CSG0: Continual Urban Scene Generation with Zero Forgetting

Himalaya Jain[1,*] Tuan-Hung Vu[1,*] Patrick Pérez[1] Matthieu Cord[1,2]

[1]Valeo.ai, Paris, France [2]Sorbonne University, Paris, France

Abstract

With the rapid advances in generative adversarial networks (GANs), the visual quality of synthesised scenes keeps improving, including for complex urban scenes with applications to automated driving. We address in this work a continual scene generation setup in which GANs are trained on a stream of distinct domains; ideally, the learned models should eventually be able to generate new scenes in all seen domains. This setup reflects the real-life scenario where data are continuously acquired in different places at different times. In such a continual setup, we aim for learning with zero forgetting, i.e., with no degradation in synthesis quality over earlier domains due to catastrophic forgetting. To this end, we introduce a novel framework that not only (i) enables seamless knowledge transfer in continual training but also (ii) guarantees zero forgetting with a small overhead cost. While being more memory efficient, thanks to continual learning, our model obtains better synthesis quality as compared against the brute-force solution that trains one full model for each domain. Especially, under extreme low-data regimes, our approach outperforms the brute-force one by a large margin.

1. Introduction

Visual scene synthesis with generative adversarial networks (GANs) conditioned on input semantic segmentation masks is progressing fast. Since the early work of Pix2Pix [6], many architecture designs and learning strategies [8, 12, 14, 19] were proposed to push forward the synthesis quality, making generated images look more and more realistic. However, most existing works are limited to a single-domain setting, *i.e.*, once trained, the GAN can only generate images close to the distribution of the training domain. Recent works propose techniques for fine-tuning a pre-trained GAN [9, 21], training specific parameters [16] or identifying the most favourable regions on the learned manifold [20], to help transfer learning to new domains.

∗ Equal contribution

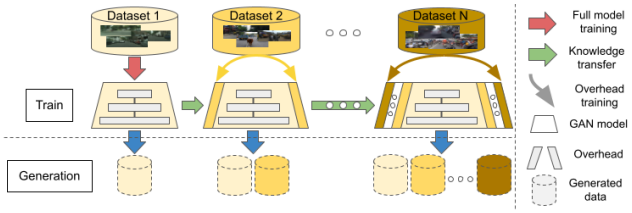

Figure 1. **Overview of the proposed framework**. Our continual setup for urban-scene generation involves a stream of datasets, with GANs trained from one dataset to another. Our framework makes use of the knowledge learned from previous domains and adapts to new ones with a small overhead. Best viewed in color.

We here tackle the task of continual urban-scene generation for multiple domains using GANs (see overview in Fig. 1). We address a realistic scenario that takes into account the continual property of driving data acquisition: data are continually collected from different places. Such a reality asks for efficient mechanisms to continuously extend generative models, which were previously trained, to newly collected datasets. The straight-forward solution is to fine-tune the current model using the new data. However this solution greatly suffers from the "catastrophic forgetting" phenomena, often met with data-driven models [15, 22, 25].

In this work, we aim for preserving at best the synthesis quality in all domains, *i.e.*, we want to avoid catastrophic forgetting altogether, seeking instead no- or zero-forgetting. To this end, one could train a separate model for each domain, at the expense of significant memory consumption when the number of domains grows. We argue that, although urban-scene datasets look different in color/texture and even have different label spaces, they share lots of structure, *e.g.*, scene arrangements and shapes of objects in shared classes. Therefore, given a GAN pre-trained on a related domain, it should be unnecessary to have all model's weights learned again for the new domain; instead, only a minimal set of parameters should need learning. Based on this rationale, we propose a novel framework for continual scene generation with zero-forgetting. Building around the idea of modulating network weights, we approach the continual task with care, analyze the requirements for suitable architecture designs and learning strategies to handle

the extension of the label space in the new domains, with minimal overhead and large adaptability. Effectively, we seek a good trade-off between the complexity overhead required by zero-forgetting and the image synthesis quality. In brief, the main contributions of this paper are as follows:

- We address the novel task of continual learning of GAN for semantic scene generation, where each new domain comes with new semantic classes and new visual styles. To the extent of our knowledge, this is the first work addressing continual scene generation.
- We propose a modular approach, named CSG0, to tackle the problem and evaluate the contribution of each module in the context of urban scenes.
- We show in various continual setups, covering both synthetic-to-real and real-to-real scenarios, that CSG0 outperforms state-of-the-art models trained on individual domains.
- We demonstrate the merit of CSG0 in low-data regimes in which learning is done with only a few tens of samples.

2. Related work

Scene generation with GANs. Image synthesis with GANs conditioned on semantic maps has progressed significantly. Pix2Pix [6] is the first work to address this problem, using an encoder-decoder generator with PatchGAN discriminator. Pix2PixHD [19] proposes a coarse-to-fine generator architecture and multiple PatchGAN discriminators to generate high-resolution images. SPADE [12] proposes a spatially-adaptive normalization layer that modulates the feature maps of the generator. The modulating parameters are predicted based on the input semantic map. This explicit use of the input semantic map to control the structure of the generated images improves the fidelity to it. Similarly, to have an explicit impact of the semantic map, CC-FPSE [8] proposes to predict convolutional kernels of the generator conditioned on it. OASIS [14] proposes a segmentation-based discriminator and shows that only the adversarial loss is sufficient to get high-fidelity generation unlike previous works that require a perceptual loss. OASIS generator extends the SPADE layers to take jointly noise and semantic map as input; this enhances the impact of noise and thus the diversity of the generated images. In our work, we use OASIS as our base framework for urban-scene generator and explore various directions for continual learning with it.

Continual learning for GANs. Continual learning for GANs was first addressed in [15], where elastic weight consolidation (EWC) [7] is used to avoid catastrophic forgetting. LifelongGAN [25] proposes to use knowledge distillation from the previous generator to the current one to address forgetting of the previous tasks. Memory replay GAN [22] uses the generated images of the previous tasks with the current task images to train the current generator. In [22, 25], all weights of the generator are fine-tuned, thus still suffering from some forgetting. In HyperlifelongGAN [24], convolutional filters of the generator are decomposed into the dynamic task-specific base filters and a deterministic generic weight matrix. Knowledge distillation is used to ensure good performance on the previous tasks. Recent works proposed to learn additional task-specific parameters while keeping the remaining generator frozen to preserve the performance on the previous tasks. Piggybank GAN [23] learns new task-specific filters. For the current task, these filters are used in combination with filters from a bank of filters learned on the previous tasks. GAN memory [1] proposes to learn task-specific weight modulation parameters, that is, task-specific mean and standard deviation of the weight matrices of the generator.

The existing continual GAN approaches are addressing continual learning for either unsupervised [1], class-conditioned [1,22,25] or image-conditioned [23–25] GANs. In this case, for each new task, the conditioning input domain (set of classes or images) is completely replaced by a new domain; thus there is no overlap or sharing between the input domains across the tasks. In this work, we focus on continual learning for the task of semantic scene generation. This brings some new aspects to continual GAN learning. In particular, the label space of the previous tasks should be extended rather than replaced when accommodating a new incoming domain. To our best knowledge, our work is the first to address continual learning for semantic scene generation. As will be explained in the next section, we take the no-forgetting approach where we only learn some task-specific new parameters while preserving the performance on the previous tasks.

Fine-tuning GANs. Fine-tuning refers to continuing on a new target dataset the training of a pre-trained model. The objective is to achieve best performance on the new domain or task, regardless of catastrophic forgetting. The first work to propose GAN fine-tuning is [21], which shows that a pre-trained generator can indeed be fine-tuned on a new dataset, thus requiring less data and learning iterations. FreezeD [9] proposes to freeze a few initial layers of the pre-trained discriminator while fine-tuning the rest on a new dataset. [11] learns new BatchNorm parameters to fine-tune the generator on a very small dataset. [16] proposes to learn class-specific BatchNorm parameters by using knowledge from BatchNorm parameters of the pre-trained conditional GAN. MineGAN [20] learns a miner network that produces latent codes for the pre-trained generator so as to gear it toward the new target distribution. In the second stage, all the networks –the miner, the generator and the discriminator– are fine-tuned on the target dataset.

While GAN fine-tuning is important, it is not directly applicable to continual learning without forgetting, which is our goal here for semantic urban-scene generation.

3. CSG0 for continual scene generation

We address the task of continual scene generation, conditioned on input semantic segmentation masks. On a continuous stream of N datasets, GAN training is done from one dataset to another. At inference, the main goal is to purposely synthesize images coming from any of the N domains. Starting from a GAN model pre-trained on previous domains, we want to reuse most of the learned weights and extend the model with small overhead (i) to leverage the knowledge learned from the previous domains and (ii) to handle the new domain with new classes using the added parameters.

To this end, several challenges must be overcome. First, as we are dealing with new classes in the new domain, the continual model must be re-designed so that it can accept those classes as inputs. Second, we need an efficient mechanism to reuse most of the parameters learned from the previous domains while allowing sufficient degrees of freedom so that the continual model can adapt to the new domain with a different style. Furthermore, by default, most GAN networks adopt Batch Normalization to stabilize training, which may not be ideal for our task where we want a more explicit control and adaptation of the domain's "style". In what follows, we present all technical details of the base generative model, in Section 3.1, and our proposed continual strategies, in Section 3.2.

3.1. GAN OASIS framework

We use the recent scene generator OASIS [14] as our base framework. As input it takes a 3D noise concatenated with one-hot segmentation maps. The input is fed at various levels using SPADE blocks [12]. The generator G has six ResNet blocks, each block is structured as SPADE-Conv-SPADE-Conv with a skip connection. The key idea of OASIS training lies in the design of the discriminator D, which is a segmentation-like network. This discriminator is trained not only to discriminate real/fake pixels, but also to classify real pixels into the correct semantic classes.

Denoting C the number of semantic classes at hand, $D(\cdot)$ thus outputs maps of size $H\times W\times(C+1)$ given an input image $\boldsymbol{x}$ of size $H\times W$; Accordingly, the generator $G(\cdot)$ produces images of the same size, given a binary semantic tensor $\boldsymbol{S}$ of size $H\times W\times C$ and a noise vector $\boldsymbol{z}$ (turned into a 3D tensor by replication over the pixel grid) as inputs.

For training, we use the objectives proposed in OASIS, that is, an adversarial loss for the generator and a segmentation loss (with an additional *fake* class) combined with LabelMix regularization for the discriminator. We provide below an abridged description of the loss terms; for more details readers could refer to [14].

Let $\boldsymbol{S}$ denote a one-hot input segmentation map, where $\boldsymbol{S}_{i,j,c} \in \{0,1\}$ is non zero only if pixel (i,j) is labelled as class c. The generator is trained to minimize,

$$\mathcal{L}_G = -\mathbb{E}_{(\boldsymbol{z},\boldsymbol{S})} \sum_{c=1}^{C} \alpha_c \sum_{i=1}^{H}\sum_{j=1}^{W} \boldsymbol{S}_{i,j,c} \log D(G(\boldsymbol{z},\boldsymbol{S}))_{i,j,c}, \tag{1}$$

where the α_c's are class-balancing weights, defined as inverse class frequencies, and expectation is approximated over pairs of random noise vectors and true semantic maps.

The discriminator loss is defined as

$$\begin{aligned}\mathcal{L}_D = &-\mathbb{E}_{(\boldsymbol{x},\boldsymbol{S})} \sum_{c=1}^{C} \alpha_c \sum_{i=1}^{H}\sum_{j=1}^{W} \boldsymbol{S}_{i,j,c} \log D(\boldsymbol{x})_{i,j,c} \\ &- \mathbb{E}_{(\boldsymbol{z},\boldsymbol{S})} \sum_{i=1}^{H}\sum_{j=1}^{W} \log D(G(\boldsymbol{z},\boldsymbol{S}))_{i,j,c=N+1},\end{aligned} \tag{2}$$

that is, the combination of the C-class cross-entropy loss for real pixels and the binary cross-entropy loss for fake ones; the first expectation is taken over real images and associated ground-truth segmentation maps.

The discriminator is further trained with LabelMix regularization. A LabelMix image is formed by mixing real and generated images associated with the same ground-truth segmentation map according to a binary mask $\boldsymbol{M}$, *i.e.*, $\text{LabelMix}(\boldsymbol{x}, G(\boldsymbol{z},\boldsymbol{S}), \boldsymbol{M}) = \boldsymbol{M}\boldsymbol{x} + (1-\boldsymbol{M})G(\boldsymbol{z},\boldsymbol{S})$, where operations are meant entry-wise. The discriminator is regularized to be pixel-wise consistent in its prediction on LabelMix images and on corresponding real and generated images before mixing.

While we use OASIS as the base for our continual learning scene generation framework and its evaluation, note that the approach is not limited to OASIS.

3.2. CSG0 Model

We now detail our continual strategies and introduce the continual scene generation model with zero-forgetting, namely CSG0. Figure 2 illustrates our architecture designs. We start CSG0 from the OASIS model G_o pre-trained on previous datasets with C_o semantic classes.

Extending the input label space. We now consider a new domain with C_n *new* classes. To accommodate it, we need to re-design the input layers of the generator to make them ingest semantic masks with C_o+C_n classes. To this end, we modify the SPADE blocks in OASIS to accept the new 3D semantic tensors composed of $(C_o$+$C_n)$-dim one-hot vectors. Originally, the first convolutional layer (`conv`) of SPADE contains 1024 kernels of size $C_o\times 3\times 3$. For new classes, we introduce a new `conv` layer, named "EL", standing for "Extented Labels", with similarly 1024 kernels, now of size $C_n\times 3\times 3$.

We then split the input 3D tensor into the "new" and "old" 3D tensors, corresponding to the new and old classes.

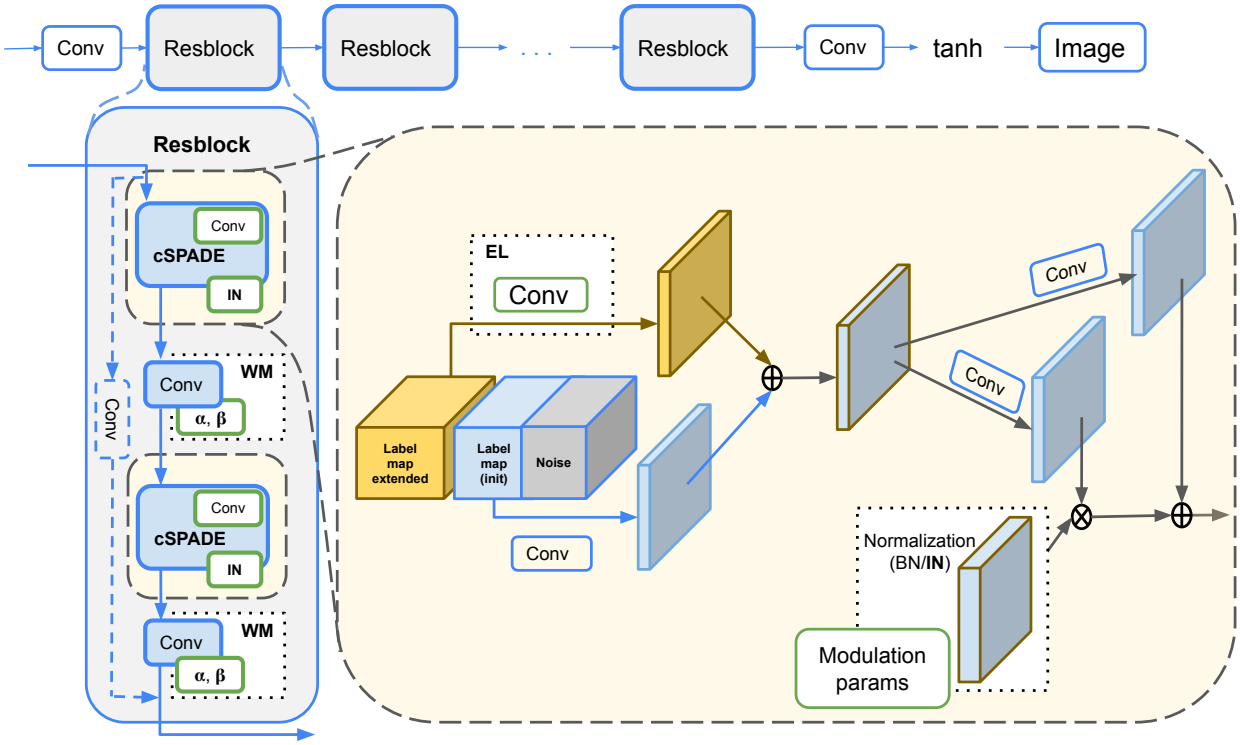

Figure 2. **Proposed framework for continual semantic image generation**. Starting off from an OASIS model shown on top, we propose new designs for continual learning. All modules drawn in blue are frozen during continual training, only green ones are learned. Yellow blocks stand for the newly introduced input in the new domain as well as its corresponding features. Details are given in Section 3.2.

Effectively, the two split tensors are composed of one-hot vectors of C_o-dim and C_n-dim respectively. Areas of old classes are encoded by zero vectors in the new 3D tensor, and vice versa. The two split 3D tensors are fed into corresponding `conv` layers, whose outputs are then summed up to obtain one final output. This new SPADE block, modified for continual learning, will be referred to as "cSPADE", to distinguish it from the original one. The structure of this block is represented in Figure 2, where the new input 3D tensor and the output of the new EL `conv` are highlighted in yellow, in contrast to the original 3D tensor and `conv` layer which are in blue.

Weight modulation. Weights of the convolutional layers encode most of the knowledge from the previous domains. We want to retain important information encoded in these weights while allowing certain adaptability to handle the new domain. To this end, we adopt the mAdaFM technique introduced in [1], which helps modulate the "style" of the `conv` layer. The technique brings to `conv` layers the idea of statistics modulation used for style transfer [3, 5]; intuitively, `conv` weights are regarded as network's "features" with domain-invariant and domain-specific parts. The key idea is to keep the domain-invariant part frozen while allowing learning in the domain-specific part, which eventually results in weight adaptation to the new domain.

In detail, given a `conv` layer with C_{in} input and C_{out} output channels, weight matrix $\mathbf{W} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times K \times K}$, bias vector $\boldsymbol{b} \in \mathbb{R}^{C_{\text{out}}}$ and kernel size K, we modulate those parameters in the new `conv` by defining the new weight matrix $\widehat{\mathbf{W}}$ and bias vector $\hat{\boldsymbol{b}}$ as:

$$\widehat{\mathbf{W}} = \boldsymbol{\alpha} \odot \frac{\mathbf{W} - \mathbf{M}}{\mathbf{S}} + \boldsymbol{\beta}, \qquad \hat{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{b}_{\text{conv}}, \tag{3}$$

where: $\mathbf{M}$ and $\mathbf{S} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ with entries $\mathbf{M}_{i,j}$ and $\mathbf{S}_{i,j}$ being the mean and standard deviation respectively of $(\mathbf{W}_{i,j,p,k})_{1 \le p,k \le K}$; $\odot$ is the Hadamard product; $\boldsymbol{\alpha} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ (scale), $\boldsymbol{\beta} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ and $\boldsymbol{b}_{\text{conv}} \in \mathbb{R}^{C_{\text{out}}}$ (shift) are new domain-specific parameters. The learning process on the new domain only updates these domain-specific parameters, while leaving the base parameters $\mathbf{W}$ and $\boldsymbol{b}$ untouched. In the Resblock of CSG0, we apply weight modulation for the two `conv` layers coming after each cSPADE block. The learnable parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are highlighted in the green boxes in Figure 2.

Instance normalization. Batch normalization (BN) and modulation of the activation in the generator have shown to have a strong influence on the style of the generated images. This is also especially true for Instance Normalization

(IN) [17]. Inspired by this observation that IN provides better control of the style, we extend our minimal EL setup by replacing BN with IN with affine transform. Note that using IN with affine transformation does not require any additional parameters compared to BN. The normalization block in Figure 2 shows this module in the full framework. With 'Modulation params' in the figure, we mark that these are dataset-specific parameters which are used to modulate the activation in the normalization layer.

4. Experiments

4.1. Experimental details

Continual set-ups. In this work, we focus on generating urban scenes, which leaves us a few dataset choices. Our selection criterion is to have datasets that have rather different visual styles and "private" semantic classes that do not exist in others. The final shortlist contains GTA5 [13], composed of synthetic urban scenes, Cityscapes [2], collected in Germany and Switzerland, Indian Driving Dataset (IDD) [18], acquired in India, and Mapillary [10] with scenes from all around the world. Addressed setups cover both synthetic-to-real and real-to-real scenarios, each with different numbers of domains. We address in our experiments the six following continual training sequences of 2 or 3 datasets:

- synth-to-real: GTA5→IDD and GTA5→Mapillary
- real-to-real: Cityscapes→IDD and Cityscapes→Mapillary
- synth-to-real-to-real: GTA5→IDD→Mapillary
- real-to-real-to-real: Cityscapes→IDD→Mapillary.

Continual training. In each continual set-up, we start off with a vanilla OASIS model trained on the first dataset. Pre-trained weights of this OASIS are used to initialize the continual CSG0 models. All new CSG0 parameters, which are dedicated to the next datasets, are newly initialized.

Evaluation protocol. In this work, we are mainly concerned with transferring with minimal overhead the knowledge learned from previous datasets to the current one, so as to achieve good synthesis quality. The most important is thus finding a sweet spot in the trade-off between complexity overhead and image quality. The former is measured by the number of newly introduced parameters, and the total number of parameters needed for generation in all domains. To assess the latter, we regard FID [4] as the main metric, similarly to [8, 12, 14, 19]. Following [14], we additionally report the mean intersection-over-union (mIoU) obtained when testing a pre-trained PSPNet [26] segmentation model on the generated data; this metric is named 'GAN-test'. We note the difficulty of quantitatively assessing the synthesis quality: on some metrics like GAN-test, good scores do not always match visual quality. In this regard, FID appears to be the most meaningful metric.

4.2. Main results

We organize results by the final target dataset in the continual stream, *i.e.*, IDD in Tab. 1 and Fig. 3, Mapillary in Tab. 2 and Fig. 4. In each table, we report results of ablated CSG0 models and the brute-force approach where one full-blown OASIS model is trained for each dataset.

IDD results. With IDD as final target domain, we consider two 2-domain scenarios: GTA5→IDD (synthetic to real) and Cityscapes→IDD (real to real). From the base OASIS model with 35 classes, pre-trained either on GTA5 or Cityscapes, CSG0 extends in both cases to 44 classes of which 9 are new, only existing in IDD.

The main results are reported in Tab. 1. In this table, 'cSPADE' stands for the basic CSG0 model that adopts the vanilla cSPADE block to accommodate new classes, *e.g.*, only introducing the EL `conv` layers. Training of the cSPADE model only learns parameters in these EL layers and keeps everything else untouched. This basic model, though only introducing a small overhead of 0.3M parameters, does not achieve good results in terms of FID and mIoU; the first column of Figure 3 visualizes some qualitative results. We still notice the colors and patterns of the previous domain, *i.e.*, GTA5 in this case; This is most noticeable for *street* and *vegetable* classes. The synthesized *sky* does not have the cloudy tone as in IDD. For new classes like *tuk-tuk*, the model cannot generate satisfactory results. When replacing the original BatchNorm layers in cSPADE block with InstanceNorm ones, denoted as 'cSPADE + IN' in Table 1, the continual model gets better in learning the new style of IDD. We observe significant improvements in the aforementioned classes. However, the overall realism level is quite limited, as reflected in the FID score, as well as in the qualitative results shown in column 2 of Figure 3.

With a larger overhead cost of 10.8M parameters, the final CSG0 model having the weight modulation strategy, denoted as 'cSPADE + IN + WM', obtains much better synthesis quality. With more learnable parameters, we observe a significant boost as compared to 'cSPADE + IN'. As shown in the third column of Figure 3, scenes generated by CSG0 are more realistic with similar color tones as in the IDD dataset. The *street* areas, sometimes mixing soil and asphalt, look very much like typical Indian roads.

We make some interesting findings when comparing CSG0 with the OASIS model that is trained on IDD, 'OASIS (I)' in Table 1. The OASIS model obtains a much worse FID of 55.3 while our CSG0 is able to reach 39.0 (GTA5→IDD) and 37.0 (Cityscapes→IDD). We posit that such improvements are brought by the knowledge transferred from the previous domains that our continual models are based on, *i.e.*, GTA5 and Cityscapes. We note that the OASIS model reported in Table 1 corresponds to the brute-force zero-forgetting solution, which resorts to training a

	Model	New params	Total params	G → I FID	G → I mIoU	C → I FID	C → I mIoU
CSG0	cSPADE	0.3M	71.5M	63.1	28.6	75.2	21.9
	cSPADE + IN	0.3M	71.5M	50.7	32.2	55.1	29.3
	cSPADE + IN + WM	10.8M	82.0M	**39.1**	**39.4**	**38.4**	**35.2**
	OASIS (I) [14]	71.4M	142.6M	55.3	41.0	*idem*	*idem*

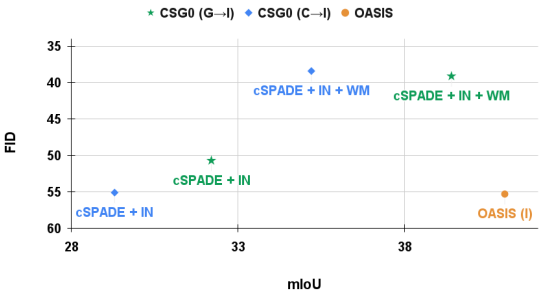

Table 1. **Performance on IDD of continual models**. All continual models are initialized from an OASIS model, trained either on GTA5 (G) or Cityscapes (C), having 72M parameters. Table to the *left*: the first four rows report results of ablated CSG0 models on different modules: cSPADE stands for the use of EL `conv` layer, IN and WM stand for InstanceNorm and 'Weight Modulation' strategies; the last row is for the 'OASIS (I)' model only trained on IDD. Some colored results in the table are plotted in the figure to the *right* with the corresponding colors. The OASIS model, though having good mIoU score, obtains worse FID as compared to CSG0 models. Such results are matched with synthesis quality of examples shown in Figure 3.

Figure 3. **Qualitative results on IDD**. Each row visualizes images synthesized from the same input semantic segmentation mask using different models. All continual models are initialized from the OASIS model that was trained on GTA5. Results of the basic CSG0 model that only has the vanilla cSPADE still retain the color and texture of GTA5. The instance normalization strategy helps facilitate style learning, resulting in images with more IDD-like tone. Having "Weight Modulation" (WM) improves further the quality. The 'OASIS (I)' model introduces visible artifacts, especially in "tree" and "sky" areas; also textures are quite repetitive. Our full CSG0 model produces the most realistic images with natural color tone and texture. Best viewed in color.

full OASIS model to handle a new dataset.

Our CSG0 model, with much smaller overhead cost, outperforms the 'OASIS (I)' in terms of FID. Though on the GAN-test metric, CSG0 models have slightly worse mIoU than the 'OASIS (I)', qualitative results of CSG0 are much more convincing. As visualized in the last column of Figure 3, the 'OASIS (I)' images have lots of artifacts; most visible are with the trees and the roads. In general, OASIS images exhibit weird color tone and contrast, making them look less realistic as compared to CSG0's.

In addition to the reported metrics, we have tried using generated images for data augmentation. In detail, we train PSPNet models using both real training IDD data and synthesized data. Compared to the model trained only on real IDD, which achieves a validation mIoU of 38.7%, the model using both real and CSG0 data achieves 39.5% validation mIoU, hence a slight gain of 0.8%.

Mapillary results. Mapillary has 64 semantic classes in total. We show results of two continual set-ups with either two or three datasets, respectively in Table 2 (a) and (b). We observe similar results among all CSG0 variants, proving the usefulness of proposed strategies. The CSG0 models achieve better FID than the brute-force OASIS model. To learn scene generation for the three datasets, note that

(a) Sequences of two datasets

	Model	New params	Total params	G → M FID	G → M mIoU	C → M FID	C → M mIoU
CSG0	cSPADE	0.8M	72.0M	48.6	21.6	57.6	17.6
CSG0	cSPADE + IN	0.8M	72.0M	32.6	23.3	38.9	21.8
CSG0	cSPADE + IN + WM	11.3M	82.5M	**25.1**	**26.5**	**24.5**	**25.5**
	OASIS (M) [14]	72M	143.2M	27.1	30.6	*idem*	*idem*

(b) Sequences of three datasets

	Model	New params	Total params	G→I → M FID	G→I → M mIoU	C→I → M FID	C→I → M mIoU
CSG0	cSPADE	0.8M	82.8M	56.2	21.6	65.8	16.9
CSG0	cSPADE + IN	0.8M	82.8M	34.1	23.9	39.4	22.0
CSG0	cSPADE + IN + WM	11.3M	93.3M	**24.0**	**26.6**	**25.4**	**26.1**
	OASIS (M) [14]	72M	214.6M	27.1	30.6	*idem*	*idem*

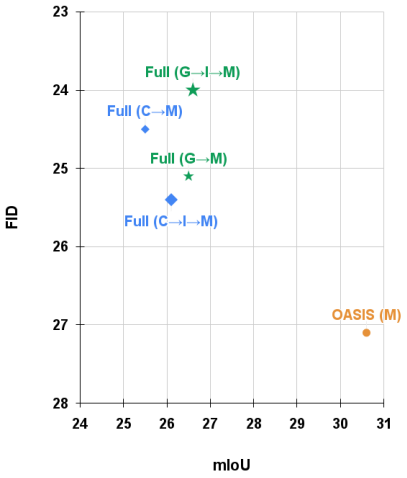

Table 2. **Performance on Mapillary of continual models**. Results of the two continual setups with sequences of (a) two datasets and (b) three datasets are reported. For each sub-table, the structure is the same as in Table 1. To the right, we plot the results of our full CSGO models (cSPADE + IN + WM) in both setups as well as the brute-force model 'OASIS (M)'. CSG0 models outperform OASIS in terms of FID, which shows in the image quality in Figure 4. Colors are matched between the table and figure.

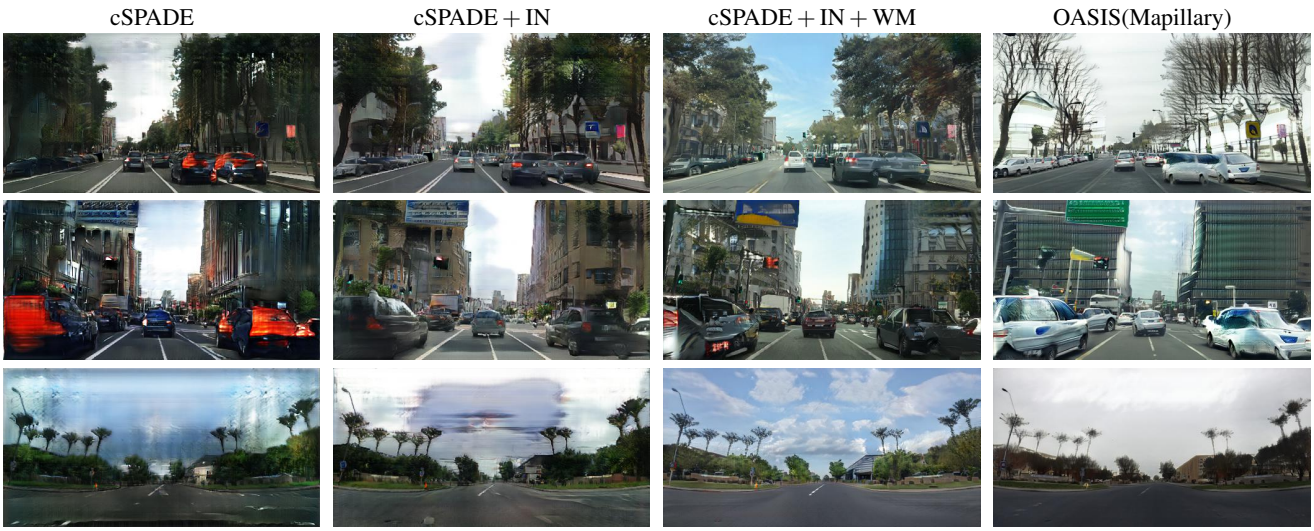

Figure 4. **Qualitative results on Mapillary**. Each row visualizes images synthesized from the same input semantic segmentation mask using different models. All continual models are initialized from the OASIS model that was trained on Cityscapes. The basic cSPADE model, with very small overhead cost, struggles to adapt to the style of Mapillary; indeed we still see the gloomy tone of the previous Cityscapes dataset. Having instance norm helps bring more style transfer effect. Our full CSG0 results look more natural and have less visible artifacts than other models. Best viewed in color.

our full CSG0 only needs 93.3M parameters in total, less than half of the parameters needed in the brute-force solution. Comparing results between two and three datasets, we do not see much difference in the real-to-real scenario, *i.e.*, Cityscapes→Mapillary *vs.* Cityscapes→IDD→Mapillary. In the synthetic-to-real scenario, with three datasets we notice small FID improvements for some CSG0 models, while GAN-test mIoUs are more or less the same. Under the same comparison, no significant changes are observed for the brute-force OASIS model. We conjecture that the number of domains the starting model has been trained on is not a very important factor in zero-forgetting continual scene generation. In Figure 4, we visualize some generated Mapillary-like images to demonstrate the differences in synthesis quality.

Advantage in cross-dataset sampling. One interesting property of CSG0 is its advantage in cross-dataset sampling, thanks to continual learning. In Figure 6, we illustrate some

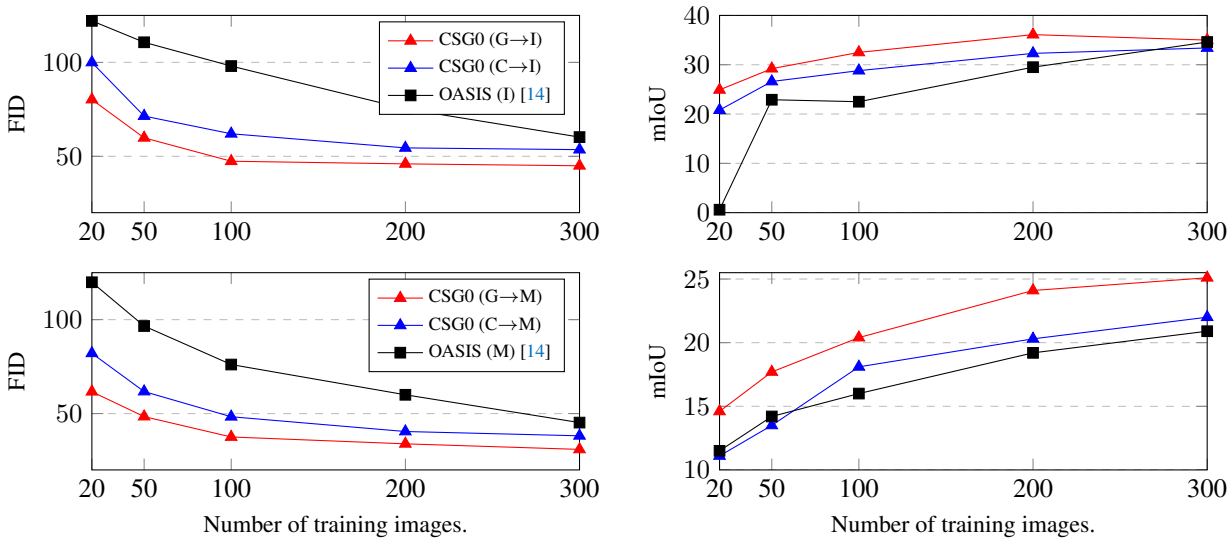

Figure 5. **Low-data regime training**. We compare our CSG0 ('cSPADE + IN + WM') models to 'OASIS (I)' and 'OASIS (M)'. Two sub-figures in the same row share the same legend. While the OASIS models overfit to the small subsets, thus getting comparatively higher FID, our CSG0 models benefit from transfer learning and achieve better scores. Similar gaps are observed in terms of GAN-test mIoU.

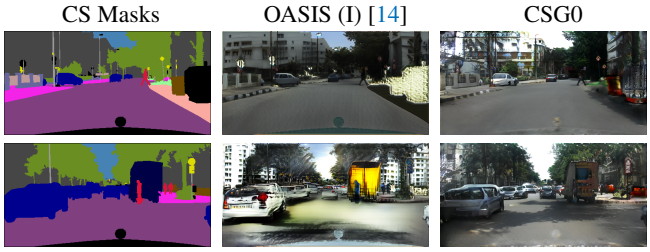

Figure 6. **Cross dataset sampling**. The figure shows images generated from IDD generators conditioned on Cityscapes (CS) masks. 'OASIS (I)' does not work well with CS masks thus producing visible artifacts, even on the shared classes.

results in which we sample images having Cityscapes layouts but with IDD style. That is made possible by feeding Cityscapes segmentation masks into models trained on IDD. The task is challenging: If the GAN models cannot handle well the shift in semantic distributions between Cityscapes and IDD, the synthesis quality would be greatly degraded. Compared to 'OASIS (I)', CSG0 produces more realistic results with fewer artifacts.

4.3. Low-data regime

Not only memory efficient, our continual framework allows seamless knowledge transfer from the previous dataset, encoded in the learned parameters, to a new yet related dataset. In this experiment, we showcase the merit of CSG0 when training with limited supervision, *i.e.*, the low-data regime, in the target dataset. In detail, we trained CSG0 models using IDD and Mapillary subsets of 20, 50, 100, 200 and 300 data samples respectively. Our models are initialized using the OASIS model pre-trained on either GTA5 or Cityscapes. We compare against the full OASIS model trained only on similar subsets.

Figure 5 plots performance curves of different models. CSG0 outperforms the OASIS models by significant margin, especially in the extreme set-ups with very little supervision, *e.g.*, only 20 and 50 training samples. Having more training data further closes the performance gap between CSG0 and OASIS. Results in the low-data regime confirm the benefit of transfer learning in our continual framework.

5. Conclusion

This work addresses a pragmatic task of continual semantic scene generation with zero-forgetting. We propose a modular framework, named CSG0, with novel architecture designs and strategies for this task. To showcase the merit of our framework, we conduct intensive experiments on various continual urban scene setups, covering both synthetic-to-real and real-to-real scenarios. Quantitative evaluations and qualitative visualizations demonstrate the interest of our CSG0 framework, which operates with minimal overhead cost (in terms of architecture size and training). Benefiting from continual learning, CSG0 outperforms the state-of-the-art OASIS model trained on single domains. We also provide experiments with three datasets to emphasize how well our strategy generalizes despite its cost constraints.

References

[1] Yulai Cong, Miaoyun Zhao, Jianqiao Li, Sijia Wang, and Lawrence Carin. Gan memory with no forgetting. *NeurIPS*, 2020. 2, 4

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 5

[3] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *ICLR*, 2017. 4

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeuRIPS*, 2017. 5

[5] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *CVPR*, 2017. 4

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 1, 2

[7] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 2017. 2

[8] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, et al. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. *NeurIPS*, 2019. 1, 2, 5

[9] Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Freeze the discriminator: a simple baseline for fine-tuning gans. In *CVPR AI for Content Creation Workshop*, 2020. 1, 2

[10] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *CVPR*, 2017. 5

[11] Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. In *ICCV*, 2019. 2

[12] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 1, 2, 3, 5

[13] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 5

[14] Edgar Schönfeld, Vadim Sushko, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. In *ICLR*, 2020. 1, 2, 3, 5, 6, 7, 8

[15] Ari Seff, Alex Beatson, Daniel Suo, and Han Liu. Continual learning in generative adversarial nets. *NeuRIPS*, 2017. 1, 2

[16] Mohamad Shahbazi, Zhiwu Huang, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Efficient conditional gan transfer with knowledge propagation across classes. In *CVPR*, 2021. 1, 2

[17] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint*, 2016. 5

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

[19] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 1, 2, 5

[20] Yaxing Wang, Abel Gonzalez-Garcia, David Berga, Luis Herranz, Fahad Shahbaz Khan, and Joost van de Weijer. Minegan: effective knowledge transfer from gans to target domains with few images. In *CVPR*, 2020. 1, 2

[21] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. Transferring gans: generating images from limited data. In *ECCV*, 2018. 1, 2

[22] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost Van de Weijer, and Bogdan Raducanu. Memory replay gans: learning to generate images from new categories without forgetting. *NeuRIPS*, 2018. 1, 2

[23] Mengyao Zhai, Lei Chen, Jiawei He, Megha Nawhal, Frederick Tung, and Greg Mori. Piggyback gan: Efficient lifelong learning for image conditioned generation. In *ECCV*, 2020. 2

[24] Mengyao Zhai, Lei Chen, and Greg Mori. Hyper-lifelonggan: Scalable lifelong learning for image conditioned generation. In *CVPR*, 2021. 2

[25] Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *ICCV*, 2019. 1, 2

[26] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 5