Layout-to-Image Translation with Double Pooling Generative Adversarial Networks

Hao Tang and Nicu Sebe

Abstract—In this paper, we address the task of layout-to-image translation, which aims to translate an input semantic layout to a realistic image. One open challenge widely observed in existing methods is the lack of effective semantic constraints during the image translation process, leading to models that cannot preserve the semantic information and ignore the semantic dependencies within the same object. To address this issue, we propose a novel Double Pooling GAN (DPGAN) for generating photo-realistic and semantically-consistent results from the input layout. We also propose a novel Double Pooling Module (DPM), which consists of the Square-shape Pooling Module (SPM) and the Rectangle-shape Pooling Module (RPM). Specifically, SPM aims to capture short-range semantic dependencies of the input layout with different spatial scales, while RPM aims to capture long-range semantic dependencies from both horizontal and vertical directions. We then effectively fuse both outputs of SPM and RPM to further enlarge the receptive field of our generator. Extensive experiments on five popular datasets show that the proposed DPGAN achieves better results than state-of-the-art methods. Finally, both SPM and SPM are general and can be seamlessly integrated into any GAN-based architectures to strengthen the feature representation. The code is available at https://github.com/Ha0Tang/DPGAN.

Index Terms—GANs, Pooling, Layout-to-Image Translation

I. INTRODUCTION

In Figure 1 we show a ‘Real vs. Fake’ game, in which a mix of ‘real’ images are collected from the real world and ‘fake’ images are generated by our GAN model. The goal is to guess which image is real and which one has been generated by the proposed GAN model. Now you can check your answers below.\footnote{Answers: (a) real; (b) fake; (c) real; (d) real; (e) fake; (f) fake.}

![Fig. 1: ‘Real vs. Fake’ game: Can you guess which image is real and which has been generated by the proposed DPGAN?](https://github.com/Ha0Tang/DPGAN)

In this paper, we aim to address the challenging layout-to-image translation task, which has a wide range of real-world applications such as content generation and image editing [2], [3], [4]. This task has been widely investigated in recent years [4], [5], [6], [7], [8], [9]. For example, Park et al. [5] proposed the GauGAN model with a novel spatially-adaptive normalization to generate realistic images from semantic layouts. Tang et al. [9] proposed the LGGAN framework with a novel local generator for generating realistic small objects and detailed local texture. Despite the interesting exploration of these methods, we can still observe blurriness and artifacts in their generated results because the existing methods lack an effective semantic dependency modeling to maintain the semantic information of the input layout, causing intra-object semantic inconsistencies such as the fence, buses, and pole generated by GauGAN in Figure 2.

To solve this limitation, we propose a novel Double Pooling GAN (DPGAN) and a novel Double Pooling Module (DPM). The proposed DPM consists of two sub-modules, i.e., Square-shape Pooling Module (SPM) and Rectangle-shape Pooling Module (RPM). In particular, SPM aims to capture short-range and local semantic dependencies, leading pixels within the same object to be correlated. Simultaneously, RPM aims to capture long-range and global semantic dependencies from both horizontal and vertical directions. Finally, we propose seven image-level and feature-level fusion strategies to effectively combine the outputs of both SPM and RPM for generating high-quality and semantically-consistent images.

Overall, the contributions of our paper are:

- We propose a novel Double Pooling GAN (DPGAN) for the challenging task of layout-to-image translation, which can effectively capture semantic dependencies among different locations of the input layout for generating photo-realistic and semantically-consistent images.
- We design a novel Double Pooling Module (DPM), which consists of the Square-shape Pooling Module (SPM) and...
Fig. 2: Visualization of our generated semantic maps compared with those from GauGAN [5] on Cityscapes. The proposed SPM (yellow grids) captures short-range and local dependencies, while the proposed RPM (i.e., HRPM and VRPM) captures long-range and global semantic correlations from both horizontal direction (green grids) and vertical direction (red grids), respectively. Equipped with both modules, the proposed DPGAN can enlarge the receptive field, thus improves the intra-object semantic consistency. Most improved regions are highlighted in the ground truths with white dash boxes.

- We conduct extensive experiments on five popular datasets with different image resolutions, i.e., ADE20K [10], DeepFashion [11], Cityscapes [12], CelebAMask-HQ [13], and Facades [14]. Both qualitative and quantitative results demonstrate that the proposed DPGAN is able to produce better results than state-of-the-art approaches.
dependencies, and a Rectangle-shape Pooling Module (RPM) which aims to capture short-range and local semantic dependencies. Our SPM is a novel spatially-adaptive normalization to generate realistic images [5], [6], [8], [9], [37], [38], [39], [40], [41]. For example, Park et al. [5] proposed GauGAN with a novel spatially-adaptive normalization to generate realistic images. Although GauGAN [5] has achieved promising results, we still observe unsatisfactory aspects mainly in the generated scene details and intra-object completions (see Figure 2), which we believe are mainly due to the lack of short-range and long-range semantic constraints in the input layout. The proposed SPM and RPM explicitly address this problem.

Pooling operations are commonly used in semantic segmentation tasks [42], [43], [44], [45] to improve the receptive field. For example, Zhao et al. [42] proposed a pyramid pooling module to capture the global context information in the scene parsing task. Hou et al. [44] proposed a new strip pooling, which considers a long but narrow kernel, for the scene parsing task. However, to the best of our knowledge, our idea of using pooling modules to capture both short-range and long-range semantic dependencies has not been investigated by any existing layout-to-image translation or even GAN-based image generation approaches.

III. DOUBLE POOLING GANs

Overview. We start by presenting the details of the proposed Dual Pooling GANs (DPGAN), which consists of a generator $G$ and discriminator $D$. An illustration of the proposed generator $G$ is shown in Figure 3, which mainly consists of three components, i.e., a feature extraction network $E$ extracting deep features from the input layout $L$, a Square-shape Pooling Module (SPM) modeling short-range and local semantic dependencies, and a Rectangle-shape Pooling Module (RPM) capturing long-range and global semantic dependencies from both horizontal and vertical directions. SPM and RPM together form our proposed Double Pooling Module (DPM). Moreover, we propose seven image-level and feature-level fusion methods to combine both the outputs of SPM and RPM.

**Feature Extraction Network.** As shown in Figure 3, the network $E$ receives the semantic layout $L$ as input and outputs the deep feature $f$, which can be formulated as,

$$f = E(L), \quad (1)$$

Then, $f$ is fed into the proposed SPM and RPM for learning short-range and long-range semantic dependencies, respectively.

**Square-Shape Pooling Module.** Existing layout-to-image translation methods such as [5], [6], [9], [40] directly use deep features generated by convolutional operations, leading to limited effective fields-of-views and thus generating different textures in the pixels with the same label. To model short-range and local semantic dependencies over the deep feature $f$, we propose a Square-shape Pooling Module (SPM). Note that the idea of the proposed SPM is inspired by the pyramid pooling module proposed in [42] and we extend the original module used in image segmentation to a completely different image generation task.

The framework of SPM is elaborated in Figure 4. Specifically, we first separately feed $f$ into $n$ square-shape pooling layers to produce $n$ new feature maps with different spatial scales. Consider the $n$-th pooling layer in Figure 4, whose
Fig. 5: The proposed Rectangle-shape Pooling Module (RPM) which consists of a Horizontal Rectangle-shape Pooling Module (HRPM) and a Vertical Rectangle-shape Pooling Module (VRPM), aiming to capture long-range and global semantic dependencies from horizontal and vertical direction, respectively. The yellow, green, and red grids represent short-dependency, horizontal long-dependency, and vertical long-dependency, respectively. The symbols ⊕ and ⊙ denote element-wise addition, and channel-wise concatenation, respectively.

input is the deep feature \( f_s \in \mathbb{R}^{C \times H \times W} \). The output is the pooled feature map \( f^n_s \in \mathbb{R}^{C/4 \times h_n \times w_n} \), where \((h_n, w_n)\) is the targeted output size of the \( n \)-th pooling layer. We then feed the pooled feature \( f^n_s \) through a convolutional layer for reducing the number of channels, leading to a new feature map \( \hat{f}_s^{1} \in \mathbb{R}^{C/4 \times 1 \times W} \). After that, we perform an upsampling operation on \( \hat{f}_s^{1} \) to obtain the feature map \( \tilde{f}_s^{1} \), which has the same spatial size with \( f_s \). Mathematically, \[ \tilde{f}_s^{1} = \text{up}_s(\text{conv}(\text{pl}_s(f_s))), \] (2)

where \( \text{up}_s(\cdot) \), \( \text{conv}(\cdot) \) and \( \text{pl}(\cdot) \) denote upsampling, convolutional layer, and square-pooling layer, respectively.

Next, we concatenate all \( n \) learned feature maps and the input feature \( f \) to produce the final feature \( f_s \in \mathbb{R}^{(n/4+1)C \times H \times W} \). The computation process can be expressed as follow,

\[ f_s = \text{concat}(\hat{f}_s^{1}, \hat{f}_s^{2}, \hat{f}_s^{3}, \cdots, \hat{f}_s^{n}, f), \] (3)

where \( \text{concat}(\cdot) \) denotes channel-wise concatenation. By doing so, the feature \( f_s \) has short-range and local semantic dependencies with different spatial scales. Therefore, the pixels with the same semantic label can achieve mutual gains, thus we are improving intra-object semantic consistency (see the first to fourth rows in Figure 2).

**Rectangle-Shape Pooling Module.** The proposed SPM captures only short-range semantic dependencies, as shown in Figure 2. To capture long-range and global semantic dependencies, we can increase the kernel size of the square pooling. However, this inevitably incorporates lots of irrelevant regions when processing rectangle-shaped and narrow objects such as the bus and pole shown in Figure 2.

To alleviate this limitation, we propose a novel Rectangle-shape Pooling Module (RPM), which aims to capture long-range and global semantic dependencies from both horizontal and vertical directions. The idea of the proposed RPM is inspired by the strip pooling module proposed in [44] and the framework of RPM is illustrated in Figure 5. It consists of a Horizontal Rectangle-shape Pooling Module (HRPM) and a Vertical Rectangle-shape Pooling Module (VRPM). HRPM captures long-range dependencies from horizontal and narrow objects (e.g., the fifth to seventh rows of Figure 2), while VRPM captures long-range correlations from vertical and narrow objects (e.g., the eighth to tenth rows of Figure 2).

As shown in Figure 5, given the feature \( f_s \in \mathbb{R}^{C \times H \times W} \) produced by SPM, we first feed it into a convolution layer to reduce the number of the channels and obtain a new feature \( f_s^{1} \in \mathbb{R}^{C/4 \times H \times W} \). Note that we use \( C \) to represent the number of channels of \( f_s \) for simplicity, which is different from the one used in Figure 4. Then \( f_s^{1} \) is separately fed into HRPM and VRPM to capture both horizontal and vertical long-range dependencies. Specifically, in HRPM, \( f_s^{1} \) is first fed into a horizontal rectangle-shape pooling layer to obtain a new feature \( f_h^{1} \in \mathbb{R}^{C/4 \times 1 \times W} \). After that, we put \( f_h^{1} \) through a 1D convolutional layer to obtain the feature \( f_h \). Next, an upsampling operation is performed on \( f_h \) to expand the spatial size and then output the feature \( f_h \). Similarly, in VRPM, \( f_s^{1} \) is first fed into a vertical rectangle-shape pooling layer...
to obtain a new feature \( f_s \in \mathbb{C}/4 \times H \times W \). After that, we use a 1D convolutional layer to obtain the feature \( f_v \). Next, an upsampling operation is performed on \( f_v \) to expand the spatial size and then output the feature \( \tilde{f}_v \). We finally sum both \( \tilde{f}_h \) and \( \tilde{f}_v \) incorporating both horizontal and vertical long-range dependencies,

\[
\tilde{f}_s = \tilde{f}_h + \tilde{f}_v.
\]  

Taking into account the advantages of short-range dependency modeling, we also consider incorporating SPM in our PRM to make the feature representations more discriminative. As shown in Figure 5, we first feed \( f_s \) into another convolution layer to reduce the number of the channels and obtain a new feature \( f'_s \in \mathbb{C}/4 \times H \times W \). We then capture short-range semantic dependencies by using Equation (5).

\[
f'_s = \text{conv}(\text{conv}(f'_s) + \text{SPM}(f'_s)),
\]  

where SPM(·) is the model proposed in Figure 4. After that, we combine both short-range and long-range semantic dependencies by using,

\[
f'_r = \text{conv}(\text{concat}(f'_s, \tilde{f}_s)).
\]  

In this way, \( f'_r \) is more discriminative than \( f_s \) by aggregating different types of contextual information via various pooling operations, leading to better results. Finally, we also propose two methods to add the input feature \( f_s \), constituting a residual connection [46]. The first one (RPM-I) is performing an element-wise addition:

\[
f_r = f'_r + f_s.
\]  

The second one (RPM-II) is performing a channel-wise concatenation:

\[
f_r = \text{concat}(f'_r, f_s).
\]  

Fusion of SPM and RPM. To take full advantage of both short-range and long-range semantic information, we further aggregate the outputs from these two pooling modules. Specifically, we propose seven aggregation methods as shown in Figure 6:

- (1) F-I is an image-level fusion strategy based on the attention fusion method proposed in LGGAN [9]. The formulation of F-I can be expressed as:

\[
\mathbf{I} = \text{conv}(f_s) \times A_1 + \text{conv}(f_r) \times A_2,
\]  

where \( A_1 \) and \( A_2 \) are attention masks produced by an attention decoder.

- (2) F-II is also an image-level fusion method represented as:

\[
\mathbf{I} = \text{conv}(f_s) + \text{conv}(f_r).
\]  

- (3) The other five methods are feature-level based fusion: F-III, F-IV and F-V are parallel structures, while F-VI and F-VII are cascading structures. Mathematically, F-III can be written as \( \mathbf{I} = \text{conv}(f_s + f_r) \).

- (4) The difference between F-IV and F-III is that F-IV uses two ‘RPM-II’ to produce the feature \( f_r \), while F-III adopts one ‘RPM-II’ and one ‘RPM-I’ to generate \( f_r \).

- (5) The difference between F-V and F-IV is that F-V uses a channel-wise concatenation operation to combine both \( f_s \) and \( f_r \), thus it can be expressed as:

\[
\mathbf{I} = \text{conv}(\text{concat}(f_s, f_r)).
\]  

- (6) Both F-VI and VII are cascading structures, and the
difference between them is the order of RPM and SPM. We can represent F-VI as: \( I' = \text{conv}(f_s) \).

- (7) F-VII is also shown in Figure 3, and it can be expressed as: \( I' = \text{conv}(f_r) \).

Optimization Objective. We follow GauGAN [5] and employ three different losses as our optimization objective, i.e., \( \mathcal{L} = \lambda_{\text{gan}} \mathcal{L}_{\text{gan}} + \lambda_f \mathcal{L}_f + \lambda_p \mathcal{L}_p \), where \( \mathcal{L}_{\text{gan}} \), \( \mathcal{L}_f \), and \( \mathcal{L}_p \) denote adversarial loss, discriminator feature matching loss, and perceptual loss, respectively. We set \( \lambda_{\text{gan}} = 1 \), \( \lambda_f = 10 \), and \( \lambda_p = 10 \) in our experiments.

Training Details. We use the multi-scale discriminator [5] as our discriminator \( D \). We use the Adam solver [47] and set \( \beta_1 = 0 \), \( \beta_2 = 0.999 \). Moreover, we set \( n = 4 \) in the proposed SPM, and set \( h_1 = w_1 = 1 \), \( h_2 = w_2 = 2 \), \( h_3 = w_3 = 3 \), and \( h_4 = w_4 = 6 \), respectively. The kernel size of convolutional layers in the proposed SPM is set to \( 1 \times 1 \). We set \( n = 2 \) for SPM used in RPM, and set \( h_1 = w_1 = 12 \) and \( h_2 = w_2 = 20 \). The kernel size of 1D convolutional layers in RPM is \( 1 \times 3 \) and \( 3 \times 1 \), respectively.

The proposed DPGAN is implemented by using PyTorch [48]. We conduct the experiments on NVIDIA DGX1 with 8 32GB V100 GPUs.

IV. EXPERIMENTS

Datasets. We follow GauGAN [5] and firstly conduct experiments on Cityscapes [12] and ADE20K [10] datasets. Cityscapes contains street scene images, and ADE20K contains both indoor and outdoor scenes. To further evaluate the robustness of our method, we conduct experiments on three more datasets with diverse scenarios, i.e., DeepFashion [11], CelebAMask-HQ [13], and Facades [14]. DeepFashion contains human body images, CelebAMask-HQ contains human facial images, and Facades contains facade images with diverse architectural styles. Experiments are conducted using different image resolutions to validate that our DPGAN can also generate high-resolution images, i.e., ADE20K (256×256), DeepFashion (256×256), Cityscapes (512×256), Facades (512×512), and CelebAMask-HQ (512×512).
Evaluation Metrics. We follow GauGAN [5] and adopt mean Intersection-over-Union (mIoU), pixel accuracy (Acc), and Fréchet Inception Distance (FID) [49] as the evaluation metrics on Cityscapes and ADE20K. For DeepFashion, CelebAMask-HQ, and Facades datasets, we use FID and Learned Perceptual Image Patch Similarity (LPIPS) [50] to evaluate the quality of the generated images.

A. Comparisons with State-of-the-Art

We adopt GauGAN [5] as our backbone and insert the proposed Double Pooling Module (DPM) before the last convolution layer to form our final model, i.e., DPGAN.

Qualitative Comparisons. We first compare the proposed DPGAN with GauGAN [5] and CC-FPSE [6] on DeepFashion, CelebAMask-HQ, and Facades datasets. Note that we used the source code provided by the authors to generate the results of GauGAN and CC-FPSE on these three datasets for fair comparisons. Visualization results are shown in Figures 7, 8, and 9. We can see that the proposed DPGAN generates more photo-realistic and semantically-consistent results than both GauGAN and CC-FPSE.

Moreover, we compare DPGAN with two leading methods on both Cityscapes and ADE20K datasets, i.e., GauGAN [5] and CC-FPSE [6]. Comparison results are shown in Figures 10 and 11. We can see that our DPGAN produces more clear and visually plausible results than both leading methods, further validating the effectiveness of our proposed DPGAN.

User Study. We follow the same evaluation protocol of GauGAN and also perform a user study. The results compared with GauGAN and CC-FPSE are shown in Table I. We see that users strongly favor the results generated by our proposed DPGAN, further validating that the generated images by our DPGAN are more photo-realistic.

Quantitative Comparisons. Although the user study is more suitable for evaluating the quality of the generated image in this task, we also follow GauGAN and use mIoU, Acc, FID, and LPIPS for quantitative evaluation. The results compared
Fig. 11: Qualitative comparison on ADE20K. From left to right: Input, GauGAN [5], CC-FPSE [6], DPGAN (Ours), and Ground Truth. We see that DPGAN produces realistic images while respecting the spatial semantic layout at the same time.

TABLE II: Quantitative comparison of different methods on DeepFashion, Facades, and CelebAMask-HQ.

| Method          | DeepFashion | Facades | CelebAMask-HQ |
|-----------------|-------------|---------|---------------|
|                 | FID ↓       | LPIPS ↓ | FID ↓         | LPIPS ↓ | FID ↓       | LPIPS ↓ |
| GauGAN          | 22.8        | 0.2476  | 116.8         | 0.5437  | 42.2        | 0.4870  |
| + DPM (Ours)    | **20.8**    | **0.2455** | **115.1**                      | **0.5503** | **25.1** | **0.4823** |

with several leading methods are shown in Tables II and III. Firstly, we observe that the proposed DPGAN achieves the best results compared with GauGAN on DeepFashion, CelebAMask-HQ, and Facades datasets, as shown in Table II. Moreover, we can see that DPGAN achieves competitive results compared with other leading methods on both Cityscapes and ADE20K datasets in Table III.

Visualization of Generated Semantic Maps. We follow GauGAN and adopt the pretrained DRN-D-105 [54] on the generated Cityscapes images to produce semantic maps. The results compared with those produced by GauGAN are shown in Figure 2. We clearly see that the proposed SPM and RPM can capture short-range and long-range semantic dependencies, leading to more semantically-consistent and realistic results than GauGAN.
### Table III: Quantitative comparison of different methods on Cityscapes and ADE20K. The results of other methods are reported from their papers.

| Method             | Cityscapes |        |        | ADE20K |        |        |
|--------------------|------------|--------|--------|--------|--------|--------|
|                    | mIoU ↑     | Acc ↑  | FID ↓  | mIoU ↑ | Acc ↑  | FID ↓  |
| CRN [2]            | 52.4       | 77.1   | 104.7  | 22.4   | 68.8   | 73.3   |
| SIMS [51]          | 47.2       | 75.5   | 49.7   | -      | -      | -      |
| Pix2pixHD [4]      | 58.3       | 81.4   | 95.0   | 20.3   | 69.2   | 81.8   |
| GAN Compression [52]| 61.2       | -      | -      | -      | -      | -      |
| BachGAN [53]       | 63.8       | 82.4   | 65.2   | 40.1   | 81.2   | 33.1   |
| PIS [7]            | 64.8       | 82.4   | 96.4   | -      | -      | -      |
| SelectionGAN [34]  | 66.1       | 82.6   | 60.3   | 40.5   | 81.6   | 31.9   |
| DAGAN [40]         | 68.4       | 83.0   | 57.7   | 41.6   | 81.8   | 31.6   |
| LGGAN [9]          | -          | -      | -      | -      | -      | -      |
| GauGAN [5]         | 62.3       | 81.9   | 71.8   | 38.5   | 79.9   | 33.9   |
| + DPM (Ours)       | 65.2 (+2.9)| 82.6 (+0.7) | 53.0 (-18.8) | 39.2 (+0.7) | 80.4 (+0.5) | 31.7 (-2.2) |
| CC-FPSE [6]        | 65.5       | 82.3   | 54.3   | 43.7   | 82.9   | 31.7   |
| + DPM (Ours)       | 66.9 (+1.4)| 82.8 (+0.5) | 51.9 (-2.4) | 44.8 (+1.1) | 83.2 (+0.3) | 30.3 (-1.4) |
| TSIT [8]           | 65.9       | 82.7   | 59.2   | 38.6   | 80.8   | 31.6   |
| + DPM (Ours)       | 66.7 (+0.8)| 83.1 (+0.4) | 56.1 (-3.1) | 39.9 (+1.3) | 81.2 (+0.4) | 30.5 (-1.1) |

### Table IV: Ablation study of our DPGAN on Cityscapes.

| No. Setting | mIoU ↑ | Acc ↑ | FID ↓ |
|-------------|--------|-------|-------|
| B1 GauGAN [5] | 62.3   | 81.9  | 71.8  |
| B2 B1 + SPM  | 64.9   | 82.5  | 55.9  |
| B3 B1 + RPM-I| 63.9   | 82.3  | 54.3  |
| B4 B1 + RPM-II| 63.8  | 82.4  | 54.9  |
| B5 B1 + 2 RPM-I| 64.5  | 82.4  | 54.2  |
| B6 B1 + 2 RPM-II| 64.7  | 82.5  | 53.2  |
| B7 B6 + SPM + F-I| 64.2  | 82.5  | 53.2  |
| B8 B6 + SPM + F-II| 65.2  | 82.5  | 54.5  |
| B9 B6 + SPM + F-III| 64.8  | 82.6  | 53.3  |
| B10 B6 + SPM + F-IV| 65.0  | 82.6  | 53.6  |
| B11 B6 + SPM + F-V| 63.1  | 82.4  | 53.3  |
| B12 B6 + SPM + F-VI| 64.8  | 82.4  | 53.4  |
| B13 B6 + SPM + F-VII| 65.2  | 82.6  | 53.0  |

### B. Ablation Study

#### Baselines of DPGAN

We conduct an extensive ablation study on Cityscapes to evaluate each component of the proposed DPGAN. DPGAN has 13 baselines (i.e., B1-B13) as shown in Table IV.

- (1) B1 is our baseline and uses a GauGAN structure [5].
- (2) B2 uses the proposed Square-shape Pooling Module (SPM) to capture short-range semantic dependencies.
- (3) B3 employs the proposed Rectangle-shape Pooling Module (RPM) to capture long-range semantic dependencies from both horizontal and vertical directions. Note that B3 uses Equation (7) to generate the feature $f_r$.
- (4) The difference between B4 and B3 is that B4 uses Equation (8) to generate $f_r$.
- (5) B5 is based on B3 and uses the proposed RPM twice.
- (6) B6 is based on B4 and uses the proposed RPM twice.
- (7)-(13) B7 to B13 are seven fusion methods proposed in Figure 6, which aim to effectively combine both short-range and long-range dependencies for further enlarging the receptive field of our model.

#### Ablation Analysis

The results of the ablation study are shown in Table IV. We can see that B2 achieves better results than B1 on all metrics, which confirms the importance of modeling short-range semantic dependencies. Both B3 and B4 outperform B1, confirming the effectiveness of modeling long-range semantic dependencies. B6 outperforms B4, and B5 outperforms B3, showing that adding more layers of RPM will further enlarge the receptive field. Moreover, we observe that B13 outperforms B6 on all metrics, showing that both short-range and long-range semantic dependencies are essential for generating high-quality results. Lastly, we observe that B13 achieves better results than B7-B12, demonstrating the effectiveness of the fusion strategy F-VII.

We also note that B7 and B11 achieve slightly deteriorated results compared with B6 on the mIoU metric, which means both F-I and F-V are not very suitable fusion strategies for the proposed SPM and RPM. Other fusion strategies such as F-II, F-IV, and F-VII achieve slightly different results in all the evaluation metrics. This is because the proposed SPM and RPM are powerful to capture both short-term and long-term dependencies and a simple fusion strategy such as F-II, F-IV, or F-VII can achieve good generation performance. This also further illustrates the effectiveness of both SPM and RPM.

#### Generalization of DPM

The proposed Double Pooling Module (DPM) is general and can be seamlessly integrated into any existing GAN-based architecture to improve the image translation performance. Therefore, to validate the generalization ability of the proposed DPM, we further conduct more experiments on both Cityscapes and ADE20K datasets. Specifically, we adopt CC-FPSE as our $E$ and then combine CC-FPSE and our DPM to form the final model. We observe that the CC-FPSE model with our DPM (i.e., CC-FPSE +
DPM) further improves all the three metrics, as shown in Table III. In the visualization results shown in Figure 12, we clearly observe that the CC-FPSE model with our DPM generates more realistic images with fewer artifacts than the CC-FPSE model without using our DPM on Cityscapes.

Moreover, we employ TSIT as our $E$ and then combine TSIT and our DPM to form the final model. We observe that the TSIT model with our DPM (i.e., TSIT + DPM) further improves all the three metrics compared with the original TSIT, as shown in Table III. We also provide the visualization results in Figure 13, we clearly observe that the TSIT model with our DPM generates more realistic images with fewer artifacts than the TSIT model without using our DPM on ADE20K. Both experimental results validate that the proposed DPM can be integrated with other methods to further boost the translation performance.

V. CONCLUSIONS

We propose a novel Double Pooling Module (DPM) for the challenging layout-to-image translation task. Specifically, we present a novel Double Pooling Module, which consists of the Square-shape Pooling Module (SPM) and the Rectangle-shape Pooling Module (RPM). SPM is used to capture short-range and local semantic dependencies. RPM is used to capture long-range and global semantic dependencies from both horizontal and vertical directions. The outputs of SPM and RPM are combined with the proposed fusion strategies to further effectively enlarge the receptive field of our model. Extensive experiments on five popular datasets demonstrate that DPGAN establishes new state-of-the-art results.

ACKNOWLEDGMENTS

This work was supported by the EU H2020 AI4Media No. 951911 project, by the Italy-China collaboration project TALENT:2018YFE0118400, and by the PRIN project PREVUE.

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in NeurIPS, 2014.

[2] Q. Chen and V. Koltun, “Photographic image synthesis with cascaded refinement networks,” in ICCV, 2017.

[3] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in CVPR, 2017.

[4] 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 CVPR, 2018.

[5] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” in CVPR, 2019.

[6] X. Liu, G. Yin, J. Shao, X. Wang et al., “Learning to predict layout-to-image conditional convolutions for semantic image synthesis,” in NeurIPS, 2019.

[7] A. Dundar, K. Sapra, G. Liu, A. Tao, and B. Catanzaro, “Panoptic-based image synthesis,” in CVPR, 2019.

[8] L. Jiang, C. Zhang, M. Huang, C. Liu, J. Shi, and C. C. Loy, “Tsit: A simple and versatile framework for image-to-image translation,” in ECCV, 2020.

[9] H. Tang, D. Xu, Y. Yan, P. H. Torr, and N. Sebe, “Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation,” in CVPR, 2020.

[10] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in CVPR, 2017.

[11] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, “Deepfashion: Powering robust clothes recognition and retrieval with rich annotations,” in CVPR, 2016.

[12] 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 CVPR, 2016.

[13] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, “Maskgan: Towards diverse and interactive facial image manipulation,” in CVPR, 2020.
