IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 22, 2021, accepted January 12, 2022, date of publication January 14, 2022, date of current version January 20, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3143538

Enhanced Class-Specific Spatial Normalization for Image Generation

MINGLE XU[1], YONGCHAE JEONG[2], (Senior Member, IEEE), DONG SUN PARK[1,3], AND SOOK YOON[4]

[1]Department of Electronics Engineering, Jeonbuk National University, Jeonju-si, Jeonbuk 54896, South Korea
[2]Division of Electronics and Information Engineering, Jeonbuk National University, Jeonju-si, Jeonbuk 54896, South Korea
[3]Core Research Institute of Intelligent Robots, Jeonbuk National University, Jeonju-si, Jeonbuk 54896, South Korea
[4]Department of Computer Engineering, Mokpo National University, Muan-gun, Jeonnam 58554, South Korea

Corresponding authors: Dong Sun Park (dspark@jbnu.ac.kr) and Sook Yoon (syoon@mokpo.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2019R1A6A1A09031717; in part by the National Research Foundation of Korea (NRF) funded by the Korean Government (MSIT) under Grant 2020R1A2C2013060; in part by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET); in part by the Korea Smart Farm Research and Development Foundation (KosFarm) through the Smart Farm Innovation Technology Development Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA); and in part by the Ministry of Science and ICT (MSIT), Rural Development Administration (RDA), under Grant 421044-04.

ABSTRACT We propose an enhanced class-specific spatial normalization, a simple yet effective layer to generate a photorealistic image given a spatial-class map. Under the assumption that pixels belonging to the same class share the same distribution in the feature space, we intuitively split an image into classes according to the map. By learning the class-specific distributions, our generator can distinguish one class from other classes. Further, our spatial normalization combines the spatial-class map and the class-specific distributions, by which our generator can produce instances in the desired locations. We apply the proposed normalization not only in semantic image generation but also in object transfiguration. The experimental results demonstrate that the spatial-class map can be efficiently utilized with our proposed method, which results in competing performances with much fewer parameters.

INDEX TERMS Semantic image synthesis, object transfiguration, image translation, image generation, class-specific spatial normalization.

I. INTRODUCTION

Conditional image generation aims to produce photorealistic images given conditions. Seminal works synthesize images from conditional images, called image translation (I2I) [5], [13], [18], [19], [25]. Current works can produce images from labels or sentences [22], [31], [32]. Conditional image generation allows the output to be controlled, and this process has witnessed profound improvements in recent years.

We focus on a specific form, spatial-class conditional image generation, in which the generated image is expected to be aligned with the spatial-class map where every pixel is assigned to its corresponding class. This form owns a wide range of applications, such as semantic image generation [7], [11], [18], [19], [33], layout image generation [10], [29], and object transfiguration [12], [14], [16]. While the output image in the previous two cases is expected to have the same class as the input semantic segmentation [19] or bounding box [10], object transfiguration splits the image into two parts, foreground and background, according to a binary mask. The first part is desired to be translated from one domain to another domain, such as a horse to zebra. In contrast, the second part is just expected to be the same as the input image. Since the conditions used in the three cases are spatial-class maps, we refer to *spatial-class conditional image generation.*

Compared to other conditions, using the spatial-class conditional map efficiently is still a challenge. Although pix2pixHD [19] managed to translate a semantic segmentation into a photorealistic image, the ''wash-away'' issue exists [7]. To address the issue, Park et.al introduced a spatially-adaptive normalization (SPADE) layer [7] in which three convolution layers are adopted to encode the spatial segmentation to scale and shift. The scale and shift latter

The associate editor coordinating the review of this manuscript and approving it for publication was Shovan Barma.

are utilized to rescale and reshift the input feature map, by which the spatial segmentation is embedded into the input feature map. Though the SPADE discards the encoder in the pix2pixHD, its computation is still huge, because of the three convolution layers with large channel sizes. In addition, how the SPADE contributed is still vague.

In this paper, motivated by adaptive instance normalization [8], [9], which assumes that all pixels in one *image* share the same distribution in the feature space to perform style transfer, we propose class-specific spatial normalization to perform conditional image generation on the assumption that the pixels belonging to one *class* share specific distribution. Under the generalized assumption, models can learn the specific distributions for each class from a semantic-class map. Moreover, we find that our method explains the SPADE [7] but we can achieve the same idea with many fewer parameters. To be specific, we analyze the process of the SPADE, and find that the SPADE implicitly embraces our generalized assumption. Extensive ablation studies validate our proposed class-specific spatial normalization layer. Further, we extend this idea from semantic image synthesis to object transfiguration [12], [14], [16]. Experimental results on horse2zebra collected from COCO-Stuff [3] demonstrate that our method outperforms the state-of-the-art by a clear margin.

To sum up, our contributions are as follows:

- We declare explicitly a generalized assumption, from adaptive instance normalization [8], [9] that the pixels belonging to each class share specific distribution, which unveils the SPADE [7];
- We propose class-specific spatial normalization to perform spatial-class conditional image generation and achieve competing performance with less parameters than the state-of-the-art (our basic method uses only around 76% of the parameters and 40% of the FLOPs of the SPADE);
- Our extensive ablation studies give understanding towards semantic image generation, including that the learned distribution spaces in conditional normalization are evolving over layers, independent scale and shift inside a layer contribute better performance, and context further contributes to the quality of the generated images;
- We extend our generalized assumption from semantic image synthesis to object transfiguration, proving that our proposed class-specific spatial normalization is suitable to utilize the spatial-class map in different applications, which gives us an integrated viewpoint for both applications.

II. RELATED WORK

A. DEEP GENERATIVE MODELS

can be adopted to generate images. Our work is established in generative adversarial networks [17], but aims at image generation task given a semantic-class map. The GANs can be split into a generator and a discriminator, in which the generator is to create realistic images so that the discriminator cannot distinguish the generated image from the real image. In this paper, we propose a class-specific spatial normalization layer for the generator to align the generated images to the given spatial-class map.

B. CONDITIONAL IMAGE GENERATION

appears in a great many forms that differ from the kinds of condition. For example, label-conditional networks produce an image from a given label [21], [22], [32]. Except for labels, words and sentences have been explored as conditional input [31]. Images translation [5], [12], [14]–[16], [23] focuses on synthesizing an image from another image in which object transfiguration requires an extra input, the location of the object to be translated. Similar to object transfiguration, image generation from semantic segmentation requires the generator to synthesize fake images according to spatial condition [7], [11], [18], [19], [33]. In this paper, we are especially interested in how to use the spatial-class map efficiently and propose a class-specific spatial normalization layer. Resorting to the normalization layer, our generator obtains better results with less parameters in semantic segmentation-based image generation and object transfiguration.

C. CONDITIONAL NORMALIZATION LAYERS

is developed from the unconditional normalization method in which input is first converted to a standard normal distribution, and then recast into another norm distribution with learned scale and shift [20]. Unconditional normalization can be categorized by one factor, along which dimension(s) the normalization and recast are performed. For example, batch normalization [20] is along all mini-batch and spatial pixels, while layer normalization [34], is along all channels and spatial pixels. But they both compute the shift and scale unconditionally. In contrast, conditional normalization layers learn the scale and shift from conditional input. Apart from dimension, conditional normalization can also be classified by the types of conditional input. Conditional batch normalization is widely deployed in label condition, in which we wish the label of the generated image to be controlled by the input label [21], [22] and conditional instance normalization is developed for style transfer, in which the produced image is expected to have the given global style [8], [9]. While successfully letting the generator use the condition, neither of them can be leveraged for a semantic-class map, since they assume that all pixels share the same label or same style. To address this challenge, we propose a class-specific spatial normalization layer, in which the condition is taken to compute the scale and shift. To be specific, we loosen the assumption to become that an image can be grouped into classes, and all pixels in a class share the same distribution, such as scale and shift. Two recent works are related to ours. The spatially-adaptive normalization layer [35] is firstly adopted to achieve image super-resolution, but we focus on image generation given a spatial-class map. Although the SPADE [7] is designed for semantic image synthesis, two distinct points exist. First, we explain the function of the SPADE and achieve the core idea, but with fewer parameters. Second, we extend the idea from semantic image synthesis to a sub-field of image translation, namely, object transfiguration.

III. CLASS-SPECIFIC SPATIAL NORMALIZATION LAYER

A. GENERAL IDEA

We aim to perform spatial-class conditional image generation, in which conditions are spatial-class map, such as semantic segmentation in semantic image generation [7]. To achieve this, we propose class-specific spatial normalization inspired by adaptive instance normalization [8], [9] and conditional batch normalization [21], [22]. While adaptive instance normalization has been utilized to perform style transfer and assumes that all pixels in one image share the same style distribution in the feature space, conditional batch normalization has been leveraged to perform image generation from noise. Inspired by them both, we use a generalized assumption that pixels belonging to each class share specific distribution and embrace the conditional normalization paradigm, learning the scale and shift in the normalization layer to control the feature distribution.

B. CLASS-SPECIFIC SPATIAL NORMALIZATION (CSN)

Let $s \in \mathbb{S}^{H\times W}$ be a spatial-class map where $\mathbb{S}$ is a range of integers indicating classes, and H and W denote the input image height and width, respectively. Each entry in $s_{h,w}$ gives the class for the pixel with spatial coordinate (h, w) in one-hot manner. We aim to learn a normalization layer that can combine the spatial-class map to a feature efficiently and synthesize a photorealistic image aligning the map in the end.

Fig. 1 shows that the CSN layer consists of four stages. Below we show how to execute the class-specific spatial normalization, taking the learnable scale γ as an example (learnable shift β is executed in the same way). In the first stage s_a, class embedding stage, the one-hot label $l \in \mathbb{L}^{N\times N}$ (N is the total number of labels) is embedded into vector $z^{\gamma} \in \mathbb{Z}^{N\times M}$ (M is a hyper-parameter), and we expect that the network can learn the representation for each label. Formally, given the one-hot label l, the class embedding stage can be formatted as follows:

$$z^{\gamma} = ReLU(FC(l)), \tag{1}$$

where the FC is a fully connected function and $ReLU$ is an activation function. Here, we use only one FC and $ReLU$, but we can employ more stacks of them. With this class embedding process, one-hot labels are encoded into a new space in which the connections between each pair of classes can be learned, instead of being equal in the one-hot label space as the SPADE adopted [7]. We expect that better connections make learning easier.

The second stage s_b is a sampling step according to the input spatial-class map s and the vectors z^{γ}. Specifically, if $s_{h,w}$ is label c, then $\gamma'_{h,w} = z^{\gamma}_c$. In other words, we get the feature γ' by selecting z^{γ} according to spatial-class map s. Therefore, we easily arrive at the following theorem:

Theorem 1: $\forall h_1, h_2$ and $\forall w_1, w_2$, if $s_{h_1,w_1} = s_{h_2,w_2}$; then $\gamma'_{h_1,w_1} = \gamma'_{h_2,w_2}$.

We emphasize that the second stage shows the generalized assumption, that pixels belonging to the same class share specific distribution in the feature space. Before introducing the third stage, we point out a requirement that the learned γ and β should have the same number of channels as the input feature f^i and in the beginning stage of image generation, the channel number of f^i are very high. Although it can be omitted, the third stage s_c is employed to balance the model's capacity and the number of parameters. In our basic version, s_c employs two 1×1 convolution layers, while in our developed version, s_c employs two 3×3 convolution layers with a lesser number of channels to use the context, followed by a 1×1 convolution to keep the same number of channels as the input feature. In contrast, two 3×3 convolutions with the same number of channels as the input feature are used to utilize the context information which increases the number of parameters. Hence the third stage s_c achieves a balance between using context information and light architecture and therefore, is called the balance stage. The ablation study in the next section validates that stage s_c dedicates the model's performance yet asks for much fewer parameters.

Finally, the normalized input feature are rescaled and reshifted in the last stage s_d with the learned γ and β. Let $f^i \in \mathbb{F}^{B\times C^i\times H^i\times W^i}$ be the feature of the i-th layer of a deep neural network with B batch samples. C^i denotes the number of channels in the layer and H^i and W^i are the height and width, respectively. In the last stage, the input feature is firstly normalized over channels and then is rescaled and reshifted with the learned values γ and β. Mathematically, the value of the output feature of the CSN layer with coordinate (b, c, h, w) is:

$$\gamma_{c,h,w}(s,l) * \frac{f^i_{b,c,h,w} - \mu^i_c}{\sigma^i_c} + \beta_{c,h,w}(s,l), \tag{2}$$

where, μ^i_c and σ^i_c are the mean and variance of the input over the channels:

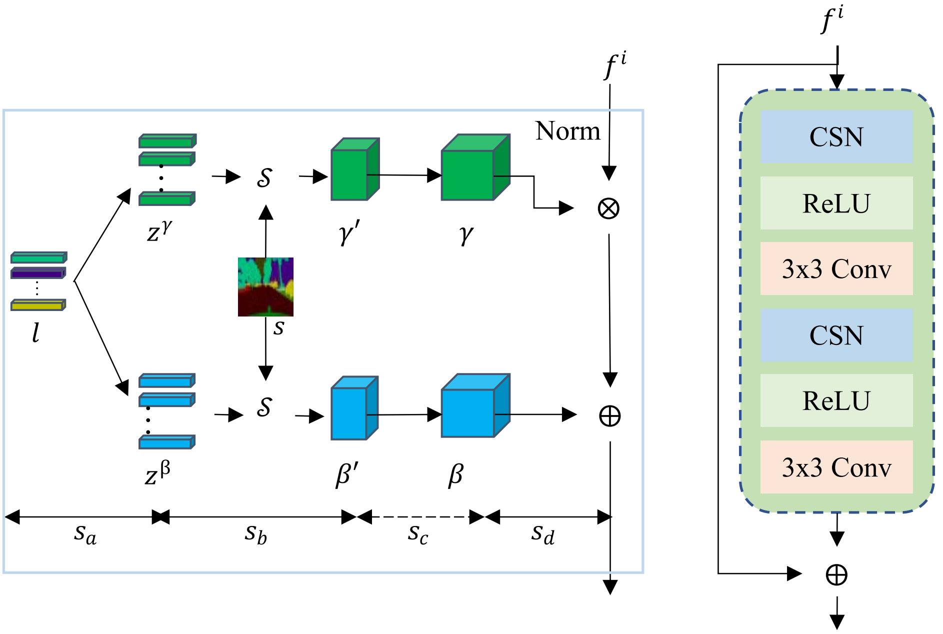
$$\mu^i_c = \frac{1}{B\cdot H^i\cdot W^i}\sum_{b,h,w} f^i_{b,c,h,w}, \tag{3}$$

$$\sigma^i_c = \sqrt{\frac{1}{B\cdot H^i\cdot W^i}\sum_{b,h,w}(f^i_{b,c,h,w})^2 - (\mu^i_c)^2}. \tag{4}$$

Fig. 1 (b) shows our CSN ResBlk (CSN residual block) that the CSN can be deployed in ResNet block [6]. Since we only use the CSN among the features with the same height and width, plain input is directly used in this paper, while a more complex version is still possible.

1) RELATION TO THE SPADE

Although SPADE [7] showed a decent performance, its function has not been explained but we find that our motivation



(a) Class-specific spatial normalization (CSN)

(b) CSN ResBlk

FIGURE 1. (a) The class-specific spatial normalization (CSN) layer takes a one-hot label l and semantic-class map s as input. It consists of four stages, class embedding stage s_a, sampling stage s_b, balance stage s_c and normalization stage s_d to perform rescale and reshift, in which s_c in the dotted line is not necessary but contributes. The second stage suggests the generalized assumption that pixels belonging to the same class share specific distribution in the feature space. $\mathcal{S}$ denotes the sampling process. (b) CSN ResBlk: ResNet block with the CSN layer.

unveils it. In the SPADE, semantic segmentation firstly undergoes a convolution layer to a middle feature map, followed by two specific convolution layers to form the learned shift and scale. As the convolution layer embraces local connection and sharing parameters, pixels in the same class result in the same output (except the boundary where the convolution kernel sees more than one class). In this way, we can understand the SPADE which embraces the same assumption.

However, our proposed CSN is different from the SPADE in three ways. First, γ and β are more independent than in the SPADE where they share the first convolution layer, the ablation study showing that the independence leads to slightly better performance. Second, we use the class embedding and sampling stage to learn the relationships among classes and combine the spatial-class map. In contrast, the SPADE adopts an equal relation among all classes because of the one-hot label space. Finally, our stage s_c makes the model achieve a balance between context information and less learnable parameters. The ablation study in the next section provides extensive insights.

2) CSN GENERATOR FOR SEMANTIC IMAGE GENERATION

Semantic image generation requires semantic segmentation as input and outputs images aligning the segmentation. The segmentation is commonly taken as the input of a neural network and undergoes a stack of convolution layers, nonlinear function, and normalization. Since the CSN can merge the segmentation to features, we cast the segmentation encoder away and down-sample the semantic segmentation as input to the generator. In contrast, the spatial-class map is fed to the generator immediately, which results in a much lighter network.

Fig. 2 illustrates the generator with CSN for semantic image generation where several ResNet blocks and up-sampling layers are employed. Similar to SPADE, the semantic segmentation is downsampled into a fixed width and height to be the input of the generator and also downsampled to match the spatial resolution of the feature maps in each scale as the width and height are increased. The generator undergoes stacks of CSN blocks and upsampling layers. Specifically, one convolution layer is adopted to the input semantic segmentation to increase the number of channels, which is followed by a CSN block. As the CSN block does not change the size of the feature map, an upsampling layer is leveraged to increase the size and reduce half the number of channels. According to the output size of the generated images, the stacks of CSN and upsampling layers are utilized four or five times, which is followed by another convolution layer to reduce the number of channels and a Tanh function to form an RGB image. The generator is trained with multi-scale discriminator and the loss functions leveraged in pix2pixHD [19] except for two settings, two discriminators with different scales and replacing the least square GAN loss [24] with hinge loss. Simultaneously, feature matching

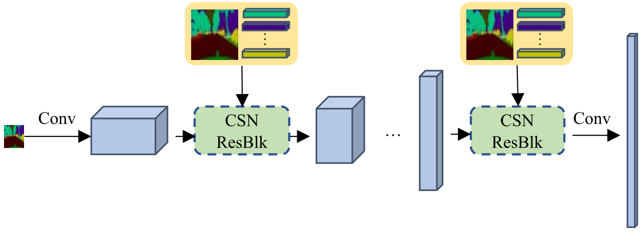


FIGURE 2. The architecture of the generator for semantic image generation. We downsample the semantic segmentation as input and employ the CSN in every feature scale until an image is formed. While the first Conv layer in the figure is utilized to increase the number of channels, the last Conv layer is leveraged to reduce the number of channels to three, which is followed by a Tanh function to form an RGB image.

loss and VGG loss are adopted to train the generator [7], [11] and we refer the readers to SPADE [7] for the details as we do not change them.

3) CSN GENERATOR FOR OBJECT TRANSFIGURATION

Object transfiguration is located in image-to-image (I2I) translation [12], [14] which takes a conditional image and binary mask. The binary mask splits the holistic image into two parts, a foreground to be translated and a background to be retained. Fig. 3 shows that the generator is expected to translate the foreground, corresponding to the white area, into the target domain and retain the background, corresponding to the dark area.

As shown in Fig. 3, our generator to perform image to image translation consists of three main parts, encoder, residual block, and decoder, which is similar to CycleGAN [13]. The residual blocks are leveraged to make rich features and a high receptive field. While the encoder is used to extract the necessary features from the conditional image, the decoder is employed to synthesize a photorealistic image given a specific feature map. The CSN block is supposed to enable the encoder to extract the foreground and background separately and enable the decoder to know where to translate and where to reconstruct. The literature commonly employs concatenation to combine the conditional image and the mask, which possibly leads to losing the spatial relation, and incurs more parameters, but in our paper, the operation is replaced with the CSN module without losing the spatial relation. Besides, our algorithm can not only perform object transfiguration, translating all instances belonging to the one object, but also instance transfiguration, only translating part of the instance(s), as shown in Fig. 3.

To train the generator, we employ three kinds of losses. Firstly, mask-guided discriminator, taking as input the multiplication of images and binary mask, is borrowed [5] which forces the discriminator to focus on the assigned content. A similar idea appears in AGGAN [12] in which attention to be translated object is produced by a sub-module, and the discriminator takes as input the multiplication of image and the attention after threshold. Secondly, background consistent loss is borrowed from InstaGAN [16] and Ref [5]. The loss tries to let the generator retain the background of the original image but allows the generator to smooth the margin of the translated area to make the synthesized image nature. Thirdly, we also adopt identity loss to ease the training process as CycleGAN [13] does.

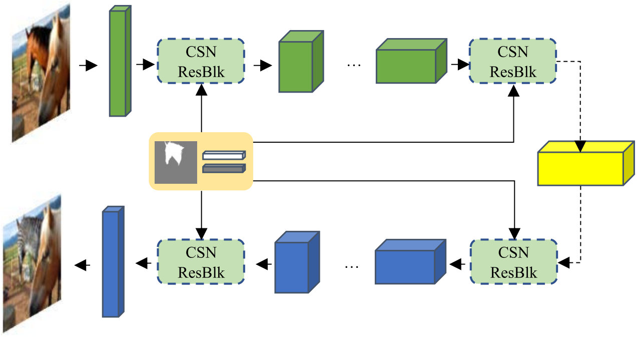


FIGURE 3. The architecture of the generator for object transfiguration. This consists of a CSN-based encoder, residual blocks, and a CSN-based decoder, which are shown in green, yellow, and blue, respectively. The CSN block is supposed to enable the encoder to extract the foreground and background separately and enable the decoder to know where to translate and where to reconstruct.

IV. EXPERIMENTS

A. SEMANTIC IMAGE GENERATION

Implementation details: We apply the spectral normalization [26] in both generator and discriminator. The learning rates for updating generator and discriminator are different, at 0.0001 and 0.0004 respectively. We select Adam [28] as optimizer with $\beta_1 = 0$ and $\beta_2 = 0.900$. We borrow synchronized BatchNorm where mean and variance are collected from all GPUs.

Datasets: We design our experiments on two datasets. Cityscapes [1] gives 2,975 images for training and 500 images for testing with 35 labels related to street scenes. ADE20K [2] contains 20,210 training and 2,000 validation images, with 150 semantic classes. We directly use the original training and validation dataset to train and validate all methods.

Baselines: We compare our algorithm to three leading models using neural networks: pix2pixHD [19], SPADE [7] and SEAN [11]. Pix2pixHD is the state-of-the-art model with encoder and decoder, while the SPADE and SEAN are the current state-of-the-art models with conditional normalization layer. The SPADE uses two convolution layers to produce scale and shift from semantic mask starting from the 'washaway' problem [7]. Based on the SPADE, SEAN assumes that different areas in one image have their style, and hence has designed a joint-conditional normalization layer.

Evaluation metrics: We take the evaluation metrics from the literature. To be clear, we adopt a pretrained semantic segmentation model on the generated images and compare the results with the semantic input for the generator. Intuitively, if the generator uses the semantic input correctly to produce natural images and the segmentation model is trained very well, the predicted segmentation will match well with the input segmentation. By matching the segmentation, we adopt pixel accuracy (accu) and mean Intersection-over-Union (mIoU). The bigger the accu and the bigger the mIoU, the better the generator. Apart from accu and mIoU from semantic segmentation metrics, we compute the Frechet Inception Distance (FID), a distance between the feature distribution of generated images and real images.

1) ABLATION STUDY

To understand the CSN better, we design extensive ablation studies in Cityscapes [1], TABLE 1 showing the results. Since we found that mIoU and FID are hard to reach the best simultaneously, we display two situations for every experiment, the best mIoU, and the best FID. We set the baseline (Ours-base) as follows. In stage s_a, we adopt a linear and ReLU function to embed one-hot label l to vectors z^γ and z^β with dimension $M = 128$. In stage s_c, we leverage 1×1 convolution with M as the number of channels and ReLU function, followed by another 1×1 convolution with the same number of channels as the input feature. To have a smaller number of parameters, the number of channels for all experiments is set as M in the middle layer convolution layer but set as the number of channels of the input feature only in the last layer.

Our basic assumption is valid: To validate our assumption first, all pixels belonging to one class sharing the same distribution on the feature space, we use *only* two linear functions to form the shift and scale. The decent performance in index 1 validates that our assumption is reliable. During the image generation process, the generator should firstly distinguish each class from other classes and secondly produce images aligning the input spatial-class map. Learning class-specific distribution for each class is useful to the first while spatial normalization is useful for the second. Besides, one advantage of this setting is fewer computations (FLOPs)), which may be desirable for mobile devices.

Class embedding stage s_a should be independent for each layer, and big scale one improves the quality of the generated image: To know the relationship of the learned distribution space between every layer, we share z^γ and z^β for every CSN layer, but we get lower mIoU yet similar FID as the results in index 2 show. We infer that for image generation, when the feature scale increases, the distribution should evolve and be independent, as suggested in [36] for feature extraction with convolution layer. To verify the assumption, we further add one more linear and ReLU function to obtain z^γ and z^β. The result in index 3 suggests that this reduces the FID while slightly impairs the mIoU.

Independent γ and β dedicate both FID and mIoU: Thirdly, we try to probe the relation of the shift β and scale γ by making them less independent by removing the non-linear ReLU function. The result in index 4 indicates that this harms the image quality because of the higher FID, which means that the independent scale and shift result in better-quality images but slightly result in less mIoU. We found this kind of independence has not been noticed in the literature, such as the SPADE [7] or SEAN [11].

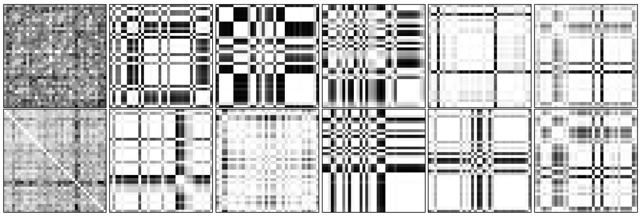


FIGURE 4. Visualization of cosine similarities of z^γ (first row) and z^β (second row). Two observations: z^γ and z^β are independent in most of the layers, while similarities between classes tend to be higher in the latter layers for both z^γ and z^β.

The context largely benefits FID yet slightly harms mIoU: To consider the context impact of semantic labels, we leverage 3×3 convolution layers. Only one 3×3 convolution layer (index 5) shows its inferiority on almost all metrics but two 3×3 convolutions (index 6) show their superiority on FID. We infer that the 3×3 convolution dilutes and smooths the boundaries among the class-map, but one 3×3 convolution could not provide a sufficient receptive field. The results show that the context slightly harms the mIoU, but contributes FID, probably because FID scores in a natural and smooth boundary but mIoU scores in a sharp boundary. Although the context information have been utilized in the SPADE [7] and SEAN [11], 3×3 convolution is adopted with higher channels, resulting in big-scale parameters. We want to emphasize that we replace the 3×3 convolution with many fewer channels (the same as $M = 128$), followed by a 1×1 convolution with the required channels, which gives the same receptive field yet many less parameters and computations.

The optimal dimension M could vary in different applications: In addition, we vary the hyper-parameter M from 128 to 96 and 256. As displayed in indices 7 and 8 in TABLE 1, a smaller dimension slightly benefits a lower FID but harms mIoU. In contrast, a higher dimension impairs both FID and mIoU mainly because more parameters require more epochs to train. We conjecture that the best dimension varies as the complexity of the application changes.

The improved version of our method embraces independent and evolving embedding stage and context information: To have a better quality of generated images, we finally update our basic version to a powerful version with two linear functions in the class embedding stage, and two 3×3 convolution layers to extract context. As suggested in TABLE 1, this obtains the best FID 49.9 and a competing mIoU, which proves that the combination of better stage s_a and s_c improves the performance further.

2) ANALYSIS OF THE CLASS EMBEDDING STAGE

One of the main difference between our method and the SPADE [7] is the class embedding stage s_a. In this subsection, we aim to analyze the function of the class embedding stage. To do so, we compute the cosine similarities between each pair of label after embedding. Fig. 4 shows the similarities of the embedded vectors in z^γ and z^β. We observe that the

TABLE 1. Ablation study on the CSN layer. s_a and s_c are the stages in Fig. 1 and M is a hyper-parameter. n, r and c denote linear, ReLU and convolution. M is the dimension of the middle layer and is a hyper-parameter. The default number of channels in the linear function is M and # suggests the same number of channels as the input features since no convolution layer is employed. The value behind c means the kernel size of the convolution layer and 2× suggests using the functions twice. ↑ means that higher is better, while ↓ means that lower is better.

Index	s_a	s_c	M	para	Best mIoU			Best FID		
					accu ↑	mIoU ↑	FID ↓	accu ↑	mIoU ↑	FID ↓
Ours-base	n+r	c1+r+c1	128	70.7 M	81.9	63.2	59.6	81.8	60.4	54.5
1	n+r+n#	/	128	70.0 M	82.0	62.7	56.4	81.8	60.7	53.0
2	share(n+r)	c1+r+c1	128	69.9 M	81.6	59.9	59.7	81.7	58.3	54.3
3	2×(n+r)	c1+r+c1	128	71.2 M	81.8	61.2	51.3	81.5	59.0	50.9
4	n+r	c1	128	70.1 M	81.9	61.5	60.3	81.8	60.8	57.0
5	n+r	c3+r+c1	128	75.4 M	81.8	61.2	58.2	81.6	59.3	56.5
6	n+r	2×(c3+r)+c1	128	80.7 M	81.8	61.9	51.9	81.6	59.7	51.4
7	n+r	c1+r+c1	96	69.9 M	81.9	62.3	57.3	81.9	61.6	53.2
8	n+r	c1+r+c1	256	77.6 M	81.4	58.6	67.4	81.4	57.9	63.9
Ours-v2	2×(n+r)	2×(c3+r)+c1	128	81.2 M	81.7	61.0	51.8	81.7	60.0	49.9

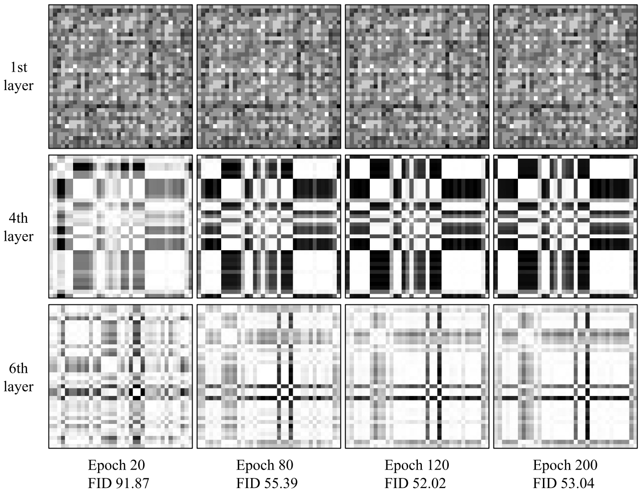


FIGURE 5. Visualization of cosine similarities of z^{γ} in different epochs. We see that the similarity converges and the change of the similarity is similar to the change of FID.

similarities between labels change over the layers and become higher in the latter layers. One of the interesting findings is that the class embedding layer learns the semantic connections between semantic labels. For example, the vehicle class becomes closer to traffic light class and traffic signal class, but farther from the sky class as training epochs increase. Another observation is that the similarities of z^{γ} and z^{β} in most of the layers are distinct which proves that they are more independent in our previous case, the result of index 2 in TABLE 1.

While Fig. 4 shows the cosine similarity of the embedding stage in different layers, figure 5 illustrates the change of the last layer during the training processes. There are also two observations. First, the similarity converges in the latter training stage. For example, the similarity in the sixth layer varies greatly from epoch 20 to epoch 80 but varies little from epoch 80 to 200. Second, the convergence of the similarity resembles the convergence of FID, such as little change of FID from epoch 80 to epoch 200.

3) COMPARISON TO THE STATE-OF-THE-ART

Quantitative comparison: TABLE 2 shows the three evaluation metrics and the number of parameters in different models. Our algorithm achieves decent performances with fewer parameters. The comparison suggests that the CSN reaches a balance between FID and mIoU while the SPADE sets the highest mIoU but the worst FID and SEAN shows a different face. Our basic scheme achieves competing results with few parameters and FLOPs. Cost at big FLOPs, ours-v2 proves that context impact is useful in the semantic image generation and achieves the best FID in Cityscapes.

Qualitative results: As shown in Fig. 6, the generator with ours-v2 CSN can generate decent images with fewer artifacts. Although the SPADE [7] shows its higher mIoU and pixel accuracy, its produced images give worse FID. SEAN sets the best FID but costs at a very heavy architecture. In contrast, the CSN costs less yet obtains a competitive performance. As displayed in the figure, the generator with the CSN can generate photorealistic images aligning to the given semantic labels.

B. OBJECT TRANSFIGURATION

Implementation details: To train the generator with the CSN, we use the discriminator in patchGAN [13], [30] with a 70 × 70 receptive field. Least-square loss [24] is leveraged because it shows stable training for image translation. To further ease the training process, a history of fake images is adopted. Similar to CycleGAN [13], Adam [28] is used with a learning rate at 0.0002, and is deployed in the first 100 epoch and linearly decayed to zero in the second 100 epoch.

Datasets: We derive instance mask for horse and zebra from the COCO [4] dataset. Tiny masks are removed that the human eyes cannot distinguish. We collect 1,276 and 996 images, split into 80% and 20% for training and testing, respectively.

Baselines: We compare our algorithm to several image-to-image algorithms. CycleGAN [13] is one of the seminal works to perform an unsupervised image translation method applying a cycle-consistent loss. AGGAN [12] aims

TABLE 2. Our method achieves competing results with much fewer parameters. We retrain for the SPADE and the SEAN but reuse the values from the SPADE [7] for the pix2pixHD. The red, blue, and green denote the best, second, and third values in each column, except for para and FLOPs.

Method	Cityscapes					ADE20K				
	para(M)	accu ↑	mIoU ↑	FID ↓	FLOPs(G) ↓	para(M)	accu ↑	mIoU ↑	FID ↓	FLOPs(G) ↓
pix2pixHD [19]	182.5	81.4	58.3	95.0	151.3	182.9	69.2	20.3	81.8	99.3
SEAN [11]	267.4	80.3	57.3	51.5	584.3	269.1	76.4	33.7	29.4	240.8
SPADE [7]	93.0	81.8	61.9	59.2	281.6	96.5	79.4	37.6	40.7	181.3
Ours	70.7	81.6	59.9	54.3	113.3	72.2	77.8	35.2	39.3	61.5
Ours-v2	81.2	81.7	60.0	49.9	405.2	82.5	78.1	36.1	38.7	207.8

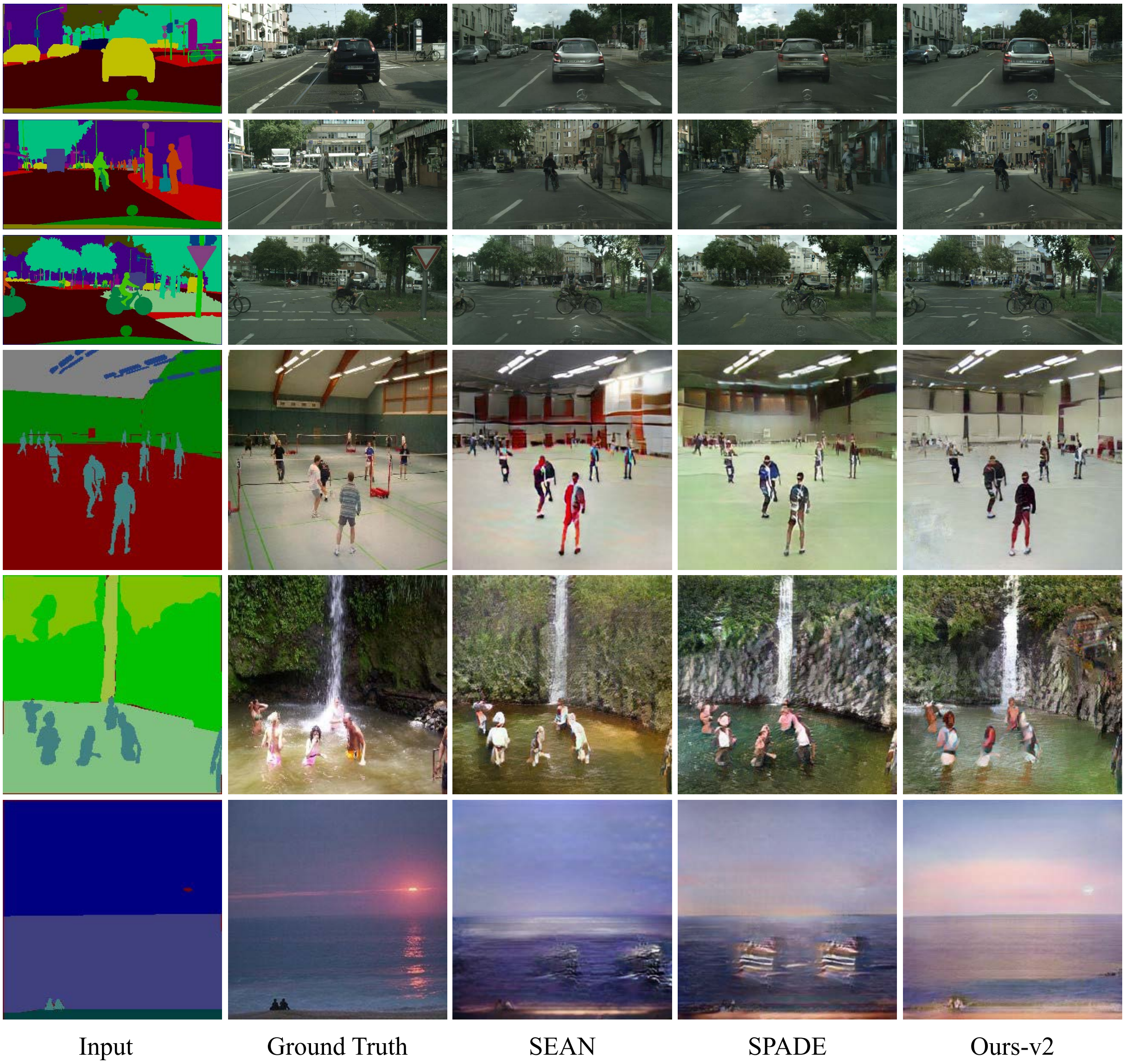


FIGURE 6. Visual comparison of semantic image generation on the Cityscapes and ADE20K dataset. Our generator with the proposed CSN produces realistic images aligning the given spatial-class map with fewer artifacts.

to perform object transfiguration and employs additional sub-modules to predict the area to be translated in the generated image. InstaGAN [16] directly employs segmentation as input. For a fair comparison, we adapt AGGAN with a binary mask as input, and the final compared images take background from the input image and foreground from the generator's output. For InstaGAN, the shape of the instance to be translated is assumed to be the same as the translated instance. As it did not perform object transfiguration, the original SPADE is not compared for this application.

Evaluation metrics: Since object transfiguration requires no change in the background, we compute mask FID (mFID)

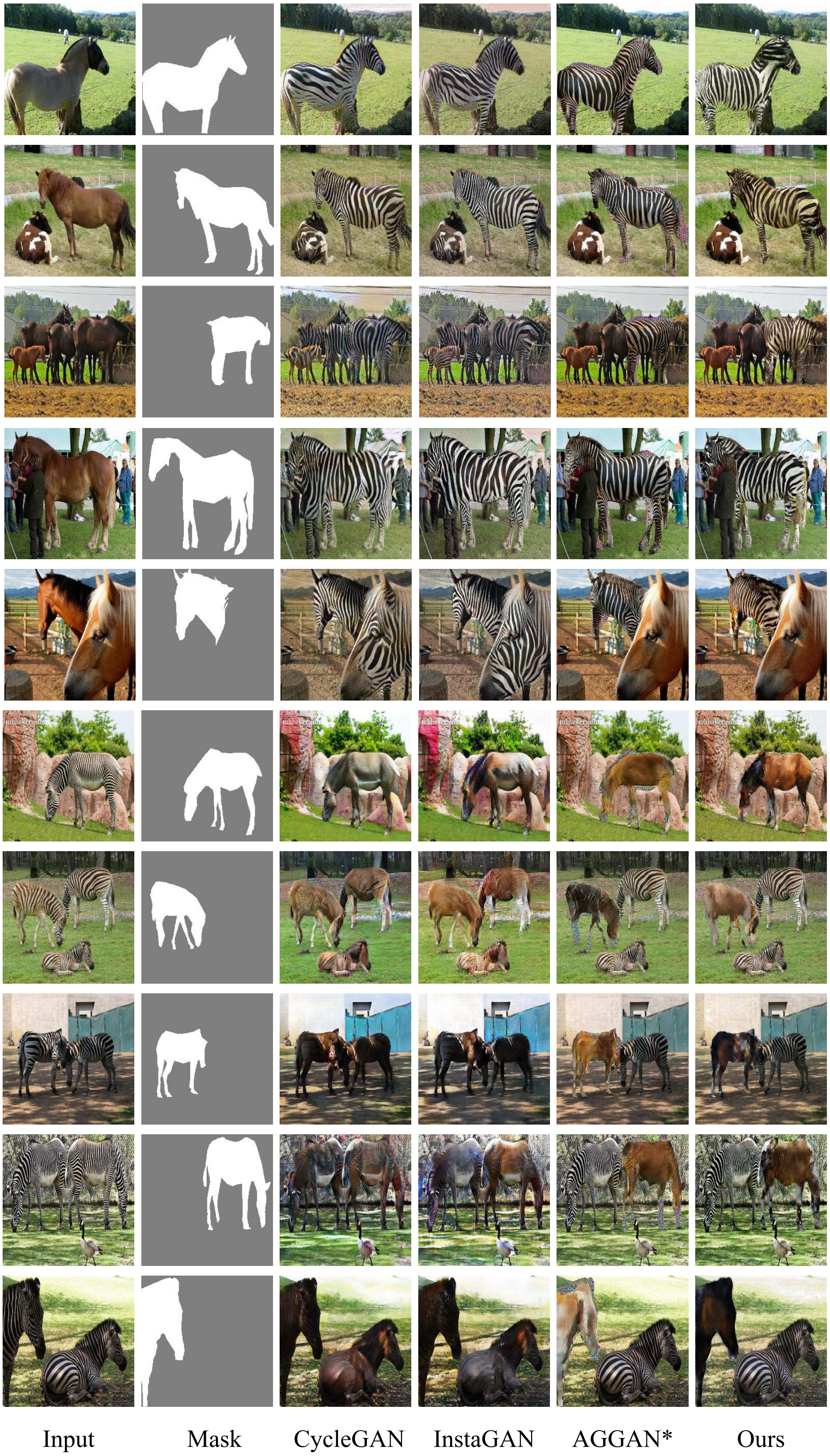


FIGURE 7. Visual comparison of object transfiguration on horse and zebra translation dataset. Our method stably produces a decent target instance with the desired background. Other algorithms either change the background or incur artifacts.

TABLE 3. Our method outperforms the state-of-the-art in object transfiguration by a clear margin except for mFID in generating fake horses. ↑ means that higher is better and ↓ means that lower is better. The best results in every column are boldfaced. Since AGGAN adopts the original background, its PSNR and SSIM are not compared.

Method	Fake zebra				Fake horse			
	mFID ↓	mIoU ↑	PSNR ↑	SSIM ↑	mFID ↓	mIoU ↑	PSNR ↑	SSIM ↑
CycleGAN [13]	64.4	0.524	20.62	0.844	**117.6**	0.448	20.52	0.810
AGGAN [12]	47.1	0.735	N/A	N/A	154.3	0.450	N/A	N/A
InstaGAN [16]	92.2	0.545	20.96	0.871	126.5	0.463	19.70	0.834
Ours	**41.5**	**0.739**	**26.40**	**0.931**	124.0	**0.529**	**27.01**	**0.958**

to evaluate the generated images. Specifically, we multiply the input mask by the generated images and the result is sent to compute FID. In addition, we also use mIoU to evaluate the translated object and instance, with the same intuition in semantic image generation [19]. But here we employ an instance segmentation as prediction model, Mask R-CNN [27]. To compare the ability to keep the background, PSNR and SSIM are used [14].

Quantitative comparison: TABLE 3 shows that our method outperforms the current state-of-the-art method by clear margins. Our method is the best to maintain the background since our algorithm achieves the best PSNR and SSIM. We note that CycleGAN produces the best mFID in translating a zebra to a horse but they are the worst to maintain the background. In generating zebra instance, our method achieves the best mFID and mIoU, which suggests that the CSN lets the generator know where to be translated and where to be retained. In contrast, current state-of-the-art models show their inferiority.

Qualitative results: Fig. 7 shows that our algorithm produces decent horses and zebras with the desired background, which is proven in TABLE 3, with much higher PSNR and SSIM than the current state-of-the-art methods. The results validate that our generator with the CSN efficiently uses the condition, input binary mask. Conversely, CycleGAN and InstaGAN tend to change the background, as shown in the second and sixth rows, and the horse and zebra are converted into another domain. Using input background, AGGAN always leads to the artifact on the boundary of the translated instance. The qualitative results suggest that our generator with the CSN outperforms the literature in terms of using the binary mask, class-conditional map, to perform object and instance transfiguration.

V. CONCLUSION

We proposed the class-specific spatial normalization (CSN) layer that efficiently adopts a semantic-class map to generate a natural image. Theoretically, we leveraged a loosened assumption, from adaptive instance normalization, that pixels belonging to each class share specific distribution. The CSN layer benefits semantic image generation aligning given segmentation, which is verified on the Cityscapes and ADE120K datasets with a decent performance, but with few parameters and FLOPs. It also contributes to object transfiguration, translating the given instance and maintain the background between horse and zebra, surpassing the state-of-the-art by a clear margin.

REFERENCES

[1] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[2] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 633–641.

[3] H. Caesar, J. Uijlings, and V. Ferrari, "COCO-stuff: Thing and stuff classes in context," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1209–1218.

[4] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[5] M. Xu, J. Lee, A. Fuentes, D. S. Park, J. Yang, and S. Yoon, "Instance-level image translation with a local discriminator," *IEEE Access*, vol. 9, pp. 111802–111813, 2021.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[7] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2337–2346.

[8] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," 2016, *arXiv:1610.07629*.

[9] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1501–1510.

[10] B. Zhao, L. Meng, W. Yin, and L. Sigal, "Image generation from layout," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8584–8593.

[11] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "SEAN: Image synthesis with semantic region-adaptive normalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5104–5113.

[12] Y. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image-to-image translation," in *Proc. NIPS*, 2018, pp. 1–22.

[13] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2223–2232.

[14] X. Chen, C. Xu, X. Yang, and D. Tao, "Attention-GAN for object transfiguration in wild images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 164–180.

[15] S. Lee and N. U. Islam, "Robust image translation and completion based on dual auto-encoder with bidirectional latent space regression," *IEEE Access*, vol. 7, pp. 58695–58703, 2019.

[16] S. Mo, M. Cho, and J. Shin, "InstaGAN: Instance-aware image-to-image translation," 2018, *arXiv:1812.10889*.

[17] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014, *arXiv:1406.2661*.

[18] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.

[19] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8798–8807.

[20] S. Ioffe and C. Szegedy, ''Batch normalization: Accelerating deep network training by reducing internal covariate shift,'' in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[21] A. Brock, J. Donahue, and K. Simonyan, ''Large scale GAN training for high fidelity natural image synthesis,'' in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–35.

[22] T. Karras, S. Laine, and T. Aila, ''A style-based generator architecture for generative adversarial networks,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.

[23] Y. Shen, M. Luo, Y. Chen, X. Shao, Z. Wang, X. Hao, and Y.-L. Hou, ''Cross-view image translation based on local and global information guidance,'' *IEEE Access*, vol. 9, pp. 12955–12967, 2021.

[24] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, ''Least squares generative adversarial networks,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2794–2802.

[25] L. Wang, W. Chen, W. Yang, F. Bi, and F. R. Yu, ''A state-of-the-art review on image synthesis with generative adversarial networks,'' *IEEE Access*, vol. 8, pp. 63514–63537, 2020.

[26] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, ''Spectral normalization for generative adversarial networks,'' 2018, *arXiv:1802.05957*.

[27] K. He, G. Gkioxari, P. Dollar, and R. Girshick, ''Mask R-CNN,'' in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[28] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' 2014, *arXiv:1412.6980*.

[29] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, ''Learning what and where to draw,'' 2016, *arXiv:1610.02454*.

[30] C. Li and M. Wand, ''Precomputed real-time texture synthesis with Markovian generative adversarial networks,'' in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 702–716.

[31] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, ''StackGAN++: Realistic image synthesis with stacked generative adversarial networks,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1947–1962, Aug. 2019.

[32] A. Odena, C. Olah, and J. Shlens, ''Conditional image synthesis with auxiliary classifier GANs,'' in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.

[33] V. Sushko, E. Schönfeld, D. Zhang, J. Gall, B. Schiele, and A. Khoreva, ''You only need adversarial supervision for semantic image synthesis,'' 2020, *arXiv:2012.04781*.

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

[35] X. Wang, K. Yu, C. Dong, and C. Change Loy, ''Recovering realistic texture in image super-resolution by deep spatial feature transform,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 606–615.

[36] M. D. Zeiler and R. Fergus, ''Visualizing and understanding convolutional networks,'' in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

MINGLE XU received the B.S. degree from Jiangxi Agricultural University, in 2015, and the M.S. degree from the Shanghai University of Engineering Science, in 2018. He is currently pursuing the Ph.D. degree majored in electronic engineering with Jeonbuk National University. His research interests include artificial intelligence and machine learning, computer vision, and image understanding.

YONGCHAE JEONG (Senior Member, IEEE) received the B.S.E.E., M.S.E.E., and Ph.D. degrees in electronics engineering from Sogang University, Seoul, South Korea, in 1989, 1991, and 1996, respectively. From 1991 to 1998, he worked as a Senior Engineer with Samsung Electronics, Seoul. In 1998, he joined the Division of Electronics Engineering, Jeonbuk National University, Jeonju, South Korea. From July 2006 to December 2007, he was a Visiting Professor with the Georgia Institute of Technology, Atlanta, GA, USA. He had also worked as the Director of the HOPE-IT Human Resource Development Center of BK21 PLUS, Jeonbuk National University, where he is currently a Professor and also a member of the IT Convergence Research Center. He is currently teaching and conducting research in microwave passive and active circuits, mobile and satellite base-station RF system, design of periodic defected transmission line (TL), negative group delay circuits and its applications, in-band full duplex radio, and RFIC design. He has authored or coauthored over 250 articles in international journals and conference proceedings. He is a member of the Korea Institute of Electromagnetic Engineering and Science (KIEES).

DONG SUN PARK received the B.S. degree from Korea University, Republic of Korea, in 1979, and the M.S. and Ph.D. degrees from the University of Missouri, USA, in 1984 and 1990, respectively. He is currently a Professor with Jeonbuk National University, Republic of Korea. He has published many papers in international conferences and journals. His research interests include computer vision and artificial neural networks, especially deep learning.

SOOK YOON received the Ph.D. degree in electronics engineering from Jeonbuk National University, South Korea, in 2003. She is currently a Professor with the Department of Computer Engineering, Mokpo National University, South Korea. She was a Researcher in electrical engineering and computer sciences with the University of California at Berkeley, Berkeley, USA, until June 2006. She joined Mokpo National University, in September 2006. She was a Visiting Scholar with the Utah Center of Advanced Imaging Research, University of Utah, USA, from 2013 to 2015. Her research interests include computer vision, object recognition, machine learning, and biometrics.