Research Article

DAGAN: A Domain-Aware Method for Image-to-Image Translations

Xu Yin , **Yan Li**, and **Byeong-Seok Shin**

Department of Computer Engineering, Inha University, Incheon 082, Republic of Korea

Correspondence should be addressed to Byeong-Seok Shin; bsshin@inha.ac.kr

Received 19 November 2019; Revised 30 January 2020; Accepted 25 February 2020; Published 28 March 2020

Academic Editor: Mahardhika Pratama

Copyright © 2020 Xu Yin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The image-to-image translation method aims to learn inter-domain mappings from paired/unpaired data. Although this technique has been widely used for visual predication tasks—such as classification and image segmentation—and achieved great results, we still failed to perform flexible translations when attempting to learn different mappings, especially for images containing multiple instances. To tackle this problem, we propose a generative framework DAGAN (Domain-aware Generative Adversarial etwork) that enables domains to learn diverse mapping relationships. We assumed that an image is composed with background and instance domain and then fed them into different translation networks. Lastly, we integrated the translated domains into a complete image with smoothed labels to maintain realism. We examined the instance-aware framework on datasets generated by YOLO and confirmed that this is capable of generating images of equal or better diversity compared to current translation models.

1. Introduction

Image-to-image translation methods [1, 2] have received increased attention in recent years. This type of generative model can be applied to many tasks related to vision, such as art restoration, image synthesis, and resolution enhancement [3, 4]. With the development of deep learning techniques, many interesting problems about this topic have been put forth and solved [5], such as noise reduction [6] and brightness enhancement [7]. Multiple output generation [8–10] and image realism improvements are examples of this technology. However, almost all research focuses on the translation of full images instead of domains. In this work, with presaved identity matrices, we propose a generative framework DAGAN, which can flexibly translate the instance domain of the original image. As shown in Figure 1, we use identity matrices to successfully segregate instances from original images and translate, respectively. The results demonstrate our model can make domain-aware translations and generate diverse and realistic generations.

Motivated by research concerning variational autoencoders (VAEs) and generative adversarial networks (GAN), existing translation models [10–12] can compare multiple

maps of one image to produce possible translation outputs. When looking at these solutions, especially under an unpaired setting in recent research, a common solution is to view the image-to-image (I2I) problem as a process of learning the joint distribution of the original and target domains [9]. By using VAEs and weight-sharing scheme, an image can be represented with common but low-dimensional latent codes. In this way, neural networks can be trained to produce images that contain styles or specificities of both domains.

One major limitation of models of current research is that we cannot totally control image translations. We still fail to manipulate the translation level and areas. For example, if an image contains multiple instances, is it possible to translate a certain instance into a style that is different from other instances? This sounds like what is usually performed using commercial software such as Photoshop, but until now, making a domain-aware translation is still challenging.

To tackle this problem, we assume that images are composed of different domains, and for each domain, we built a subtranslation model. At the same time, we followed the pipeline of “segregation → integration,” whereby generations from domain-translation models would finally

FIGURE 1: Domain-aware translation of unpaired images. We perform translations of the background and instance domains, respectively, and then integrate both domains with smoothed labels. First, each image is segregated into background and instance domains (as shown in the “background” and “instance” row). After translating, respectively, we integrated both translation generations into a full image (the integration row). (a) Integration. (b) Instance. (c) Background.

be integrated into a full image. Generally speaking, we used the UNIT [11] (Unsupervised Image-to-image Translation) framework as the basis of our overall translation works and then treated a full image as a combination of instance and background domains. First, identity matrices were used to record location information of instance domains that needed to be translated individually, and we successfully segregated the instances from the given image and assigned the original location (instance areas) with the mean pixel value of the original image [13]. Next, we translated the segregated background and instance domains, respectively. Last, we utilized identity matrices to integrate the output produced from the background networks with the reconstructed/translated instances.

Our contributions in this work are summarized as follows:

- (1) We built a domain-aware I2I framework, DAGAN, and applied a background and instance network to facilitate translations for both domains. Then, we specially designed two different modes for the

instance part, which helped users flexibly control translations in training.

- (2) With label-smoothing training, we made reintegrated images more realistic and comfortable.
- (3) Compared to current image translation research, DAGAN enables domain-aware translations but maintains the quality of the generations. Extensive qualitative experiments on benchmarks show that our framework compares favorably against current translation methods. The results from our method maintain realism and great diversity.

2. Related Works

Since implementation in [14, 15], GAN models have achieved encouraging results in many vision tasks [5, 6]. The most common use of GAN [16, 17] is to enforce the mapping of generated images to target domains using adversarial processes. This kind of generative model can be trained to produce realistic images from random noise vectors.

Furthermore, research studies [18–20] have explored combining VAEs with GANs. Common VAE architecture consists of encoder and decoder networks where the encoder learns an interpretable representation z (called latent space) from given images x . With the reliable representation of input, we can control the direction of image processing by choosing the added visual attribute vectors. This kind of operation helps decoders produce better reconstructions or transformations. The core of VAE is that it regularizes the encoder by enforcing the variational posterior $q(x|z)$ as close to the true posterior $p(x|z)$ as possible. The methods presented in this work have been built on conditional VAE and GAN models in both background and instance parts, and we aim to learn visual attributes from target domains. By jointly optimizing the objectives of the instance network and the background network, we learned of a shared latent space C , where a trade-off occurs between the source and target domains when transforming.

As for practical application research in this field, we conclude that these general translation problems aim to learn the mapping relationship from a given image to target domains, but retaining content attributes and semantic consistency in training brings great challenges. In working with Pix2Pix [1], the authors constructed models using paired data to enforce mapping. Although transformation results from Pix2Pix are very realistic, considering the lack of training pairs, solutions using unpaired data are more general and applicable in industrial applications. Zhu et al.'s work [2] used cycle-GANs under the same conditions (with unpaired data) to successfully produce high-quality images. Overall, cycle-GANs use high frequencies to hide information and make it imperceptible to humans to ensure that the generator can recover samples later [21]. By cycle-like losses, which encourage inverse translations (bidirectional), Zhu enforced mapping in training. While in the work of UNIT [11], Liu provided another perspective of the learning translation problem. Translation can be seen as a process of learning joint distributions, so they created a common space for both domains. The idea of UNIT inspired many further research efforts [9, 10], but when looking back on that research, although many problems have been solved and generations perfectly fit their new distributions, we still cannot perform flexible translations wherever we want. For example, we cannot change the background but retain the instance area's specifics or perform a special instance translation. To summarize, we still cannot manipulate the extent and direction of transformations.

In this work, with the help of identity matrices, we separate background and instance parts from source images. Overall, we follow the assumptions of UNIT, and set up a common latent space to maintain the stability of the models. With improvements in the framework and training strategies, we enabled the proposed model to flexibly translate different domains from unpaired inputs.

3. Proposed Models

The proposed framework aims to perform an instance-aware translation of a given image A and a target B . If we assume that each image is composed of the background (bgr) and

instance (ins) areas, then image A and B can be represented as follows: $A: \{A_{\text{bgr}}, A_{\text{ins}}\}$, and $B: \{B_{\text{bgr}}, B_{\text{ins}}\}$. As illustrated in Figure 2, before training the entire network, we cropped instance areas A_{ins} and B_{ins} from A and B and saved both areas' location values as identity matrices called labels. Cropped areas are then replaced by the mean value of the rest of the image. After training both background and instance networks with labels (not used during training), we can recover the translated images (this step is called integration).

In Section 3.1, we discuss the background and instance parts of images, and in Section 3.2, we further modify the instance domain to produce diverse translated outputs.

3.1. Background Network. Having separated instance area from the input image, the rest part (called background area) is processed by padding with mean pixel values. The background model can be seen as an independent part of the entire translation framework and this model will learn a visual translation between the two domains. Following this assumption, we set an encoder (denoted with E) and generator (denoted with G) for each side (here, for each parameter appearing in this work, we marked the type of domain with its subscript):

$$\begin{aligned} N_{\text{bgr}}^A: \{E_{\text{bgr}}^A, G_{\text{bgr}}^A\}, \quad E_{\text{bgr}}^A \in C_{\text{bgr}}, \\ N_{\text{bgr}}^B: \{E_{\text{bgr}}^B, G_{\text{bgr}}^B\}, \quad E_{\text{bgr}}^B \in C_{\text{bgr}}. \end{aligned} \quad (1)$$

Similar to UNIT, we assume that, with encoder E_{bgr}^A and E_{bgr}^B , we can map a given target background into a common latent space $C_{\text{bgr}}: \{C_{\text{bgr}}^A, C_{\text{bgr}}^B\}$, where $C_{\text{bgr}}^A = E_{\text{bgr}}^A(A_{\text{bgr}})$ and $C_{\text{bgr}}^B = E_{\text{bgr}}^B(B_{\text{bgr}})$. Then, C_{bgr}^A and C_{bgr}^B represent the latent codes of domains A_{bgr} and B_{bgr} , respectively. In our work, we shared the weight of the last two layers of E_{bgr}^A and E_{bgr}^B and the first layer of G_{bgr}^A and G_{bgr}^B . At the same time, we add two discriminators, D_{bgr}^A and D_{bgr}^B , which ensure that the translation between two background domains can be performed in an adversarial process, where $G_{\text{bgr}}^A(C_{\text{bgr}}^B)$ and A_{bgr} are distinguished by D_{bgr}^A while $G_{\text{bgr}}^B(C_{\text{bgr}}^A)$ and B_{bgr} are distinguished by D_{bgr}^B . Figure 3 illustrates the background model in a visual way.

3.2. Instance Network. We designed two modes for instance network: *reconstruction* and *multioutput*. Overall, the goal of the instance model is to keep the instance area relatively independent from background areas, so as to not influence the instance area with the background transformation.

First, we adopt similar DCGAN (Deep Convolution Generative Adversarial Networks) [22] architecture for simple instance reconstructions, drawing noise vectors from the Gaussian distribution $N(0, 1)$. This enables the entire framework (including the background) to successfully translate the background area into another style but leave the instance unchanged. Then, inspired by [23] and based on the assumption that images are composed with style and content codes, we adapt the instance part into a multioutput model.

FIGURE 2: Architecture overview, our proposed method consists of two subparts, background and instance models.

FIGURE 3: Background model overview; this model aims to learn a cross-domain transformation between both background inputs.

3.2.1. Reconstruction Mode. This mode aims to keep the instance part unchanged and the final integrated images as realistic as possible (the loss functions used will be discussed in later sections). Considering the size of the instance, we have three convolutional layers in both the generator and discriminator, which takes the normal distribution noise $Z \sim N(0, 1)$ as the input vector. The process is illustrated in Figure 4.

3.2.2. Multioutput Mode. Having been cropped from the original image, the instance area can be considered an independent image. Under such conditions, we can perform any desired translation work on this domain. A common approach to achieve diverse outputs lies in treating images as a combination of style and content information. In principle, we can translate images into any style if we add suitable attributes/style

codes. Similar to the setting in MUNIT [9] (Multimodal Unsupervised Image-to-Image Translation), we produced multiple instance generations by randomly sampling style codes that drawn from the target instance and then recombined them with content codes. Assuming we are given instance domain A_{ins} and target instance B_{ins} ; first, we map A_{ins} and B_{ins} to the style and content space, respectively (stand by S_{ins}^A , C_{ins}^A , S_{ins}^B , and C_{ins}^B , representing style and content codes), where the corresponding encoders are represented as $E_{A_{ins}}^S$, $E_{A_{ins}}^C$, $E_{B_{ins}}^S$, and $E_{B_{ins}}^C$:

$$\begin{aligned} \{S_{ins}^A, C_{ins}^A\} &= \{E_{A_{ins}}^S(A_{ins}), E_{A_{ins}}^C(A_{ins})\}, & C_{ins}^A &\in C_{ins}, \\ \{S_{ins}^B, C_{ins}^B\} &= \{E_{B_{ins}}^S(B_{ins}), E_{B_{ins}}^C(B_{ins})\}, & C_{ins}^B &\in C_{ins}. \end{aligned} \quad (2)$$

Like the common space setting in the background part, content space is also shared by both instances. Then, we

FIGURE 4: Reconstruction mode. Three-blocked discriminators aim to distinguish the original input and generator (z). For the generator, its block consists of “Transposed convolution + Batch normalization + ReLU activation,” while the discriminator’s block is composed of “Convolution + Batch normalization + Leaky ReLU activation.”

integrated content codes C_{ins}^A and C_{ins}^B with the randomly sampled style codes $\sigma(S_{ins}^B)$ and $\sigma(S_{ins}^A)$ from S_{ins}^B and S_{ins}^A . With cycle semantics consisting of enforcing training, we achieved multiple bidirectional instance translations. The detailed architecture is shown in Figure 5.

4. Loss Functions and Training

Considering the model difference between the background and instance parts, the loss functions used, especially for different modes, should be adjusted accordingly. We jointly solve background and instance translation problems overall by using the full objective function, and then we discussed the loss functions used for both parts (denoted with L_{ins} and L_{bgr}):

$$\text{Loss}_G = \min_D \max \left[\min_{G_{ins}, E_{ins}} \max_{D_{ins}} L_{ins} + \min_{G_{bgr}, E_{bgr}} \max_{D_{bgr}} L_{bgr} \right]. \quad (3)$$

4.1. Instance Loss. We designed two modes that make the instance part a flexible transformation. The first mode focuses on ensuring that instances remain unchanged when the background is translated, and the second mode concentrates on producing a range of instance translations. Theoretically speaking, having been cropped from the given images, we can then flexibly add any code to these instances to manipulate their style.

4.1.1. Reconstruction Mode. As shown in Figure 3, the instance network in this mode consists of a noise vectors Z_A , and Z_B , two generators, and two discriminators, each with three layers (represented by G_{ins}^A , G_{ins}^B , D_{ins}^A , and D_{ins}^B). The adversarial loss is expressed as follows, where M denotes the batch size when training:

$$\begin{aligned} L_{ins}^A &= E_{A_{ins} \sim p(A_{ins}), Z_B \sim p(Z_B)} \left[\frac{1}{2} \log D_{ins}^A(G_{ins}^A(Z_A)) + \frac{1}{2} \log(1 - D_{ins}^A(A_{ins})) \right] \\ &= -\frac{1}{2M} \left(\sum_{G_{ins}^A(Z_A)}^M \log D_{ins}^A(G_{ins}^A(Z_A)) + \sum_{D_{ins}^A(A_{ins})}^M \log(1 - D_{ins}^A(A_{ins})) \right). \end{aligned} \quad (4)$$

4.1.2. Multioutput Mode. As illustrated in Figure 5, the encoder-generator collection $\{E_{A-ins}^S, E_{A-ins}^C, G_{ins}^A\}$ constitutes the single part A_{ins} . The L_{ins} in this mode comprises a reconstruction loss and an adversarial loss.

4.2. Reconstruction Loss. Generally speaking, instances are processed in translation as images to latent codes to target directions, as shown in Figure 5. However, in opposition to [14, 24], style codes are not drawn from the normal

FIGURE 5: Example on Pascal VOC [50] using multioutput mode.

distribution $N(0, 1)$. We use two encoders, style and content, which causes translated instances to more or less possess the target domain's attributes. The goal of using this kind of loss

is to ensure that images keep the ability to recover in terms of both latent codes and semantic consistency after training:

$$\begin{aligned}
 L_{\text{recon}}^{A_{\text{ins}}} &= L_{s_{\text{re}}}^{A_{\text{ins}}} + L_{C_{\text{re}}}^{A_{\text{ins}}}, \\
 L_{s_{\text{re}}}^{A_{\text{ins}}} &= E_{A_{\text{ins}} \sim p(A_{\text{ins}}), C_{\text{ins}}^A \sim p(C_{\text{ins}}^A)} \left[\left\| G_{\text{ins}}^A(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A) - A_{\text{ins}} \right\|_1 \right], \\
 L_{C_{\text{re}}}^{A_{\text{ins}}} &= E_{A_{\text{ins}}, C_{\text{ins}}^A, S_{\text{ins}}^A \sim p(A_{\text{ins}}), p(C_{\text{ins}}^A), p(S_{\text{ins}}^A)} \left[\left\| E_{A_{\text{ins}}}^C(G_{\text{ins}}^B(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A)) - C_{\text{ins}}^A \right\|_1 \right] + E_{C_{\text{ins}}^A} \left[\left\| E_{A_{\text{ins}}}^S(G_{\text{ins}}^B(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A)) - S_{\text{ins}}^A \right\|_1 \right].
 \end{aligned} \tag{5}$$

Then, $L_{s_{\text{re}}}^{A_{\text{ins}}}$ and $L_{C_{\text{re}}}^{A_{\text{ins}}}$ can be represented as follows:

$$\begin{aligned}
 L_{s_{\text{re}}}^{A_{\text{ins}}} &= -\frac{1}{2M} \sum_{\sigma(S_{\text{ins}}^A)}^M \log \left(\left\| G_{\text{ins}}^A(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A) - A_{\text{ins}} \right\| \right), \\
 L_{C_{\text{re}}}^{A_{\text{ins}}} &= -\frac{1}{2M} \left(\sum_{\sigma(S_{\text{ins}}^A)}^M \log \left\| E_{A_{\text{ins}}}^C \left[G_{\text{ins}}^B(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A) \right] - C_{\text{ins}}^A \right\| + \sum_{\sigma(S_{\text{ins}}^A)}^M \log \left\| E_{A_{\text{ins}}}^S \left[G_{\text{ins}}^B(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A) \right] - S_{\text{ins}}^A \right\| \right).
 \end{aligned} \tag{6}$$

4.3. Adversarial Loss. The main use of GANs lies in their ability to match the target distribution as much as possible in

the adversarial process. We use the generators D_{ins}^A and D_{ins}^B to distinguish between translations and real instances:

$$\begin{aligned}
 L_{\text{adv}}^{A_{\text{ins}}} &= E_{A_{\text{ins}} \sim p(A_{\text{ins}}), C_{\text{ins}}^A \sim p(C_{\text{ins}}^A)} \left[\frac{1}{2} \log D_{\text{ins}}^A(G_{\text{ins}}^A(\sigma(S_{\text{ins}}^A), C_{\text{ins}}^A)) + \frac{1}{2} \log(1 - D_{\text{ins}}^A(A_{\text{ins}})) \right] \\
 &= -\frac{1}{2M} \left(\sum_{G_{\text{ins}}^A(Z_A)}^M \log D_{\text{ins}}^A(G_{\text{ins}}^A(Z_A)) + \sum_{D_{\text{ins}}^A(A_{\text{ins}})}^M \log(1 - D_{\text{ins}}^A(A_{\text{ins}})) \right).
 \end{aligned} \tag{7}$$

The L_{ins}^A term is the sum of $L_{\text{recon}}^{A_{\text{ins}}}$ and $L_{\text{adv}}^{A_{\text{ins}}}$; furthermore, L_{ins}^B is defined similarly:

$$\begin{aligned} L_{\text{ins}}^A &= L_{\text{recon}}^{A_{\text{ins}}} + L_{\text{adv}}^{A_{\text{ins}}} \\ &= \left[L_{s_{\text{re}}}^{A_{\text{ins}}} + L_{t_{\text{re}}}^{A_{\text{ins}}} \right] + L_{\text{adv}}^{A_{\text{ins}}} \end{aligned} \quad (8)$$

4.4. Background Loss. The background part is composed of N_{bgr}^A , N_{bgr}^B , C_{bgr}^A , and C_{bgr}^B . As mentioned in previous

sections, both the background and instance parts are independent of each other before their integration. Visual domains in this part still follow the reconstruction and translation stream. According to the components in the background part, we used L_{VAE} , L_{GAN} , and L_{SC} to represent VAEs, GANs, and semantic-consistency loss. Three weight parameters— λ_v , λ_G , and λ_{sc} —are applied to measure the impact of each component. For example, the L_{bgr}^A can be formulated as follows:

$$L_{\text{bgr}}^A = \lambda_v L_{\text{VAE}}^A(E_{\text{bgr}}^A, G_{\text{bgr}}^A) + \lambda_G L_{\text{GAN}}^A(C_{\text{bgr}}^A, G_{\text{bgr}}^A, D_{\text{bgr}}^A) + \lambda_{\text{SC}} L_{\text{SC}}^A(C_{\text{bgr}}^A, G_{\text{bgr}}^A, E_{\text{bgr}}^B, G_{\text{bgr}}^B). \quad (9)$$

The VAE architecture aims to learn a latent model by approximating the marginal log-likelihood of training data (ELBO algorithm [19] (the lower bound of latent codes)). Its objective function is

$$\begin{aligned} L_{\text{VAE}}^A(E_{\text{bgr}}^A, G_{\text{bgr}}^A) &= \lambda_1 KL(q_{\text{bgr}}^A(C_{\text{bgr}}^A | A_{\text{bgr}}) | p_C(C_{\text{bgr}})) - \\ &\quad \lambda_2 E_{A_{\text{bgr}} \sim p(A_{\text{bgr}}), C_{\text{bgr}}^A \sim p(C_{\text{bgr}}^A)} [\log p_{G_{\text{bgr}}^A}(A_{\text{bgr}} | C_{\text{bgr}}^A)]. \end{aligned} \quad (10)$$

In this function, the weight parameters λ_1 and λ_2 control the impact of the objective function and KL (meaning *kullback leibler divergence*) measures how well $q_{\text{bgr}}^A(C_{\text{bgr}}^A | A_{\text{bgr}})$ matches prior distribution $p_C(C_{\text{bgr}})$, which denotes the distribution of common content space C_{bgr}^A . To better perform sampling from spaces, we model $\{q_{\text{bgr}}^A(C_{\text{bgr}}^A | A_{\text{bgr}}), p_C(C_{\text{bgr}})\}$ with normal distribution and $p_{G_{\text{bgr}}^A}$ with Laplacian distributions, respectively:

$$\begin{aligned} L_{\text{GAN}}^A(C_{\text{bgr}}^A, G_{\text{bgr}}^A, D_{\text{bgr}}^A) &= E[\log D_{\text{bgr}}^A(A_{\text{bgr}})] + E[\log(1 - D_{\text{bgr}}^A(G_{\text{bgr}}^A(C_{\text{bgr}}^A)))] \\ &= -\frac{1}{2M} \left(\sum_{A_{\text{bgr}}}^M \log D_{\text{bgr}}^A(A_{\text{bgr}}) + \sum_{D_{\text{bgr}}^A(G_{\text{bgr}}^A(C_{\text{bgr}}^A))}^M \log(1 - D_{\text{bgr}}^A(G_{\text{bgr}}^A(C_{\text{bgr}}^A))) \right). \end{aligned} \quad (11)$$

The GAN's objective function aims to translate and reconstruct images in the adversarial process:

$$\begin{aligned} L_{\text{SC}}^A(E_{\text{bgr}}^A, G_{\text{bgr}}^A, E_{\text{bgr}}^B, G_{\text{bgr}}^B) &= E[\|E_{A_{\text{bgr}}}^S(G_{\text{bgr}}^A(C_{\text{bgr}}^A)) - C_{\text{bgr}}^A\|_1] + E[\|E_{\text{bgr}}^B(G_{\text{bgr}}^B(C_{\text{bgr}}^A)) - C_{\text{bgr}}^B\|_1] \\ &= -\frac{1}{2M} \left(\sum_{G_{\text{bgr}}^A(C_{\text{bgr}}^A)}^M \log \|E_{A_{\text{bgr}}}^S(G_{\text{bgr}}^A(C_{\text{bgr}}^A)) - C_{\text{bgr}}^A\|_1 + \sum_{G_{\text{bgr}}^B(C_{\text{bgr}}^A)}^M \log \|E_{\text{bgr}}^B(G_{\text{bgr}}^B(C_{\text{bgr}}^A)) - C_{\text{bgr}}^B\|_1 \right). \end{aligned} \quad (12)$$

The semantic-consistency objective function makes sure that images can be mapped back to the original latent space, but possesses characteristics from the target domains. The significant modification to this function is using the l_1 norm to directly compare semantic differences instead of using KL terms to measure the distance in latent spaces (such as UNIT [11]).

5. Training Techniques

The proposed framework tends to decompose an image into its background and instance constituents and then feeds these parts into independent translation networks. Finally, using saved labels, we integrate both translated parts. Such a

solution raises another question: how can integrations be made to look real? If we simply crop, translate, respectively, and integrate, the integration looks strange and uncomfortable, since the background and instance parts are translated in totally different directions (as shown in Figure 6).

To maintain the realism of the images after integration and make them more comfortable to look at, a technique called “label smoothing” [25] is used to further improve the proposed framework.

5.1. Methods. We know that GANs work effectively when discriminators can estimate the ratio of input and model data, even at any point, represented by x :

$$\frac{P_{\text{model}}(x)}{P_{\text{img}}(x)}. \quad (13)$$

Under the previous conditions, we would train discriminators to estimate:

$$\begin{aligned} D_{\text{bgr}}^A &\longrightarrow \frac{P_{G_{\text{bgr}}^A}(B_{\text{bgr}})}{P(A_{\text{bgr}})}, \\ D_{\text{bgr}}^B &\longrightarrow \frac{P_{G_{\text{bgr}}^B}(A_{\text{bgr}})}{P(B_{\text{bgr}})}. \end{aligned} \quad (14)$$

(Instance) reconstruction mode:

$$\begin{aligned} D_{\text{ins}}^A &\longrightarrow \frac{P_{G_{\text{ins}}^A}(A_{\text{ins}})}{P(A_{\text{ins}})}, \\ D_{\text{ins}}^B &\longrightarrow \frac{P_{G_{\text{ins}}^B}(A_{\text{ins}})}{P(B_{\text{ins}})}. \end{aligned} \quad (15)$$

(Instance) multioutput mode:

$$\begin{aligned} D_{\text{ins}}^A &\longrightarrow \frac{P_{G_{\text{ins}}^A}(B_{\text{ins}})}{P_{\text{ins}}(A_{\text{ins}})}, \\ D_{\text{ins}}^B &\longrightarrow \frac{P_{G_{\text{ins}}^B}(A_{\text{ins}})}{P_{\text{ins}}(B_{\text{ins}})}. \end{aligned} \quad (16)$$

Let $\text{model}_{\text{bgr}}$ and $\text{model}_{\text{ins}}$ represent the background and instance networks. Then, outputs produced by the models will be integrated (represented by I_A and I_B) as follows:

$$\begin{aligned} I_A &: \{\text{model}_{\text{bgr}}(A_{\text{bgr}}), \text{model}_{\text{bgr}}(A_{\text{ins}})\}, \\ I_B &: \{\text{model}_{\text{bgr}}(B_{\text{bgr}}), \text{model}_{\text{bgr}}(B_{\text{ins}})\}. \end{aligned} \quad (17)$$

We add another pair of discriminators $\{D_A$ and $D_B\}$ to distinguish B , I_A and A , I_B after integration. Since integrations are not exactly the same as the originals, A and B , we add parameter α to smooth the training data. The parameters are given as follows:

$$\begin{aligned} \alpha^A &= \frac{\text{Area}(A_{\text{ins}})}{\text{Area}(A)}, \\ \alpha^B &= \frac{\text{Area}(B_{\text{ins}})}{\text{Area}(B)}. \end{aligned} \quad (18)$$

Then, to add α into the distinguishing process, we use $\{D_A$, and $D_B\}$ to estimate the ratio of

$$\begin{aligned} D_A &\longrightarrow \frac{P_{G_{\text{bgr}}^A}(B_{\text{bgr}})}{(1 - \alpha^A)p(A) + \alpha^A P_{G_{\text{bgr}}^A}(B_{\text{bgr}})}, \\ D_B &\longrightarrow \frac{P_{G_{\text{bgr}}^B}(A_{\text{bgr}})}{(1 - \alpha^B)p(B) + \alpha^B P_{G_{\text{bgr}}^B}(A_{\text{bgr}})}. \end{aligned} \quad (19)$$

Although smoothing parameters may encourage less confident outputs (compared to the original images) and influence the style of outputs, this adjustment makes integrations much more realistic, so the image does not obviously resemble the combination of two completely different images. The detailed discrimination method is illustrated in Figure 7. A comparison of the results from before and after implementing the smoothing technique will be presented in the results section.

6. Experiments and Discussion

6.1. Datasets. The framework is tested on a pair of benchmarks: cityscape [26] $\leftrightarrow$ GTA [27] (a bidirectional translation). Before feeding training images into proposed frameworks, we crop instances from the original images. Limited by memory resources, all instances are resized to 64×64 , while background parts are resized to 256×256 .

6.2. Car Translation (Cityscape $\leftrightarrow$ GTA). Having been trained in YOLO [28], we achieved all cars’ location values and saved this information in bound boxes. For clearer visuals, we cropped instances that were size 300×300 (these instances later resized to 64×64).

As shown in Figure 8, we clearly see that during reconstruction, the instance domains are reconstructed well, being almost identical to their input. In Figure 9, we observe that the translated background domains immediately have the style and attributes of the target domains. While in Figure 10, style codes sampled from the other space allowed the instance part to produce four different translations, which definitely proves that by matching fixed content with different style information, we can successfully produce diverse translations.

When looking at the final integrations consisting of both the background and instance domains, we clearly observe that although both parts produce high-quality translations, the unsmoothed integrations (the second and forth row in Figure 11) look mismatched, such as a car of a totally different style has been attached to the background, as illustrated in Figure 11. When we apply smoothed labels at the end of training, both smoothed translations fit well

FIGURE 6: Translation without smoothing. The input image is separated into Bgr and Ins areas. Both areas are then fed into the background network $model_{bgr}$ and instance network $model_{ins}$. After translations, generations from both methods are integrated to become a full image.

FIGURE 7: Smoothed discriminator. To separate instance A_{ins} and background A_{bgr} from image A after processed by $model_{bgr}$ and $model_{ins}$, they are then integrated to be a full image I_A by labels. We added a new discriminator D_A to distinguish I_A and the target image B .

FIGURE 8: Continued.

FIGURE 8: Instance reconstruction. In reconstruction mode, for instance inputs (the first and third column) from Cityscapes and GTA, we fed them into instance networks and then reconstructed them (as demonstrated in the second and fourth row). (a) Cityscapes. (b) GTA.

FIGURE 9: Background translations. This figure shows translation generations (Cityscapes $\leftrightarrow$ GTA) from the background network. Inputs from both datasets are shown in the first and third column, and the outputs are shown in the second and fourth row. (a) Cityscapes. (b) GTA.

with each other and look like a full image. Although several results still preserve boundaries, the boundaries are extremely close in proximity to the translated content and color (see the 3rd, 4th, and 5th maps in the city integrations presented in Figure 12). Based on comparisons with Figure 11, we conclude that applying smoothing labels is a feasible and useful technique for integrating translations.

By contrast, we also experimented with MUNIT [9] and DRIT [8]. As demonstrated in Figures 13 and 14, although both models achieved good translation

performances on Cityscape $\leftrightarrow$ GTA, they still failed to manipulate instance-level translations, which means that we are unable to segregate an instance (such as objects and areas) from others, meaning we can only treat all elements as a whole.

When it comes to our generations, with prerecorded location information and smoothing techniques, we make the domain/instance-level translation possible. Instances of “car” were selected as objects and remaining domains were translated to cityscape/GTA style, whereby cars can be maintained or translated into another destination.

FIGURE 10: Instance translation using diverse modes. When in the multioutput mode, for single instance outputs (the first row), we are able to produce four translation outputs (from the second to the fifth column).

FIGURE 11: Integrated Translations (without smoothing). For inputs (the first and third row) from both datasets (Cityscapes and GTA), we selected a certain car domain as the instance and then fed the remaining domain and instance into networks. The application of the smoothing technique is shown in the final integration in the second and fourth rows. (a) City inputs. (b) Integration. (c) GTA inputs. (d) Integration.

FIGURE 12: Integrated translations (smoothed). (a) City inputs. (b) Integration. (c) GTA inputs. (d) Integration.

FIGURE 13: Continued.

FIGURE 13: Performance of MUNIT (Cityscape $\leftrightarrow$ GTA). (a) City inputs. (b) To GTA. (c) GTA inputs. (d) To city.

FIGURE 14: Performance of DRIT (cityscape $\leftrightarrow$ GTA). (a) City inputs. (b) To GTA. (c) GTA inputs. (d) To city.

7. Qualitative Evaluation

7.1. Questionnaire. A widely used method to evaluate the realism of generated images is to distribute a questionnaire. We selected two translation methods, DRIT [8] and MUNIT [9], as baselines and randomly sampled 100 images as inputs to compare generations from methods. When conducting questionnaires, each participant would be given five images, the real one that has the target style, generations (from source inputs) from DRIT, MUNIT, and ours (including the smoothed one). Compared to the real one, participants have to select the most realistic generation from four groups.

We gathered 20 students as users from the department of Computer Science. Each person is expected to answer the

question “Compared to the real one, among four outputs, which image do you think is the most realistic?” During the questionnaire, we counted the number of people who made the selection and summarized them in Figure 15.

7.2. Metrics

7.2.1. Learned Perceptual Image Patch Similarity Distance (LPIPS Distance). Since our research focuses on instance-aware and flexible translations, we used LPIPS Distance [9, 11] to measure the diversity of generations, which has been proven to correlate well with human preference. Similar to MUNIT [9], we use trained AlexNet [25] as a

FIGURE 15: Realism comparison with other methods.

TABLE 1: Comparison of the state-of-art I2I translation methods. Our framework enables a domain-aware diverse translation on unpaired inputs.

Method	Unpaired	Diversity	Domain-aware
BIGAN [12]	—	—	—
MUNIT [9]	✓	✓	—
DRIT [8]	✓	✓	—
OURS	✓	✓	✓

feature extractor and selected 150 target images to be one input pair and sampled 15 output pairs per input pair.

7.2.2. Conditional Inception Score (CIS). Put in [9], this kind of modified metric can better measure the diversity of outputs. We used fine-tuned Inception-v3 [29] on our datasets as the classifier and calculated the CIS score based on 200 input pairs and 400 translations per pair.

We conclude feature comparisons with other methods in Table 1. It can be clearly seen that, unlike existing works, our method enables flexible translations with diverse outputs under unpaired training settings.

Figure 15 and Table 2 show the results for the realism and diversity comparison with other methods. For realism, our method does not outperform any previous research, but we noted that the realism of outputs improved greatly since the application of the smoothing technique, and there is not much difference in realism between MUNIT [11] and the smoothed method.

Furthermore, the diversity comparisons show that the proposed framework (with smoothing) obtains the second place in terms of LPIPS and CIS. DRIT [8] achieves the best diversity, but it is not domain-aware. Overall, our method achieves great diversity and produces realistic results.

TABLE 2: Diversity. In this table, a generations’ diversity is measured using LPIPS and CIS metrics. For the LPIPS metric, lower is better, while for CIS, the higher the number, the better. All experiments are conducted using the cityscape $\leftrightarrow$ GTA (car translations), and all numbers here are the average values of three experiments.

Method	LPIPS	CIS
MUNIT	0.493	0.981
DRIT	0.295	1.205
DAGAN (ins $A \leftarrow \rightarrow B$)	0.469	1.087
DAGAN (integrated $A \leftarrow \rightarrow B$)	0.520	1.005
DAGAN (smoothed ins)	0.402	1.098
DAGAN (smoothed integrated)	0.447	1.084

8. Conclusions and Future Work

In this work, we presented a flexible framework for domain-aware image-to-image translations. With smoothed training, we achieved great translations and better integration in terms of diversity and realism. Current research focuses on full-image translation but overlooks domain level translations and the models’ flexibility.

With the proposed framework, we enabled networks to learn instance-level mapping. Compared with generations from other baselines, our method did not perform poorly and allowed users to choose areas and styles they want to change.

At the same time, when observing the midterm results during training, we noticed that the pretreatment step that replaces the cropped instance area with the mean pixel value has a great influence on the final translation. Although this operation does not totally change the distribution of images and perfectly fits background areas, it still presents a great challenge for later translation works.

From our experiment results we clearly see that the midterm outputs from the given or target domains also learned the replaced area’s distribution from the other’s domain, and this badly influenced later generations, especially when performing integration operations. Under such conditions, we need a smoother replacement or restoration when cropping instances.

On the other hand, our framework achieved a 1: n translation on the background and instance domains. However, as previously mentioned, images are integrations of content and style information, and they are separable by deep-learning techniques. We can also perform multioutput translations on background domains as we did on instances. We can then achieve $m:n$ generations that would be more applicable for industrial business.

Data Availability

The codes used to support the findings of this study were supplied by the Korean government (MSIT) under license, and therefore cannot be made freely available. Requests for access to this data should be made to Xu Yin, Department of Computer Engineering, Inha University, Incheon, 082, South Korea.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korean government (MSIT) (no. 2017-0-018715, Development of AR-based Surgery Toolkit and Applications). It was also supported by Research Program to Solve Social Issues of the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (no. NRF-2019R1A2C1090713)

References

- [1] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125–1134, Honolulu, HI, USA, July 2017.
- [2] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, Venice, Italy, October 2017.
- [3] J. Park, J. Y. Lee, D. Yoo, and I. S. Kweon, "Distort-and-recover: color enhancement using deep reinforcement learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5928–5936, Salt Lake City, UT, USA, June 2018.
- [4] C. Fabbri, M. J. Islam, and J. Satter, "Enhancing underwater imagery using generative adversarial networks," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7159–7165, IEEE, Brisbane, Australia, May 2018.
- [5] S.-W. Huang, C.-T. Lin, S.-P. Chen, Y.-Y. Wu, P.-H. Hsu, and S.-H. Lai, "AugGAN: cross domain adaptation with GAN-based data augmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 718–731, Munich, Germany, September 2018.
- [6] J. M. Wolterink, T. Leiner, M. A. Viergever, and I. Išgum, "Generative adversarial networks for noise reduction in low-dose CT," *IEEE Transactions on Medical Imaging*, vol. 36, no. 12, pp. 2536–2545, 2017.
- [7] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to see in the dark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3291–3300, Salt Lake City, UT, USA, June 2018.
- [8] H. Y. Lee, H. Y. Tseng, J. B. Huang, M. Singh, and M. H. Yang, "Diverse image-to-image translation via disentangled representations," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 35–51, Munich, Germany, September 2018.
- [9] X. Huang, M. Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 172–189, Munich, Germany, September 2018.
- [10] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797, Salt Lake City, UT, USA, June 2018.
- [11] M. Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 700–708, Long Beach, CA, USA, December 2017.
- [12] J. Y. Zhu, R. Zhang, D. Pathak et al., "Toward multimodal image-to-image translation," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 465–476, Long Beach, CA, USA, December 2017.
- [13] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," 2017, <http://arxiv.org/abs/1708.04896>.
- [14] I. Goodfellow, "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, Montreal, Canada, December 2014.
- [15] I. Goodfellow, "NIPS 2016 tutorial: generative adversarial networks," 2016, <http://arxiv.org/abs/1701.00160>.
- [16] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein GANs," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5767–5777, Long Beach, CA, USA, December 2017.
- [17] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, <http://arxiv.org/abs/1701.07875v3>.
- [18] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, <http://arxiv.org/abs/1312.6114>.
- [19] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: interpretable representation learning by information maximizing generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2172–2180, Barcelona, Spain, December 2016.
- [20] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, and S. Mohamed, "Variational approaches for auto-encoding generative adversarial networks," 2017, <http://arxiv.org/abs/1706.04987>.
- [21] A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. Courville, "Augmented CycleGAN: learning many-to-many mappings from unpaired data," 2018, <http://arxiv.org/abs/1802.10151>.
- [22] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, <http://arxiv.org/abs/1511.06434>.
- [23] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, October 2017.
- [24] A. Krizhevsky, S. Ilya, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems*, Lake Tahoe, CA, USA, December 2012.
- [25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proceedings of the Advances in Neural Information Processing Systems*, Barcelona, Spain, December 2016.
- [26] M. Cordts, M. Omran, S. Ramos et al., "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.
- [27] R. Richter, "Playing for data: ground truth from computer games," in *Proceedings of the European Conference on Computer Vision*, Springer, Amsterdam, The Netherlands, October 2016.
- [28] J. Redmon and F. Ali, "Yolov3: an incremental improvement," 2018, <http://arxiv.org/abs/1804.02767>.
- [29] S. Christian, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.