


Multi-Mapping Image-to-Image Translation with Central Biasing Normalization

Xiaoming Yu, Zhenqiang Ying, Thomas Li, Shan Liu, and Ge Li , *Member, IEEE*,

Abstract—Recent advances in image-to-image translation have seen a rise in approaches generating diverse images through a single network. To indicate the target domain for a one-to-many mapping, the latent code is injected into the generator network. However, we found that the injection method leads to mode collapse because of normalization strategies. Existing normalization strategies might either cause the inconsistency of feature distribution or eliminate the effect of the latent code. To solve these problems, we propose the consistency within diversity criteria for designing the multi-mapping model. Based on the criteria, we propose central biasing normalization to inject the latent code information. Experiments show that our method can improve the quality and diversity of existing image-to-image translation models, such as StarGAN, BicycleGAN, and pix2pix.

Index Terms—normalization, multiple mappings, latent code injection, image-to-image translation.

I. INTRODUCTION

MANY image processing and computer vision problems can be framed as image-to-image translation tasks [1], such as facial synthesis [2]–[4], photo to sketch [5], [6], and image colorization [7]. This can also be viewed as mapping an image from one specific domain to another. Many studies have shown remarkable success in image-to-image translation between two domains, *e.g.* image synthesis [1], inpainting [8], colorization [7] and super-resolution [9]. In these methods, the generative model tries to learn a specific mapping from the source domain to the target domain. However, these one-to-one mapping methods are not suitable for multi-mapping problems, such as the transfer of facial attributes, art styles, or textures. To achieve multi-mapping translation, they need to be built for different pairs of mappings, even though some mappings share common semantics. To overcome this limitation, recent studies [10]–[12] take both image and latent code as input to the generator to learn diverse translations. Specifically, the

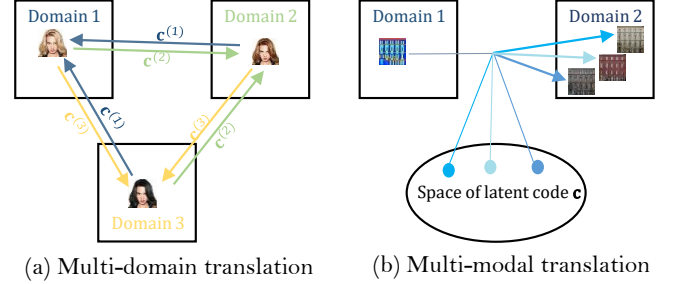


Fig. 1. An illustration of multi-mapping indicated by the latent code c . (a) Facial attribute transfer indicated by the attribute label. (b) `label2photo` indicated by style embedding.

latent code can be the attribute (domain) label for multi-domain translation [10], [11], or the style embedding for multi-modal translation [12].

As shown in Fig. 1, the facial attribute transfer [11] is a typical multi-domain translation task that aims to learn mappings among different attributes, *e.g.*, black/blond/brown for hair color. As for the multi-modal translation, the latent code is usually sampled from a latent space with prior distribution (*e.g.* Uniform or Gaussian priors) to indicate the cross-domain style, such as the facade textures in the `label2photo` task [12]. Both of them attempt to capture the joint output distribution between the input image and latent code by a single generator. But previous works [1], [12] note that trivially injecting a latent code into the network did not help produce diverse results. To prevent this mode collapse phenomenon, recent studies focus on enforcing the generator to make use of the latent code, such as latent regression [12] or domain classification [10], [11]. However, as illustrated in Fig. 2, these methods are sensitive to the choice of network structure, *e.g.* the padding strategies or normalization operations. To tackle this issue, we explore the working mechanism of the multi-mapping models from the perspective of latent code injection (LCI). Through mathematical analysis, we show how latent code can control the target mapping by affecting the mean value of convolutional outputs. Besides, we find that using batch or instance normalizations in multi-mapping models results in ambiguous outputs for different mappings. Thus the performance of the generator is sensitive to the choice of network structures. To tackle this issue, we introduce the *consistency within diversity criteria* for multi-mapping model. With the criteria, we propose *central biasing normalization* (CBN) as an alternative for injecting the latent code into the multi-mapping model. The main idea of CBN is to eliminate

This work was supported by the Project of National Engineering Laboratory for Video Technology-Shenzhen Division, Shenzhen Municipal Science and Technology Program under Grant (JCYJ20170818141146428), and National Natural Science Foundation of China and Guangdong Province Scientific Research on Big Data (No. U1611461). This paper was recommended by Associate Editor X. XX.

X. Yu, Z. Ying, T. Li, and G. Li are with the School of Electronic and Computer Engineering, Shenzhen Graduate School, Peking University, 518055 Shenzhen, China (e-mail: xiaomingyu@pku.edu.cn; zqying@pku.edu.cn; thomasli@pkusz.edu.cn; geli@ece.pku.edu.cn). S. Liu is with the Media Lab, Tencent (e-mail: shanli@tencent.com).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier XX.XXXX/XXX.20XX.XXXXXXX

Manuscript received XXX XX, 20XX; revised XXX XX, 20XX.



Fig. 2. Edge2photo results sampled by BicycleGAN [12]. The first column shows the input image and ground truth. In the remaining columns, the images with the same configuration are generated by sampling different latent codes. We can observe that the diversity and quality of BicycleGAN are sensitive to the choice of normalization and padding strategies.

the inconsistency of feature maps and align them according to the target mapping. By replacing the existing LCI generator with the *central biasing generator* (CBG), we show that our method can improve the stability and performance of multi-mapping translation. In summary, this paper makes the following contributions:

- We show how latent code affects the mean value of the feature maps to control the target mapping in the multi-mapping model.
- We point out the potential problems of common latent code injection and propose the *consistency within diversity criteria*.
- Based on the criteria, we propose central biasing normalization as an alternative to the common latent code injection strategy.

II. RELATED WORK

Benefiting from large public image repositories and high-performance computing systems, convolutional neural networks (CNNs) have been widely used in various image processing problems in recent years. By minimizing the loss function that evaluates the quality of results, CNNs attempt to model the mapping between the source and target domain. However, it is difficult to manually design an effective and universal loss function for different tasks. To overcome this problem, recent studies apply generative adversarial networks (GANs) for different generation tasks because they use metric that adapts to the data rather than the task-specific evaluation.

A. Image-to-Image Translation using GANs

By staging a zero-sum game, GANs have shown impressive results in image generation [13]–[18]. The extensions of this kind of networks with conditional settings (cGAN) [19] have achieved remarkable results in various conditional generation tasks such as image inpainting [8], [20], super-resolution [9], text2image [21], facial synthesis [2]–[4], [22], and photo editing [23]. For more details of GANs, we refer the readers to [24], [25] for excellent overviews.

To extend cGAN as a general-purpose solution for image processing problems, Isola *et al.* [1] define the problem of image-to-image translation and propose pix2pix for tasks with data pairs. However, many image processing tasks are ill-posed due to the lack of paired training data. Thus, CycleGAN

[26], DiscoGAN [27], and DualGAN [28] introduce cycle consistency to achieve unsupervised image translation. To further regularize the unsupervised learning, DistanceGAN [29] proposes the distance constraints to maintain the distance between the samples before and after the mapping. UNIT [30] combines variational autoencoders [31] with CoGAN [32] to learn a joint distribution of images in different domains. These studies have promoted the development of one-to-one mapping translation, but have shown limited scalability for multi-mapping translation.

B. Multi-Mapping Translation

To achieve a more scalable approach for image-to-image translation, researchers have recently made significant progress in multi-mapping translation [10]–[12], as compared in Table I. For instance, StarGAN [11] uses a single model and latent code to achieve multiple domain translations. It learns multiple mappings by the auxiliary classifier [33]. Fader Networks [10] learns the attribute-invariant representation for manipulating the image. BicycleGAN [12] combines VAE-GAN objects [34] and LR-GAN objects [15], [35], [36] for a bijective mapping between the latent code and output spaces. The common feature of these methods is that they encourage the generator to learn a joint distribution between the input image and latent code.

C. Latent Code for Multi-Mapping

For controlling multiple attributes of the generated image, latent code [10]–[12], [15] is introduced for targeting the salient structured semantic features. For instance, in the facial

TABLE I
MODEL COMPARISON OF IMAGE-TO-IMAGE TRANSLATION

Model	Paired Data	Multi-Domain	Multi-Modal	Main Idea
Pix2pix	Need	×	×	Conditional GANs
CycleGAN	No need	×	×	Cycle consistency
DiscoGAN	No need	×	×	Discover cross-domain relations
DualGAN	No need	×	×	Dual learning
DistanceGAN	No need	×	×	Distance constraints
UNIT	No need	×	×	Shared latent space assumption
Fader Networks	No need	✓	×	Invariant latent representation
StarGAN	No need	✓	×	Auxiliary classifier
BicycleGAN	Need	×	✓	Bijective consistency

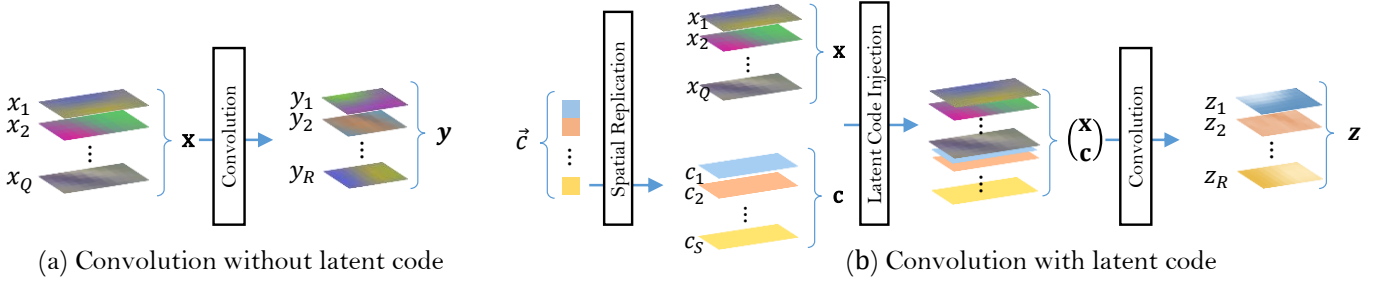


Fig. 3. Convolution operation without/with latent code. (a) The common convolution without latent code. (b) The convolution with latent code injection.

attribute transfer task, the latent code \vec{c} indicates the specific features, such as gender, expression, or hair color. In existing multi-mapping models [10]–[12], latent code \vec{c} is used as an input to the convolution layer by spatial replication. However, this naive injection strategy is unreliable and may lead to mode collapse. We discuss this problem in Section IV and compare our method with StarGAN and BicycleGAN in Section VI.

III. COMMON LATENT CODE INJECTION

In this section, we first explore the existing injection mechanism by formulating the convolution operation. Then we revisit the normalization to facilitate the later analysis in Section IV.

A. Convolution Operation without Latent Code

Following the notation of Convolutional Matrix Multiplication [37], we extend the matrix of numbers to the matrix of feature maps or convolution kernels. *Here, each element is a feature map or a convolution kernel instead of a number.*

Let x_1, x_2, \dots, x_Q be the Q input feature maps (each sized $M \times N$) and $w_{r,q}$ be the $R \times Q$ convolution kernels (each sized $K \times L$) where $r = 1, 2, \dots, R$ and $q = 1, 2, \dots, Q$. Then the R output feature maps y_1, y_2, \dots, y_R can be represented as

$$\begin{aligned} y_1 &= w_{1,1} * x_1 + w_{1,2} * x_2 + \dots + w_{1,Q} * x_Q \\ y_2 &= w_{2,1} * x_1 + w_{2,2} * x_2 + \dots + w_{2,Q} * x_Q \\ &\vdots \\ y_R &= w_{R,1} * x_1 + w_{R,2} * x_2 + \dots + w_{R,Q} * x_Q, \end{aligned} \quad (1)$$

where $*$ is the convolution operation. Further, these equations can be redefined as a special matrix/vector multiplication

$$\mathbf{y} = \mathbf{W} \times \mathbf{x}, \quad (2)$$

where $\mathbf{x} = (x_1, x_2, \dots, x_Q)^T$, $\mathbf{y} = (y_1, y_2, \dots, y_R)^T$, and

$$\mathbf{W} = \begin{pmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,Q} \\ w_{2,1} & w_{2,2} & \dots & w_{2,Q} \\ \vdots & \vdots & \ddots & \vdots \\ w_{R,1} & w_{R,2} & \dots & w_{R,Q} \end{pmatrix}.$$

B. Convolution Operation with Latent Code

As shown in Fig. 3, we denote the special vector \mathbf{c} as the S latent code feature maps, where $\mathbf{c} = (c_1, c_2, \dots, c_S)^T$. The elements of \mathbf{c} are replicated from the numerical element of original latent code \vec{c}

$$c_s(m, n) = \vec{c}_s,$$

where $s = 1, 2, \dots, S$; $m = 1, 2, \dots, M$; $n = 1, 2, \dots, N$ and c_s is a constant feature map in which every element has the same value. We denote $v_{r,s}$ as the $R \times S$ convolution kernel that is associated with the latent code. Then

$$\mathbf{o} = \mathbf{V} \times \mathbf{c}, \quad (3)$$

where $\mathbf{o} = (o_1, o_2, \dots, o_R)^T$, and

$$\mathbf{V} = \begin{pmatrix} v_{1,1} & v_{1,2} & \dots & v_{1,S} \\ v_{2,1} & v_{2,2} & \dots & v_{2,S} \\ \vdots & \vdots & \ddots & \vdots \\ v_{R,1} & v_{R,2} & \dots & v_{R,S} \end{pmatrix}.$$

Note each feature map o_r is a constant channel as it is the linear combination of feature maps from \mathbf{c} :

$$o_r = v_{r,1} * c_1 + v_{r,2} * c_2 + \dots + v_{r,S} * c_S. \quad (4)$$

The whole convolution operation from input \mathbf{x} and \mathbf{c} can be represented as

$$\mathbf{z} = (\mathbf{W}, \mathbf{V}) \times \begin{pmatrix} \mathbf{x} \\ \mathbf{c} \end{pmatrix} = \mathbf{y} + \mathbf{o}, \quad (5)$$

where $\mathbf{z} = (z_1, z_2, \dots, z_R)^T$ is the final convolution output.

We make two observations about the convolution with latent code. First, the final convolution output \mathbf{z} can be decomposed into two separate parts \mathbf{y} and \mathbf{o} . It means that the target mapping is only determined by the latent code \mathbf{c} . Second, different latent codes only provide different offsets to the outputs, since the elements of \mathbf{o} are constant feature maps. Thus different mappings of the same input image differ only in the mean value. It implies that the network needs to distinguish between different mappings of an input image based on the mean value of feature maps. Hence, we refer to the consistency of mean value as the consistency of mapping in this work.

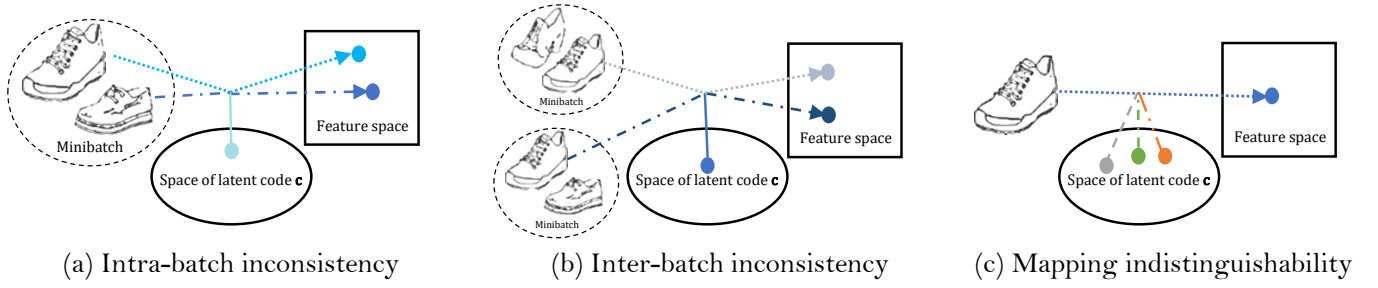


Fig. 4. The potential problems of existing LCI models. (a) The intra-batch inconsistency problem of BN. (b) The inter-batch inconsistency problem of BN. (c) The Mapping indistinguishable problem of IN.

C. Normalization

1) *Batch Normalization*: To accelerate training and improve the performance of deep networks, Ioffe and Szegedy introduced batch normalization (BN) [38] to reduce the internal covariate shift [39] of neural networks. Although BN is designed to ease the training of discriminator, it is also effective in deep generative models [18]. In the training stage, the input minibatch mean and standard deviation are used to normalize each feature map of the convolutional output:

$$\text{BN}(z_r) = \frac{z_r - \mu(Z_r)}{\sigma(Z_r)}, \quad (6)$$

where $Z_r = (z_r^{(1)}, z_r^{(2)}, \dots)$ is the batch of the r -th feature map; $\mu(\cdot)$ and $\sigma(\cdot)$ are the mean operator and standard deviation operator. During the inferencing stage, BN replaces the minibatch statistics with moving statistics.

2) *Instance Normalization*: In style transfer tasks, Ulyanov *et al.* [40] found that critical improvement could be achieved by replacing batch normalization with instance normalization (IN). Recent image transformation tasks [11], [12], [26] confirm that IN is also useful for improving the quality of transformation results. Unlike BN, which uses minibatch statistics, IN only uses the statistics of the instance itself for normalization:

$$\text{IN}(z_r) = \frac{z_r - \mu(z_r)}{\sigma(z_r)}. \quad (7)$$

Consequently, the feature statistics used by IN are consistent between the training and inference stages.

3) *Scale and Shift*: A simple normalization operation $\text{Norm}(\cdot)$ for convolution output may change what it can represent. Therefore, the affine parameters γ_r , and β_r are used for restoring the representation power of the network [38]:

$$\hat{z}_r = \gamma_r \text{Norm}(z_r) + \beta_r. \quad (8)$$

IV. ANALYSIS OF PROBLEMS WITH COMMON LCI

According to the above analyses, we know that the role of the latent code is to provide offsets for output feature maps. In this section, we further analyze the effect of latent code in the common convolution pipeline (Convolution-Normalization-Activation). First, we use some examples to explain how normalization confuses the statistical mean of output feature map and degrades the model performance. Then, we explain why the activation function cannot help

clear the ambiguity caused by the normalization. Finally, we propose the *consistency within diversity criteria* for multi-mapping model.

A. Impaired Consistency of Batch Normalization

1) *Intra-batch Inconsistency*: Consider this minibatch $\mathbf{Z}^{(1)} = (\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \mathbf{z}^{(3)})$, where

$$\begin{aligned} \mathbf{z}^{(1)} &= (\mathbf{W}, \mathbf{V}) \times \begin{pmatrix} \mathbf{x}^{(1)} \\ \mathbf{c}^{(1)} \end{pmatrix} = \mathbf{y}^{(1)} + \mathbf{o}^{(1)}, \\ \mathbf{z}^{(2)} &= (\mathbf{W}, \mathbf{V}) \times \begin{pmatrix} \mathbf{x}^{(2)} \\ \mathbf{c}^{(1)} \end{pmatrix} = \mathbf{y}^{(2)} + \mathbf{o}^{(1)}, \\ \mathbf{z}^{(3)} &= (\mathbf{W}, \mathbf{V}) \times \begin{pmatrix} \mathbf{x}^{(3)} \\ \mathbf{c}^{(1)} \end{pmatrix} = \mathbf{y}^{(3)} + \mathbf{o}^{(1)}. \end{aligned} \quad (9)$$

In batch $\mathbf{Z}^{(1)}$, different input instances $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}$ map to the same domain indicated by the latent code $\mathbf{c}^{(1)}$. It implies that the distribution of instances in $\mathbf{Z}^{(1)}$ should be similar in the feature space. But without any distribution alignment operations, the statistical mean of the feature is inconsistent in different instances

$$\begin{aligned} &\mu(\text{BN}(z_r^{(1)})) - \mu(\text{BN}(z_r^{(2)})) \\ &= \mu(\text{BN}(z_r^{(1)}) - \text{BN}(z_r^{(2)})) \\ &= \mu\left(\frac{z_r^{(1)} - z_r^{(2)}}{\sigma(Z_r^{(1)})}\right). \end{aligned} \quad (10)$$

Since $z_r^{(1)} = y_r^{(1)} + o_r^{(1)}$ and $z_r^{(2)} = y_r^{(2)} + o_r^{(1)}$. Hence

$$\begin{aligned} &\mu(\text{BN}(z_r^{(1)})) - \mu(\text{BN}(z_r^{(2)})) \\ &= \frac{\mu(y_r^{(1)} - y_r^{(2)})}{\sigma(Z_r^{(1)})}. \end{aligned} \quad (11)$$

In most cases we have $\mu(y_r^{(1)}) \neq \mu(y_r^{(2)})$, i.e., $\mu(y_r^{(1)} - y_r^{(2)}) \neq 0$, therefore

$$\mu(\text{BN}(z_r^{(1)})) \neq \mu(\text{BN}(z_r^{(2)})). \quad (12)$$

Similarly, we can derive $\mu(\text{BN}(z_r^{(1)})) \neq \mu(\text{BN}(z_r^{(2)})) \neq \mu(\text{BN}(z_r^{(3)}))$. Although these three instances have the same mapping, their statistical mean is inconsistent. As the example shown in Fig. 4(a), different input images with the same latent code cannot have a consistent mapping in the feature space because of the above problem. We call this phenomenon *intra-batch inconsistency*. This inconsistency is considered to be the shortcoming of BN in style transfer tasks [41].

2) *Inter-batch Inconsistency*: Replace $\mathbf{z}^{(3)}$ in $\mathbf{Z}^{(1)}$ with $\mathbf{z}^{(4)}$, where

$$\mathbf{z}^{(4)} = (\mathbf{W}, \mathbf{V}) \times \begin{pmatrix} \mathbf{x}^{(3)} \\ \mathbf{c}^{(2)} \end{pmatrix} = \mathbf{y}^{(3)} + \mathbf{o}^{(2)}. \quad (13)$$

The new batch is $\mathbf{Z}^{(2)} = (\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \mathbf{z}^{(4)})$. Here, we use the statistical mean of the same mapping instances to represent the main characteristic of a specific mapping. In batch $\mathbf{Z}^{(1)}$, the statistical mean of the mapping indicated by $\mathbf{c}^{(1)}$ is $\mu(\text{BN}(\mathbf{Z}^{(1)})) = 0$, but the mean value indicated by $\mathbf{c}^{(1)}$ in batch $\mathbf{Z}^{(2)}$ is

$$\begin{aligned} & \mu((\text{BN}(z_r^{(1)}), \text{BN}(z_r^{(2)}))) \\ &= \frac{1}{2} \mu(\text{BN}(z_r^{(1)})) + \frac{1}{2} \mu(\text{BN}(z_r^{(2)})) \\ &= \frac{\mu(z_r^{(1)}) + \mu(z_r^{(2)}) - 2\mu(Z_r^{(2)})}{2\sigma(Z_r^{(2)})}. \end{aligned} \quad (14)$$

Since $\mu(Z_r^{(2)}) = \frac{1}{3}\mu(z_r^{(1)}) + \frac{1}{3}\mu(z_r^{(2)}) + \frac{1}{3}\mu(z_r^{(3)})$. Hence

$$\begin{aligned} & \mu((\text{BN}(z_r^{(1)}), \text{BN}(z_r^{(2)}))) \\ &= \frac{\mu(Z_r^{(2)}) - \mu(z_r^{(3)})}{2\sigma(Z_r^{(2)})} \\ &= -\frac{1}{2} \mu(\text{BN}(z_r^{(3)})). \end{aligned} \quad (15)$$

In most cases, $\mu(\text{BN}(z_r^{(3)})) \neq 0$. Therefore, the mean value of the same mapping is always inconsistent between different batches. It means that the inconsistency also exists between different batches in the training stage. We call this phenomenon *inter-batch inconsistency*. This problem causes the same input has different results in different minibatch, as shown in Fig. 4(b). Due to the above phenomenon, the network has to continuously adapt to the changeful distribution of different mappings during the training stage. Therefore, the network will suffer from covariate shift [39] which reduces its performance. We call these problems *mapping inconsistency*.

B. Impaired Diversity of Instance Normalization

The *mapping inconsistency* of BN is caused by normalizing the feature statistics of the minibatch. IN does not have this problem since it normalizes the feature statistics of an instance, but IN will eliminate the diversity indicated by the latent code.

Consider these two instances $\mathbf{z}^{(3)}$ and $\mathbf{z}^{(4)}$. With the same input $\mathbf{x}^{(3)}$, different mappings are indicated by $\mathbf{c}^{(1)}$ and $\mathbf{c}^{(2)}$. After IN

$$\begin{aligned} & \text{IN}(z_r^{(3)}) - \text{IN}(z_r^{(4)}) \\ &= \frac{y_r^{(3)} + o_r^{(1)} - \mu(y_r^{(3)} + o_r^{(1)})}{\sigma(y_r^{(3)} + o_r^{(1)})} - \frac{y_r^{(3)} + o_r^{(2)} - \mu(y_r^{(3)} + o_r^{(2)})}{\sigma(y_r^{(3)} + o_r^{(2)})} \\ &= \frac{y_r^{(3)} - \mu(y_r^{(3)})}{\sigma(y_r^{(3)})} - \frac{y_r^{(3)} - \mu(y_r^{(3)})}{\sigma(y_r^{(3)})} = 0, \\ & \Rightarrow \text{IN}(z_r^{(3)}) = \text{IN}(z_r^{(4)}). \end{aligned} \quad (16)$$

Since o_r is the constant feature map, the diversity indicated by the latent code will be eliminated after IN. It means that the model cannot distinguish different mappings indicated by the latent code and suffer from mode collapse, i.e., the input image with different latent codes has same results despite the injecting information, as shown in Fig. 4(c). We call this problem *mapping indistinguishability*.

C. Effect of Activation Function

Before outputting the normalized results, we usually append an activation function for mapping the resulting values into the desired range. In most cases, the neural network requires an activation function to introduce the nonlinear property to solve the nonlinear problems [42], and the monotonic property to ease the optimization [43]. Although the nonlinear property will complicate the final results, the monotonic property guarantees the mapping order after the activation function. Therefore, the mapping problems caused by normalization still exist after activation.

D. Consistency Within Diversity Criteria

With the above potential problems, the common LCI model has difficulty in learning the stable mapping indicated by the latent code. Motivated by the role of latent code, we argue that the feature statistics, especially the mean value of the feature map, represents the target mapping. Therefore, we propose two criteria for designing the multi-mapping model. To simplify the expression, we define the multi-mapping result as $\phi(\mathbf{x}, \vec{c})$, where \vec{c} is the original latent code.

1) *Consistency criterion*: To reduce mapping inconsistency, different $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ should produce consistent outputs when given the same latent code \vec{c} .

$$\mu(\phi(\mathbf{x}^{(1)}, \vec{c})) = \mu(\phi(\mathbf{x}^{(2)}, \vec{c})). \quad (17)$$

It means that the statistical mean of the mapping function $\phi(\mathbf{x}, \vec{c})$ should not relate to the input \mathbf{x} .

2) *Diversity criterion*: To maintain diversity, same input \mathbf{x} should produce diversity outputs when given different $\vec{c}^{(1)}$ and $\vec{c}^{(2)}$. Therefore, the mean value of the outputs should not be identical:

$$\mu(\phi(\mathbf{x}, \vec{c}^{(1)})) \neq \mu(\phi(\mathbf{x}, \vec{c}^{(2)})). \quad (18)$$

In other words, the statistical mean of the mapping function $\phi(\mathbf{x}, \vec{c})$ should be related to the input \vec{c} .

Previous multi-mapping approaches [10]–[12] focused on finding effective loss function to generate diverse outputs. As a result, these methods meet the diversity criterion. However, due to the mapping inconsistency problem, they do not satisfy the consistency criterion. Therefore, we refer the above criteria to as *consistency within diversity criteria* to emphasize the mapping consistency.

V. PROPOSED METHOD BASED ON CENTRAL BIASING NORMALIZATION

In this section, we first propose the central biasing operation for common normalization according to the above criteria.

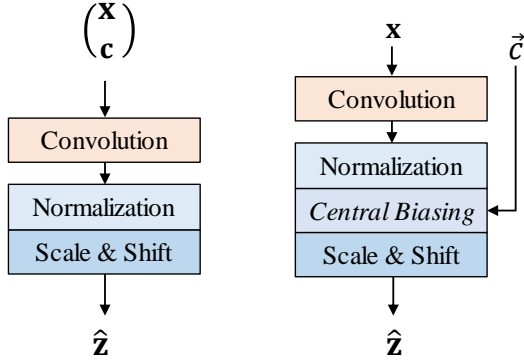


Fig. 5. Left: a traditional convolutional block with latent code injection. Right: a convolutional block with proposed central biasing normalization.

Then we apply the proposed central biasing normalization (CBN) to construct our central biasing generator (CBG) for multi-mapping translation.

A. Central Biasing Normalization

To meet the proposed criteria, an intuitive approach is to add related constraints or regularizations to the loss function in the training stage. But controlling the importance of the additional loss term is a new challenge. We can also remove the normalization to reduce the impact of potential problems. But it causes the network to be difficult to converge. Rethinking the potential problems discussed in Section IV, we observe that the mapping indicated by the latent code is confusing because of the zero-centered operation in normalization. If we can separate the mapping learning from the feature extraction of the convolution, we can avoid the above issues. Therefore, we propose a novel normalization strategy for learning the specific mapping indicated by the latent code. Our proposed method is easy to incorporate into existing multi-mapping models and orthogonal to the ongoing exploration about learning strategy.

1) *Formulation*: We first eliminate the offset of normalization feature maps to meet the consistency criterion, which aligns the distribution center of different instances. Then we append a bias calculated by the latent code to encourage the model to meet the diversity criterion. We call this method as central biasing normalization. It can be represented as

$$\text{CBN}(y_r, \vec{c}) = \text{Norm}(y_r) - \mu(\text{Norm}(y_r)) + b_r(\vec{c}), \quad (19)$$

$$b_r(\vec{c}) := \tanh(f(\vec{c})), \quad (20)$$

where $\text{Norm}(\cdot)$ represents the common normalization, $b_r(\vec{c})$ is the bias for the r -th output feature map, and $f(\cdot)$ represents the affine transformation. Using this initialized bias $f(\vec{c})$ also meets the *design criteria*, but we argue it goes against the intention of the normalization operation. In [38], the normalization layer is introduced to accelerate deep network training. It can reduce the internal covariate shift of the network by ensuring the input distribution is stable for the next convolution layer. But without any constraints, the feature distribution after adding bias $f(\vec{c})$ is unknown since there is no constraint on the range of bias. To stabilize the output

TABLE II
THE NETWORK ARCHITECTURE OF CBG. “ $C \times K \times L \text{ Conv } S_n P_m$ ” DENOTES C n -STRIDE CONVOLUTIONAL FILTERS WITH $K \times L$ KERNEL SIZE AND m SIZED ZERO PADDING OR REFLECTION PADDING. H AND W ARE THE HEIGHT AND WIDTH OF THE INPUT IMAGE, RESPECTIVELY.

Convolution	Norm	Activation	Output Size
$64 \times 7 \times 7 \text{ Conv } S1 P3$	CBN	ReLU	$64 \times H \times W$
$128 \times 4 \times 4 \text{ Conv } S2 P1$	CBN	ReLU	$128 \times \frac{H}{2} \times \frac{W}{2}$
$256 \times 4 \times 4 \text{ Conv } S2 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$256 \times 3 \times 3 \text{ Res } S1 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$256 \times 3 \times 3 \text{ Res } S1 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$256 \times 3 \times 3 \text{ Res } S1 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$256 \times 3 \times 3 \text{ Res } S1 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$256 \times 3 \times 3 \text{ Res } S1 P1$	CBN	ReLU	$256 \times \frac{H}{4} \times \frac{W}{4}$
$128 \times 4 \times 4 \text{ TrConv } S2 P1$	BN/IN	ReLU	$128 \times \frac{H}{2} \times \frac{W}{2}$
$64 \times 4 \times 4 \text{ TrConv } S2 P1$	BN/IN	ReLU	$64 \times H \times W$
$3 \times 7 \times 7 \text{ TrConv } S1 P3$	/	Tanh	$3 \times H \times W$

distribution, we append a \tanh function to constrain the range of bias. The final bias for the feature map is defined as Eq. 20. The mean value of feature is $\mu(\text{CBN}(y_r, \vec{c})) = b_r(\vec{c})$, where $b_r(\vec{c}) \in [-1, 1]$. With the scale and shift operation, the distribution after normalization is stable for the next layer.

2) *Proof*: By substituting Eq. 6 into Eq. 17, we observe that the batch normalization does not meet the consistency criterion.

$$\begin{aligned} \mu(\text{BN}(z_r)) &= \mu\left(\frac{z_r - \mu(Z_r)}{\sigma(Z_r)}\right) = \frac{\mu(z_r) - \mu(Z_r)}{\sigma(Z_r)} \\ &= \frac{\mu(y_r + o_r) - \mu(Y_r + O_r)}{\sigma(Z_r)} \\ &= \frac{\mu(y_r - \mu(Y_r))}{\sigma(Z_r)} + \frac{\mu(o_r - \mu(O_r))}{\sigma(Z_r)}. \end{aligned} \quad (21)$$

Eq. 21 shows that the mean value of the feature map after BN relates to the input \mathbf{x} and minibatch, which results in *mapping inconsistency*.

According to Eq. 7 and Eq. 18, we observe that instance normalization violates the diversity criterion.

$$\mu(\text{IN}(z_r)) = \mu\left(\frac{z_r - \mu(z_r)}{\sigma(z_r)}\right) = \frac{\mu(z_r) - \mu(z_r)}{\sigma(z_r)} = 0. \quad (22)$$

Eq. 22 indicates that IN will eliminate the effects of the latent code, which leads to *mapping indistinguishability*.

By contrast, these normalizations will meet the criteria by applying the central biasing operation.

$$\begin{aligned} \mu(\text{CBN}(y_r, \vec{c})) &= \mu(\text{Norm}(y_r) - \mu(\text{Norm}(y_r)) + b_r(\vec{c})) \\ &= \mu(\text{Norm}(y_r)) - \mu(\mu(\text{Norm}(y_r))) + \mu(b_r(\vec{c})) \\ &= \mu(b_r(\vec{c})). \end{aligned} \quad (23)$$

For specific normalizations BN and IN, CBN can be simplified as CBBN and CBIN respectively:

$$\text{CBBN}(y_r, \vec{c}) = \frac{y_r - \mu(y_r)}{\sigma(\mathbf{Y}_r)} + b_r(\vec{c}), \quad (24)$$

$$\text{CBIN}(y_r, \vec{c}) = \frac{y_r - \mu(y_r)}{\sigma(y_r)} + b_r(\vec{c}). \quad (25)$$

It should be noted that CBBN eliminates the instance mean $\mu(y_r)$ instead of batch mean $\mu(\mathbf{Y}_r)$, and that the standard deviation of the output depends on the batch instead of instances like CBIN.

B. Central Biasing Generator

Instead of common LCI strategy, we inject the latent code into the normalization layers by replacing traditional normalization with central biasing normalization, as shown in Fig. 5. We adopt the common encoder-decoder architecture [11], [26], [44] to build the generator, which contains two stride-2 convolution layers for downsampling, six residual blocks [45] and two stride-2 transposed convolution layers for upsampling. The normalization layers in the downsampling and residual blocks are replaced with central biasing normalization layers. We refer this generator to as the central biasing generator. The details of the network are shown in Table II.

VI. EXPERIMENTS

As discussed in Section IV, the existing latent code injection model has some potential problems when learning multiple mappings. Why are similar convolution pipelines in the existing works [11], [12] still working? The reason is that these networks introduce zero padding (ZP) before the convolution operation, which aims to control the spatial size of the output volume. After zero padding, the latent code channel of the input volume is no longer a constant plane but has a circle of zero boundaries. Through the convolutional operation, the convolution output o_r in Eq. 4 is not a constant feature map. The activation boundaries give the possibility of keeping diversity in non-boundary areas after instance normalization. However, the problems still exist if we remove the zero padding or use other padding strategies, such as reflection padding (RP). In this section, we first verify the potential problems of LCI models and then analyze the effects of padding strategies in multi-mapping models. Finally, we further explore the proposed central biasing normalization in several ablation studies.

A. Baselines

To verify the potential problems, we apply different normalization (BN&IN) and padding strategies (ZP&RP) to the state-of-the-art multi-mapping translation models.

1) *StarGAN*: To model multi-domain mappings with a single model, StarGAN introduces the auxiliary classifier [33] for the discriminator. Since StarGAN uses the attribute label as the latent code \vec{c} , the latent space is discrete, and the state of \vec{c} is limited.

2) *BicycleGAN*: To model the distribution of possible outputs, BicycleGAN combines the VAE-GAN [34] and LR-GAN [15], [35], [36] objectives for encouraging a bijective mapping between the latent and output spaces. Since BicycleGAN uses the encoder with a Gaussian assumption to extract the latent code \vec{c} , the latent space is continuous and the state of \vec{c} is varied. To further compare the effect between LCI and CBN, we also remove the encoder of BicycleGAN and use the noise vector as the latent code. Without the incentive for the generator to make use of the latent code, BicycleGAN degenerates into pix2pix [1].

B. Datasets

To evaluate the performance of StarGAN, we study the facial attribute transfer on the CelebA¹ dataset [46]. According to [11], we select 2,000 test images from 202,599 face images and retain the rest to training. Seven facial attributes are selected for experimentations: hair color (black, blond, brown), gender (male/female), and age (young/old). Besides, we compare the performance of BicycleGAN-based models on several multi-model translation tasks, including edge2photo² [47], [48] and label2photo³ [49]. The default training and test sets of these datasets are used for all experiments. For each task, we resize the image to 128×128 resolution and follow the suggested training strategy in [11], [12].

C. Metrics

1) *Classification Accuracy*: To judge the translation accuracy of the multi-domain translation, we compare the classification accuracy of facial attributes on synthesized images. Firstly, we train a ResNet-18 [45] as the multi-label classifier on the CelebA dataset. Then we translate each test image to all possible targets (12 domain of facial attributes). Finally, we classify the translated images using the ResNet classifier.

2) *Diversity and Consistency Score*: To compare the diversity of different multi-modal models, we compute the averaged perceptual distance in feature space. As suggested in [12], we use the cosine similarity (CosSim) to evaluate the feature distance of the VGG-16 network [50] pre-trained on ImageNet. We average across spatial dimensions and sum across the five convolution layers preceding the pool layers. As in Zhu *et al.* [12], the perceptual distance is defined as $(5.0 - \sum_{i=1}^5 \text{CosSim}_i)$. The larger the perceptual distance, the greater the difference between the two images. To evaluate the diversity of different models, we randomly sample an input image and use a pair of random latent code to generate images. Then, we compute the average distance between 2,000 pairs of generated images. In addition to the diversity, the mapping consistency is also important to the multi-modal translation. We first use the latent code encoded by ground truth to indicate the image generation. Then we calculate the average reconstruction loss to measure the consistency of models. More specifically, the consistency score is calculated

¹<http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html>

²<http://people.csail.mit.edu/junyanz/projects/gvm>

³<http://cmp.felk.cvut.cz/~tylecr1/facade>

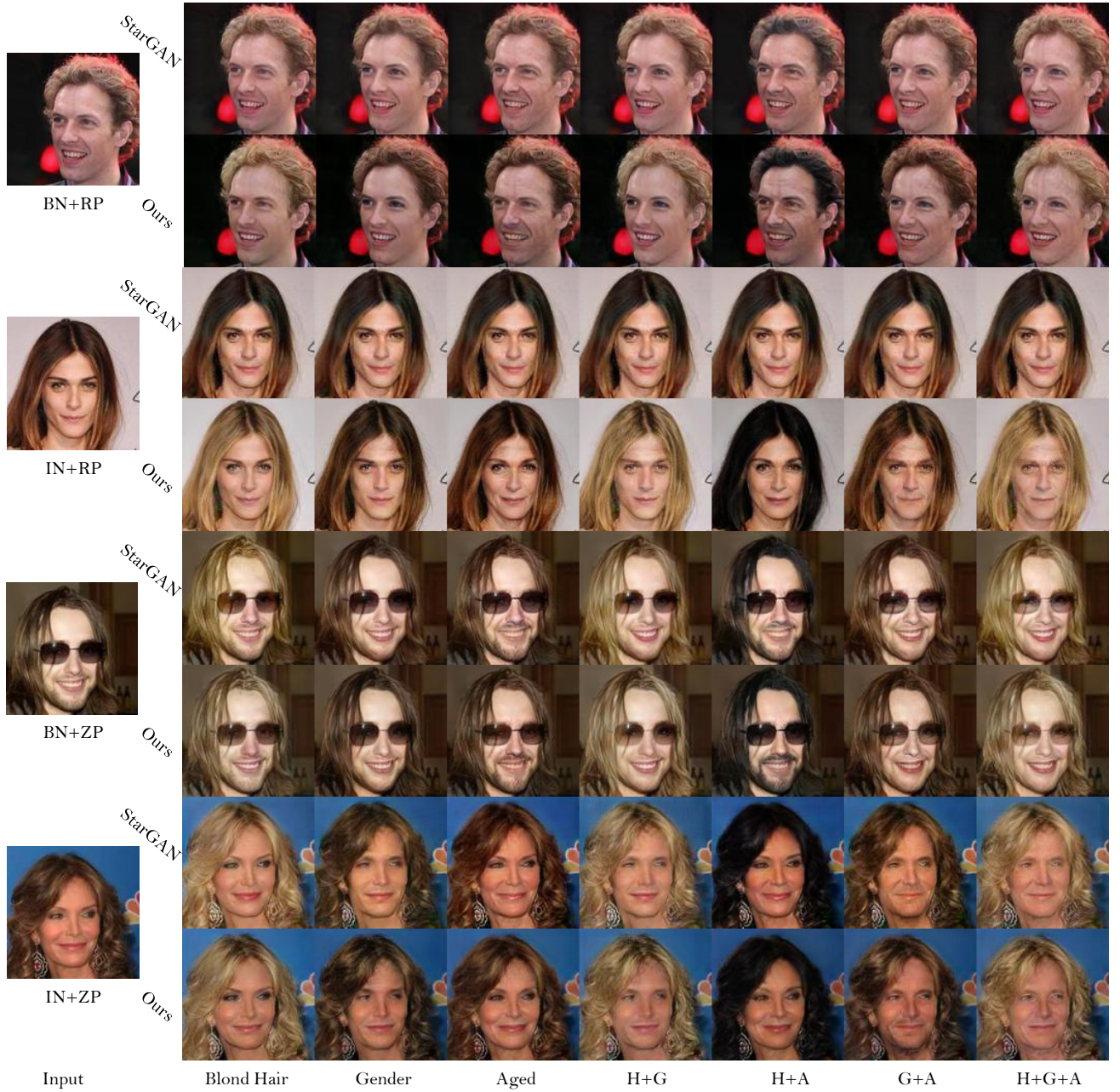


Fig. 6. Facial attribute transfer results. The first column shows the input images, the next three columns show the single attribute transformation results, and the rest of the columns show the multi-attribute transfer results. H: Hair color, G: Gender, A: Aged.

TABLE III

THE CLASSIFICATION ACCURACY FOR FACIAL ATTRIBUTE TRANSFER.

Method		Hair color	Gender	Aged	Average
BN+RP	StarGAN	70.90%	61.16%	61.57%	64.54%
	Ours	81.93%	68.00%	63.21%	71.05%
IN+RP	StarGAN	33.33%	50.00%	50.00%	44.44%
	Ours	96.46%	88.55%	86.05%	90.36%
BN+ZP	StarGAN	84.25%	79.53%	66.09%	76.62%
	Ours	88.63%	76.32%	67.76%	77.57%
IN+ZP	StarGAN	95.25%	88.38%	86.05%	89.89%
	Ours	95.74%	90.20%	86.11%	90.68%
Real images		91.56%	97.25%	88.74%	92.52%

TABLE IV

THE FRÉCHET INCEPTION DISTANCE AND HUMAN EVALUATION SCORE FOR FACIAL ATTRIBUTE TRANSFER.

Method		FID	Human evaluation
BN+RP	StarGAN	20.71	34.67%
	Ours	19.83	65.33%
IN+RP	StarGAN	20.26	15.63%
	Ours	16.50	84.37%
BN+ZP	StarGAN	17.53	48.20%
	Ours	17.45	51.80%
IN+ZP	StarGAN	16.72	49.67%
	Ours	16.62	50.33%

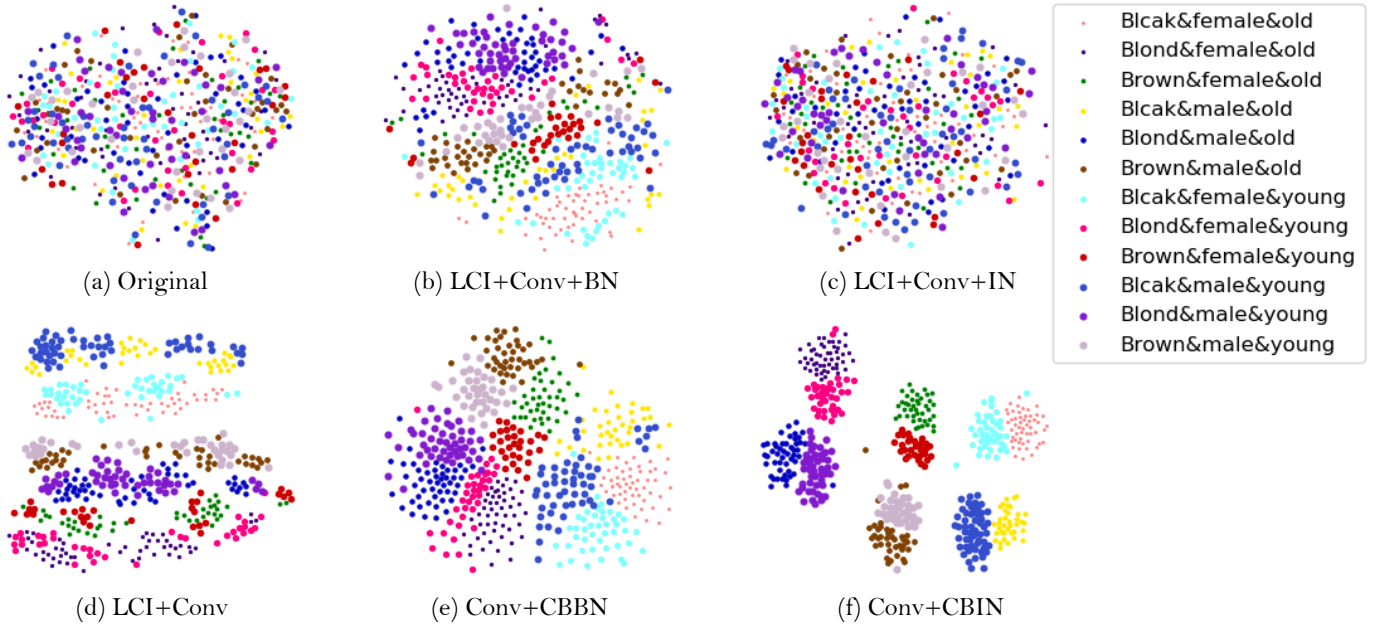


Fig. 7. Feature visualization of facial attribute transfer. Different colors represent different mappings indicated by the latent code.

by $(1.0 - \|G(Enc(y), x) - y\|_1)$, where x and y are the input and target images, Enc is the encoder, and G represents the generator. Intuitively, the higher the consistency score, the stronger the ability of the model to generate images with the specified style.

3) *Fréchet Inception Distance*: To measure the quality of the generated images, we adopt Fréchet Inception Distance (FID) [51] to evaluate the similarity between the generated images and the real images. Specifically, we translate each facial image to all possible combinations of attributes in the multi-domain translation task, and randomly generate 10 different samples for each image in the multi-modal translation task. Then we compute the FID score between the distribution of generated images and real images. The lower the FID score, the better the quality of the generated images.

4) *Human Perception*: To compare the realism of translation outputs, we conduct a human perceptual study on Amazon Mechanical Turk (AMT). In the multi-domain translation task, the workers are given a description of attributes and two translation outputs from the methods with and without CBN. They are given unlimited time to select which image looks more realistic and fits the given attributes. Similarly, in the multi-modal translation task, the workers are required to choose a more realistic one from the images generated by different methods. We randomly generate 500 questions for each comparison, and each question is finished by 5 different workers. A higher score indicates that the generated images are more realistic.

D. Verification of Potential Problems

1) *Experiments on CelebA Dataset*: The qualitative comparison results are shown in Fig. 6. We observe that StarGAN fails to generate diversity outputs when it is equipped with reflection padding. These results have improved after it is

equipped with zero padding. By replacing the original generator of StarGAN with the proposed CBG model, we observe that the results in all settings are both diverse and realistic.

The quantitative results further confirm the observations above. As shown in Table III, we observe that the classification accuracy of the generated results is improved after StarGAN is equipped with CBN. The FID scores and human perception results, shown in Table IV, also suggest that the proposed CBN can help the model to learn multiple mappings more efficiently and to improve the quality of the generated images.

In order to illustrate the potential problems of the LCI model more intuitively, we visualize the feature embeddings by using t-SNE [52] in Fig. 7. All of the models are equipped with reflection padding to satisfy the constant assumption in Eq. 4. Since each input image is randomly translated to a possible target domain, we can observe that the t-SNE embeddings of original inputs are chaotic in Fig. 7(a). Despite the injection of latent code, the embeddings cannot be well clustered, as shown in Fig. 7(d). This observation also holds after batch normalization because of the mapping inconsistency problem. After instance normalization, the embeddings are chaotic again because of the mapping indistinguishable problem. In contrast, the embeddings of different mappings are clustered well after central biasing normalization.

2) *Experiments on Edge2photo Dataset*: To verify the potential problems of LCI model in multi-modal translation task, we evaluate BicycleGAN by the edge2photo task. As the qualitative comparison shows in Fig. 8, we can get observations similar to the facial attribute transfer above: the CBN based models produce more realistic results while maintaining diversity. As shown in Table V, we can observe that our method obtains higher diversity and consistency scores than BicycleGAN under the same settings. The FID scores and human perception results shown in Table VI further



Fig. 8. Edge2photo results. The first column shows the input images and the remaining columns are the transfer results indicated by random latent codes.

TABLE V
THE DIVERSITY AND CONSISTENCY SCORES FOR EDGE2PHOTO TASK.

Method	Diversity	Consistency
BN+RP BicycleGAN	0.069	0.906
Ours	0.867	0.951
IN+RP BicycleGAN	1.025	0.949
Ours	1.652	0.966
BN+ZP BicycleGAN	0.492	0.939
Ours	0.754	0.951
IN+ZP BicycleGAN	1.106	0.952
Ours	1.637	0.966
Real images	3.481	N/A

TABLE VI
THE FRÉCHET INCEPTION DISTANCE AND HUMAN EVALUATION SCORE FOR EDGE2PHOTO TASK.

Method	FID	Human evaluation
BN+RP BicycleGAN	85.67	43.43%
Ours	41.64	56.57%
IN+RP BicycleGAN	57.29	45.27%
Ours	34.23	54.73%
BN+ZP BicycleGAN	68.18	45.93%
Ours	42.32	54.07%
IN+ZP BicycleGAN	44.30	49.10%
Ours	32.19	50.90%

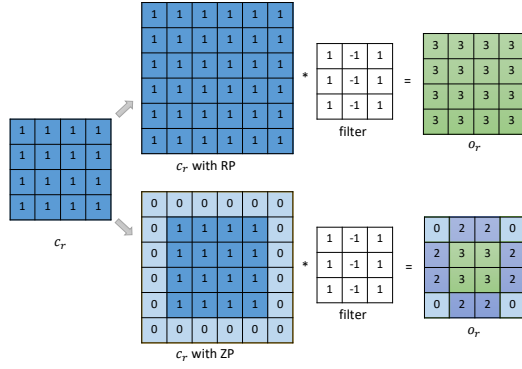


Fig. 9. Different padding operations. The first column is the source latent code channel c_r . The first row in the rest of the columns is the convolution with reflection padding, and the second row is the convolution with zero padding.

demonstrate our analysis of the potential problems.

E. Analysis of Padding Strategies

In the above experiments, we observe that the improvements of the model with ZP are smaller than the model with RP after applying CBN, especially in facial attribute transfer task. To analyze this phenomenon, let us revisit the padding strategy through Fig. 9. As discussed in the previous sections, the latent code just provides a constant offset in RP mode, so there are mapping inconsistency and mapping elimination problems. For ZP mode, the convolution of latent code provides the feature map which contains non-constant boundaries and a constant center region. Therefore, the network has the ability to control the distribution of features for different mappings after normalization. Since the mean value of the feature is normalized to zero after IN, the common LCI model with ZP is inconsistent with our *diversity criteria*. But apart from the non-constant boundaries, the input latent code does provide the constant offsets for different output feature maps. We think the key to distinguishing different mappings is also the mean value in non-boundary feature regions.

To verify the above conjecture, we test the StarGAN under the settings of IN+ZP and find that the statistics of non-boundary feature regions are strongly related to the target mapping. We first sample the feature maps z which are the normalized outputs after convolution operation. Then, we calculate the mean value of features without considering the boundary area affected by padding operation. To reduce the computational complexity, we use principal component analysis (PCA) to reduce the dimension of the statistical embeddings. While retaining 96% variance, we reduce the data from 64 to 6 dimensions. After performing k-means ($k=12$) clustering in the low dimensional data, we find that each cluster represents a kind of mapping, as shown in Fig. 10. According to the above experiment, we confirm that ZP can alleviate the mapping problems discussed above. Therefore, when the model is equipped with ZP, the improvement of our method is not obvious in the facial attribute transfer task. But unlike CBN, which can align the statistical mean of the same mapping, we observe the clusters are not compact in Fig. 10.

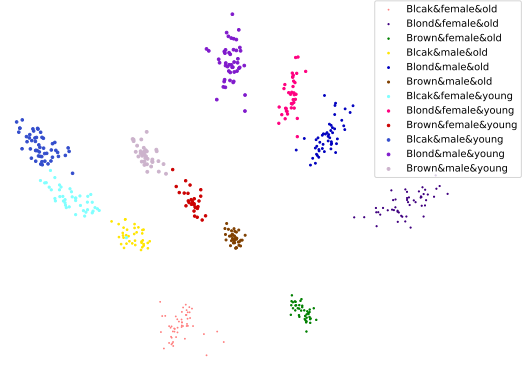


Fig. 10. The results of k-means clustering from 2-dimensional statistical data. Different colors represent different mappings indicated by the latent code.

TABLE VII
THE PERCEPTUAL DISTANCE FOR CBG WITH DIFFERENT CONSTRAINT FUNCTIONS.

Constraint	edge→photo	label→photo	Average
None	1.577 ^a	1.609	1.593
Tanh	1.637	1.562	1.600
Sigmoid	1.427	1.166	1.297

^a The results were calculated from the generated images before mode collapse.

It implies the decision boundaries of different mappings are not sharp. When there are the countless target mappings, *e.g.* edge2photo task, the models without CBN suffer from mapping inconsistency again and produce monotonous outputs.

F. Ablation Studies and Analyses

1) *Range of Bias*: As we analyzed in Section V-A1, adding the unconstrained bias goes against the intention of the normalization. Therefore, it is necessary to constrain the range of bias. Here, we further explore the impact of different bias ranges by BicycleGAN. We extend the range of bias by removing the constraint function and narrow the range by replacing tanh with sigmoid. As shown in Table VII, we find that sigmoid is inferior to tanh in diversity performance. It implies that the narrow range of bias will limit the representation of the network. This finding is further validated when we remove the constraint function to expand the range in the label→photo task. But we also observe that the model without constraint function is unstable in the training stage because the feature distribution is unbounded. This phenomenon is obvious in the edge→photo task due to its diverse target styles. In this task, we find that the generator is easy to collapse when we remove the constraint function. Therefore, we propose to apply the tanh function to constrain the range of bias while maintaining the capacity of the network.

2) *The Length of Latent Code*: As in Zhu *et al.* [12], we explore the effect of latent code length on model performance. Under varying numbers of dimensions of latent codes $\{2, 8, 128, 256\}$, we test the default BicycleGAN, which uses IN and ZP, and CBG with same settings. Similar to the results of Zhu *et al.* [12], a high-dimensional latent code can potentially encode more information for image generation at the cost of making sampling quite difficult for the common



Fig. 11. Label2photo results with varying length of the latent code. The images in each pair show randomly generated samples. We observe that larger $|z|$ can encode more information for expressive results but is not conducive to BicycleGAN output densely fill results.

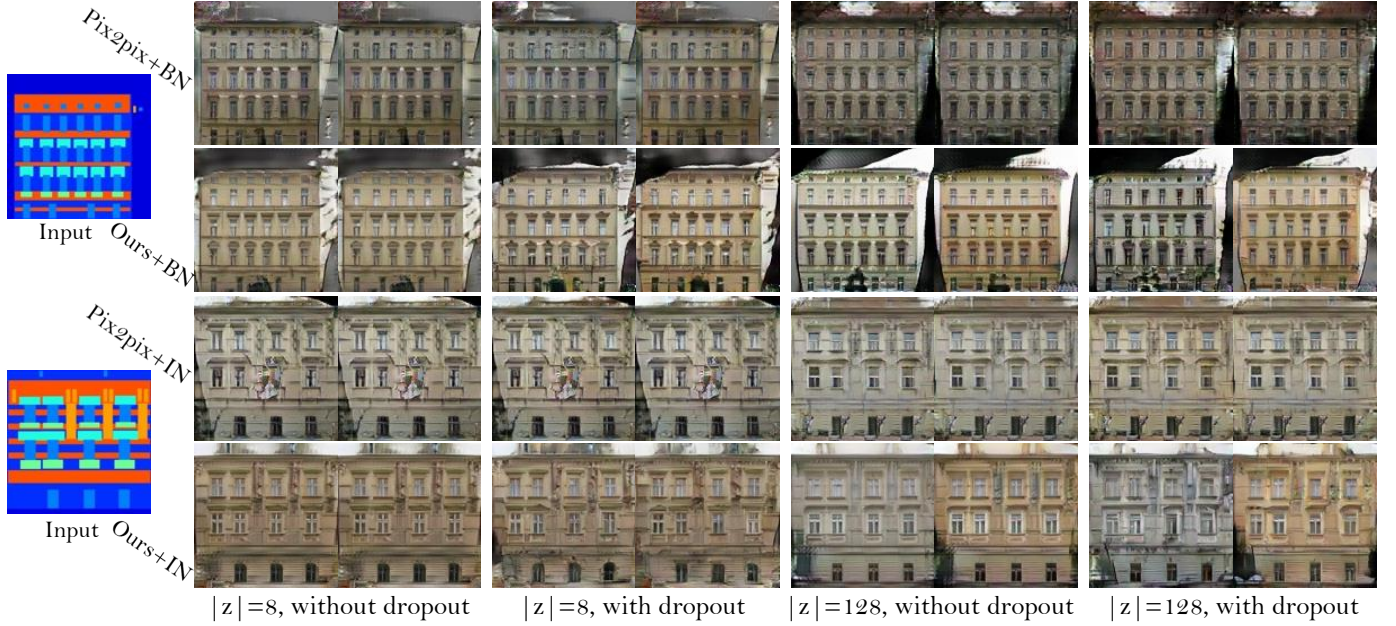


Fig. 12. Qualitative comparison of pix2pix based models. The images in each pair are generated by sampling different latent codes.

TABLE VIII
GENERATOR PARAMETERS WITH DIFFERENT LENGTH OF LATENT CODE.

Base model	BicycleGAN	CBG
	39.9M	8M
$ c = 2$	+78K	+6.9K
$ c = 8$	+312K	+27.5K
$ c = 128$	+4.9M	+440K
$ c = 256$	+9.75M	+880K

LCI model. In contrast, our method shows stable performance when the dimension of the latent code is high enough, as shown in Fig. 11. Besides, the parameters introduced by CBN are typically negligible. Table VIII shows the comparison of parameters between BicycleGAN and CBG. Due to the replication of latent code in the common LCI model, the convolutional parameters for latent code (convolution matrix V in Eq. 3) are redundant. For a fair comparison, we just use the suggested latent code with dimension 8 [12] in the above experiments.

3) *The Incentive of Training:* To further explore the effectiveness of CBN when it lacks the incentive to make use of the latent code, we randomly sample Gaussian noise as the latent code injects into pix2pix [1], which can be considered as the degradation of BicycleGAN. Similar to [1], we apply dropout with a rate of 50% to the generator (Convolution-Norm-Dropout-ReLU) to increase the stochasticity in the output. We also extend the dimension of the latent code to reinforce its effectiveness. The diversity results are shown in Fig. 13, and the qualitative comparison results are presented in Fig. 12. Similar to the results in [1], [12], we observe that injecting noise into the original pix2pix does not produce a large variation. The situation has not improved even though we applied dropout and extended the dimension of the latent code. The same results also appear when we directly replace the generator in pix2pix with CBG. After applying the dropout operation, we observe that the diversity score is improved by introducing stochasticity in the image structure. By extending the dimension of latent code, the outputs become diverse and the score is further improved. When we apply both strategies

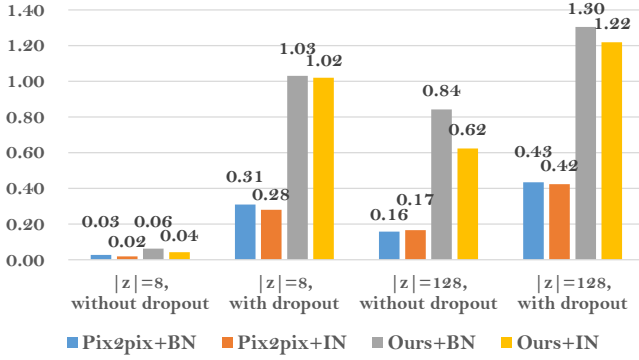


Fig. 13. The perceptual distance of pix2pix based models. We compare diversity results on the `label2photo` across different construction strategies.

to CBG, we get the best diversity performance in the pix2pix based model.

4) *Convergence Rate*: In this section, we give empirical evidence to demonstrate that the proposed CBN can accelerate the training of the multi-mapping model. By comparing the reconstruction loss of BicycleGAN with and without CBN in Fig. 14, we can observe that CBN accelerates the model convergence and reduces the final reconstruction loss. The reason is that CBN can reduce the internal covariate shift of the generator as mentioned in Section V.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we study the latent code injection in the convolution neural network and show the potential problems through different multi-mapping models, *e.g.* StarGAN, BicycleGAN, and pix2pix. By decomposing the convolution output, we show that the latent code controls the target mapping by providing offsets to the output feature maps. With further analysis, we find the *mapping inconsistency* and *mapping indistinguishability* in the existing methods. To overcome these problems, we propose the *consistency within diversity criteria* as a guide to designing the multi-mapping model. Based on the criteria, we propose central biasing normalization to replace the existing latent code injection strategy. There are many advantages of our method:

- It solves the mapping inconsistency and indistinguishability caused by normalization.
- It is insensitive to the selection of network structure, *e.g.* the length latent code, padding strategies, or normalization operations.
- It has fewer parameters and faster convergence than common LCI model.
- It is easy to integrate into the latent-code-based models.

Among a variety of multi-mapping translation tasks, CBN provides a solid improvement over baselines. Our CBN can be used in a variety of image processing tasks that need to inject auxiliary information into the convolution neural network, such as arbitrary image stylization, multi-view generation, and diverse image inpainting. For more specific applications, we refer the reader to our related works SingleGAN⁴ [53] and

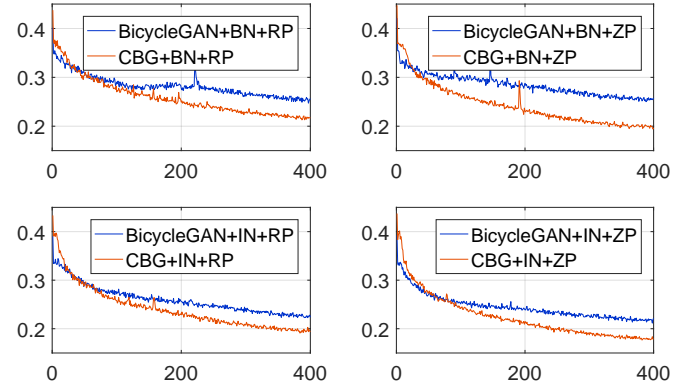


Fig. 14. Comparison of convergence rate on `label2photo`. The horizontal axis shows the iteration and the vertical axis represents the reconstruction loss.

DMIT⁵ [54]. In our future work, we will investigate whether CBN can help extend the representation ability of the discriminative model. For example, whether the domain-related information can be injected into the network for tackling the dataset bias problem [55] will be explored. Besides, the bias provided by central biasing normalization is indiscriminate for the entire feature map, even though the transfer is non-global, *e.g.* facial expression transfer. Combining our method with spatial attention [56], [57] or saliency mechanisms [58]–[61] would be an interesting work to improve the local transformation. Finally, we believe that this work is valuable for studying the multi-mapping translation. Further exploration will allow the proposed method to be more universal and effective in different applications.

REFERENCES

- [1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Computer Vision and Pattern Recognition*, 2017.
- [2] Y. Tian, X. Peng, L. Zhao, S. Zhang, and D. N. Metaxas, “Crgan: Learning complete representations for multi-view generation,” in *International Joint Conference on Artificial Intelligence*, 2018.
- [3] Y. Taigman, A. Polyak, and L. Wolf, “Unsupervised cross-domain image generation,” in *International Conference on Learning Representations*, 2017.
- [4] J. Huang, M. Tan, Y. Yan, C. Qing, Q. Wu, and Z. Yu, “Cartoon-to-photo facial translation with generative adversarial networks,” in *Asian Conference on Machine Learning*, 2018, pp. 566–581.
- [5] S. Zhang, X. Gao, N. Wang, J. Li, and M. Zhang, “Face sketch synthesis via sparse representation-based greedy search,” *IEEE Transactions on Image Processing*, 2015.
- [6] D.-P. Fan, S. Zhang, Y.-H. Wu, Y. Liu, M.-M. Cheng, B. Ren, P. L. Rosin, and R. Ji, “Scoot: A perceptual metric for facial sketches,” in *International Conference on Computer Vision*, 2019.
- [7] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *European Conference on Computer Vision*, 2016.
- [8] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, “Context encoders: Feature learning by inpainting,” in *Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.
- [9] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Computer Vision and Pattern Recognition*, 2017.
- [10] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer *et al.*, “Fader networks: Manipulating images by sliding attributes,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5969–5978.

⁴<https://github.com/Xiaoming-Yu/SingleGAN>

⁵<https://github.com/Xiaoming-Yu/DMIT>

- [11] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Computer Vision and Pattern Recognition*, 2018.
- [12] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Advances in Neural Information Processing Systems* 30, 2017.
- [13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [14] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *International Conference on Computer Vision*, 2017, pp. 2813–2821.
- [15] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2016, pp. 2172–2180.
- [16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223.
- [17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5769–5779.
- [18] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations*, 2016.
- [19] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014.
- [20] Y. Ren, X. Yu, R. Zhang, T. H. Li, S. Liu, and G. Li, "Structureflow: Image inpainting via structure-aware appearance flow," in *International Conference on Computer Vision*, 2019.
- [21] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *International Conference on Machine Learning*, 2016, pp. 1060–1069.
- [22] Y. Ren, X. Yu, J. Chen, T. H. Li, and G. Li, "Deep image spatial transformation for person image generation," *Computer Vision and Pattern Recognition*, 2020.
- [23] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Neural photo editing with introspective adversarial networks," in *International Conference on Learning Representations*, 2017.
- [24] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.
- [25] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, 2018.
- [26] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *International Conference on Computer Vision*, 2017.
- [27] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 1857–1865.
- [28] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *International Conference on Computer Vision*, 2017, pp. 2868–2876.
- [29] S. Benaim and L. Wolf, "One-sided unsupervised domain mapping," in *Advances in neural information processing systems*, 2017, pp. 752–762.
- [30] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 700–708.
- [31] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.
- [32] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in neural information processing systems*, 2016, pp. 469–477.
- [33] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," 2016.
- [34] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *International Conference on Machine Learning*, 2016.
- [35] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *International Conference on Learning Representations*, 2016.
- [36] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," in *International Conference on Learning Representations*, 2016.
- [37] J. Cong and B. Xiao, "Minimizing computation in convolutional neural networks," in *International conference on artificial neural networks*, 2014, pp. 281–290.
- [38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.
- [39] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of statistical planning and inference*, vol. 90, no. 2, pp. 227–244, 2000.
- [40] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," in *Computer Vision and Pattern Recognition*, 2017.
- [41] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *International Conference on Computer Vision*, 2017.
- [42] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.
- [43] H. Wu, "Global stability analysis of a general class of discontinuous neural networks with linear growth activation functions," *Information Sciences*, vol. 179, no. 19, pp. 3432–3441, 2009.
- [44] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016, pp. 694–711.
- [45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [46] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *International Conference on Computer Vision*, 2015, pp. 3730–3738.
- [47] A. Yu and K. Grauman, "Fine-grained visual comparisons with local learning," in *Computer Vision and Pattern Recognition*, 2014, pp. 192–199.
- [48] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *European Conference on Computer Vision*, 2016, pp. 597–613.
- [49] R. Tyleček and R. Šára, "Spatial pattern templates for recognition of objects with regular structure," in *German Conference on Pattern Recognition*, 2013, pp. 364–374.
- [50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computer Science*, 2014.
- [51] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.
- [52] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [53] X. Yu, X. Cai, Z. Ying, T. Li, and G. Li, "Singlegan: Image-to-image translation by a single-generator network using multiple generative adversarial learning," in *Asian Conference on Computer Vision*, 2018.
- [54] X. Yu, Y. Chen, S. Liu, T. Li, and G. Li, "Multi-mapping image-to-image translation via learning disentanglement," in *Advances in Neural Information Processing Systems*, 2019.
- [55] A. Torralba, A. A. Efros *et al.*, "Unbiased look at dataset bias," in *Computer Vision and Pattern Recognition*, 2011.
- [56] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image-to-image translation," in *Advances in Neural Information Processing Systems*, 2018.
- [57] C. Yang, T. Kim, R. Wang, H. Peng, and C.-C. J. Kuo, "Show, attend and translate: Unsupervised image translation with self-regularization and attention," *Transactions on Image Processing*, 2019.
- [58] D.-P. Fan, M.-M. Cheng, J.-J. Liu, S.-H. Gao, Q. Hou, and A. Borji, "Salient objects in clutter: Bringing salient object detection to the foreground," in *European Conference on Computer Vision*, 2018.
- [59] J. Zhao, J. Liu, D. Fan, Y. Cao, J. Yang, and M.-M. Cheng, "Egnet: edge guidance network for salient object detection," in *International Conference on Computer Vision*, 2019.
- [60] J.-X. Zhao, Y. Cao, D.-P. Fan, M.-M. Cheng, X.-Y. Li, and L. Zhang, "Contrast prior and fluid pyramid integration for rgb-d salient object detection," in *Computer Vision and Pattern Recognition*, 2019.
- [61] W. Wang, J. Shen, H. Sun, and L. Shao, "Video co-saliency guided co-segmentation," *Transactions on Circuits and Systems for Video Technology*, 2017.