Semantic Image Synthesis with Spatially-Adaptive Normalization

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu

UC Berkeley, NVIDIA, and MIT CSAIL

CVPR 2019

Presenter: Minho Park

Semantic Image Synthesis

- Demo: http://gaugan.org/gaugan2/ (GauGAN2)

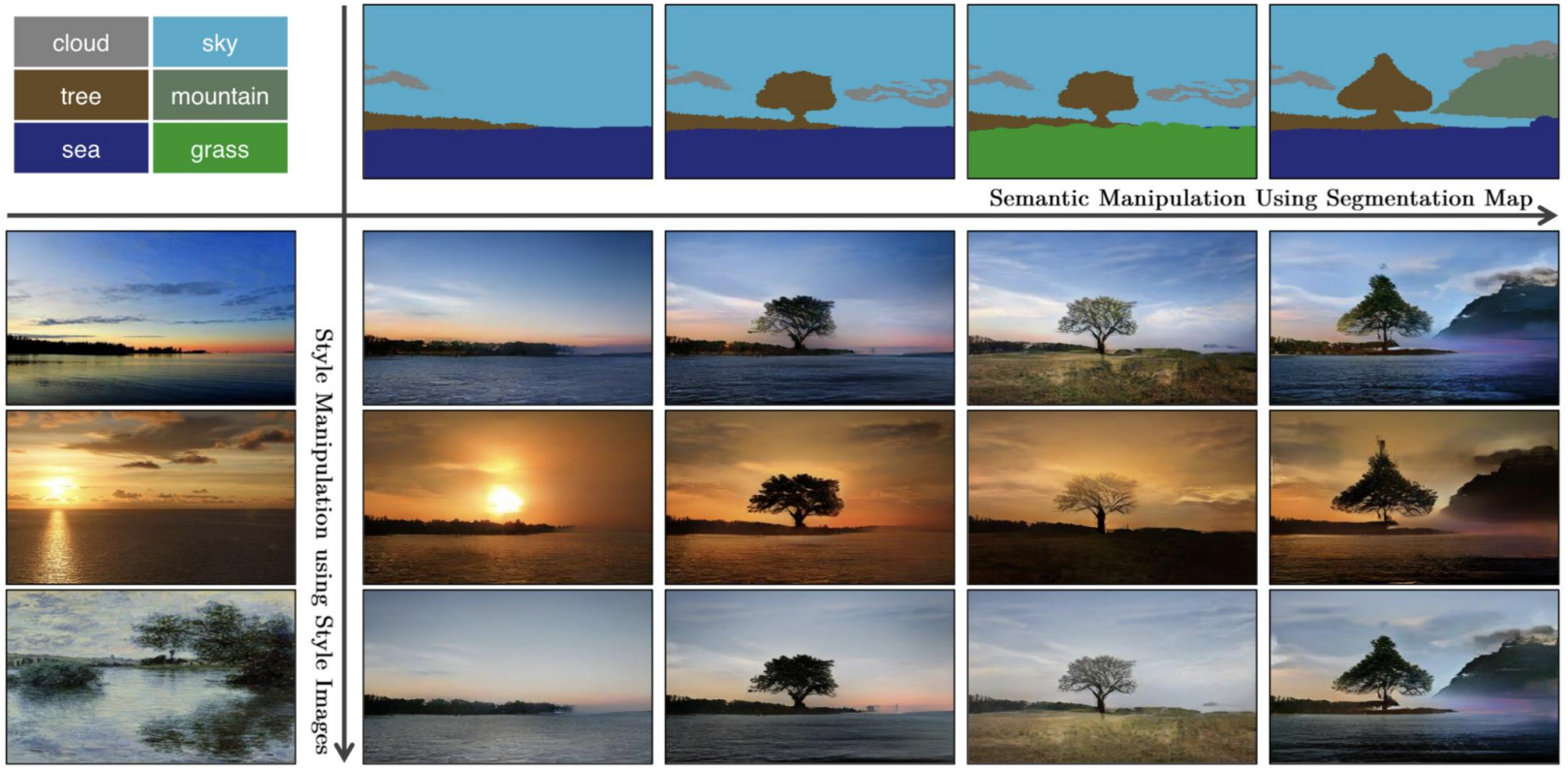


Figure 1: Our model allows user control over both semantic and style as synthesizing an image. The semantic (e.g., the existence of a tree) is controlled via a label map (the top row), while the style is controlled via the reference style image (the leftmost column). Please visit our website for interactive image synthesis demos.

Previous Work

- pix2pixHD: Conditional GAN with semantic condition.

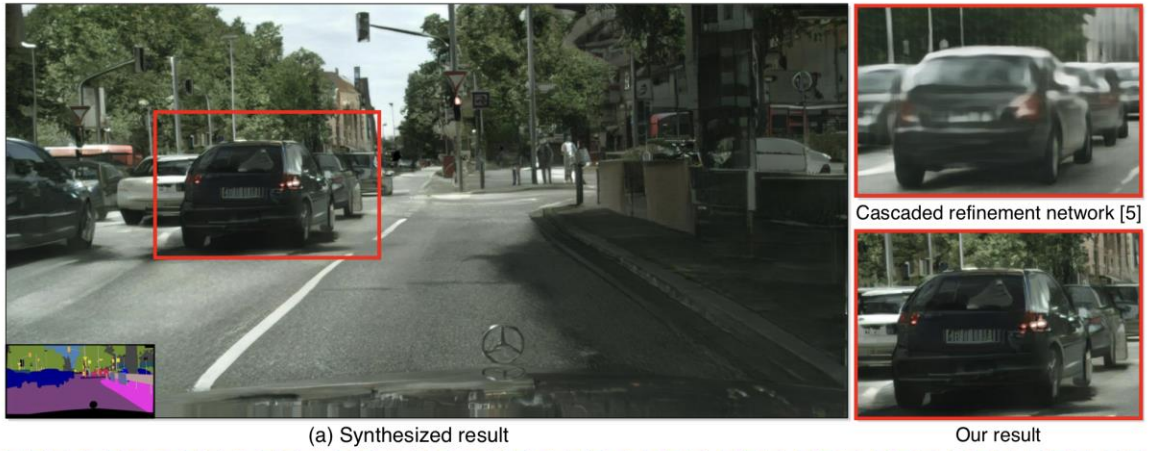


Figure 1: We propose a generative adversarial framework for synthesizing 2048×1024 images from semantic label maps (lower left corner in (a)). Compared to previous work [5], our results express more natural textures and details. (b) We can change labels in the original label map to create new scenes, like replacing trees with buildings. (c) Our framework also allows a user to edit the appearance of individual objects in the scene, e.g. changing the color of a car or the texture of a road. Please visit our website for more side-by-side comparisons as well as interactive editing demos.

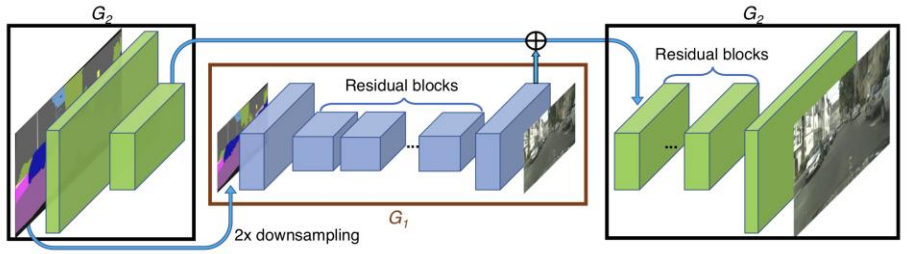


Figure 3: Network architecture of our generator. We first train a residual network G_1 on lower resolution images. Then, another residual network G_2 is appended to G_1 and the two networks are trained jointly on high resolution images. Specifically, the input to the residual blocks in G_2 is the element-wise sum of the feature map from G_2 and the last feature map from G_1.

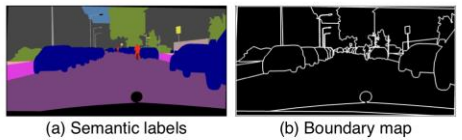


Figure 4: Using instance maps: (a) a typical semantic label map. Note that all connected cars have the same label, which makes it hard to tell them apart. (b) The extracted instance boundary map. With this information, separating different objects becomes much easier.

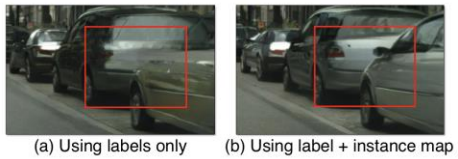
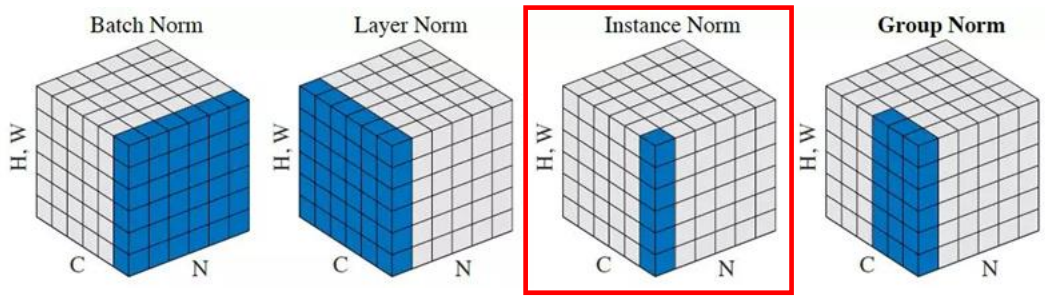


Figure 5: Comparison between results without and with instance maps. It can be seen that when instance boundary information is added, adjacent cars have sharper boundaries.

Wang, Ting-Chun, et al. "High-resolution image synthesis and semantic manipulation with conditional gans." *Proceedings of the IEEE conference on co mputer vision and pattern recognition*. 2018.

Wash Away

- Instance normalization layers tend to “wash away” semantic information.
- $y_{bchw} = \frac{x_{bchw} - \mu_{bc}}{\sigma_{bc}}$.



Various types of normalization.

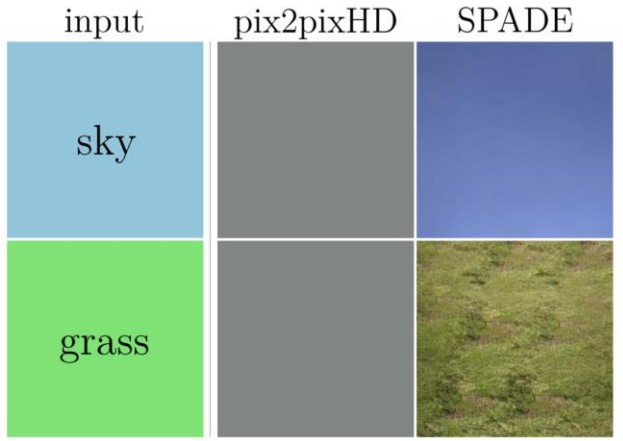
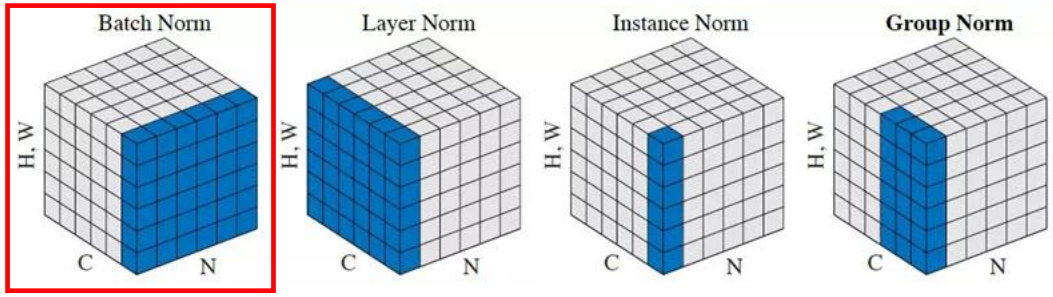


Figure 3: Comparing results given uniform segmentation maps: while the SPADE generator produces plausible textures, the pix2pixHD generator [48] produces two identical outputs due to the loss of the semantic information after the normalization layer.

Wu, Yuxin, and Kaiming He. "Group normalization." *Proceedings of the European conference on computer vision (ECCV)*. 2018.

SPatially-Adaptive (DE)normalization

- We have to inject semantic information using normalization.
- Batch normalization w/ spatial adaptive parameter γ, β
- $\gamma_{chw}(m) \cdot \frac{h_{bchw} - \mu_c}{\sigma_c} + \beta_{chw}(m)$.
- m: semantic segmentation mask.



Various types of normalization.

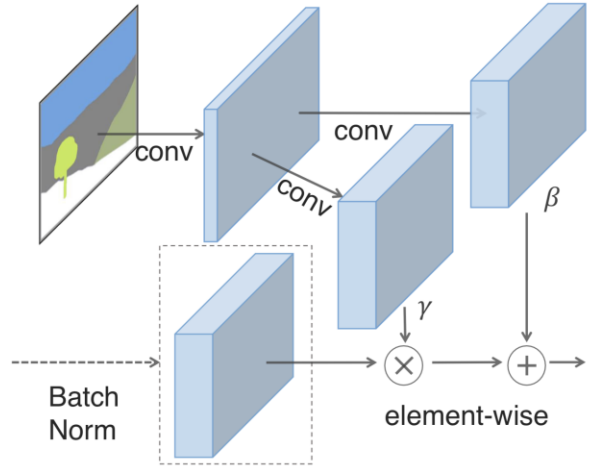


Figure 2: In the SPADE, the mask is first projected onto an embedding space and then convolved to produce the modulation parameters γ and β. Unlike prior conditional normalization methods, γ and β are not vectors, but tensors with spatial dimensions. The produced γ and β are multiplied and added to the normalized activation element-wise.

SPADE generator (GauGAN)

- SPADE generator does not need encoder.

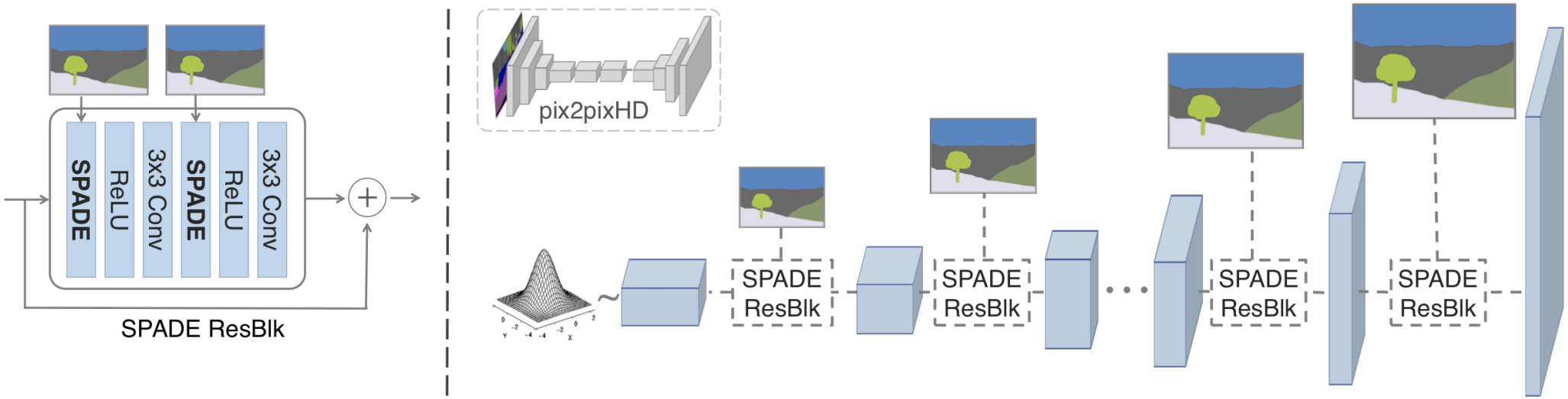


Figure 4: In the SPADE generator, each normalization layer uses the segmentation mask to modulate the layer activations. *(left)* Structure of one residual block with the SPADE. *(right)* The generator contains a series of the SPADE residual blocks with upsampling layers. Our architecture achieves better performance with a smaller number of parameters by removing the downsampling layers of leading image-to-image translation networks such as the pix2pixHD model [48].

Quantitative Results

- mIoU, accuracy, FID, and user study.

Method	COCO-Stuff			ADE20K			ADE20K-outdoor			Cityscapes		
	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID
CRN [6]	23.7	40.4	70.4	22.4	68.8	73.3	16.5	68.6	99.0	52.4	77.1	104.7
SIMS [43]	N/A	N/A	N/A	N/A	N/A	N/A	13.1	74.7	67.7	47.2	75.5	**49.7**
pix2pixHD [48]	14.6	45.8	111.5	20.3	69.2	81.8	17.4	71.6	97.8	58.3	81.4	95.0
Ours	**37.4**	**67.9**	**22.6**	**38.5**	**79.9**	**33.9**	**30.8**	**82.9**	**63.3**	**62.3**	**81.9**	71.8

Table 1: Our method outperforms the current leading methods in semantic segmentation (mIoU and accu) and FID [17] scores on all the benchmark datasets. For the mIoU and accu, higher is better. For the FID, lower is better.

Dataset	Ours vs. CRN	Ours vs. pix2pixHD	Ours vs. SIMS
COCO-Stuff	79.76	86.64	N/A
ADE20K	76.66	83.74	N/A
ADE20K-outdoor	66.04	79.34	85.70
Cityscapes	63.60	53.64	51.52

Table 2: User preference study. The numbers indicate the percentage of users who favor the results of the proposed method over those of the competing method.

Qualitative Results: Semantic Image Synthesis

Figure 5: Visual comparison of semantic image synthesis results on the COCO-Stuff dataset. Our method successfully synthesizes realistic details from semantic labels.

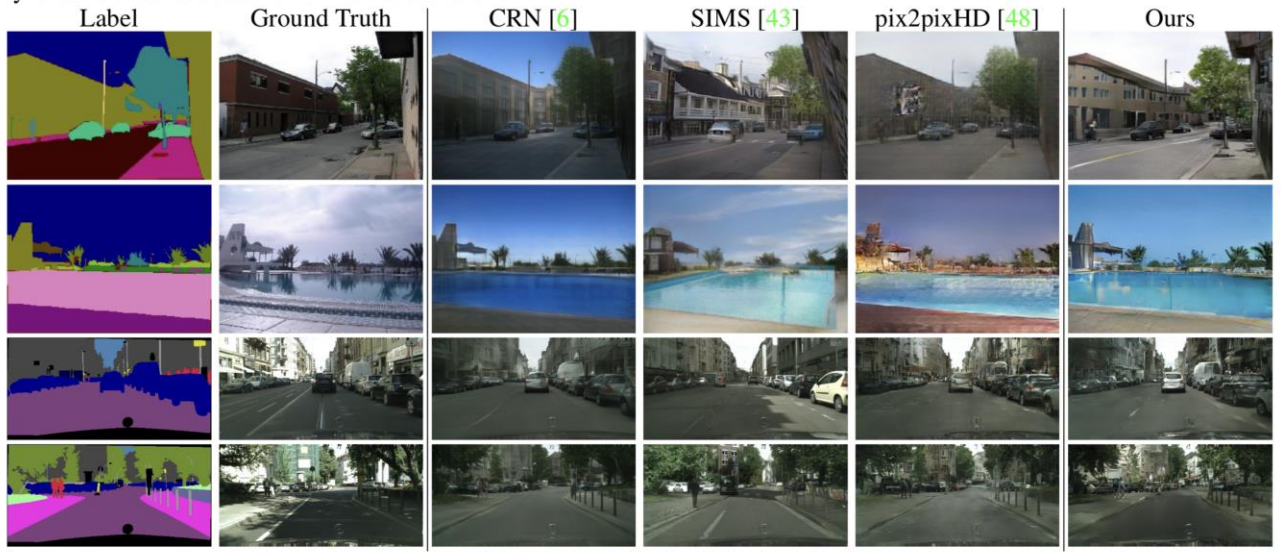


Figure 6: Visual comparison of semantic image synthesis results on the ADE20K outdoor and Cityscapes datasets. Our method produces realistic images while respecting the spatial semantic layout at the same time.

Qualitative Results: SIS (Flickr)

- Train with synthetic semantic segmentation mask predicted by Deeplabv2.

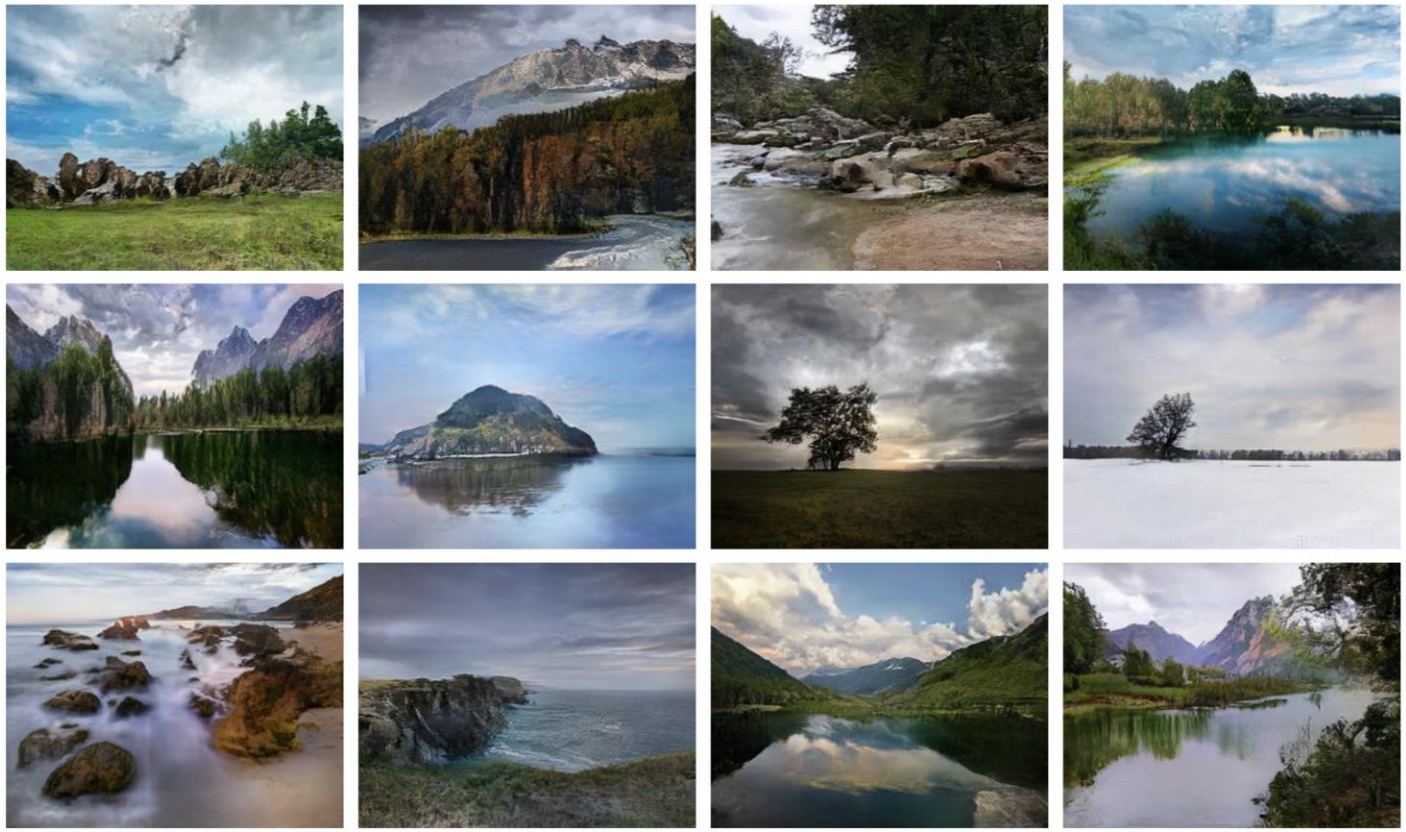


Figure 7: Semantic image synthesis results on the Flickr Landscapes dataset. The images were generated from semantic layout of photographs on the Flickr website.

Image Encoder

- Use image encoder with KL Divergence loss for style manipulation.

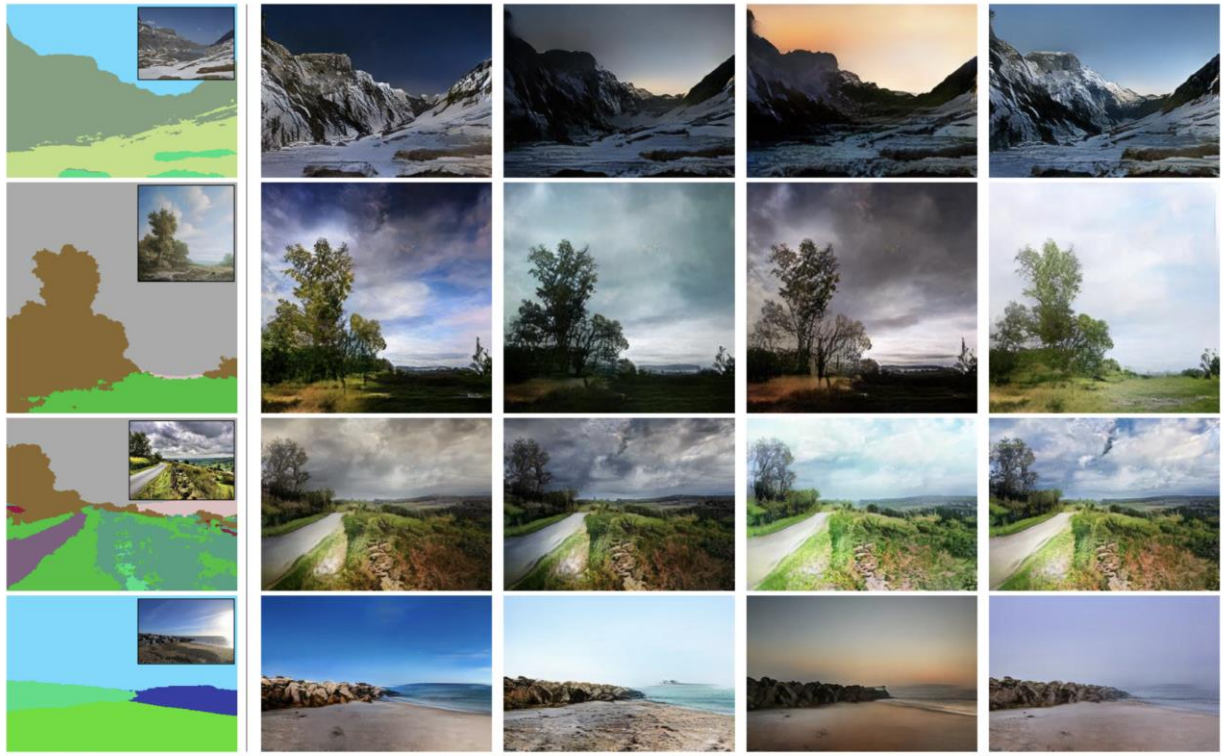


Figure 9: Our model attains multimodal synthesis capability when trained with the image encoder. During deployment, by using different random noise, our model synthesizes outputs with diverse appearances but all having the same semantic layouts depicted in the input mask. For reference, the ground truth image is shown inside the input segmentation mask.

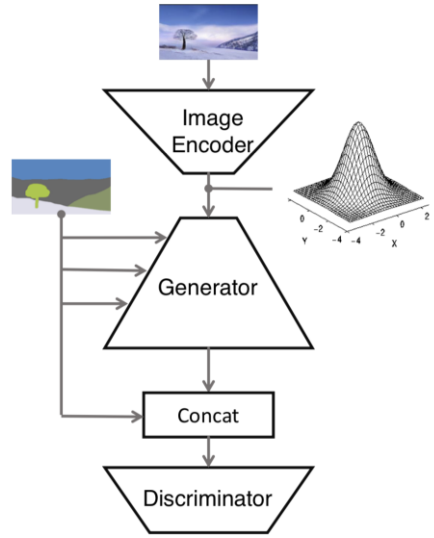


Figure 15: The image encoder encodes a real image to a latent representation for generating a mean vector and a variance vector. They are used to compute the noise input to the generator via the reparameterization trick [28]. The generator also takes the segmentation mask of the input image as input via the proposed SPADE ResBlks. The discriminator takes concatenation of the segmentation mask and the output image from the generator as input and aims to classify that as fake.