol{m}^\top \mathbf{1}_C = \boldsymbol{r}\,, \quad (3)$$

where (2) warrants that soft pixel-to-class assignments respect the input class proportions ($\frac{1}{N}\sum_n \boldsymbol{m}_{c,n} = \boldsymbol{t}_c$) and (3) ensures that at each pixel location there is a valid class distribution ($\sum_c \boldsymbol{m}_{c,n} = 1$). If $\boldsymbol{m}$ verifies both, then $N^{-1}\boldsymbol{m} \in U(\boldsymbol{r}, \boldsymbol{t})$. Note that, in practice, only (3) is a hard constraint.

We can now formulate the task of finding $\frac{1}{N}\boldsymbol{m}$ as solving an entropy-regularized optimal-transport problem [3]:

$$\boldsymbol{P}^* = \underset{\boldsymbol{P}\in U(\boldsymbol{r},\boldsymbol{t})}{\operatorname{argmin}} \langle \boldsymbol{P}, \boldsymbol{K}\rangle - \frac{1}{\lambda} h(\boldsymbol{P}), \quad (4)$$

where $\boldsymbol{K} \in \mathbb{R}^{C\times N}$ is a suitable transport-cost matrix, $h(\boldsymbol{P})$ is the entropy of $\boldsymbol{P}$, $\lambda$ is a weight (fixed as 1 next) and $\langle\,,\rangle$ denotes the Frobenius dot-product.

In the SAA module, the cost matrix $\boldsymbol{K}$ is defined as $-\boldsymbol{f}$. Intuitively $\boldsymbol{f}$, the "raw" output of our network, indicates the initial class preference of each pixel $i$; its opposite $-\boldsymbol{f}$ can be seen as the transportation cost, *i.e.*, the higher the chance to assign pixel $i$ to class $c$, the lower the cost to "transport" from pixel $i$ to class $c$ is.

To find the optimal plan $\boldsymbol{P}^*$, one can adopt the Sinkhorn algorithm, initializing $\boldsymbol{P}$ as $\exp(-\boldsymbol{K}) = \exp(\boldsymbol{f})$ and alternating row-wise and column-wise normalization/scaling steps [3]:

$$\boldsymbol{P} \leftarrow \operatorname{diag}\big[\boldsymbol{t} \oslash (\boldsymbol{P}\mathbf{1}_N)\big]\boldsymbol{P}\,, \quad (5)$$

$$\boldsymbol{P} \leftarrow \boldsymbol{P}\operatorname{diag}\big[\boldsymbol{r} \oslash (\boldsymbol{P}^\top \mathbf{1}_C)\big]\,, \quad (6)$$

where $\oslash$ denotes the Hadamard entry-wise division. Eq. 5 amounts to successively normalizing each of the $C$ rows and then multiplying each by its target probability in $\boldsymbol{t}$ – exactly how $\boldsymbol{\omega}$ is derived from $\boldsymbol{f}$; Eq. 6 amounts to normalizing each of the $N$ columns – exactly how $\boldsymbol{m}$ is derived from $\boldsymbol{\omega}$ (since $\boldsymbol{m}$ corresponds to $N\boldsymbol{P}$).

Effectively, the steps of the SAA presented in Section 3.1 of the paper correspond to a single step of this Sinkhorn algorithm. Having more steps is possible, yet we opted to a single one as to allow certain slacks in the final scene composition, *i.e.*, not forcing an exact matching to the input semantic palette.

## B. Direct matching loss in Baseline 1

The *baseline 1* introduced in Section 5.1 uses a direct matching loss to enforce conditioning constraints. We provide here the detail of this loss.

The conditional layout generator $G$ produces semantic soft probability masks $\boldsymbol{m} \in [0,1]^{C\times H\times W}$. Let us define $\phi : [0,1]^{C\times H\times W} \rightarrow \Delta^C$ the function that computes the class histogram of the final semantic map derived from soft mask $\boldsymbol{m}$, where $\Delta^C := \{\boldsymbol{x} \in \mathbb{R}_+^C : \boldsymbol{x}^\top \mathbf{1}_C = 1\}$ is the probability simplex. For each class $c \in [\![1, C]\!]$, the proportion of pixels assigned to this class in the image is given by:

$$\phi_c(\boldsymbol{m}) = \frac{1}{HW}\sum_{(i,j)\in\Omega} [\operatorname*{argmax}_k \boldsymbol{m}_{k,i,j} = c]. \quad (7)$$

This function $\phi$ being non-differentiable, it cannot be easily used to define a training loss. Instead, we propose to use $\widehat{\phi}$, a differentiable *soft* estimation of the semantic histogram, defined as:

$$\widehat{\phi}_c(\boldsymbol{m}) = \frac{1}{HW} \sum_{(i,j)\in\Omega} \boldsymbol{m}_{c,i,j}, \tag{8}$$

for each $c \in [\![1, C]\!]$. The matching loss in *baseline 1* is finally defined as the KL-divergence between target and estimated histograms:

$$\mathcal{L}_{\text{MATCH}}(G) = \mathbb{E}_{(\boldsymbol{z},\boldsymbol{t})}\Big[ \sum_{c\in[\![1,C]\!]} \boldsymbol{t}_c \cdot \log\Big(\frac{\boldsymbol{t}_c}{\widehat{\phi}_c(G(\boldsymbol{z},\boldsymbol{t}))}\Big)\Big]. \tag{9}$$

## C. Domain adaptation for data augmentation

We explain here how AdvEnt, the domain-adaptation technique in [12], is mobilized in Section 5.2 when using both real and synthetic data. In effect, we adopt the main ingredient of AdvEnt: an adversarial training procedure to perform alignment on the so-called weighted self-information space. While the segmenter is trained as usual, an additional discriminator, taking segmenter's prediction as input, is trained in parallel to determine from which domain (real or synthetic) the prediction originates. Playing the adversarial game, the segmenter tries to fool the discriminator, eventually resulting in closing the domain gap.

Such a technique has been proven effective for unsupervised domain adaptation in semantic segmentation, where images are annotated only in one domain. We revisit it in a different context, where full annotations are available in both domains. Empirical results (Table 3 of the paper) demonstrate the benefit of addressing domain gap this way when using synthesized data for data augmentation. Since for DA we use the default hyper-parameters from [12], fine-tuning them might yield even higher performance.

## D. Novelty loss for face editing

In the case of partial editing, the conditional layout generator $G$ takes a semantic layout $\boldsymbol{l}$ as input, in addition to the noise $\boldsymbol{z}$ and the target palette $\boldsymbol{t}$. We denote $G(\boldsymbol{z}, \boldsymbol{t}, \boldsymbol{l}) = \boldsymbol{m}$ the final edited layout produced by the generator, after the generated partial layout and the input layout have been merged. For the face editing task, different from the partial-editing method proposed for data augmentation in urban scenes, the input layout is not cropped. In addition, we introduce a novelty loss on top of the conditional and adversarial losses, to ensure that the edits do modify the original content. It is defined by:

$$\mathcal{L}_{\text{NOV}}(G) = \frac{1}{|E|}\mathbb{E}_{(\boldsymbol{z},\boldsymbol{t},\boldsymbol{l})}\Big[ \sum_{(i,j)\in E} \sum_{c\in[\![1,C]\!]} \boldsymbol{l}_{c,i,j}\boldsymbol{m}_{c,i,j}\Big], \tag{10}$$

where $E \subset \Omega$ is the set of pixel locations where an edit has been made, *i.e.*, the dominant class in the partial layout is not the *background* class. This loss is, at every edited pixel location, the scalar product between the generated soft probability distribution and the input one-hot one, and therefore promotes orthogonal content between the two.

## E. Better base generative frameworks

In this work, we built the layout synthesizer upon ProGAN [5] as to guarantee a fair comparison to SBGAN [1] and to highlight the merits of the proposed architecture designs and learning objectives. We note that this part of the Semantic Palette's pipeline can leverage any other hierarchical GAN architecture, for example StyleGAN [6] or StyleGAN2 [7]. In fact, the choice of the base generative framework is orthogonal to our contributions and any improvement on it should increase the performance of both the Semantic Palette and the considered baselines. Similarly, for the image generation part, we adopted GauGAN [10] as done in SB-GAN [1] to ensure fair comparison while noting that the choice of this generator is orthogonal to our contributions; in particular, if using a different framework like CC-FPSE [9] was to bring improvements, they would benefit to all compared pipelines.

## F. Implementation details

**Weights for losses.** All the introduced losses are equally weighted in the experiments. However, a particular weighting may prove useful for specific applications such as to improve further the semantic control at the expense of a slight degradation of the image realism or the other way around.

**Layout synthesis model.** The layout generator is trained with ADAM [8], an initial learning rate of $10^{-3}$, $\beta = (0, 0.99)$ and specific epochs (600 to 150) and batch sizes (1024 to 8) for every resolution ($4\times8$ to $128\times256$).

**Image synthesis model.** The image generator is trained with ADAM [8], an initial learning rate of $2 \cdot 10^{-4}$, $\beta = (0.5, 0.999)$ for 200 epochs and a batch size of 8.

**Segmenter.** We train a DeeplabV3 [2] model with Stochastic Gradient Descent, an initial learning rate of $10^{-2}$, 0.9 momentum, $5\cdot10^{-4}$ weight decay, for 300 epochs and a batch size of 16.

**Palette generator.** The GMM model is trained on the semantic proportions from the real training dataset, using the expectation-maximization (EM) algorithm. The number of Gaussian components control the trade-off between approximation and generalization. We select their number using the Akaike Information Criterion (AIC), which balances these two objectives.

In practice, to ensure that the vectors sampled from the GMM are true proportions, *i.e.*, non-negative and $L_1$-normalized, one has to project them onto the probability

| | | | (a) Cityscapes | | (b) Cityscapes-25k | | (c) IDD | |
|---|---|---|---|---|---|---|---|---|
| Data | Method | Crop | mIoU* | mIoU | mIoU* | mIoU | mIoU* | mIoU |
| Real | Baseline | | 36.9 | 48.1 | 36.5 | 53.0 | 33.8 | 43.8 |
| | | ✓ | 35.7 $_{\downarrow 1.2}$ | 51.4 $_{\uparrow 3.3}$ | 35.5 $_{\downarrow 1.0}$ | 59.6 $_{\uparrow 6.6}$ | 32.7 $_{\downarrow 1.1}$ | 40.0 $_{\downarrow 3.8}$ |
| Real + Syn | Sem. Palette (DA) | | 38.6 | 51.6 | 38.6 | 57.3 | 34.5 | 44.7 |
| | | ✓ | 38.7 $_{\uparrow 0.1}$ | 52.2 $_{\uparrow 0.6}$ | 32.9 $_{\downarrow 5.7}$ | 57.0 $_{\downarrow 0.3}$ | 29.8 $_{\downarrow 4.7}$ | 38.5 $_{\downarrow 6.2}$ |
| | Sem. Palette (Part. + DA) | | **40.7** | 52.6 | **42.5** | **60.5** | **35.3** | **45.8** |
| | | ✓ | 39.8 $_{\downarrow 0.9}$ | **54.4** $_{\uparrow 1.8}$ | 37.0 $_{\downarrow 5.5}$ | 56.4 $_{\downarrow 4.1}$ | 31.1 $_{\downarrow 4.2}$ | 39.7 $_{\downarrow 6.1}$ |

Table 1: **Using cropping to augment real data.** Same notations as in Table 3 of the paper. In each group, models using cropping are compared against the ones without.

$C$–simplex. As the projection is not easy to compute analytically, one can get a good approximation with constrained minimization methods such as the trust-region constrained algorithm. However, their convergence is slow, making them impractical in our case. Instead, we chose to compute a rough estimate of the projection by first clipping the sampled vectors to $[0, 1]$ and then normalizing them.

## G. Standard data augmentation.

We used random horizontal flipping in all experiments done in the main paper.

Cropping is another standard data augmentation strategy used in semantic segmentation. We provide here an ablation study where we additionally perform cropping to augment real data while training the baseline and Semantic Palette models. We report in Table 1 the performance on the three benchmarks. In terms of mIoU, we observe only on Cityscapes that cropping helps improve all methods and achieves best scores when combined with our augmentation strategy; yet, on the other datasets, having cropping degrades the performance. In terms of mIoU*, the performance drops in most cases. These results reveal different behaviors in the three datasets when including cropping in the data augmentation procedure during training. We note that, as cropping is done only on real data, increase or decrease in performance by using it is orthogonal to our proposed framework. Overall, the best results are obtained using Semantic Palette.

## H. Additional experiments.

We provide results of a few additional experiments aimed at evaluating the Semantic Palette in different setups, namely with other types of data and at higher resolution.

**Effect of Semantic Palette on non-urban scenes.** To verify that the proposed method generalizes well to other types of natural images, we trained the Semantic Palette and the unconditional baselines on the ADE-Indoor dataset [13] at 128×256 resolution. The results, as shown in Table 2, confirm the advantage of our model over the unconditional baselines.

**Ability to scale to higher resolutions.** To afford an extensive evaluation of the proposed methods compared to the different baselines, all experiments were conducted at the 128×256 resolution. To evaluate the performance at higher resolution, we now compare the Semantic Palette to SB-GAN at the 256×512 resolution on Cityscapes. The results, in Table 3, turn out to be consistent with the ones reported at 128×256.

| Method | Layout | Image | GAN-test | GAN-train |
|---|---|---|---|---|
| | FSD ↓ | FID ↓ | mIoU | mIoU |
| PCGAN [4] | 211.7 | 96.6 | 11.1 | 6.8 |
| SB-GAN [1] | 211.7 | 93.7 | 12.2 | 7.0 |
| Sem. Palette | **76.1** | **88.4** | **20.1** | **10.5** |
| SB-GAN $_{e2e}$ [1] | 93.3 | 82.7 | 15.3 | 10.0 |
| Sem. Palette $_{e2e}$ | **19.0** | **76.5** | **21.4** | **11.7** |

Table 2: **Results on ADE-Indoor**.

| Method | Layout | Image | GAN-test | | GAN-train | |
|---|---|---|---|---|---|---|
| | FSD ↓ | FID ↓ | mIoU* | mIoU | mIoU* | mIoU |
| SB-GAN [1] | 66.0 | 74.8 | 34.9 | 46.0 | 28.0 | 35.7 |
| Sem. Palette | **32.4** | **66.7** | **38.0** | **50.0** | **32.1** | **42.3** |

Table 3: **Results on Cityscapes at resolution** 256×512.

## I. Qualitative results

We provide additional qualitative results of scene generation in Figures 1 and 2, and of face editing in Figures 3, 4 and 5. These figures are best viewed in color.

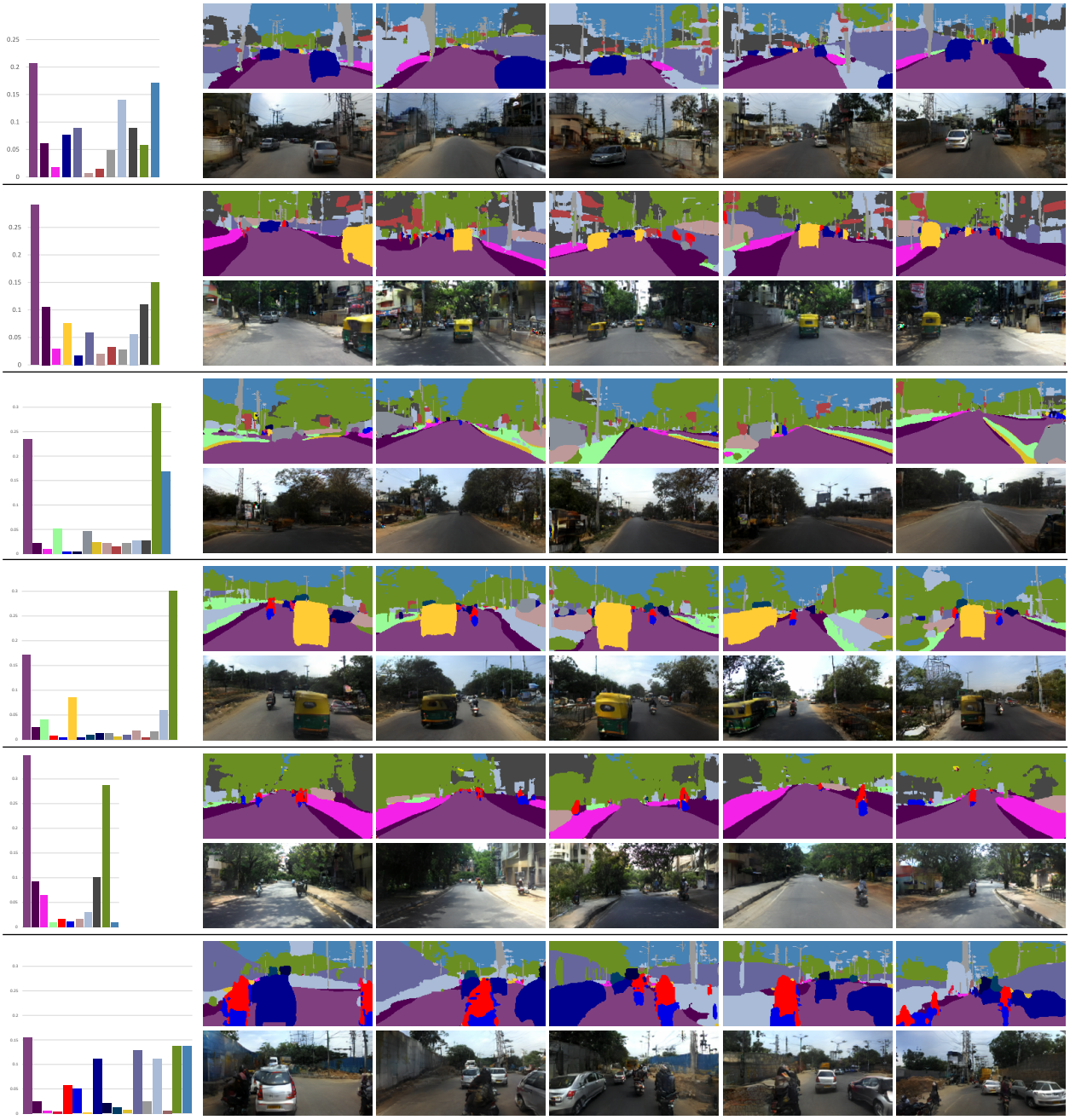

Figure 1: **Conditional layout-and-scene generation.** Various layout-scene pairs sampled from the same semantic code (left).

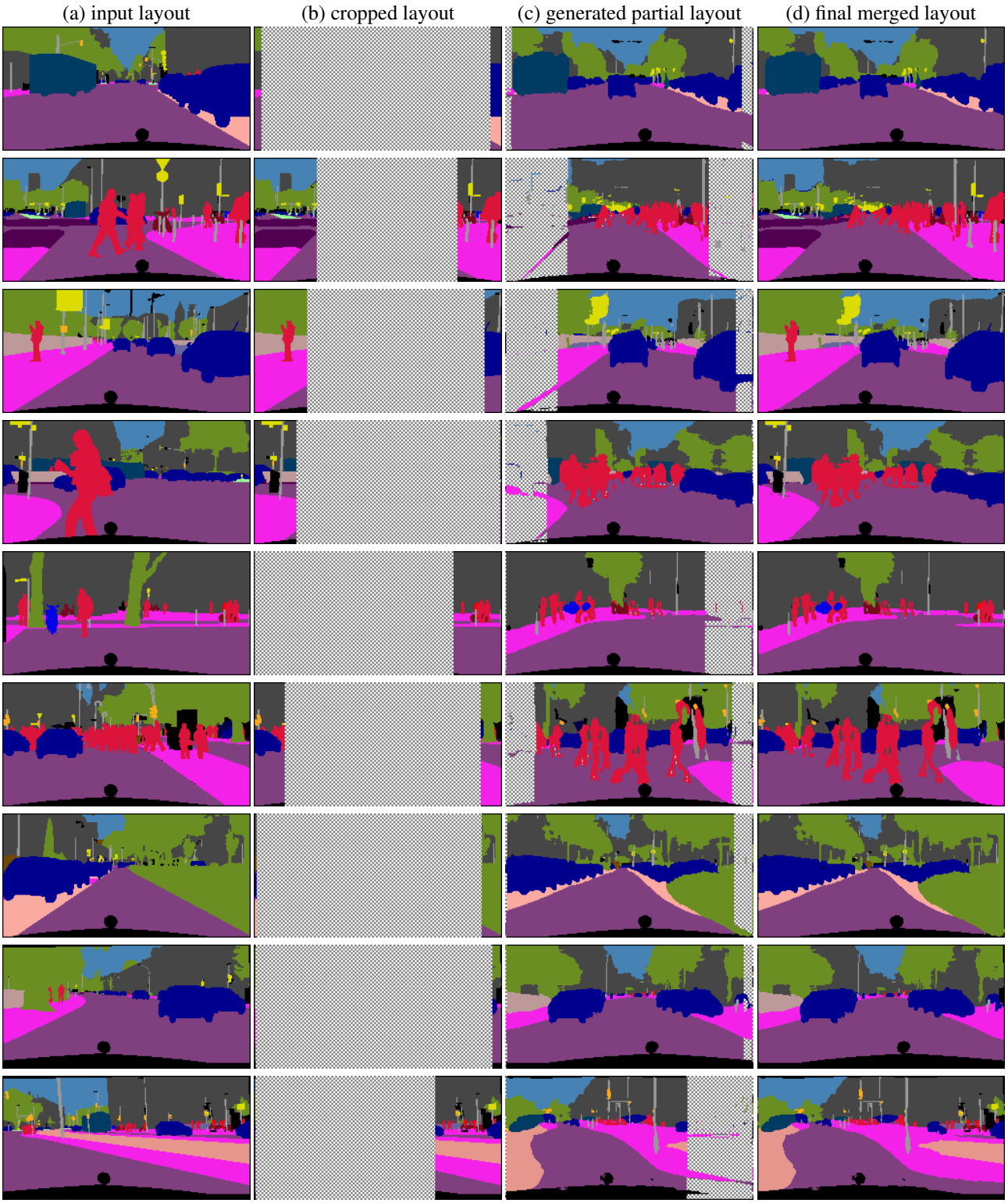


Figure 2: **Partial editing of layouts.** The procedure consists in cropping ground-truth layouts and then synthesizing new objects within the cropped area, guided by the initial semantic proportions.

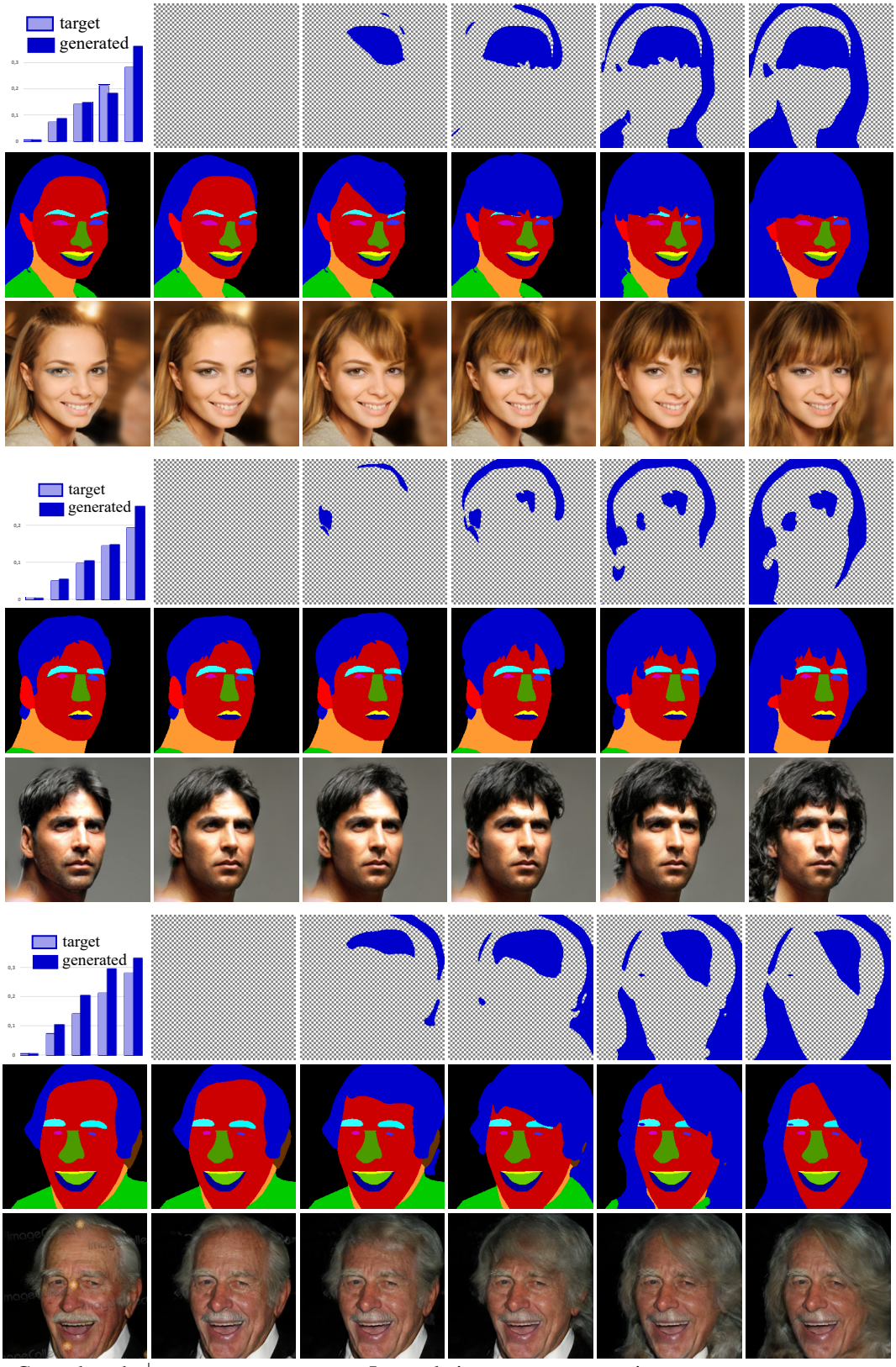


Figure 3: **Hair manipulation 1: Grow existing hair**. We select subjects with short hair and progressively increase the hair budget. The hair style corresponds to the input ground-truth image-layout pair. Please, zoom in for details.

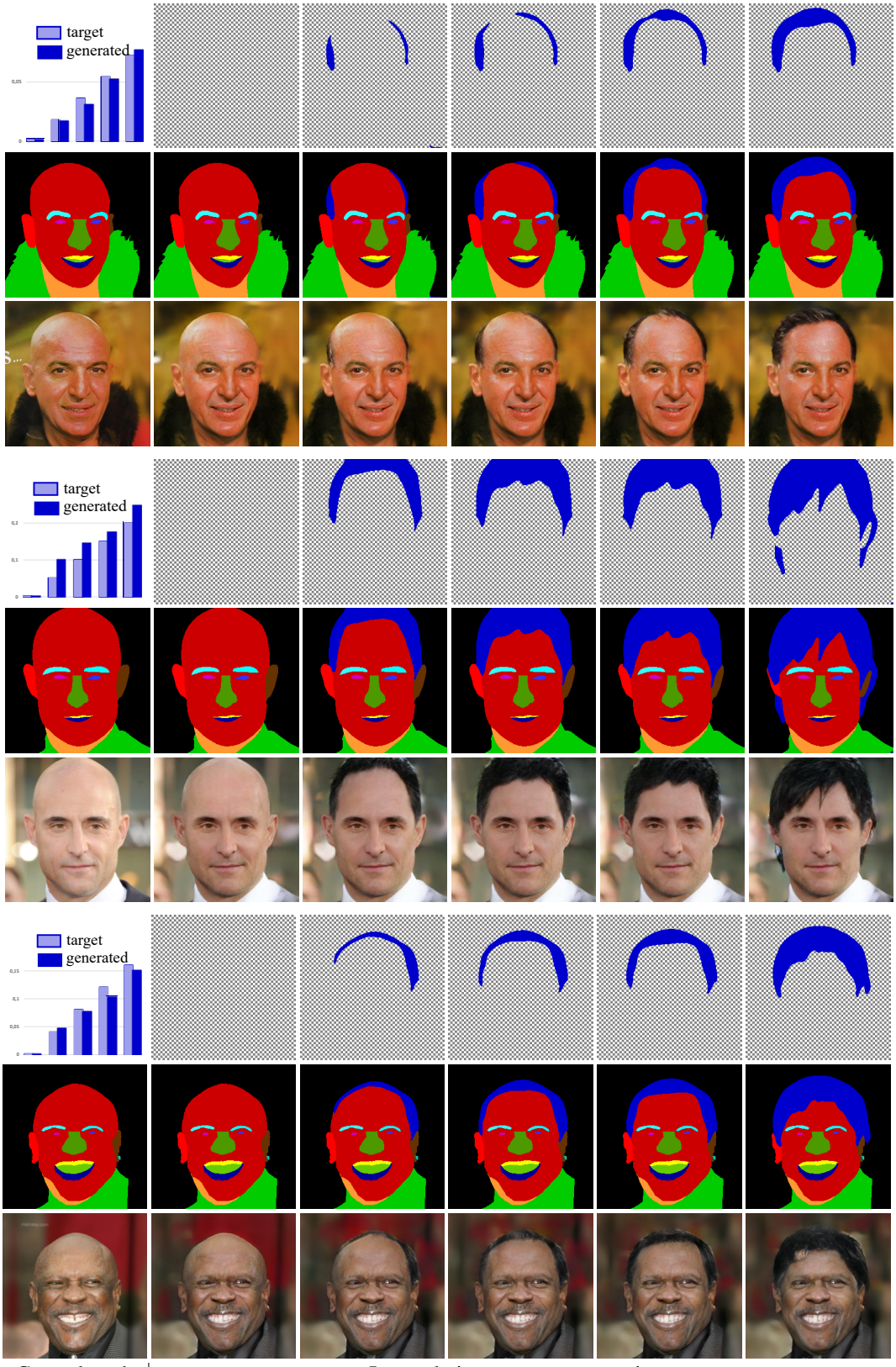


Figure 4: **Hair manipulation 2: Bald to not bald**. We select bald subjects and progressively increase the hair budget. Since there is no hair initially, the hair style is randomly burrowed from another subject in the training set. Please, zoom in for details.

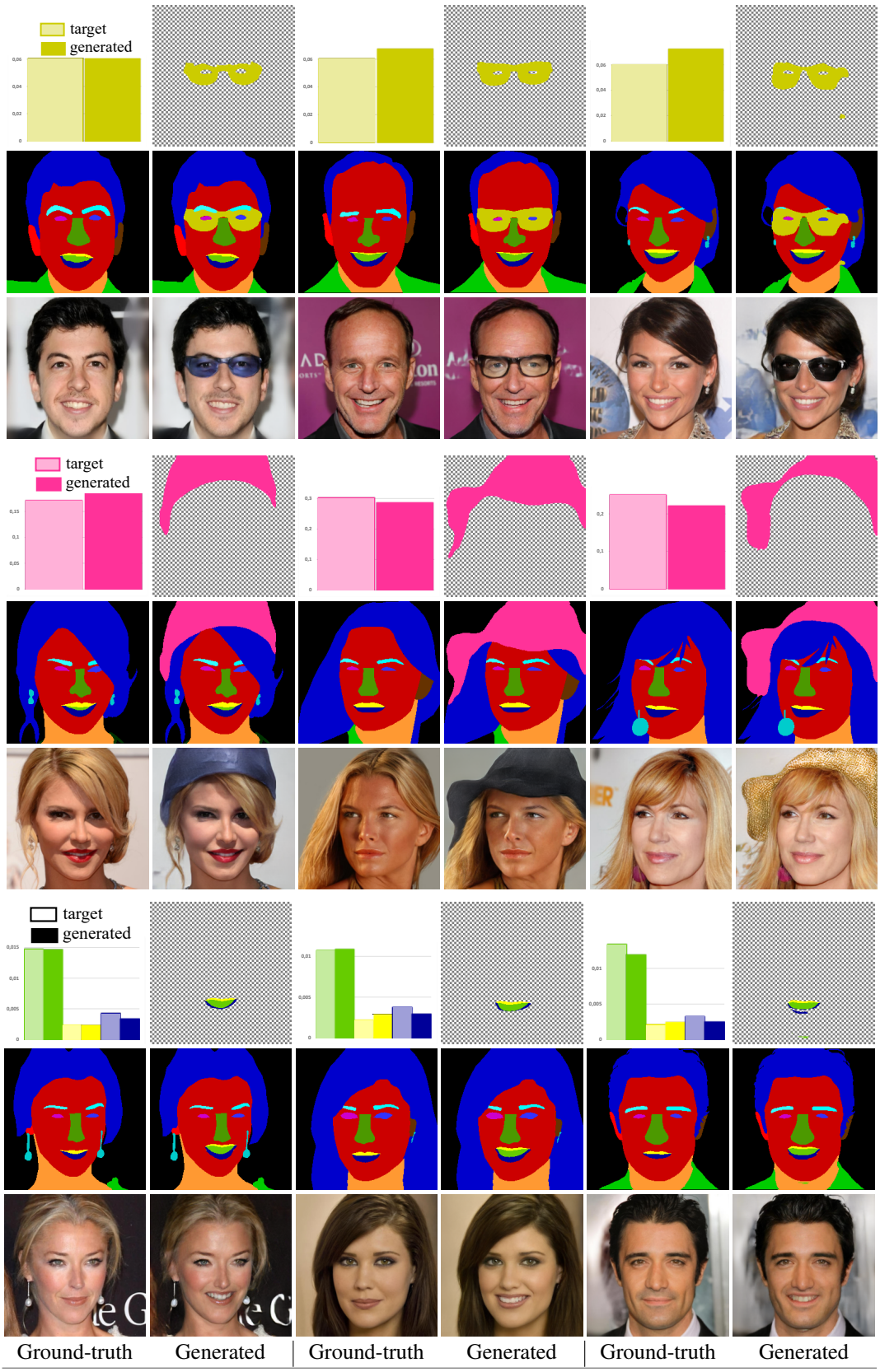


Figure 5: **Manipulation of diverse semantic attributes.** Glasses (1st row), hat (2nd), teeth (3rd). Please, zoom in for details.

## References


[1] Samaneh Azadi, Michael Tschannen, Eric Tzeng, Sylvain Gelly, Trevor Darrell, and Mario Lucic. Semantic bottleneck scene generation. *arXiv preprint arXiv:1911.11357*, 2019. 2, 3

[2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2

[3] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013. 1

[4] 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. 3

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

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2

[7] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of styleGAN. In *CVPR*, 2020. 2

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[9] 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

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

[11] Richard Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 1967. 1

[12] 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. 2

[13] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20k dataset. In *CVPR*, 2017. 3