# Diverse Image Synthesis from Semantic Layouts via Conditional IMLE

## Supplementary Material


Ke Li*
UC Berkeley
ke.li@eecs.berkeley.edu

Tianhao Zhang*
Nanjing University
bryanzhang@smail.nju.edu.cn

Jitendra Malik
UC Berkeley
malik@eecs.berkeley.edu


## Appendix A. Implementation Details

**One-Hot Embedding of Semantic Layout** The segmentation map $\mathbf{x}$ that is fed as input to the neural net $T_\theta$ is encoded in a one-hot fashion. Specifically, each pixel in the segmentation map is represented as a one-hot encoding of the semantic category it belongs to, that is

$$\mathbf{x}^{i,j,p} = \begin{cases} 1, \text{pixel at location } (i,j) \text{ belongs to class } p \\ 0, \text{otherwise} \end{cases}$$

**Distance Metric** For the distance metric $\mathcal{L}(\cdot,\cdot)$, we use the loss function used by CRN, namely a perceptual loss function based on VGG-19 features [2]:

$$\mathcal{L}(\mathbf{y},\widetilde{\mathbf{y}}) = \sum_{i=1}^{l} \lambda_i \left\| \Phi_i(\mathbf{y}) - \Phi_i(\widetilde{\mathbf{y}}) \right\|_1 \tag{1}$$

Here $\Phi_1(\cdot), \cdots, \Phi_l(\cdot)$ represents the feature outputs of the following layers in VGG-19: "conv1_2", "conv2_2", "conv3_2", "conv4_2", and "conv5_2". Hyperparameters $\{\lambda_i\}_{i=1}^{l}$ are set such that the loss of each layer makes the same contribution to the total loss. We use this loss function as the distance metric in the IMLE objective.

**Dataset Rebalancing** We first rebalance the dataset to increase the chance of rare images being sampled when populating the training batch $S$. To this end, for each training image, we calculate the average colour of each category in that image. Then for each category, we estimate the distribution of the average colour of the category over different training images using a Gaussian kernel density estimate (KDE).

More concretely, we compute the average colour of each semantic category $p$ for each image $k$, which is a three-dimensional vector:

$$\mathbf{c}_k(p) = \frac{\sum_{i=1}^{h}\sum_{j=1}^{w} \mathbf{1}\left[\mathbf{x}_k^{i,j} = p\right] \mathbf{y}_k^{i,j}}{\sum_{i=1}^{h}\sum_{j=1}^{w} \mathbf{1}\left[\mathbf{x}_k^{i,j} = p\right]} = \frac{\sum_{i=1}^{h}\sum_{j=1}^{w} \mathbf{x}_k^{i,j,p}\mathbf{y}_k^{i,j}}{\sum_{i=1}^{h}\sum_{j=1}^{w} \mathbf{x}_k^{i,j,p}}$$

For each category $p$, we consider the set of average colours for that category in all training images, i.e.: $\{\mathbf{c}_k(p) | k \in \{1,\ldots,n\}$ such that class $p$ appears in $\mathbf{x}_k\}$. We then fit a Gaussian kernel density estimate to this set of vectors and obtain an estimate of the distribution of average colours of category $p$. Let $D_p(\cdot)$ denote the estimated probability density function (PDF) for category $p$. We define the *rarity score* of category $p$ in the $k^{\text{th}}$ training image as follows:

$$R_p(k) = \begin{cases} \frac{1}{D_p(\mathbf{c}_k(p))} & \text{class } p \text{ appears in } \mathbf{x}_k \\ 0 & \text{otherwise} \end{cases}$$

When populating training batch $S$, we allocate a portion of the batch to each of the top five categories that have the largest overall area across the dataset. For each category, we sample training images with a probability in proportion of their rarity scores. Effectively, we upweight images containing objects with rare appearance.

*Equal contribution.

The rationale for selecting the categories with the largest areas is because they tend to appear more frequently and be visually more prominent. If we were to allocate a fixed portion of the batch to rare categories, we would risk overfitting to images containing those categories.

**Loss Rebalancing** The same training image can contain both common and rare objects. Therefore, we modify the loss function so that the objects with rare appearance are upweighted. For each training pair $(\mathbf{x}_k, \mathbf{y}_k)$, we define a rarity score mask $\mathcal{M}^k \in \mathbb{R}^{h \times w \times 1}$:

$$\mathcal{M}^k_{i,j} = R_p(k) \quad \text{if pixel } (i,j) \text{ belongs to class } p$$

We then normalize $\mathcal{M}^k$ so that every entry lies in $(0, 1]$:

$$\widehat{\mathcal{M}}^k = \frac{1}{\max_{i,j} \mathcal{M}^k_{i,j}} \mathcal{M}^k$$

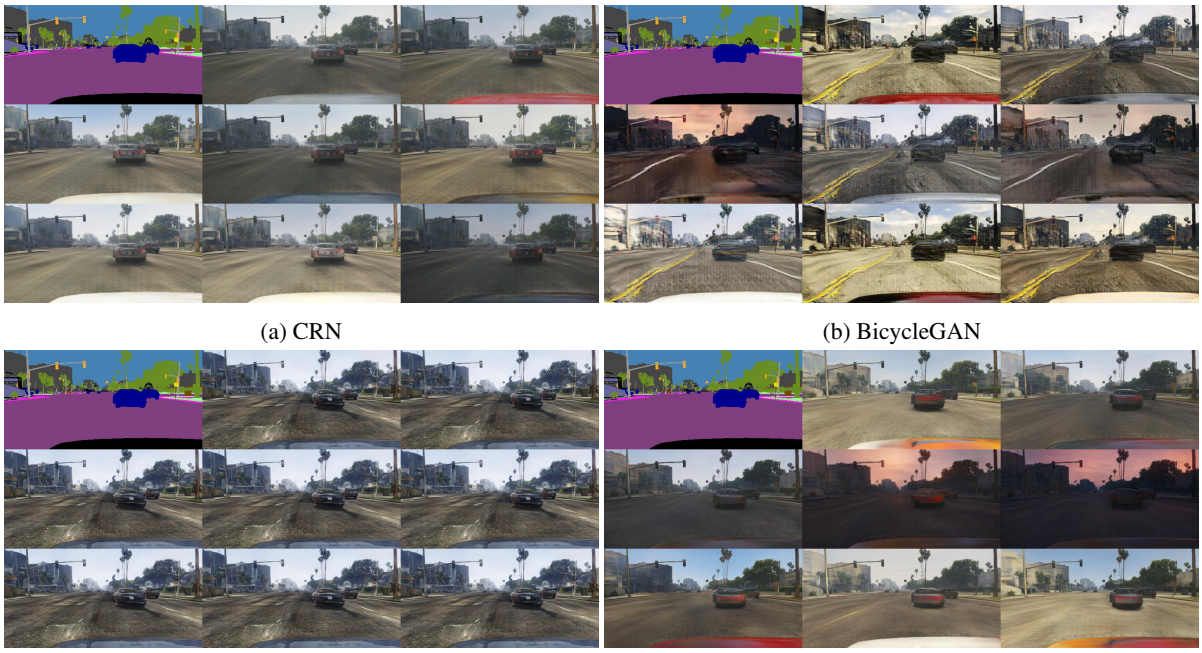
The mask is then used to weight different pixels differently the loss function (1). Let $\widehat{\mathcal{M}}$ be the normalized rarity score mask associated with the training pair $(\mathbf{x}, \mathbf{y})$. The new loss $\mathcal{L}$ becomes:

$$\mathcal{L}(\mathbf{y}, \widetilde{\mathbf{y}}) = \sum_{i=1}^{l} \lambda_l \left\| \widehat{\mathcal{M}}_i \circ [\Phi_i(\mathbf{y}) - \Phi_i(\widetilde{\mathbf{y}})] \right\|_1$$

Here $\widehat{\mathcal{M}}_i$ is the normalized rarity score mask $\widehat{\mathcal{M}}$ downsampled to match the size of $\Phi_i(\cdot)$, and $\circ$ denotes the element-wise product.

## Appendix B. Baselines with Proposed Rebalancing Scheme

A natural question is whether applying the proposed rebalancing scheme to the baselines would result in a significant improvement in the diversity of generated images. We tried this and found that the diversity is still lacking; the results are shown in Figure 1. The LPIPS score of CRN only improves slightly from 0.12 to 0.13 after dataset and loss rebalancing are applied. It still underperforms our method, which achieves a LPIPS score of 0.19. The LPIPS score of Pix2pix-HD showed no improvement after applying dataset rebalancing; it still ignores the latent input noise vector. This suggests that the baselines are not able to take advantage of the rebalancing scheme. On the other hand, our method is able to take advantage of it, demonstrating its superior capability compared to the baselines.



(a) CRN (b) BicycleGAN

(c) Pix2pix-HD (d) Ours

Figure 1: Samples generated by baselines with proposed rebalancing scheme, compared to the samples generated by our method. As shown, even with the proposed rebalancing scheme, the samples generated by CRN and Pix2pix-HD exhibit far less diversity than the samples generated by our method, and the samples generated by BicycleGAN are both less diverse and contain more artifacts than the samples generated by our method.

## Appendix C. Additional Results

All videos that we refer to below are available at http://people.eecs.berkeley.edu/~ke.li/projects/imle/scene_layouts.

### C.1. Video of Interpolations

We generated a video that shows smooth transitions between different renderings of the same scene. Frames of the generated video are shown in Figure 2.

(a)

(b)

Figure 2: Frames of video generated by smoothly interpolating between latent noise vectors.

## C.2. Video Generation from Evolving Scene Layouts

We generated videos of a car moving farther away from the camera and then back towards the camera by generating individual frames independently using our model with different semantic segmentation maps as input. For the video to have consistent appearance, we must be able to consistently select the same mode across all frames. In Figure 3, we show that our model has this capability: we are able to select a mode consistently by using the same latent noise vector across all frames.

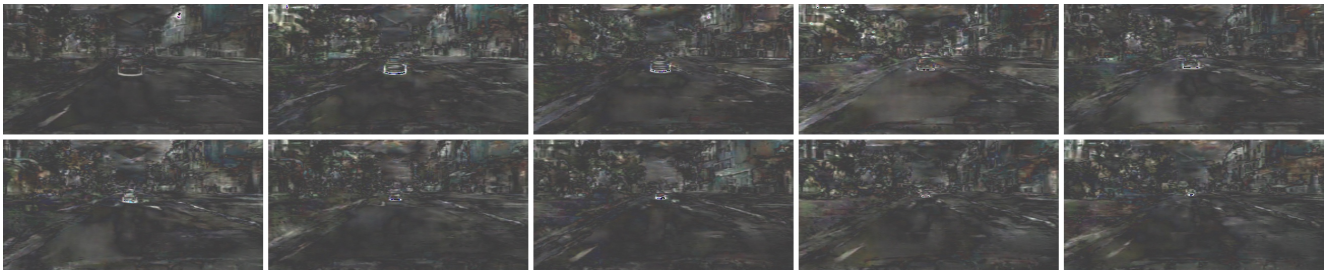
(a)

(b)

Figure 3: Frames from two videos of a moving car generated using our method. In both videos, we feed in scene layouts with cars of varying sizes to our model to generate different frames. In (a), we use the same latent noise vector across all frames. In (b), we interpolate between two latent noise vectors, one of which corresponds to a daytime scene and the other to a night time scene. The consistency of style across frames demonstrates that the learned space of latent noise vectors is semantically meaningful and that scene layout and style are successfully disentangled by our model.

Here we demonstrate one potential benefit of modelling multiple modes instead of a single mode. We tried generating a video from the same sequence of scene layouts using pix2pix [1], which only models a single mode. (For pix2pix, we used a pretrained model trained on Cityscapes, which is easier for the purposes of generating consistent frames because Cityscapes is less diverse than GTA-5.) In Figure 4, we show the difference between adjacent frames in the videos generated by our model and pix2pix. As shown, our model is able to generate consistent appearance across frames (as evidenced by the small difference between adjacent frames). On the other hand, pix2pix is not able to generate consistent appearance across frames, because it arbitrarily picks a mode to generate and does not permit control over which mode it generates.

(a)



(b)

Figure 4: Comparison of the difference between adjacent frames of synthesized moving car video. Darker pixels indicate smaller difference and lighter pixels indicate larger difference. (a) shows results for the video generated by our model. (b) shows results for the video generated by pix2pix [1].

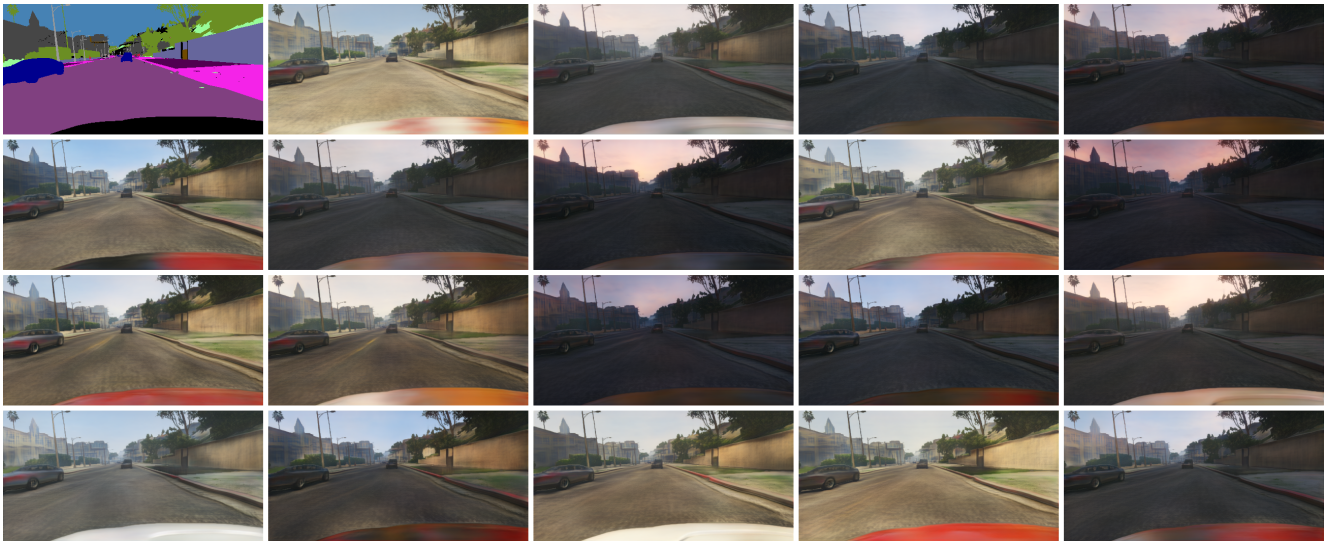
## Appendix D. More Generated Samples

(a)

(b)

Figure 5: Samples generated by our model. The image at the top-left corner is the input semantic layout and the other 19 images are samples generated by our model conditioned on the same semantic layout.

(a)



(b)

Figure 6: Samples generated by our model. The image at the top-left corner is the input semantic layout and the other 19 images are samples generated by our model conditioned on the same semantic layout.

## References


[1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017. 6

[2] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1