Semantic Image Synthesis with Spatially-Adaptive Normalization

Taesung Park[1]* Ming-Yu Liu[2] Ting-Chun Wang[2] Jun-Yan Zhu[2,3]

[1]UC Berkeley [2]NVIDIA [3]MIT CSAIL

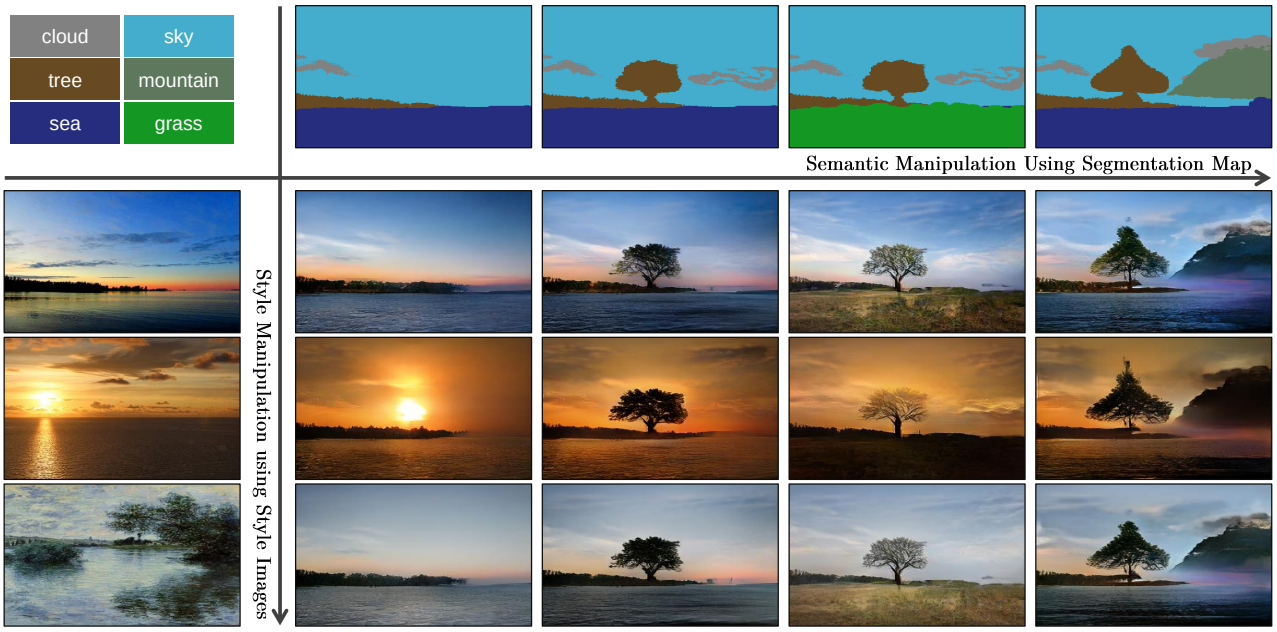

Figure 1: Our model allows user control over both semantic and style as synthesizing an image. The semantic (e.g., existence of a tree) is controlled via a label map (visualized in the top row), while the style is controlled via the reference style image (visualized in the leftmost column). Please visit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

Abstract

We propose spatially-adaptive normalization, a simple but effective layer for synthesizing photorealistic images given an input semantic layout. Previous methods directly feed the semantic layout as input to the deep network, which is then processed through stacks of convolution, normalization, and nonlinearity layers. We show that this is suboptimal as the normalization layers tend to "wash away" semantic information. To address the issue, we propose using the input layout for modulating the activations in normalization layers through a spatially-adaptive, learned transformation. Experiments on several challenging datasets demonstrate the advantage of the proposed method over existing approaches, regarding both visual fidelity and alignment with input layouts. Finally, our model allows user control over both semantic and style as synthesizing images. Code will be available at https://github.com/NVlabs/SPADE.

1. Introduction

Conditional image synthesis refers to the task of generating photorealistic images conditioning on some input data. Earlier methods compute the output image by stitching pieces from a database of images [3, 13]. Recent methods directly learn the mapping using neural networks [4, 7, 20, 39, 40, 45, 46, 47]. The latter methods are generally faster and require no external database of images.

We are interested in a specific form of conditional image synthesis, which is converting a semantic segmenta-

*Taesung Park contributed to the work during his NVIDIA internship.

tion mask to a photorealistic image. This form has a wide range of applications such as content generation and image editing [7, 20, 40]. We will refer to this form as semantic image synthesis. In this paper, we show that the conventional network architecture [20, 40], which is built by stacking convolutional, normalization, and nonlinearity layers, is at best sub-optimal, because their normalization layers tend to "wash away" information in input semantic masks. To address the issue, we propose *spatially-adaptive normalization*, a conditional normalization layer that modulates the activations using input semantic layouts through a spatially-adaptive, learned transformation and can effectively propagate the semantic information throughout the network.

We conduct experiments on several challenging datasets including the COCO-Stuff [5, 26], the ADE20K [48], and the Cityscapes [8]. We show that with the help of our spatially-adaptive normalization layer, a compact network can synthesize significantly better results compared to several state-of-the-art methods. Additionally, an extensive ablation study demonstrates the effectiveness of the proposed normalization layer against several variants for the semantic image synthesis task. Finally, our method supports multi-modal and style-guided image synthesis, enabling controllable, diverse outputs as shown in Figure 1.

2. Related Work

Deep generative models can learn to synthesize randomly sampled images. Recent methods include generative adversarial networks (GANs) [12] and variational autoencoder (VAE) [22]. Our work is built on GANs but aims for the conditional image synthesis task. The GANs consist of a generator and a discriminator where the goal of the generator is to produce realistic images so that the discriminator cannot tell the synthesized images apart from real ones.

Conditional image synthesis exists in many forms that differ in the type of input data. For example, class-conditional models [4, 29, 31, 33] learn to synthesize images given category labels. Researchers have explored various models for generating images based on text [16, 36, 43, 46]. Another widely-used form is image-to-image translation [18, 20, 23, 27, 49, 50], where both input and output are images. In this work, we focus on converting segmentation masks to photorealistic images. We assume the training dataset contains paired segmentation masks and images. With the proposed spatially-adaptive normalization, our compact network achieves better results compared to leading methods.

Unconditional normalization layers have been an important component in modern deep networks and can be found in various classifier designs, including the Local Response Normalization (LRN) in the AlexNet [24] and Batch Normalization (BN) in the Inception-v2 network [19]. Other popular normalization layers include the Instance Normalization (IN) [38], Layer Normalization (LN) [2], Group Normalization (GN) [41], and Weight Normalization (WN) [37]. We label these normalization layers as unconditional as they do not depend on external data in contrast to the conditional normalization layers discussed below.

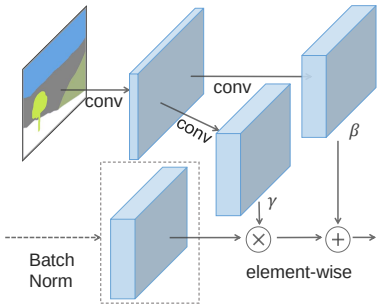

Figure 2: In SPADE, the mask is first projected onto an embedding space, and then convolved to produce the modulation parameters γ and β. Unlike prior conditional normalization methods, γ and β are not vectors, but tensors with spatial dimensions. The produced γ and β are multiplied and added to the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

Conditional normalization layers include the Conditional Batch Normalization (Conditional BN) [10] and Adaptive Instance Normalization (AdaIN) [17]. Both were first used in the style transfer task and later adopted in various vision tasks [9, 18, 29, 31, 34, 45]. Different from the earlier normalization techniques, conditional normalization layers require external data and generally operate as follows. First, layer activations are normalized to zero mean and unit deviation. Then the normalized activations are *denormalized* by modulating the activation using a learned affine transformation whose parameters are inferred from external data. For style transfer tasks [10, 17], the affine parameters are used to control the global style of the output, and hence are uniform across spatial coordinates. Different from prior work, our proposed normalization layer applies a spatially-varying affine transformation, making it suitable for image synthesis from spatially-varying semantic mask.

3. Semantic Image Synthesis

Let $\mathbf{m} \in \mathbb{L}^{H \times W}$ be a semantic segmentation mask where $\mathbb{L}$ is a set of integers denoting the semantic labels, and H and W are the image height and width. Each entry in $\mathbf{m}$ denotes the semantic label of a pixel. We aim to learn a mapping function that can convert an input segmentation mask $\mathbf{m}$ to a photorealistic image.

Spatially-adaptive denormalization. Let $\mathbf{h}^i$ denote the activations of the i-th layer of a deep convolutional network given a batch of N samples. Let C^i be the number of channels in the layer. Let H^i and W^i be the height and width of the activation map in the layer. We propose a new con-

ditional normalization method called SPatially-Adaptive (DE)normalization[1] (SPADE). Similar to Batch Normalization [19], the activation is normalized in the channel-wise manner, and then modulated with learned scale and bias. Figure 2 illustrates the SPADE design. The activation value at site $(n \in N, c \in C^i, y \in H^i, x \in W^i)$ is given by

$$\gamma^i_{c,y,x}(\mathbf{m})\frac{h^i_{n,c,y,x} - \mu^i_c}{\sigma^i_c} + \beta^i_{c,y,x}(\mathbf{m}) \tag{1}$$

where $h^i_{n,c,y,x}$ is the activation at the site before normalization, μ^i_c and σ^i_c are the mean and standard deviation of the activation in channel c:

$$\mu^i_c = \frac{1}{NH^iW^i}\sum_{n,y,x} h^i_{n,c,y,x} \tag{2}$$

$$\sigma^i_c = \sqrt{\frac{1}{NH^iW^i}\sum_{n,y,x}(h^i_{n,c,y,x})^2 - (\mu^i_c)^2}. \tag{3}$$

The variables $\gamma^i_{c,y,x}(\mathbf{m})$ and $\beta^i_{c,y,x}(\mathbf{m})$ in (1) are the learned modulation parameters of the normalization layer. In contrast to BatchNorm [19], they depend on the input segmentation mask and vary with respect to the location (y, x). We use the symbol $\gamma^i_{c,y,x}$ and $\beta^i_{c,y,x}$ to denote the functions that convert the input segmentation mask $\mathbf{m}$ to the scaling and bias values at the site (c, y, x) in the i-th activation map. We implement the functions $\gamma^i_{c,y,x}$ and $\beta^i_{c,y,x}$ using a simple two-layer convolutional network, whose detail design can be found in the appendix.

In fact, SPADE is related to, and is a generalization of several existing normalization layers. First, replacing the segmentation mask $\mathbf{m}$ with the image class label and making the modulation parameters spatially-invariant (i.e., $\gamma^i_{c,y_1,x_1} \equiv \gamma^i_{c,y_2,x_2}$ and $\beta^i_{c,y_1,x_1} \equiv \beta^i_{c,y_2,x_2}$ for any $y_1, y_2 \in \{1, 2, ..., H^i\}$ and $x_1, x_2 \in \{1, 2, ..., W^i\}$), we arrive at the form of Conditional Batch Normalization layer [10]. Indeed, for any spatially-invariant conditional data, our method reduces to Conditional BN. Similarly, we can arrive at AdaIN [17] by replacing the segmentation mask with another image, making the modulation parameters spatially-invariant and setting $N = 1$. As the modulation parameters are adaptive to the input segmentation mask, the proposed SPADE is better suited for semantic image synthesis.

SPADE generator. With SPADE, there is no need to feed the segmentation map to the first layer of the generator, since the learned modulation parameters have encoded enough information about the label layout. Therefore, we discard encoder part of the generator, which is commonly used in recent architectures [20, 40]. This simplification results in a more lightweight network. Furthermore, similarly to existing class-conditional generators [29, 31, 45], the new generator can take a random vector as input, enabling a simple and natural way for multi-modal synthesis [18, 50].

[1] Conditional normalization [10, 17] uses external data to denormalize the normalized activations; i.e., the denormalization part is conditional.

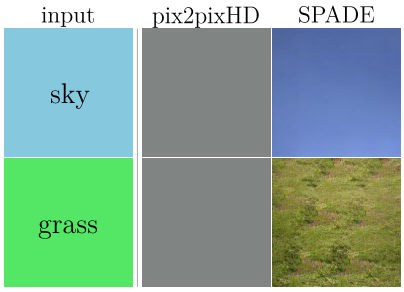

Figure 3: Comparing results given uniform segmentation maps: while SPADE generator produces plausible textures, pix2pixHD [40] produces identical outputs due to the loss of the semantic information after the normalization layer.

Figure 4 illustrates our generator architecture, which employs several ResNet blocks [14] with upsampling layers. The modulation parameters of all the normalization layers are learned using SPADE. Since each residual block operates at a different scale, SPADE downsamples the semantic mask to match the spatial resolution.

We train the generator with the same multi-scale discriminator and loss function used in pix2pixHD except that we replace the least squared loss term [28] with the hinge loss term [25, 30, 45]. We test several ResNet-based discriminators used in recent unconditional GANs [1, 29, 31] but observe similar results at the cost of a higher GPU memory requirement. Adding the SPADE to the discriminator also yields a similar performance. For the loss function, we observe that removing any loss term in the pix2pixHD loss function lead to degraded generation results.

Why does SPADE work better? A short answer is that it can better preserve semantic information against common normalization layers. Specifically, while normalization layers such as the InstanceNorm [38] are essential pieces in almost all the state-of-the-art conditional image synthesis models [40], they tend to wash away semantic information when applied to uniform or flat segmentation masks.

Let us consider a simple module that first applies convolution to a segmentation mask and then normalization. Furthermore, let us assume that a segmentation mask with a single label is given as input to the module (e.g., all the pixels have the same label such as sky or grass). Under this setting, the convolution outputs are again uniform with different labels having different uniform values. Now after we apply InstanceNorm to the output, the normalized activation will become all zeros no matter what the input semantic label is given. Therefore, semantic information is totally lost. This limitation applies to a wide range of generator architectures, including pix2pixHD and its variant that concate-

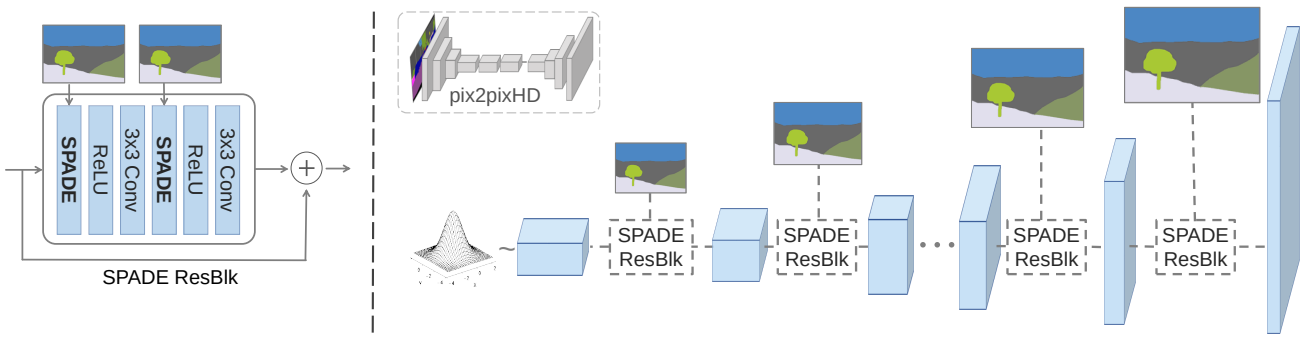

Figure 4: In the SPADE generator, each normalization layer uses the segmentation mask to modulate the layer activations. *(left)* Structure of one residual block with SPADE. *(right)* The generator contains a series of SPADE residual blocks with upsampling layers. Our architecture achieves better performance with a smaller number of parameters by removing the downsampling layers of leading image-to-image translation networks (pix2pixHD [40]).

nates the semantic mask at all intermediate layers, as long as a network applies convolution and then normalization to the semantic mask. In Figure 3, we empirically show this is precisely the case for pix2pixHD. Because a segmentation mask consists of a few uniform regions in general, the issue of information loss emerges when applying normalization.

In contrast, the segmentation mask in the SPADE Generator is fed through spatially adaptive modulation *without* normalization. Only activations from the previous layer are normalized. Hence, the SPADE generator can better preserve semantic information. It enjoys the benefit of normalization without losing the semantic input information.

Multi-modal synthesis. By using a random vector as the input of the generator, our architecture provides a simple way for multi-modal synthesis. Namely, one can attach an encoder that processes a real image into a random vector, which will be then fed to the generator. The encoder and generator form a variational autoencoder [22], in which the encoder tries to capture the style of the image, while the generator combines the encoded style and the segmentation mask information via SPADE to reconstruct the original image. The encoder also serves as a style guidance network at test time to capture the style of target images, as used in Figure 1. For training, we add a KL-Divergence loss term [22].

4. Experiments

Implementation details. We apply the Spectral Norm [30] to all the layers in both the generator and discriminator. The learning rates for the generator and discriminator are set to 0.0001 and 0.0004, respectively [15]. We use the ADAM [21] and set $\beta_1 = 0$, $\beta_2 = 0.999$. All the experiments are conducted on an NVIDIA DGX1 with 8 V100 GPUs. We use synchronized mean and variance computation, i.e., these statistics are collected from all the GPUs.

Datasets. We conduct experiments on several datasets.

- *COCO-Stuff* [5] is derived from the COCO dataset [26]. It has 118,000 training images and 5,000 validation images captured from diverse scenes. It has 182 semantic classes. Due to its large diversity, existing image synthesis models perform poorly on this dataset.
- *ADE20K* [48] consists of 20,210 training and 2,000 validation images. Similarly to COCO, the dataset contains challenging scenes with 150 semantic classes.
- *ADE20K-outdoor* is a subset of the ADE20K dataset that only contains outdoor scenes, used in Qi *et al*. [35].
- *Cityscapes* dataset [8] contains street scene images in German cities. The training and validation set sizes are 3,000 and 500, respectively. Recent work has achieved photorealistic semantic image synthesis results [35, 39] on the Cityscapes dataset.
- *Flickr Landscapes.* We collect 41,000 photos from Flickr and use 1,000 samples for the validation set. Instead of manual annotation, we use a pre-trained DeepLabV2 model [6] to compute the input segmentation masks.

We train the competing semantic image synthesis methods on the same training set and report their results on the same validation set for each dataset.

Performance metrics. We adopt the evaluation protocol from previous work [7, 40]. Specifically, we run a semantic segmentation model on the synthesized images and compare how well the predicted segmentation mask matches the ground truth input. This is based on the intuition that if the output images are realistic then a well-trained semantic segmentation model should be able to predict the ground truth label. For measuring the segmentation accuracy, we use the mean Intersection-over-Union (mIoU) and pixel accuracy (accu) metrics. We use state-of-the-art segmentation networks for each dataset: DeepLabV2 [6, 32] for COCO-Stuff, UperNet101 [42] for ADE20K, and DRN-D-105 [44] for Cityscapes. In addition to segmentation accuracy, we use the Fréchet Inception Distance (FID) [15] to measure

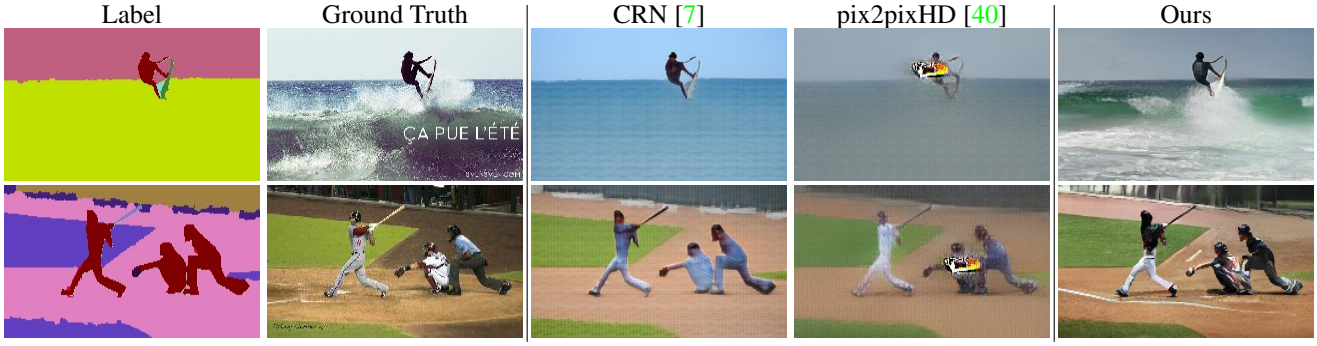

Figure 5: Visual comparison of semantic image synthesis results on the COCO-Stuff dataset. Our method successfully synthesizes realistic details from semantic labels.

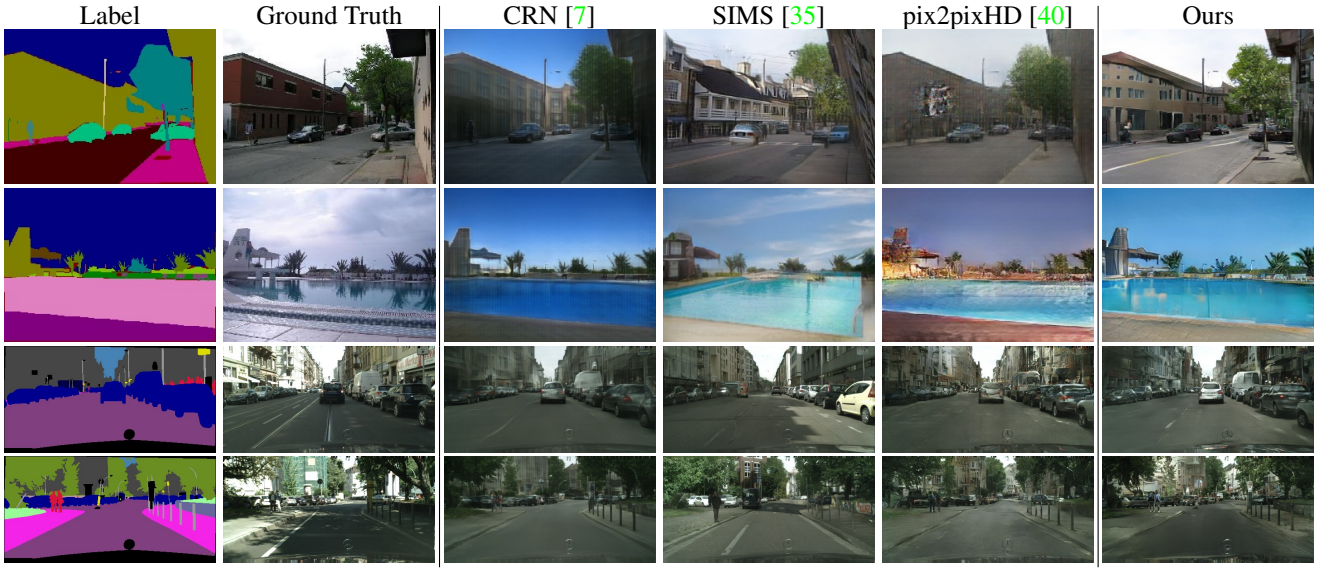

Figure 6: Visual comparison of semantic image synthesis results on the ADE20K outdoor and Cityscapes datasets. Our method produces realistic images while respecting the spatial semantic layout at the same time.

	COCO-Stuff			ADE20K			ADE20K-outdoor			Cityscapes		
Method	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID
CRN [7]	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 [35]	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 [40]	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 current leading methods in semantic segmentation scores (mean IoU and overall pixel accuracy) and FID [15] on all the benchmark datasets. For mIoU and pixel accuracy, higher is better. For FID, lower is better.

the distance between the distributions of synthesized results and the distribution of real images.

Baselines. We compare our method with three leading semantic image synthesis models: the pix2pixHD model [40], the cascaded refinement network model (CRN) [7], and the semi-parametric image synthesis model (SIMS) [35]. pix2pixHD is the current state-of-the-art GAN-based conditional image synthesis framework. CRN uses a deep network that repeatedly refines the output from low to high resolution, while the SIMS takes a semi-parametric approach that composites real segments from a training set and refines the boundaries. Both the CRN and SIMS are mainly trained using image reconstruction loss. For a fair comparison, we train the CRN and pix2pixHD models using the implementations provided by the authors. As synthesizing an image using SIMS requires many queries to the training dataset, it is computationally prohibitive for a large dataset such as COCO-stuff and the full ADE20K. Therefore, we use the result images provided by the authors whenever possible.

Quantitative comparisons. As shown in Table 1, our

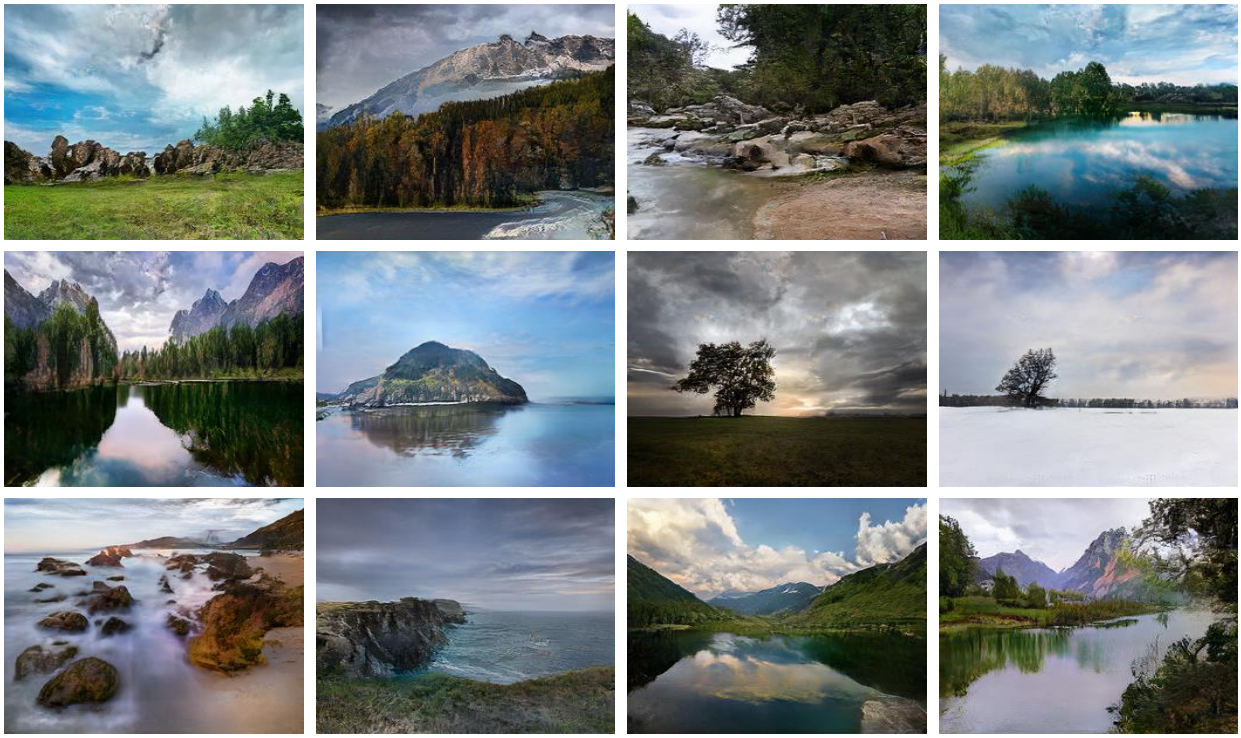

Figure 7: Semantic image synthesis results on the Flickr Landscapes dataset. The images were generated from semantic layout of photographs on Flickr.

method outperforms the current state-of-the-art methods by a large margin in all the datasets. For COCO-Stuff, our method achieves a mIoU score of 35.2, which is about 1.5 times better than the previous leading method. Our FID is also 2.2 times better than the previous leading method. We note that the SIMS model produces a lower FID score but has poor segmentation performances on the Cityscapes dataset. This is because the SIMS synthesizes an image by first stitching image patches from the training dataset. As using the real image patches, the resulting image distribution can better match the distribution of real images. However, because there is no guarantee that a perfect query (e.g., a person in a particular pose) exists in the dataset, it tends to copy objects with mismatched segments.

Qualitative results. In Figures 5 and 6, we provide a qualitative comparison of the competing methods. We find that our method produces results with much better visual quality and fewer artifacts, especially for diverse scenes in the COCO-Stuff and ADE20K dataset. When the training dataset size is small, the SIMS model also renders images with good visual quality. However, the depicted content often deviates from the input segmentation mask (e.g., the shape of the swimming pool in the second row of Figure 6).

In Figures 7 and 8, we show more example results from the Flickr Landscape and COCO-Stuff datasets. The proposed method can generate diverse scenes with high image fidelity. More results are included in the appendix.

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 the competing method.

Human evaluation. We use Amazon Mechanical Turk (AMT) to compare the perceived visual fidelity of our method against existing approaches. Specifically, we give the AMT workers an input segmentation mask and two synthesis outputs from different methods and ask them to choose the output image that looks more like a corresponding image of the segmentation mask. The workers are given unlimited time to make the selection. For each comparison, we randomly generate 500 questions for each dataset, and each question is answered by 5 different workers. For quality control, only workers with a lifetime task approval rate greater than 98% can participate in our evaluation.

Table 2 shows the evaluation results. We find that users strongly favor our results on all the datasets, especially on the challenging COCO-Stuff and ADE20K datasets. For the

Figure 8: Semantic image synthesis results on COCO-Stuff. Our method successfully generates realistic images in diverse scenes ranging from animals to sports activities.

Method	#param	COCO.	ADE.	City.
decoder w/ SPADE (Ours)	96M	**35.2**	38.5	62.3
compact decoder w/ SPADE	61M	**35.2**	38.0	**62.5**
decoder w/ Concat	79M	31.9	33.6	61.1
pix2pixHD++ w/ SPADE	237M	34.4	**39.0**	62.2
pix2pixHD++ w/ Concat	195M	32.9	38.9	57.1
pix2pixHD++	183M	32.7	38.3	58.8
compact pix2pixHD++	103M	31.6	37.3	57.6
pix2pixHD [40]	183M	14.6	20.3	58.3

Table 3: mIoU scores are boosted when SPADE layers are used, for both the decoder architecture (Figure 4) and encoder-decoder architecture of pix2pixHD++ (our improved baseline over pix2pixHD [40]). On the other hand, simply concatenating semantic input at every layer fails to do so. Moreover, our compact model with smaller depth at all layers outperforms all baselines.

Method	COCO	ADE20K	Cityscapes
segmap input	35.2	38.5	62.3
random input	35.3	38.3	61.6
kernelsize 5x5	35.0	39.3	61.8
kernelsize 3x3	35.2	38.5	62.3
kernelsize 1x1	32.7	35.9	59.9
#params 141M	35.3	38.3	62.5
#params 96M	35.2	38.5	62.3
#params 61M	35.2	38.0	62.5
Sync Batch Norm	35.0	39.3	61.8
Batch Norm	33.7	37.9	61.8
Instance Norm	33.9	37.4	58.7

Table 4: The SPADE generator works with different configurations. We change the input of the generator, the convolutional kernel size acting on the segmentation map, the capacity of the network, and the parameter-free normalization method. The settings used in the paper are boldfaced.

Cityscapes, even when all the competing methods achieve high image fidelity, users still prefer our results.

The effectiveness of SPADE. To study the importance of SPADE, we introduce a strong baseline called pix2pixHD++, which combines all the techniques we find useful for enhancing the performance of pix2pixHD except SPADE. We also train models that receive segmentation mask input at all the intermediate layers via concatenation (pix2pixHD++ w/ Concat) in the channel direction. Finally, the model that combines the strong baseline with SPADE is denoted as pix2pixHD++ w/ SPADE. Additionally, we compare models with different capacity by using a different number of convolutional filters in the generator.

As shown in Table 3 the architectures with the proposed SPADE consistently outperforms its counterparts, in both the decoder-style architecture described in Figure 4 and more traditional encoder-decoder architecture used in pix2pixHD. We also find that concatenating segmentation masks at all intermediate layers, an intuitive alternative to SPADE to provide semantic signal, does not achieve the same performance as SPADE. Furthermore, the decoder-style SPADE generator achieves better performance than

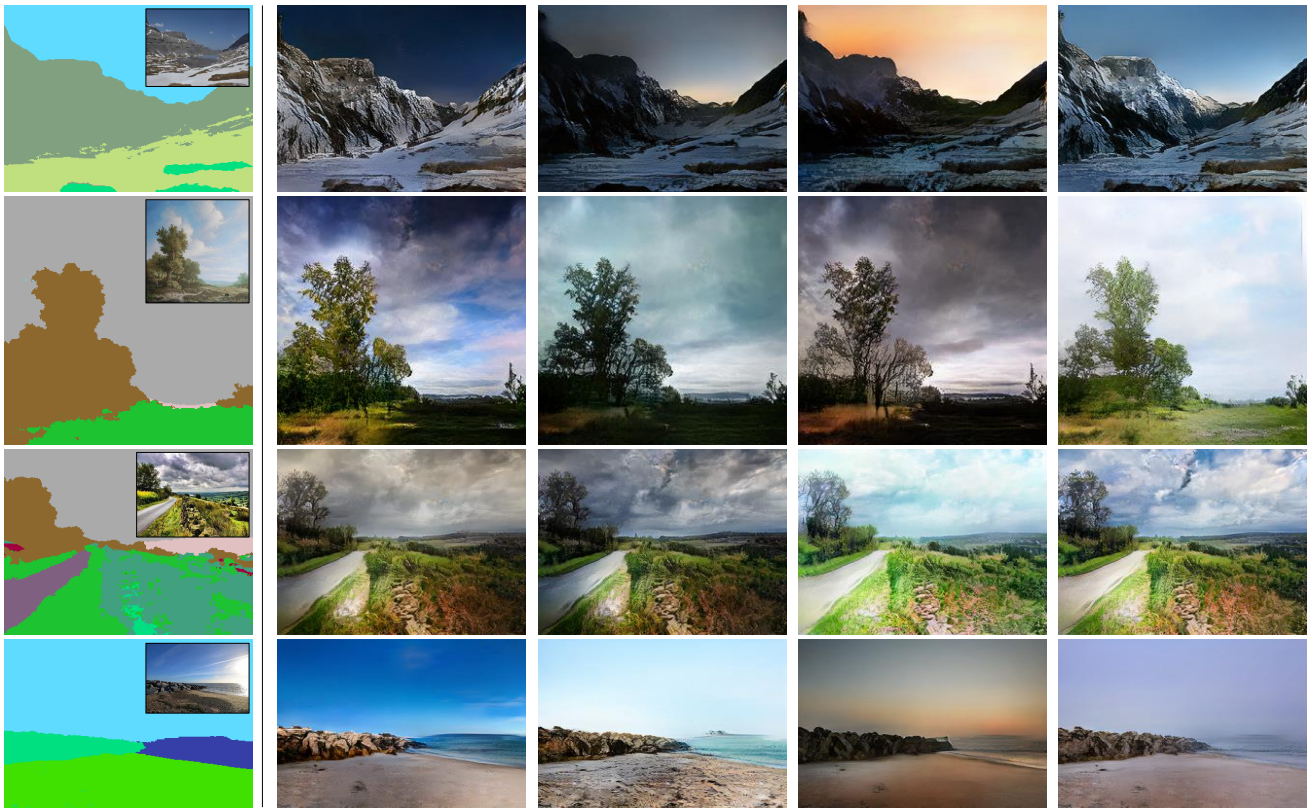

Figure 9: Our model attains multimodal synthesis capability when trained with the image encoder. During deployment, by using different random noise, our model synthesizes outputs with diverse appearances but all having the same semantic layouts depicted in the input mask. For reference, the ground truth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

the strong baselines even when using a smaller number of parameters.

Variations of SPADE generator. Table 4 reports the performance of variations of our generator. First, we compare two types of the input to the generator: random noise or downsampled segmentation maps. We find that both render similar performance, and conclude that the modulation by SPADE alone provides sufficient signal about the input mask. Second, we vary the type of parameter-free normalization layers before applying the modulation parameters. We observe that SPADE works reliably across different normalization methods. Next, we vary the convolutional kernel size acting on the label map, and find that kernel size of 1x1 hurts performance, likely because it prohibits utilizing the context of the label. Lastly, we modify the capacity of the generator network by changing the number of convolutional filters. We present more variations and ablations in the appendix for more detailed investigation.

Multi-modal synthesis. In Figure 9, we show the multimodal image synthesis results on the Flickr Landscape dataset. For the same input segmentation mask, we sample different noise inputs to achieve different outputs. More results are included in the appendix.

Semantic manipulation and guided image synthesis. In Figure 1, we show an application where a user draws different segmentation masks, and our model renders the corresponding landscape images. Moreover, our model allows users to choose an external style image to control the global appearances of the output image. We achieve it by replacing the input noise with the embedding vector of the style image computed by the image encoder.

5. Conclusion

We have proposed the spatially-adaptive normalization, which utilizes the input semantic layout while performing the affine transformation in the normalization layers. The proposed normalization leads to the first semantic image synthesis model that can produce photorealistic outputs for diverse scenes including indoor, outdoor, landscape, and street scenes. We further demonstrate its application for multi-modal synthesis and guided image synthesis.

Acknowledgments We thank Alexei A. Efros and Jan Kautz for insightful advice. Taesung Park contributed to the work during his internship at NVIDIA. His Ph.D. is supported by a Samsung Scholarship.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017. 3

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[3] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM SIGGRAPH*, 2009. 1

[4] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 2

[5] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 4

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(4):834–848, 2018. 4

[7] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2, 4, 5, 13, 14, 15, 16, 17, 18

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 4

[9] H. De Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2

[10] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. In *International Conference on Learning Representations (ICLR)*, 2016. 2, 3

[11] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010. 12, 13

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 2

[13] J. Hays and A. A. Efros. Scene completion using millions of photographs. In *ACM SIGGRAPH*, 2007. 1

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 4, 5, 13

[16] S. Hong, D. Yang, J. Choi, and H. Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[17] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 3

[18] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. *European Conference on Computer Vision (ECCV)*, 2018. 2, 3

[19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. 2, 3

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 3, 11, 12

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 4

[22] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 2, 4, 11, 12

[23] A. Kolliopoulos, J. M. Wang, and A. Hertzmann. Segmentation-based 3d artistic rendering. In *Rendering Techniques*, pages 361–370, 2006. 2

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 2

[25] J. H. Lim and J. C. Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017. 3, 11

[26] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 2, 4

[27] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2

[28] X. Mao, Q. Li, H. Xie, Y. R. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 3, 11

[29] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018. 2, 3, 11

[30] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018. 3, 4, 11

[31] T. Miyato and M. Koyama. cGANs with projection discriminator. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 3, 11

[32] K. Nakashima. Deeplab-pytorch. https://github.com/kazuto1011/deeplab-pytorch, 2018. 4

[33] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. In *International Conference on Machine Learning (ICML)*, 2017. 2

[34] E. Perez, H. De Vries, F. Strub, V. Dumoulin, and A. Courville. Learning visual reasoning without strong priors. In *International Conference on Machine Learning (ICML)*, 2017. 2

[35] X. Qi, Q. Chen, J. Jia, and V. Koltun. Semi-parametric image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 5, 13, 17, 18

[36] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning (ICML)*, 2016. 2

[37] T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 2

[38] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. arxiv 2016. *arXiv preprint arXiv:1607.08022*, 2016. 2, 3

[39] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 1, 4

[40] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 4, 5, 7, 11, 12, 13, 14, 15, 16, 17, 18

[41] Y. Wu and K. He. Group normalization. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[42] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding. In *European Conference on Computer Vision (ECCV)*, 2018. 4

[43] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[44] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[45] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 1, 2, 3, 11

[46] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2

[47] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018. 1

[48] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 4

[49] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[50] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2, 3

A. Additional Implementation Details

Generator. The architecture of the generator consists of a series of the proposed SPADE ResBlks with nearest neighbor upsampling. We train our network using 8 GPUs simultaneously and use the synchronized version of the batch normalization. We apply the spectral normalization [30] to all the convolutional layers in the generator. The architectures of the proposed SPADE and SPADE ResBlk are given in Figure 10 and Figure 11, respectively. The architecture of the generator is shown in Figure 12.

Discriminator. The architecture of the discriminator follows the one used in the pix2pixHD method [40], which uses a multi-scale design with instance normalization (IN). The only difference is that we apply the spectral normalization to all the convolutional layers of the discriminator. The details of the discriminator architecture is shown in Figure 13.

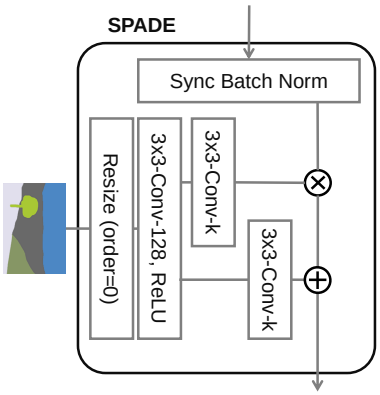

Figure 10: SPADE Design. The term 3x3-Conv-k denotes a 3-by-3 convolutional layer with k convolutional filters. The segmentation map is resized to match the resolution of the corresponding feature map using nearest-neighbor downsampling.

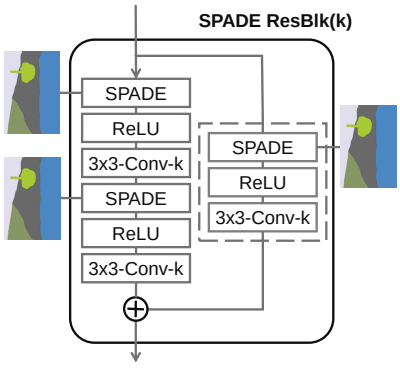

Figure 11: SPADE ResBlk. The residual block design largely follows that in [29] and [31]. We note that for the case that the number of channels before and after the residual block is different, the skip connection is also learned (dashed box in the figure).

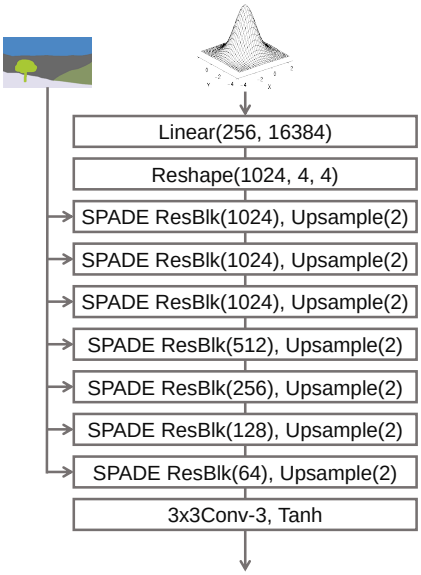

Figure 12: SPADE Generator. Different from prior image generators [20, 40], the semantic segmentation mask is passed to the generator through the proposed SPADE ResBlks in Figure 11.

Image Encoder. The image encoder consists of 6 stride-2 convolutional layers followed by two linear layers to produce the mean and variance of the output distribution as shown in Figure 14.

Learning objective. We use the learning objective function in the pix2pixHD work [40] except that we replace its LSGAN loss [28] term with the Hinge loss term [25, 30, 45]. We use the same weighting among the loss terms in the objective function as that in the pix2pixHD work.

When training the proposed framework with the image encoder for multi-modal synthesis and style-guided image synthesis, we include a KL Divergence loss:

$$\mathcal{L}_{\text{KLD}} = \mathcal{D}_{\text{KL}}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

where the prior distribution $p(\mathbf{z})$ is a standard Gaussian distribution and the variational distribution q is fully determined by a mean vector and a variance vector [22]. We use the reparamterization trick [22] for back-propagating the gradient from the generator to the image encoder. The weight for the KL Divergence loss is 0.05.

In Figure 15, we overview the training data flow. The image encoder encodes a real image to a mean vector and a variance vector. They are used to compute the noise input to the generator via the reparameterization trick [22].

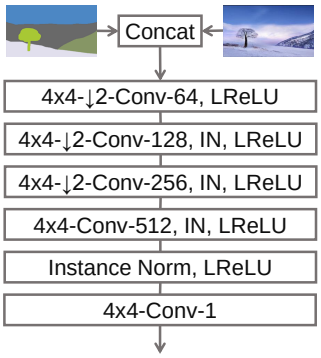

Figure 13: Our discriminator design largely follows that in the pix2pixHD [40]. It takes the concatenation the segmentation map and the image as input. It is based on the PatchGAN [20]. Hence, the last layer of the discriminator is a convolutional layer.

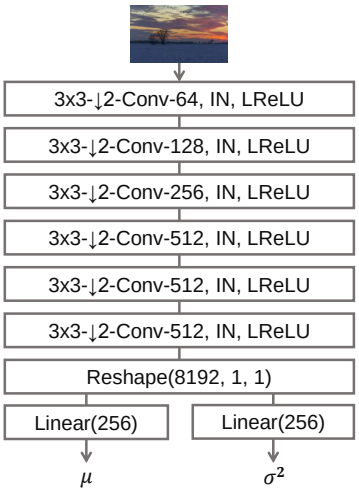

Figure 14: The image encoder consists a series of convolutional layers with stride 2 followed by two linear layers that output a mean vector μ and a variance vector σ.

The generator also takes the segmentation mask of the input image as input with 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.

Training details. We perform 200 epochs of training on the Cityscapes and ADE20K datasets, 100 epochs of training on the COCO-Stuff dataset, and 50 epochs of training on the Flickr Landscapes dataset. The image sizes are 256x256, except the Cityscapes at 512x256. We linearly decay the learning rate to 0 from epoch 100 to 200 for the Cityscapes and ADE20K datasets. The batch size is 32. We initialize the network weights using Glorot initialization [11].

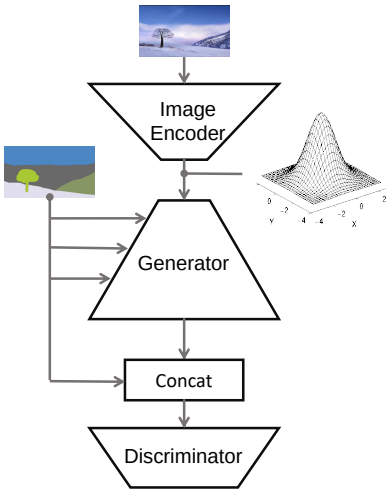

Figure 15: The image encoder encodes a real image to a latent representation for generating a mean vector and a variance vector. They are used to compute the noise input to the generator via the reparameterization trick [22]. The generator also takes the segmentation mask of the input image as input 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.

B. Additional Ablation Study

Method	COCO.	ADE.	City.
Ours	35.2	38.5	62.3
Ours w/o Perceptual loss	24.7	30.1	57.4
Ours w/o GAN feature matching loss	33.2	38.0	62.2
Ours w/ a deeper discriminator	34.9	38.3	60.9
pix2pixHD++ w/ SPADE	34.4	39.0	62.2
pix2pixHD++	32.7	38.3	58.8
pix2pixHD++ w/o Sync Batch Norm	27.4	31.8	51.1
pix2pixHD++ w/o Sync Batch Norm, and w/o Spectral Norm	26.0	31.9	52.3
pix2pixHD [40]	14.6	20.3	58.3

Table 5: Additional ablation study results regarding mIoU scores: the table shows that both the perceptual loss and GAN feature matching loss terms are important. Making the discriminator deeper does not lead to a performance boost. The table also shows that the components (Synchronized Batch Normalization, Spectral Normalization, TTUR, Hinge loss, and SPADE) used in the proposed method also helps our strong baseline, pix2pixHD++.

Table 5 provides additional ablation study results analyzing the contribution of individual components in the proposed method. We first find that both of the perceptual loss and GAN feature matching loss inherited from the learning objective function of the pix2pixHD [40] are important. Removing any of them leads to a performance drop. We also find that increasing the depth of the discriminator by inserting one more convolutional layer to the top of the pix2pixHD discriminator does not lead to a performance boost.

In Table 5, we also analyze the effectiveness of each component used in our strong baseline, the pix2pixHD++ method, derived from the pix2pixHD method. We found that the spectral norm, synchronized batch norm, TTUR [15], and hinge loss all contribute to the performance boost. However, with adding the SPADE to the strong baseline, the performance further improves. Note that pix2pixHD++ w/o Sync Batch Norm and w/o Spectral Norm still differs from pix2pixHD in that it uses the hinge loss, TTUR, a large batch size, and Glorot initialization [11].

C. Additional Results

In Figure 16, 17, and 18, we show additional synthesis results from the proposed method on the COCO-Stuff and ADE20K datasets with comparison to those from the CRN [7] and pix2pixHD [40] methods.

In Figure 19 and 20, we show additional synthesis results from the proposed method on the ADE20K-outdoor and Cityscapes datasets with comparison to those from the CRN [7], SIMS [35], and pix2pixHD [40] methods.

In Figure 21, we show additional multi-modal synthesis results from the proposed method. As sampling different $\mathbf{z}$ from a standard multivariate Gausssian distribution, we synthesize images of diverse appearances.

In the accompanying video, we demonstrate our semantic image synthesis interface. We show how a user can create photorealistic landscape images by painting semantic labels on a canvas. We also show how a user can synthesize images of diverse appearances for the same semantic segmentation mask as well as transfer the appearance of a provided style image to the synthesized one.

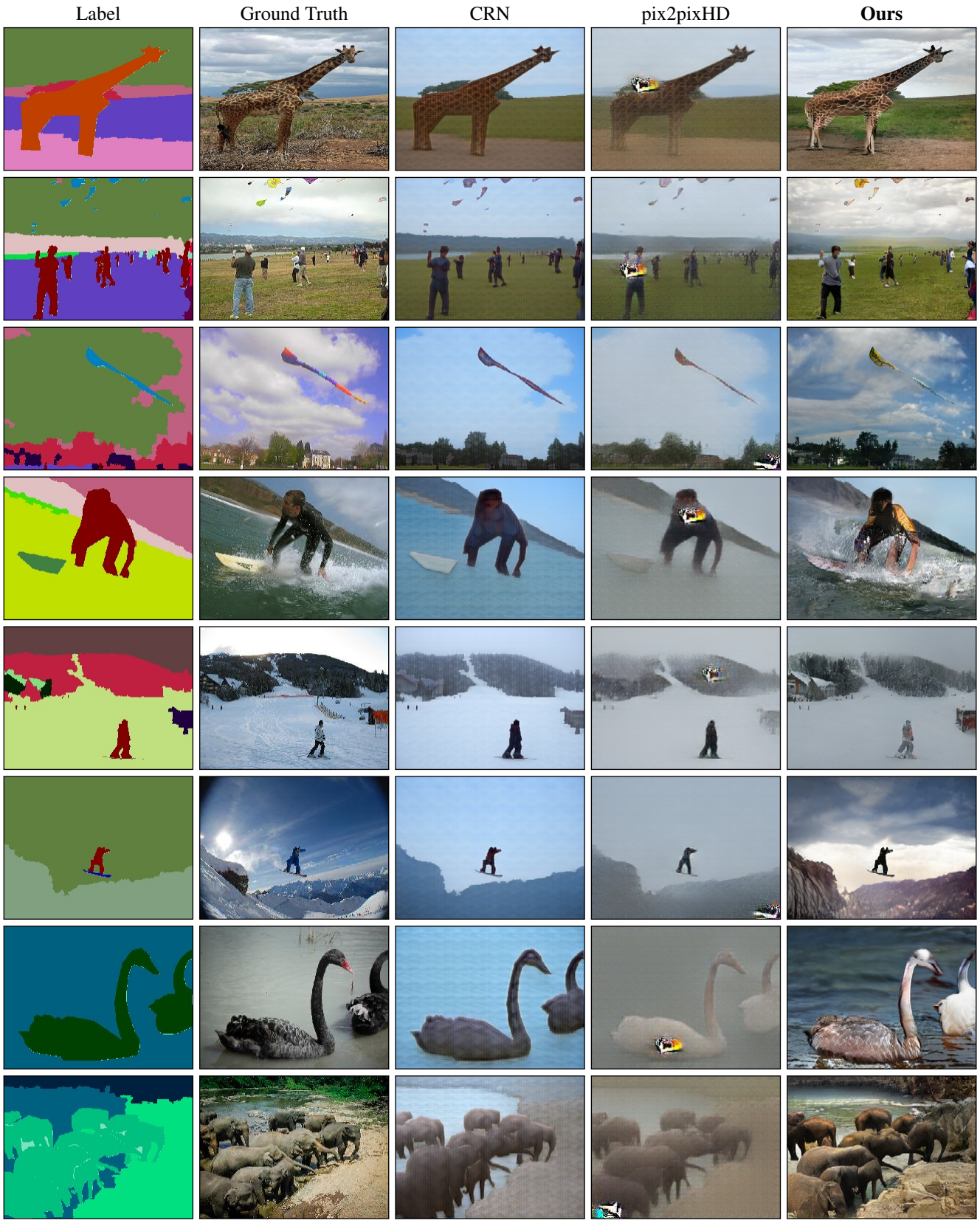

Figure 16: Additional results with comparison to those from the CRN [7] and pix2pixHD [40] methods on the COCO-Stuff dataset.

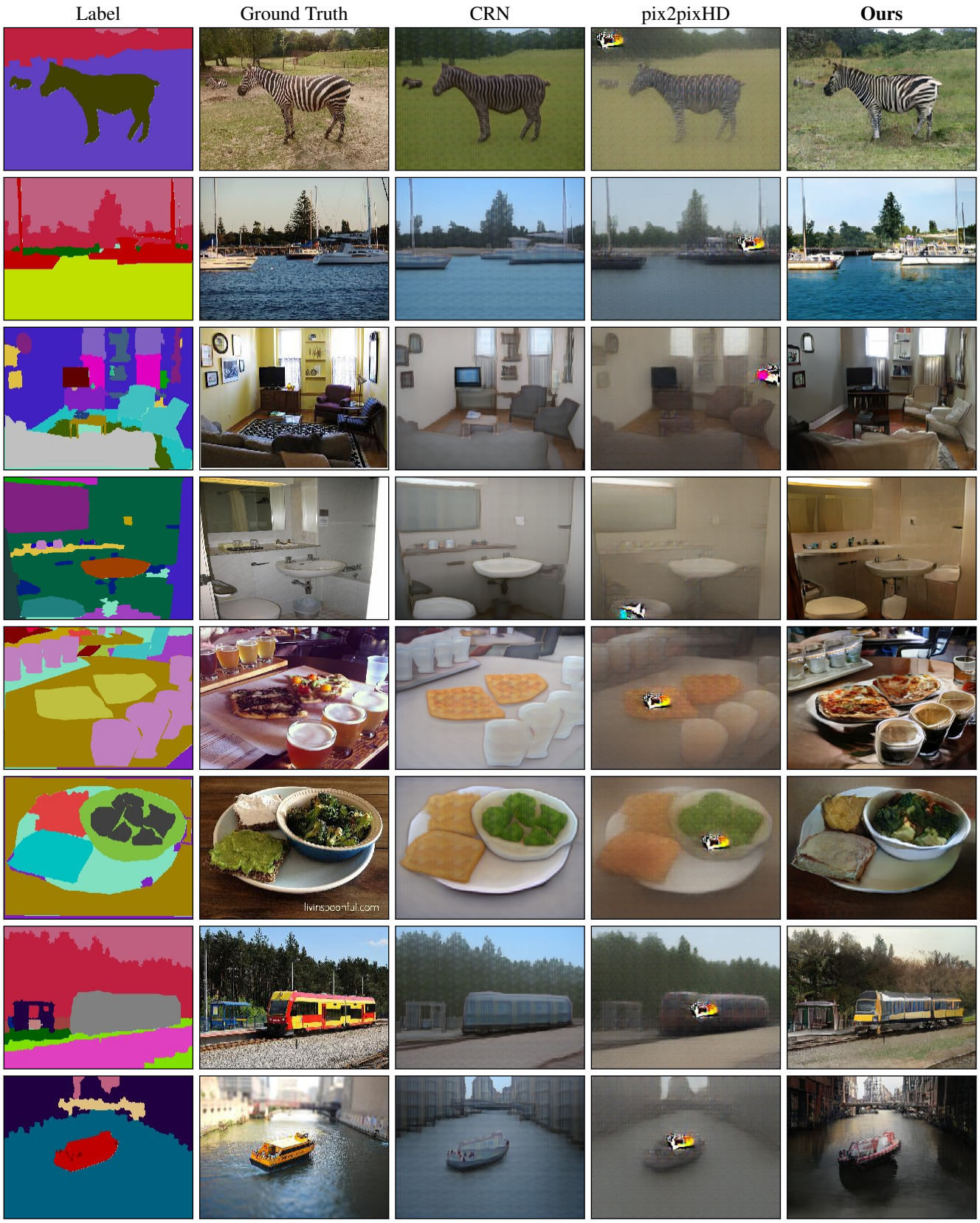

Figure 17: Additional results with comparison to those from the CRN [7] and pix2pixHD [40] methods on the COCO-Stuff dataset.

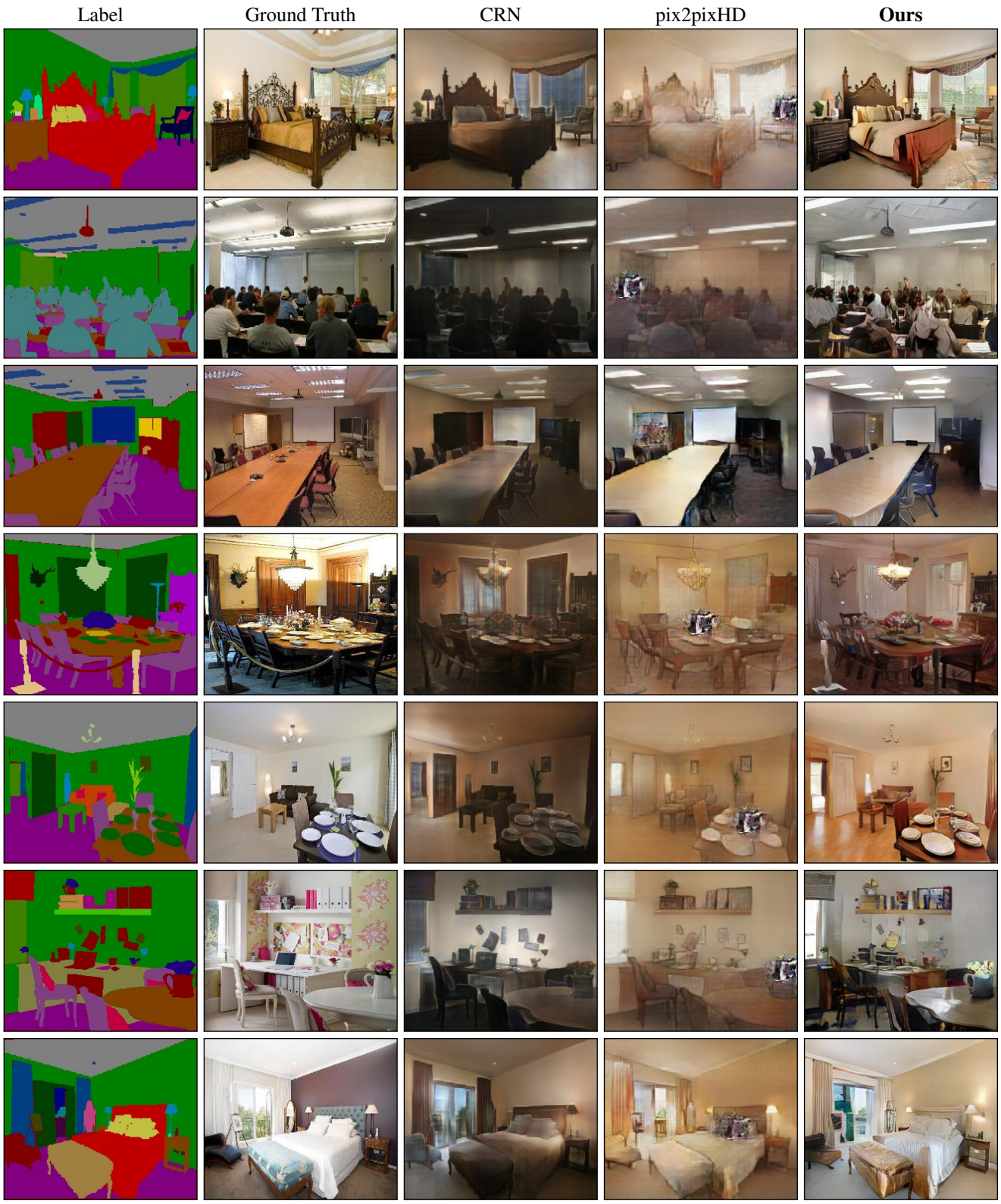

Figure 18: Additional results with comparison to those from the CRN [7] and pix2pixHD [40] methods on the ADE20K dataset.

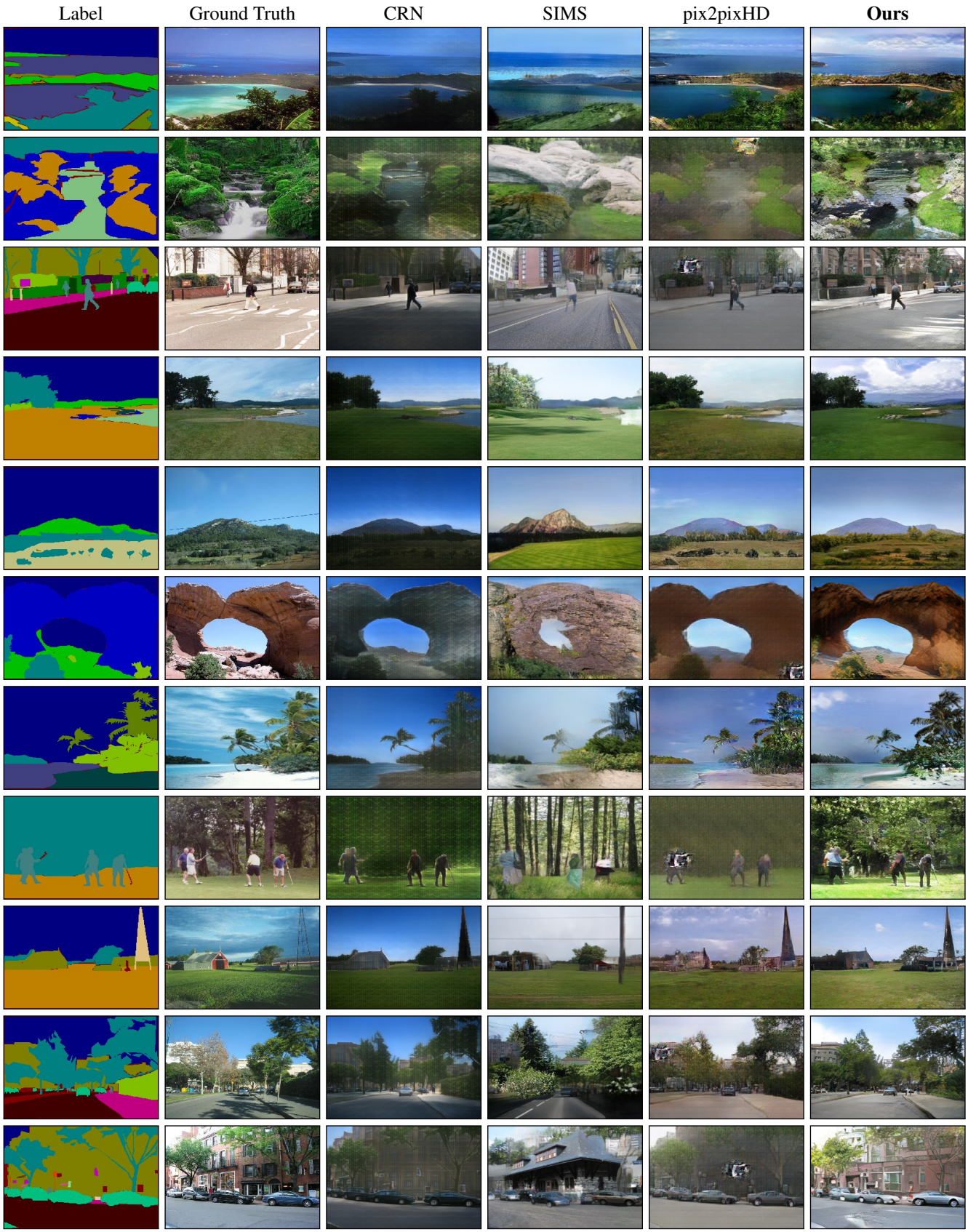

Figure 19: Additional results with comparison to those from the CRN [7], SIMS [35], and pix2pixHD [40] methods on the ADE20K-outdoor dataset.

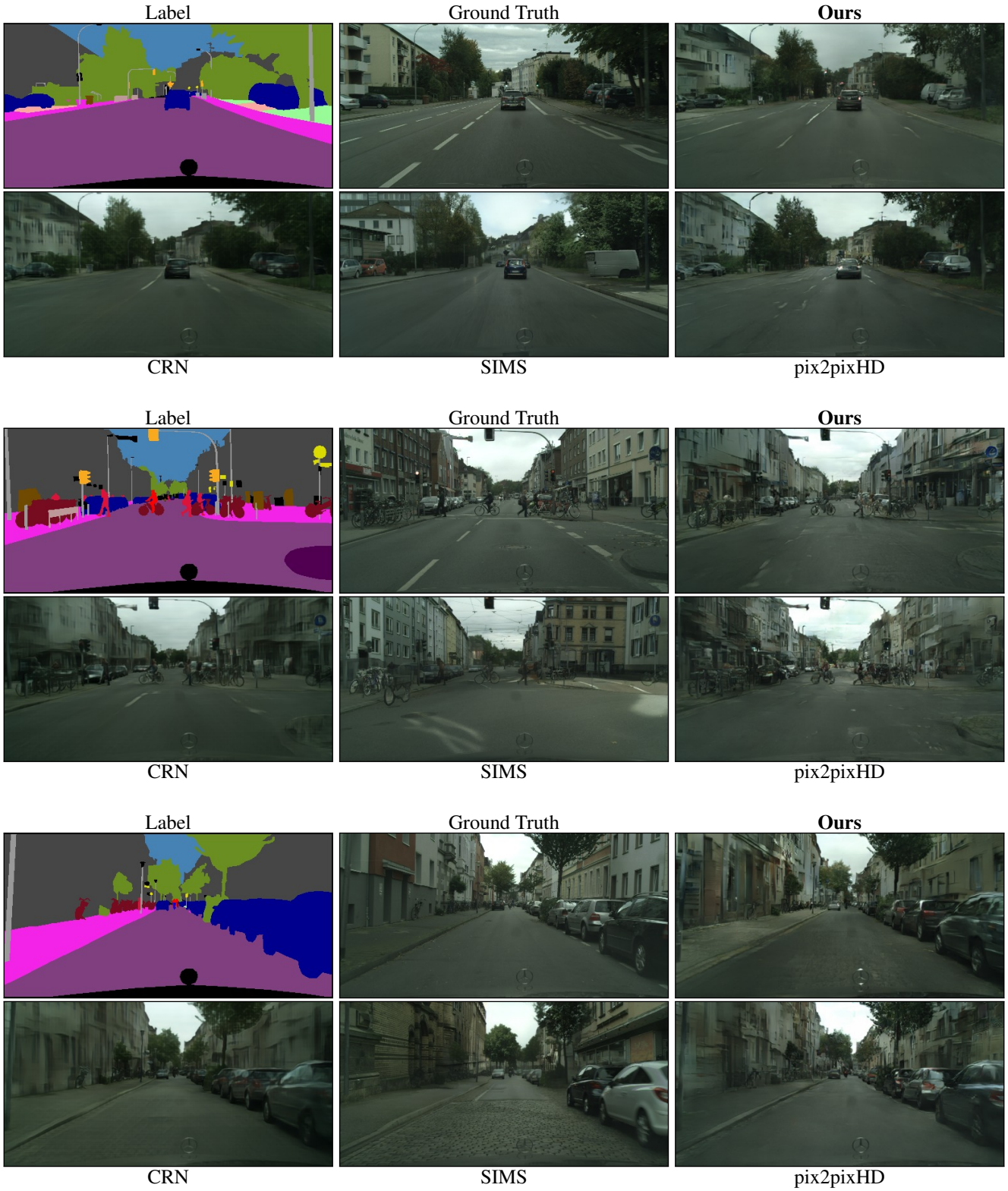

Figure 20: Additional results with comparison to those from the CRN [7], SIMS [35], and pix2pixHD [40] methods on the Cityscapes dataset.

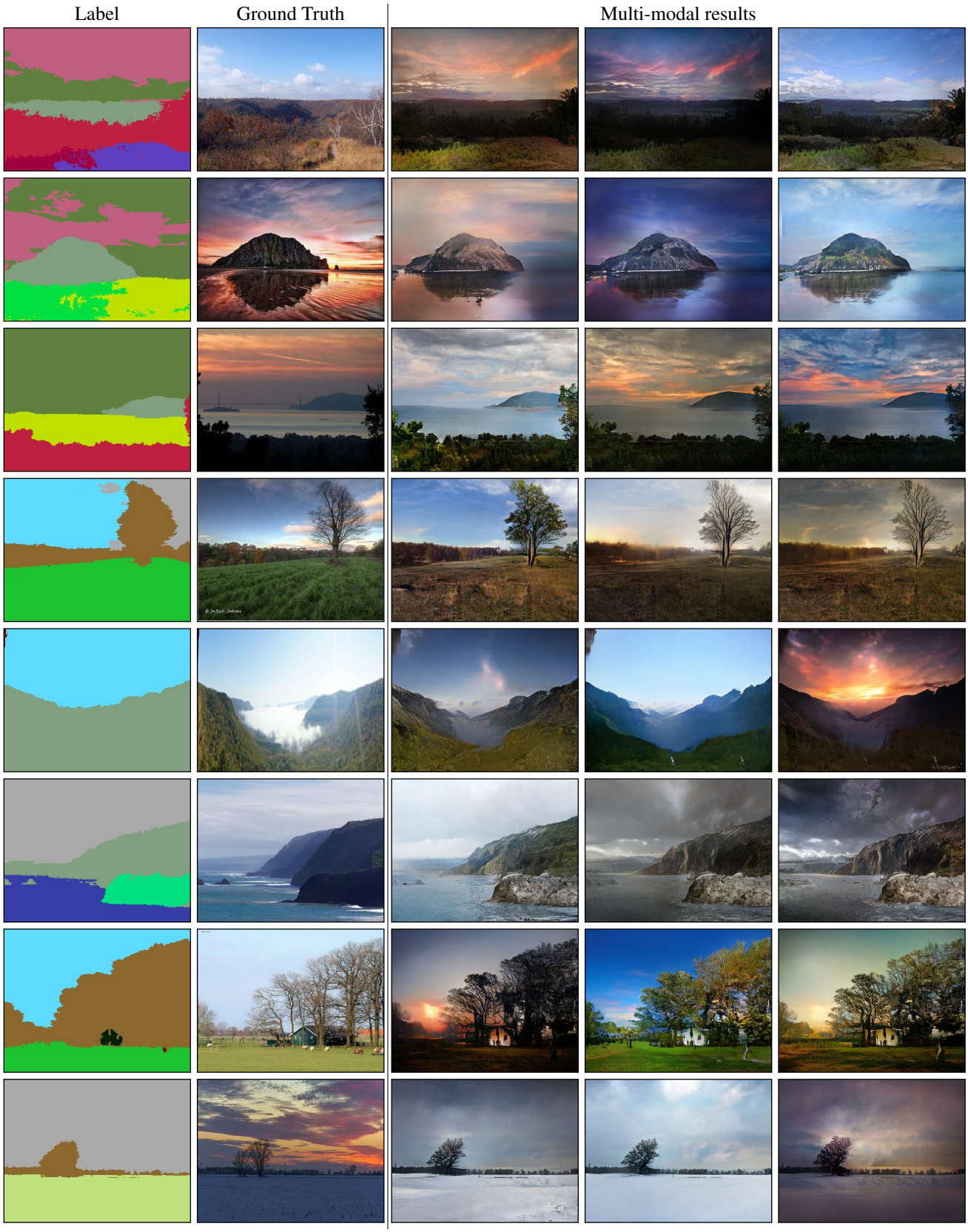

Figure 21: Additional multi-modal synthesis results on the Flickr Landscapes Dataset. By sampling latent vectors from a standard Gaussian distribution, we synthesize images of diverse appearances.