



# Style-Consistent Image Translation: A Novel Data Augmentation Paradigm to Improve Plant Disease Recognition

Mingle Xu<sup>1,2</sup>, Sook Yoon<sup>3\*</sup>, Alvaro Fuentes<sup>1,2</sup>, Jucheng Yang<sup>4</sup> and Dong Sun Park<sup>1,2\*</sup>

<sup>1</sup> Department of Electronics Engineering, Jeonbuk National University, Jeonbuk, South Korea, <sup>2</sup> Core Research Institute of Intelligent Robots, Jeonbuk National University, Jeonbuk, South Korea, <sup>3</sup> Department of Computer Engineering, Mokpo National University, Jeonnam, South Korea, <sup>4</sup> College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin, China

## OPEN ACCESS

### Edited by:

Yuzhen Lu,  
Mississippi State University,  
United States

### Reviewed by:

Borja Espejo-Garca,  
Agricultural University of Athens,  
Greece  
Dong Chen,  
Michigan State University,  
United States  
Junfeng Gao,  
University of Lincoln, United Kingdom  
Ebenezer Olaniyi,  
Mississippi State University,  
United States

### \*Correspondence:

Sook Yoon  
syoon@mokpo.ac.kr  
Dong Sun Park  
dspark@jbnu.ac.kr

### Specialty section:

This article was submitted to  
Sustainable and Intelligent  
Phytoprotection,  
a section of the journal  
Frontiers in Plant Science

**Received:** 09 September 2021

**Accepted:** 23 December 2021

**Published:** 07 February 2022

### Citation:

Xu M, Yoon S, Fuentes A, Yang J and  
Park DS (2022) Style-Consistent  
Image Translation: A Novel Data  
Augmentation Paradigm to Improve  
Plant Disease Recognition.  
*Front. Plant Sci.* 12:773142.  
doi: 10.3389/fpls.2021.773142

Deep learning shows its advantages and potentials in plant disease recognition and has witnessed a profound development in recent years. To obtain a competing performance with a deep learning algorithm, enough amount of annotated data is requested but in the natural world, scarce or imbalanced data are common, and annotated data is expensive or hard to collect. Data augmentation, aiming to create variations for training data, has shown its power for this issue. But there are still two challenges: creating more desirable variations for scarce and imbalanced data, and designing a data augmentation to ease object detection and instance segmentation. First, current algorithms made variations only inside one specific class, but more desirable variations can further promote performance. To address this issue, we propose a novel data augmentation paradigm that can adapt variations from one class to another. In the novel paradigm, an image in the source domain is translated into the target domain, while the variations unrelated to the domain are maintained. For example, an image with a healthy tomato leaf is translated into a powdery mildew image but the variations of the healthy leaf are maintained and transferred into the powdery mildew class, such as types of tomato leaf, sizes, and viewpoints. Second, current data augmentation is suitable to promote the image classification model but may not be appropriate to alleviate object detection and instance segmentation model, mainly because the necessary annotations can not be obtained. In this study, we leverage a prior mask as input to tell the area we are interested in and reuse the original annotations. In this way, our proposed algorithm can be utilized to do the three tasks simultaneously. Further, We collect 1,258 images of tomato leaves with 1,429 instance segmentation annotations as there is more than one instance in one single image, including five diseases and healthy leaves. Extensive experimental results on the collected images validate that our new data augmentation algorithm makes useful variations and contributes to improving performance for diverse deep learning-based methods.

**Keywords:** tomato disease recognition, data augmentation, image translation, image classification, instance segmentation, image style

## 1. INTRODUCTION

Food security has been raised to a high level in many countries partly because food distribution is not compatible with the distribution of the population in the world, and the number of laborers related is becoming less. While enough amount of food is required to feed our humans, many factors may harm plant growth and hence threaten the food supply. Controlling disease is one of the key challenges to keep plants healthy toward obtaining the expected yield. Artificial intelligence has recently witnessed a booming development with a decent disease recognition performance as intelligent machines deployed in farms can reduce workload. Deep learning, a core technique of artificial intelligence, has been successfully adopted to recognize diseases or abnormalities, such as tomato (Fuentes et al., 2017; Liu and Wang, 2020; Wang et al., 2021), banana (Lin et al., 2021), potato (Gao et al., 2021), corn and apple (Zhong and Zhao, 2020), and many other plants (Gao et al., 2020; Liu and Wang, 2021). Recent studies (Martineau et al., 2017; Liu and Wang, 2021; Saranya et al., 2021) show the advantages and potentialities of deep learning methods compared to other methods, such as handcraft feature, in recognizing plant diseases and related tasks.

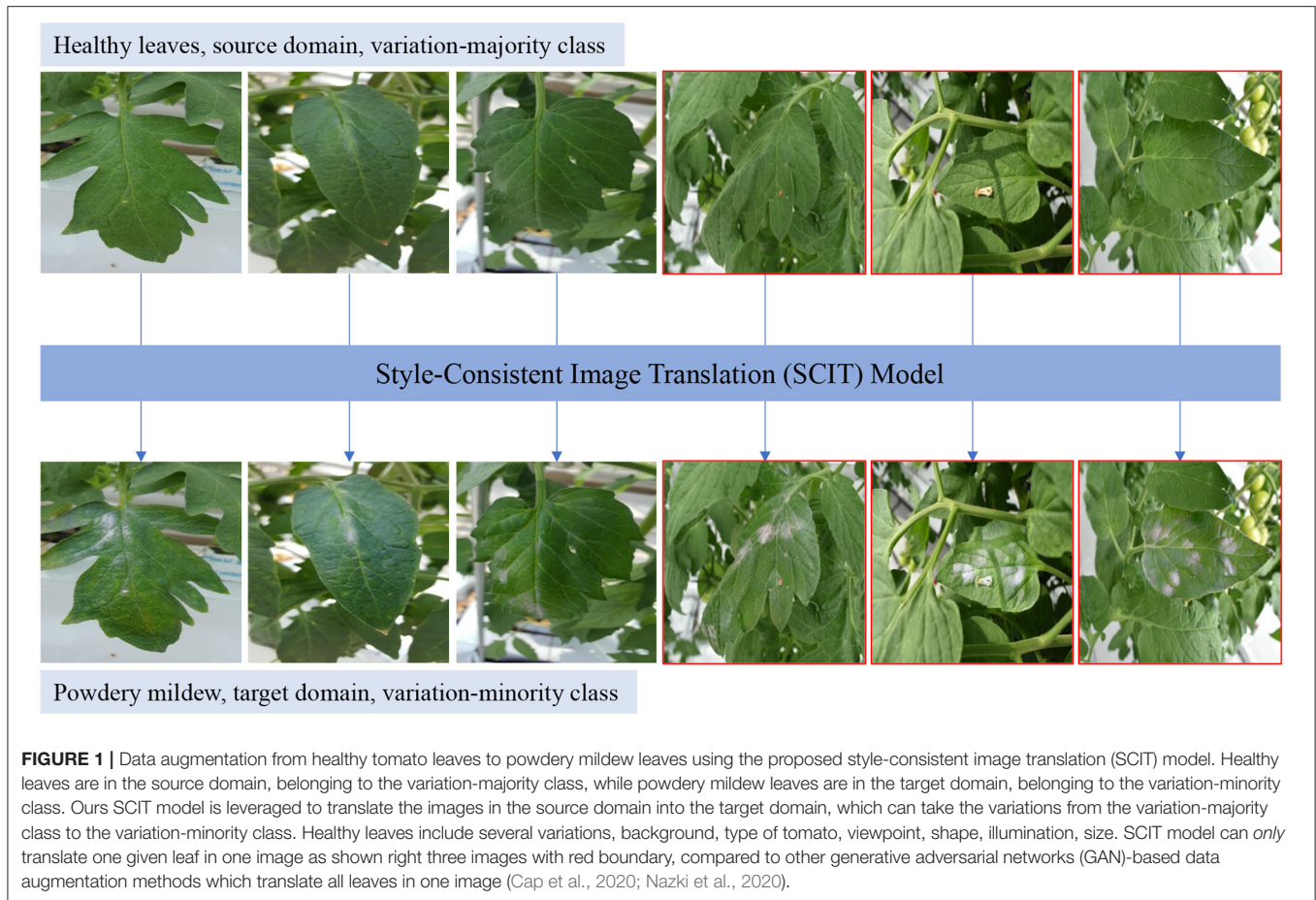
To obtain a competing performance with a deep learning algorithm, enough amount of annotated data is requested but in the natural world, scarce or imbalanced data are common, and annotated data is expensive or hard to collect. For example, some diseases rarely appear or even never appear on one farm but the healthy plant is more common, which can not lead to a convincing disease recognition performance. To address this challenge, data augmentation is one of the most potential solutions and has been utilized in the agricultural field (Zhu et al., 2018; Nazki et al., 2020; Abbas et al., 2021). Data augmentation aims to generate more data with collected *training* data to improve the deep learning model's performance in *testing* data. Previous studies (Pawara et al., 2017; Douarre et al., 2019; Pinto Sampaio Gomes and Zheng, 2020; Liu and Wang, 2021; Saranya et al., 2021) have validated that data augmentation plays a significant role to improve the performance of deep learning in the agricultural area. In this study, we are interested in image-based recognition for tomato diseases, and hence data augmentation defaults with image-based.

Traditional data augmentation methods generate new augmented data within one specific class, *within-class data augmentation*, where the appearance of the image can be changed but the corresponding class remains, such as rotating or translating an image (Hu et al., 2020; Gorad and Kotrappa, 2021). In contrast, *cross-class data augmentation* methods can translate one image from the class to another class *via image translation* that aims to translate images from a source domain to a target domain, by which the variations are desired to be borrowed from one class to another class. For example, a healthy tomato image, source domain, can be translated into a powdery mildew image, target domain. In general, healthy tomato leaf images are easy to collect with large variations, such as background, viewpoint, size of the leaf, and type of tomato, as shown in **Figure 1**. As there is a high variation for the healthy tomato images, we refer the healthy to a *variation-majority*

*class*. On the other hand, the class, hard to obtain images or enough variation, is referred to as a *variation-minority class*, such as some disease tomato leaves. To achieve the cross-class data augmentation, CycleGAN (Zhu et al., 2017), one of the state-of-the-art methods to do image translation, is plausible to be utilized. Based on CycleGAN, Nazki et al. (2020) proposed an activation reconstruction loss to improve the quality of the generated image. Except for low image quality, CycleGAN tends to change the undesired content such as background, and an attention module was proposed in LeafGAN (Cap et al., 2020) to detect the area that we are interested in. Although they achieved better results than the original CycleGAN, the following two new challenges are addressed in this study.

*First, how can we adapt the majority of variations from the source domain to the target domain?* Image translation is expected to keep the variations but there are no guarantees to keep the variations in current algorithms. To ease this issue, we propose a new paradigm to combine style consistency and image translation to maintain the variations during the image translation process, and hence, we call our algorithm *style-consistent image translation* (SCIT). Specifically, we follow the hypothesis that images can be factorized into two parts, *label-related* and *style-related* (Gonzalez-Garcia et al., 2018; Lee et al., 2018). While the label-related are the characteristics or patterns of specific classes such as healthy and powdery mildew, the style-related is independent of the labels, such as illumination, viewpoints, and background. In the translation process, we aim to translate the label-related but keep the style-related, which contributes to adapting the styles from the source domain to the target domain. In **Figure 1**, we can see that the selected healthy tomato leaves are translated into powdery mildew leaves by our method, but the styles in healthy tomato leaves are maintained, and the variations of powdery mildew domain are augmented from the healthy domain.

*Second, how can we use image translation as data augmentation to ease object detection and instance segmentation?* Traditional data augmentation and current image translation-based algorithms can be leveraged to improve the image classification but may not be appropriate to alleviate object detection and instance segmentation which are closer to our practical applications. The first reason is that object detection and instance segmentation require more annotations than image classification but current algorithms can not make those necessary annotations. Generally, we only need class information of images to train the image classification model, but the exact locations of each class are necessary to train the object detection algorithm, and both location and instance identity are required to train the instance segmentation model, **Figure 9** giving examples about the two tasks. Furthermore, classification is an image-level task but object detection and instance segmentation are at the instance level. Elusively, the input image undergoes a preprocessing to be one leaf or even part of one leaf to do image classification by which classification model is easier to be trained (Nazki et al., 2020). In contrast, preprocessing is not necessary for object detection and instance segmentation. In fact, the leaves are normally in different scales, as shown in the red boundary images in **Figure 1**. Therefore, we are



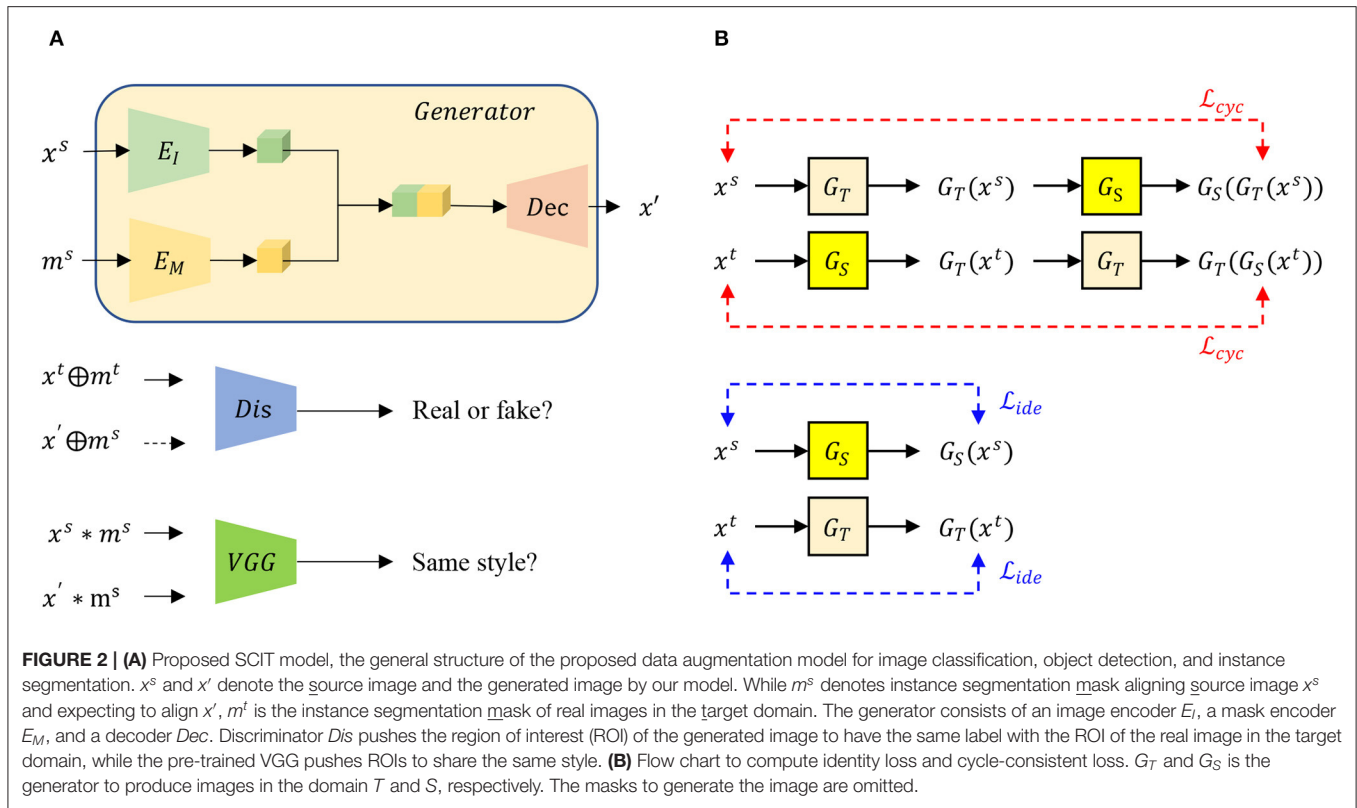
requested to translate leaf instances separately in diverse scales instead of translating all leaves as other algorithms have been doing (Cap et al., 2020; Nazki et al., 2020). To address the two issues, we employ a mask as prior knowledge in order to split an image into the region of interest (ROI) and background (BG). We aim to translate its ROI part but reuse its BG part, by which the original annotations can be reused for the produced images. Further, we design a new framework based on CycleGAN (Zhu et al., 2017). First of all, a mask encoder is designed to be incorporated with the image encoder, as shown in **Figure 2**, by which our generator knows where is interested. Although a similar idea appears in RBGAN (Xu et al., 2021), we aim to translate part of the image yet keep the other part and maintain the style during the image translation but RBGAN aims to perform instance-level image translation with a decent translated instance boundary. Besides, our discriminator absorbs both real or fake images and corresponding masks and is pushed to know where is the translated area and whether the area is real or fake. Therefore, with the new generator and discriminator along with the input mask, our algorithm can translate given leaves in diverse scales with the reused annotations, which contributes to ease object detection and instance segmentation as a data augmentation method.

To summarize, our contributions are as follows:

1. We propose a data augmentation paradigm, SCIT, which can increase the data variations for the variation-minority class by leveraging the images from the variation-majority class.
2. We propose a framework to perform image translation in diverse scales with necessary annotations as output to ease object detection and instance segmentation, which is out of current data augmentation methods.
3. Taking tomato as an example, we perform extensive experiments on three tasks, image classification, object detection, and instance segmentation. The experimental results suggest that our proposed algorithm improves the performances for diverse deep learning-based methods and outperforms the state-of-the-art data augmentation methods.

The remainder of this study is organized as follows. Related studies and our basic idea are introduced in the preliminary section. The proposed method to do data augmentation is instantiated in Section 3, including the framework and loss function. In the experiments section, we show the details about our dataset, implementation to train and test our model, ablation study to understand our algorithm, comparison to other methods in three tasks. Finally, we conclude our studies and future study in the last section.





## 2. PRELIMINARY

Data augmentation based on the image can be categorized into two main parts, basic image manipulations and deep learning-based algorithms. In this section, we try to highlight the difference between other methods and our method to achieve data augmentation.

**Image manipulations.** Image manipulations make use of image processing methods, such as pixel-wise conversion and geometrical transformations. Formally, let  $x^s$  and  $x^a$  denote the source image and the augmented image. Similarly,  $y^s$  and  $y^a$  are corresponding labels. The formulation of basic image manipulation-based data augmentation refers to Equation 1.

$$\begin{cases} x^a = f_m(x^s), \\ y^a = y^s, \end{cases} \quad (1)$$

where  $f_m$  is one of the basic image manipulation functions. The equation suggests that the label of the input image is not changed when the input image undergoes manipulations. The retainment degree of the label is called safety of data augmentation because manipulations can not always keep the label (Shorten and Khoshgoftaar, 2019). For example, after adding too much random noise, some images could not be recognized as before. With these image manipulations, prior works achieved better performance in the missions with small datasets (Hu et al., 2020; Gorad and Kotrappa, 2021). Besides, these basic image manipulation can be adapted from one image to more than one image (Dwibedi et al., 2017). Kuznichov et al.

(2019) copied the leaves from different images to form a new augmented image to promote leaf segmentation and counting. Similarly, Gao et al. (2020) produced a synthetic image by combining specific objects from different images, in which two different classes can appear in a single image. Their experiments validated that the performance can be also improved by their methods. In this study, we aim to do data augmentation from another viewpoint by using one class to augment another class, hoping that more variations can be produced.

**Deep learning-based algorithms.** Different from image manipulations, deep learning-based algorithms employ deep neural networks to generate new images. According to the condition to generate new images, deep learning-based algorithms can be split into label-condition and image-condition. Label-condition algorithms generate images from given labels by using generative adversarial networks (GANs) (Valerio Giuffrida et al., 2017; Pandian et al., 2019; Bi and Hu, 2020; Abbas et al., 2021). In contrast, image-condition algorithms produce images from given images. Style transfer is one of the possible methods (Li et al., 2017; Huang et al., 2021; Shen et al., 2021). Mathematically, it can be formalized as Equation 2.

$$\begin{cases} x^a = f_{st}(x^s), \\ y^a = y^s, \end{cases} \quad (2)$$

where  $f_{st}$  is the style transfer function. Because of the function, style transfer algorithms can introduce more variations into the dataset. For instance, images captured in sunny style can be transferred into night style (Shen et al., 2021), which is

beyond the basic image manipulations. However, the labels of the transferred images are considered as same as the source, which is the same as the basic image manipulations. Besides, the style transferring-based algorithms try to maintain the content of the image. Take the tomato leaves as an example, it aims to maintain the size, shape of leaves, viewpoints and, hence, making more variations about them is still challenging.

To deal with the challenge mentioned above, we propose a novel paradigm to achieve data augmentation, following the disentangled idea that an image can be factorized into two factors: style-related and label-related (Gonzalez-Garcia et al., 2018; Lee et al., 2018). By using image translation, label-related factors of an image in the source domain can be translated into images in the target domain. Simultaneously, the style-related factors of the source image are desired to be kept in the translated image. In the style-consistent image translation, the variations of the style in the source domain are borrowed into the target domain. Mathematically, this kind of data augmentation algorithm can be formulated as Equation 3.

$$\begin{cases} x^a = f_{it}(x^s), \\ y^a \neq y^s, \\ g(x^a) = g(x^s), \end{cases} \quad (3)$$

where  $f_{it}$  symbolizes image translation function.  $y^a$  and  $y^s$  denote the label corresponding to image  $x^s$  and  $x^a$ , respectively. The function  $g$  symbolizes the style extracting function.

We argue that this kind of data augmentation eases practical applications. When it is hard to collect the data or the collected data suffer from a lack of variations in one domain but easier to collect data in another domain, we can use this data augmentation to leverage the variations in the easier domain to promote the variations in the harder domain. For example, images of healthy tomato leaves can be easily collected from farms but images with specific diseases of abnormalities like powdery mildew could not be collected easily, mainly because farmers must do necessary measures to prevent before their appearance or make a fast remedy after their appearance to reduce financial loss. We can augment data for powdery mildew effectively by using image translation from healthy domain to powdery mildew domain. Besides, we emphasize that the proposed data augmentation method can be employed with any kind of image translation model  $f_{it}$  and domain-invariant function  $g$ .

Moreover, we also notice that data augmentation for image classification attracts much more attention than for object detection or instance segmentation. A label is globally assigned to a whole image for the image classification task but a label is locally assigned to a region of an image for object detection or instance segmentation task. Therefore, one of the challenges for object detection and instance segmentation is mainly that dealing with a bounding box or instance mask for a local region is required. To address this issue, we further spatially split one image into two parts, region of interest (ROI)  $x_{roi}$  and background  $x_{bg}$ . Formally, input image  $x$  is split into  $x_{roi}$  and  $x_{bg}$  according to the binary instance segmentation mask from source domain  $m^s$ . Then our proposed generative adversarial network takes the  $x^s$  and  $m^s$  as input and aims to translate the  $x_{roi}$  into the target domain.

Simultaneously, we reuse the background of the source image since the translation model is not interested in the background. In this way, the augmented image  $x^a$  shares the same bounding box or instance segmentation with the source image  $x^s$ , but the label is changed. Formally, the SCIT for object detection and instance segmentation can be formalized as Equation 4.

$$\begin{cases} x^a = m^s * f_{it}(x^s) + (1 - m^s) * x^s, \\ y_{roi}^a \neq y_{roi}^s, \\ g(x^a) = g(x^s), \end{cases} \quad (4)$$

where  $y_{roi}^a$  and  $y_{roi}^s$  denote the label corresponding to the ROI of  $x^a$  and  $x^s$ . We found this kind of method suitable for image classification.

### 3. STYLE-CONSISTENT IMAGE TRANSLATION

In this section, we instantiate  $f_{it}$  and  $g$  as a GAN model and VGG network and deploy them to do image classification (Simonyan and Zisserman, 2014), object detection, and instance segmentation for tomato leaves. Specifically, an updated CycleGAN (Zhu et al., 2017) is leveraged in our experiments to translate images from the source domain to the target domain. To keep the style consistent, VGG loss is employed (Huang and Belongie, 2017; Li et al., 2017). But we emphasize that other kinds of image translation methods and style losses are possible and encouraged. Since our method aims to keep the style consistent when doing image translation, it is called SCIT.

#### 3.1. Framework

Style-consistent image translation consists of three parts functionally, as shown in **Figure 2A**. The Generator,  $G$  for short, is expected to translate the image, while the discriminator  $Dis$  is assumed to push the translated image similar to the real image in the target domain, and a pre-trained VGG19 is utilized to extract the style, class-unrelated characters.

The generator  $G$  absorbs source image  $x^s$  and instance segmentation mask from source image  $m^s$  as input, in which two specific encoders,  $E_I$  and  $E_M$ , are leveraged to extract features from image and mask, respectively. The outputs of the two encoders are concatenated, followed by a decoder to produce an output image  $x'$ . Formally, the generating process can be formalized as Equation 5.

$$x' = G(x^s, m^s) = Dec(E_I(x^s) \oplus E_M(m^s)), \quad (5)$$

where  $\oplus$  denotes concatenation channel-wise. The instance segmentation mask  $m^s$  is used to let the generator know where to put attention. Similarly, our discriminator takes in real image  $x^t$  in the target domain or generated image  $x'$ , along with its instance mask  $m^s$ , to recognize whether it is fake or real.

In the inference time, the image  $x^a$  to be used as augmented data is a fusion of the source image  $x^s$  and the generated image  $x'$ . Equation 6 shows the inference process to get augmented data. Intuitively, the augmented image has the same background



**FIGURE 3** | Original tomato leaf dataset. From the first to the last row are Healthy, Powdery (powdery mildew), Canker, LMold (leaf mold), ToCV, MagDef (magnesium deficiency). We collect the dataset from different farms at different times. The variations among the dataset include background, type of tomato, the severity of disease, illumination condition, the distance between the camera and interesting leaf, viewpoint to take the picture.

as the source image but has the same foreground as the translated image.

$$x^a = m^s * Dec(E_I(x^s) \oplus E_M(m^s)) + (1 - m^s) * x^s. \quad (6)$$

### 3.2. Loss Functions

The loss functions employed to train our SCIT model are explained in this subsection. To push the generated image toward the real image in the target domain, GAN loss is used as shown in Equation 7.

$$\mathcal{L}_{GAN} = \mathbb{E} \|Dis(x^t \oplus m^t) - \mathbf{1}\|_2 + \mathbb{E} \|Dis(G(x^s, m^s) \oplus m^s)\|_2. \quad (7)$$

Except for being like a real image in the target domain, the generated image  $x^t$  is hypothesized to have the same style as the source image  $x^s$ , which is realized by a pre-trained VGG network, as Equation 8.

$$\mathcal{L}_{sty} = \mathbb{E} \sum_i^n k(\phi_i(x^s * m^s), \phi_i(x^t * m^s)), \quad (8)$$

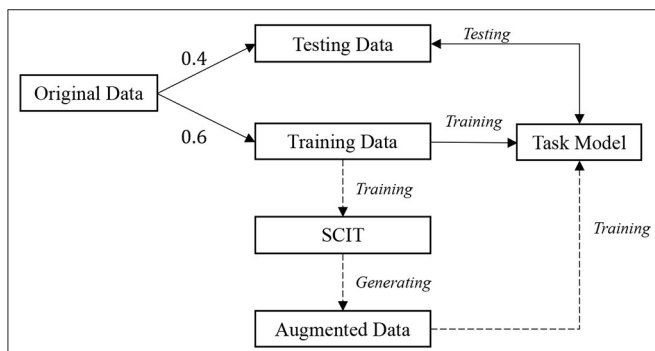
where  $k$  denotes a kernel function, and  $\phi$  is an image feature extractor. A linear kernel function is employed in this study, as it gives a decent performance and requires less computation



**TABLE 1** | Dataset used in the experiment.

Type	Original data						Augmented data	
	All		Testing data		Training data		Images	Masks
	Images	Masks	Images	Masks	Images	Masks		
Healthy	314	358	0	0	0	0	0	0
Powdery	170	185	71	73	106	112	314	358
Canker	298	309	119	122	185	187	314	358
Leaf mold	166	198	74	78	105	120	314	358
ToCV	172	227	81	89	115	138	314	358
Mag Def	138	152	57	59	86	93	314	358
All	999	1,071	402	421	597	650	1,570	1,790

It includes healthy tomato leaves and five kinds of diseases of abnormalities. To do the tasks (image classification, object detection, and instance segmentation), the original data are split into two parts according to the number of masks, 60% training and 40% as testing. The healthy images are only used to do image translation.



**FIGURE 4** | Data utilization process. The originally collected data are split into testing and training data, 0.4 and 0.6, respectively. The training data is firstly adopted to train our SCIT model that then generates the augmented data. Next, the training data along with the augmented data are leveraged to train the task model (image classification, object detection, or instance segmentation). The dotted lines suggest the data augmentation process. The SCIT model is one of our main contributions, which introduces new variations for the original data and, thus, encourages the task model to have better performance on the testing data.

denote the source domain and target domain. For instance,  $G_S$  means the generator which aims to translate an image into the source domain while  $G_T$  denotes the generator which aims to translate an image into the target domain. Identity loss, defined in Equation 9, comes from that when one instance in the target domain is given to generator  $G_T$  as input, the generator needs to output the same as its input without any change. Equation 10 shows the way to calculate the cycle-consistent loss. When we translate an image in the source domain into the target domain and translate the result back into the source domain, we want to get the same result as the original input.

$$\mathcal{L}_{ide} = \mathbb{E} \|G_S(x^t, m^t) - x^t\|_1 + \mathbb{E} \|G_T(x^s, m^s) - x^s\|_1. \quad (9)$$

$$\mathcal{L}_{cyc} = \mathbb{E} \|G_S(G_T(x^s, m^s), m^s) - x^s\|_1 + \mathbb{E} \|G_T(G_S(x^t, m^t), m^t) - x^t\|_1. \quad (10)$$

In the end, our model is trained with four loss functions described before. Mathematically, Equation 11 shows the sum of the four losses, where  $\lambda_*$  balances each loss.

$$\mathcal{L}_{full} = \mathcal{L}_{GAN} + \lambda_{sty} \mathcal{L}_{sty} + \lambda_{ide} \mathcal{L}_{ide} + \lambda_{cyc} \mathcal{L}_{cyc}. \quad (11)$$

(Li et al., 2017). We apply a pretrained VGG-19 network as the feature extractor without finetuning. Thus,  $\phi_i$  means a layer in VGG-19, which is similar to the literature. In our experiments, we use *relu1\_1*, *relu2\_1*, *relu3\_1*, *relu4\_1*, and *relu5\_1* layers with equal weights and, thus,  $n = 5$ . Intuitively, the style loss pushes the two images to share the same feature distribution. Specifically, the linear kernel function-based loss encourages them to have the same sample mean in feature distribution space. We refer to Li et al. (2017) to check the detail about the style loss and its related kernel functions. The deep learning-based style loss with a linear kernel function is adopted in our experiment, but other methods, such as different kernel functions and new style loss are theoretically possible.

To ease the training of generator model  $G$ , identity loss and cycle-consistency loss are leveraged in our experiments, as shown in Figure 2B (Zhu et al., 2017). To describe clearly the cycle-consistency loss, we use subscript with  $S$  and  $T$  to

## 4. EXPERIMENTS

### 4.1. Dataset and Implementation Details

**Dataset.** We aim to recognize tomato diseases among different farms and collect data in real farms with many variations, such as the type of tomato, the distance between the camera and tomato leaf, weather, and illumination. Figure 3 gives examples of the collected images. A total of 1,258 images of tomato leaves are collected, called original data, which covers five types of disease and healthy leaves. Table 1 displays the number of images for each class. As shown in Figure 4, the original data are first split into 40% testing and 60% training data, respectively. We adopt the training data to train data augmentation models and utilize all healthy leaves images as testing to get the augmented data for the other five diseases. Table 1 shows the number of augmented data for each class. Since we are not interested in healthy leaves,

**TABLE 2** | The architecture details adopted in our algorithm.

Network	Input size	Operation	Normalization	Active function
$E_I$	(256, 256, 3)	Conv7-C64-S1-P3	InstNorm	ReLU
	(256, 256, 64)	Conv3-C128-S2-P1	InstNorm	ReLU
	(128, 128, 128)	Conv3-C256-S2-P1	InstNorm	ReLU
	(64, 64, 256)		Residual block * 9	
$E_M$	(256, 256, 3)	Conv7-C64-S1-P3	InstNorm	ReLU
	(256, 256, 64)	Conv3-C128-S2-P1	InstNorm	ReLU
	(128, 128, 128)	Conv3-C256-S2-P1	InstNorm	ReLU
Dec	(64, 64, 512)	Conv1-C256-S1-P1	InstNorm	ReLU
	(64, 64, 256)	DeConv3-C128-S2-P1	InstNorm	ReLU
	(128, 128, 128)	DeConv3-C64-S2-P1	InstNorm	ReLU
Dis	(256, 256, 64)	Conv7-C3-S1-P0	InstNorm	Tanh
	(256, 256, 3)	Conv4-C64-S2-P1	InstNorm	LeakyReLU
	(128, 128, 64)	Conv4-C128-S2-P1	InstNorm	LeakyReLU
	(64, 64, 128)	Conv4-C256-S2-P1	InstNorm	LeakyReLU
	(32, 32, 256)	Conv4-C512-S1-P1	InstNorm	LeakyReLU
	(31, 31, 256)	Conv4-C1-S1-P1	InstNorm	Sigmoid

The input size is height, width, and channel. In the operation, Convk is a convolution layer with kernel size as k, DeConv is the deconvolution layer. Ck, Sk, and Pk denote the number of channels, stride, and padding, respectively. Instance normalization (InstNorm) is used. We utilize nine residual blocks (He et al., 2016) to extract necessary information from the image while no residual block is used in the mask encoder as the mask is much simpler than the image.

**TABLE 3** | The impact of  $\lambda_{sty}$  on FIDs. For each  $\lambda_{sty}$ , we execute five times and report the mean and SD.

	$\lambda_{sty} = 0$	$\lambda_{sty} = 0.5$	$\lambda_{sty} = 1$	$\lambda_{sty} = 2$	$\lambda_{sty} = 4$
Powdery	70.3 ± 1.35	60.2 ± 1.41	63.0 ± 1.40	65.7 ± 0.60	67.3 ± 0.84
Canker	97.5 ± 2.36	93.4 ± 2.90	93.5 ± 1.73	98.0 ± 2.81	99.3 ± 1.93
LMold	97.9 ± 3.09	98.2 ± 2.97	100.8 ± 2.34	103.1 ± 3.32	112.6 ± 5.14
ToCV	79.6 ± 5.95	72.7 ± 6.68	75.7 ± 5.35	86.6 ± 3.52	93.0 ± 2.14
MagDef	83.2 ± 4.10	83.1 ± 1.71	85.1 ± 1.77	90.2 ± 2.83	89.5 ± 3.12

we do not do data augmentation for the healthy class. A total of 314 generated images and 358 instances are generated for each disease class. Finally, the augmented data and the training data are leveraged to train task models (classification, object detection, and instance segmentation). While object detection and instance segmentation can give more than one label or one instance to one image, image classification requires one holistic label for one image. To ease image classification, we crop the original image to get a single leaf in one image.

**Implementation details.** The original training dataset is employed to train our data augmentation model. Meanwhile, basic image manipulation is adopted to enlarge the original data since the collected data are not enough to train the SCIT model. Specifically, we use three times random brightening or darkening, and three times random cropping. Hence, the training data are enlarged to six times in total. During the training process of our data augmentation model, random flip in horizontal and vertical are employed and every image is resized to 256 in both height and width. Each type of disease employs one specific SCIT model.

We use Adam optimizer to train our model for 100 epochs with a learning rate of 0.0002. The batch size is set as 6

with three TITAN V GPUs (12 GB memory). After training, the trained models are adopted to generate disease images, which are later taken as augmented data to train task models. Every training process for each class and  $\lambda_{sty}$  roughly spends 5 h. Therefore, all translation models require about 26 days (5 translation models for each class and 5  $\lambda_{sty}$  settings, each setting is executed 5 times.)

**Architectures.** The proposed SCIT model consists of three sub-models. First, the generator consists of an image encoder, mask encoder, and decoder. The image encoder, aiming to extract necessary information from the input images, leverages several stacks of convolution-ReLU-BatchNorm layers and nine residual blocks, while the mask encoder only adopts the same number of stacks of convolution-ReLU-BatchNorm layers without residual blocks since the mask is much simpler than images. In contrast, the decoder, aiming to produce bigger size images from the smaller size of the feature map, employs several stacks of deconvolution-ReLU-BatchNorm. Second, the discriminator also applies several stacks of convolution-LeakyReLU-BatchNorm. The details of our generator and discriminator are referred to in **Table 2** and our codes will



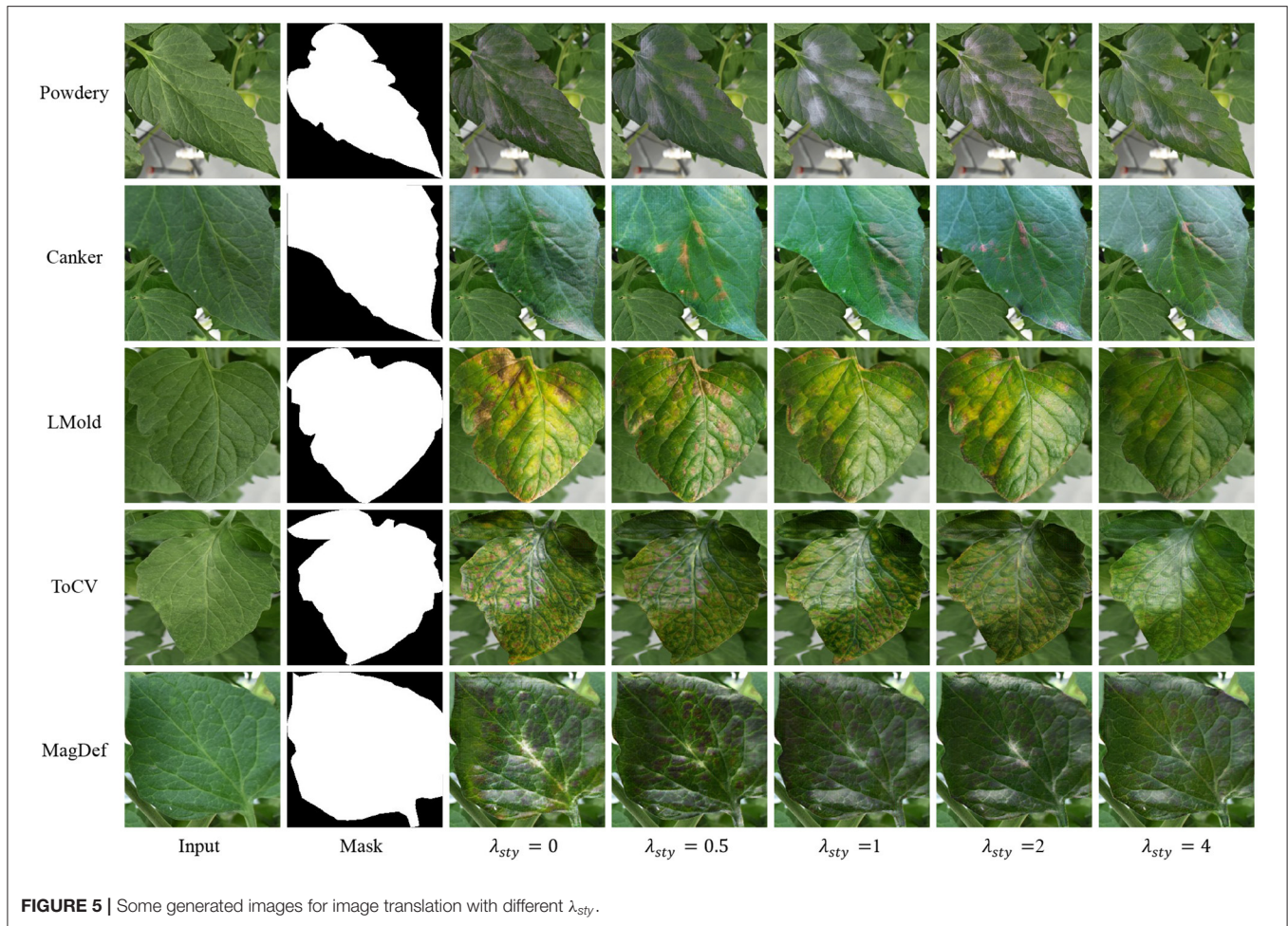


FIGURE 5 | Some generated images for image translation with different  $\lambda_{sty}$ .

TABLE 4 | Ablation study of  $\lambda_{sty}$  in image classification.

	VGG11	ResNet18	DenseNet121	MobileNet v2	ShuffleNet v2	MNASNet
$\lambda_{sty} = 0$	89.04 ± 2.12	92.68 ± 0.63	96.01 ± 0.32	94.14 ± 0.23	79.56 ± 0.42	67.65 ± 7.01
$\lambda_{sty} = 0.5$	88.03 ± 3.75	94.24 ± 0.55	96.25 ± 0.46	95.56 ± 0.32	80.37 ± 0.19	65.08 ± 6.65
$\lambda_{sty} = 1$	93.82 ± 2.34	94.10 ± 0.64	96.24 ± 0.48	96.20 ± 0.28	86.46 ± 0.69	73.11 ± 4.52
$\lambda_{sty} = 2$	94.36 ± 2.63	94.61 ± 0.38	96.48 ± 0.44	96.28 ± 0.32	87.44 ± 0.69	69.45 ± 7.48
$\lambda_{sty} = 4$	86.17 ± 3.12	93.96 ± 0.76	95.32 ± 0.31	95.57 ± 0.49	82.35 ± 0.79	64.52 ± 6.79

The red font shows the best accuracy for each classification model.

be public soon<sup>1</sup>. In terms of the style computation model, a pretrained VGG19 model<sup>2</sup> is leveraged.

## 4.2. Ablation Study

**FIDs and Visualization.** In this subsection, we analyze the impact of the style-consistent loss by changing the value of  $\lambda_{sty}$  in Equation 11. We use Fréchet inception distance (FID) (Heusel et al., 2017) to show its impact, one of the popular methods to access the quality of the generated images by computing the distance between two images distributions, real images, and the

translated images. In general, the lower the FID value, the closer distance between the distributions. We point out that FID is not suitable to access the generated images by our SCIT model since we assume that the real images in the target domain are not available. But the FID can be used to show the tendency between our generated images and all available data that we have when  $\lambda_{sty}$  changes.

To compute the FID, all original images including training and testing images are leveraged as real images while the generated images are taken as fake images. We borrow code<sup>3</sup> to compute

<sup>1</sup><https://github.com/xml94/SCIT>

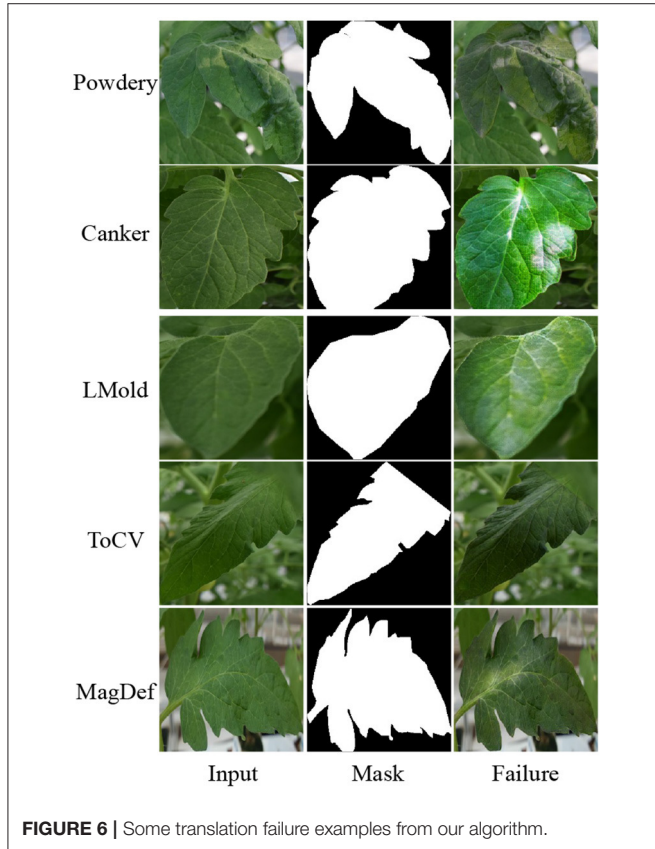
<sup>2</sup><https://pytorch.org/vision/stable/models.html#torchvision.models.vgg19>

<sup>3</sup><https://github.com/mseitzer/pytorch-fid>

**TABLE 5** | Verification of SCIT.

Training data	ResNet18	DenseNet121	MobileNet v2
$\mathcal{T}_o$	86.94 ± 0.69	91.21 ± 0.72	85.27 ± 0.47
$\mathcal{T}_t$	78.24 ± 1.05	81.38 ± 1.06	79.90 ± 1.51
$\mathcal{T}_o + \mathcal{T}_m$	90.01 ± 0.42	94.06 ± 0.41	92.37 ± 0.54
$\mathcal{T}_o + \mathcal{T}_m + \mathcal{T}_t$	94.61 ± 0.38	96.48 ± 0.44	96.28 ± 0.32

We show the performance of three popular classifier models according to differently combined training datasets with averages and standard deviations.  $\mathcal{T}_o$  denotes the original training dataset,  $\mathcal{T}_m$  denotes the augmented training data by basic image manipulation, and  $\mathcal{T}_t$  denotes the training data by our image translation method.



**FIGURE 6** | Some translation failure examples from our algorithm.

FID. **Table 3** shows the FID values for each class,  $\lambda_{sty}$  ranging from 0 to 4. From the table, we observe that FID tends to be larger as  $\lambda_{sty}$  ranges from 0.5 to 4, which proves that the generated images are farther from the real images when we try to keep the style in the image translation process. Moreover, the performance of  $\lambda_{sty} = 0$  differs in different classes. It shows lower FIDs in LMold and MagDef but higher FIDs in translated powdery mildew.

**Figure 5** shows the generated image with different  $\lambda_{sty}$ . The visual comparisons in the figure comply with the FID values in **Table 3**. First, the generated images with  $\lambda_{sty} = 0$  are clear to show the corresponding class, but the style is far from the input images. Further, the style is better to be maintained when  $\lambda_{sty}$  becomes larger while the abnormal severity tends to be less. We

argue that the variation of the severity contributes to improving task performance. As the collected data are limited to variations, the FID in big  $\lambda_{sty}$  tends to be worst for some classes, such as in LMold and MagDef. But when the variations in the collected data are bigger, such as powdery mildew, our SCIT model tends to be better.

**Image classification.** Except for FID and visualization, we conduct the ablation study on the image classification task. To compare the accuracy in image classification, three categories of classification models are utilized for our tomato leaves. VGG, ResNet, and DenseNet are often deployed in applications with big scale datasets, while MobileNet and ShuffleNet aim to save computations for mobile devices. MNASNet is designed to find the optimal model setting. As our dataset is not big, smaller architecture is the default for all models. On the other hand, since our main objective is data augmentation, we use an open code<sup>4</sup> to produce it. For each model, we execute five times independently for each augmented data. All models are trained for 400 epochs and the best performance is recorded. The initial learning rate is set as 0.02 and decreased to 0.01 and 0.005 at epoch 50 and 200 respectively. SGD is the optimizer with 0.9 as the momentum and batch size is 64 using one GPU. **Table 4** displays the comparison results. The table shows that the performance varies with different  $\lambda_{sty}$ ,  $\lambda_{sty} = 2$  showing its superiority to other values except in MNASNet, and controlling the style tends to be better than without controlling the style, such as  $\lambda_{sty} = 0$  resulting in less accuracy than other values of  $\lambda_{sty}$  with model ResNet, DenseNet, MobileNet, and ShuffleNet. It validates that our model, style controlling, is reliable to dedicate the classification in the applications with a small dataset. Moreover, the classification accuracy changes with different models. We argue that it is related to the model itself. As ResNet has fewer parameters and more powerful architecture than VGG, the performances in ResNet are better than VGG. DenseNet, an advanced version of ResNet, also obtains decent results. While MobileNet receives competing results, MNASNet behaves much lower than other methods. We guess that the optimal model setting of MNASNet learned from other datasets is not suitable for our tomato dataset.

**Verification of SCIT.** As shown in **Table 5**, performance is compared using different training datasets in three popular networks for image classification to check how the augmented data by our SCIT model work as training data. For this experiment, we use the combinations of three datasets as training datasets. Let  $\mathcal{T}_o$  denote the original training data,  $\mathcal{T}_t$  denote the training data by our image translation SCIT method, and  $\mathcal{T}_m$  denote the augmented training data by basic image manipulation.  $\mathcal{T}_t$  individually shows potential with an average accuracy of 79.84 on the testing data, even though its performance is worse than  $\mathcal{T}_o$ . One of the reasons is that there are some translation failures in our SCIT model, as shown in **Figure 6**, which is common with deep learning-based image generation, such as BigGAN (Brock et al., 2018). However,  $\mathcal{T}_t$  coupled with  $\mathcal{T}_o, \mathcal{T}_m$  improves performance averagely by 3.58% over three classification models.

<sup>4</sup><https://github.com/bearpaw/pytorch-classification>

**TABLE 6** | Comparison results to other methods to perform image classification for tomato leaves with multiple deep learning-based models.

	Training data	Accuracy	Precision	F1 Score	Specificity
VGG11	Baseline	78.72 ± 3.78	98.95 ± 0.12	86.31 ± 5.21	87.42 ± 1.51
	DCGAN	78.34 ± 6.36	98.30 ± 0.78	86.30 ± 4.67	87.46 ± 1.31
	LeafGAN	87.73 ± 2.47	99.06 ± 0.05	94.03 ± 1.37	89.28 ± 0.46
	CycleGAN*	89.04 ± 2.12	99.21 ± 0.33	95.37 ± 1.55	89.81 ± 0.55
	Ours	<b>94.36 ± 2.63</b>	<b>99.46 ± 0.13</b>	<b>96.50 ± 2.03</b>	<b>90.14 ± 0.51</b>
ResNet18	Baseline	86.75 ± 0.81	99.07 ± 0.43	92.06 ± 0.61	88.80 ± 0.83
	DCGAN	82.52 ± 0.99	98.77 ± 0.29	89.44 ± 0.74	88.17 ± 0.59
	LeafGAN	90.36 ± 0.82	99.27 ± 0.31	94.49 ± 0.56	89.93 ± 0.39
	CycleGAN*	92.68 ± 0.63	99.35 ± 0.12	95.50 ± 0.33	90.33 ± 0.22
	Ours	<b>94.61 ± 0.38</b>	<b>99.68 ± 0.07</b>	<b>96.59 ± 0.12</b>	<b>90.65 ± 0.19</b>
DenseNet121	Baseline	90.97 ± 0.40	99.59 ± 0.27	94.94 ± 0.27	90.16 ± 0.40
	DCGAN	90.45 ± 1.28	99.63 ± 0.22	94.52 ± 0.74	89.97 ± 0.39
	LeafGAN	92.02 ± 0.75	<b>99.77 ± 0.15</b>	95.57 ± 0.59	90.37 ± 0.50
	CycleGAN*	96.01 ± 0.32	99.44 ± 0.22	97.14 ± 0.22	90.16 ± 0.62
	Ours	<b>96.48 ± 0.44</b>	99.61 ± 0.14	<b>97.87 ± 0.40</b>	<b>90.76 ± 0.31</b>
MobileNet v2	Baseline	84.37 ± 2.10	99.05 ± 0.35	90.79 ± 1.34	88.35 ± 0.90
	DCGAN	82.80 ± 1.69	98.94 ± 0.37	89.68 ± 1.12	88.33 ± 0.46
	LeafGAN	91.31 ± 0.53	99.31 ± 0.19	95.17 ± 0.30	89.94 ± 0.39
	CycleGAN*	94.14 ± 0.23	<b>99.44 ± 0.39</b>	96.59 ± 0.77	<b>90.22 ± 0.42</b>
	Ours	<b>96.28 ± 0.32</b>	99.36 ± 0.22	<b>97.53 ± 0.30</b>	90.10 ± 0.52
ShuffleNet v2	Baseline	71.82 ± 0.66	98.98 ± 0.35	84.76 ± 0.31	88.12 ± 0.38
	DCGAN	78.05 ± 1.82	98.80 ± 0.18	86.43 ± 1.10	87.47 ± 0.46
	LeafGAN	77.84 ± 2.26	<b>99.15 ± 0.31</b>	88.89 ± 1.45	88.97 ± 0.70
	CycleGAN*	79.56 ± 0.42	99.12 ± 0.11	89.54 ± 0.48	<b>89.92 ± 0.17</b>
	Ours	<b>87.44 ± 0.69</b>	<b>99.15 ± 0.46</b>	<b>92.43 ± 0.87</b>	89.76 ± 0.69

The boldfaces show the best average evaluation metric for each classification model.

Hence, we conclude that our SCIT model can ease downstream applications to be a data augmentation method.

### 4.3. Image Classification

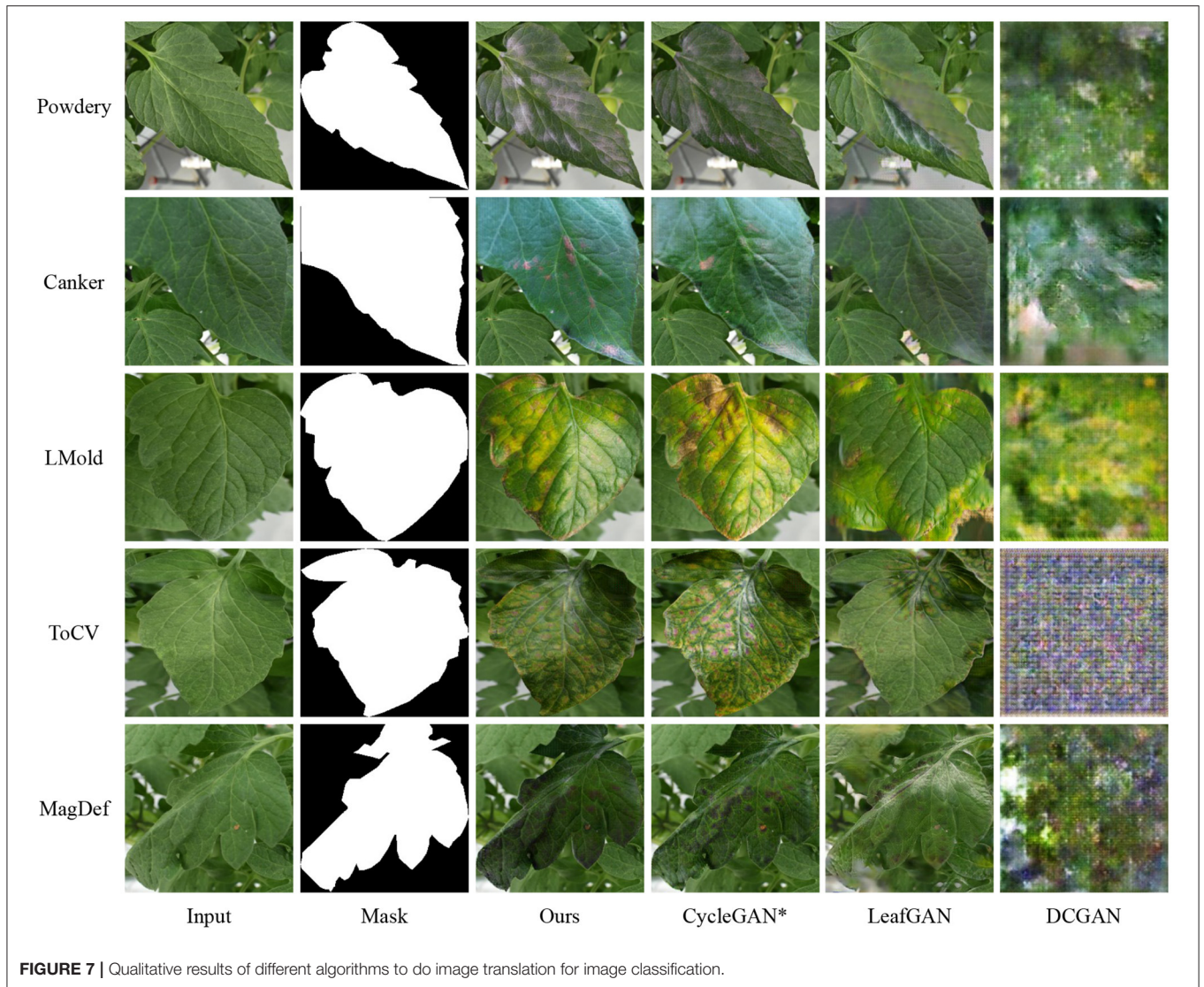
**Compared algorithms.** To validate our algorithm, we compare it to other image generation methods using GANs, as GANs have good reputations to produce clear images. The following methods are compared:

- *Baseline.* We adopt the image manipulation-based data augmentation with the original training dataset. We argue that our data augmentation is complementary to other data augmentation methods. In other words, taking more advanced data augmentation methods such as mixup (Zhang et al., 2018) and cut and paste (Dwivedi et al., 2017) could be a stronger baseline for our future study.
- *DCGAN* (Radford et al., 2015). DCGAN-based (Pandian et al., 2019) or label-condition GANs (Valerio Giuffrida et al., 2017; Pandian et al., 2019; Bi and Hu, 2020; Abbas et al., 2021) are GAN-based algorithms to do data augmentation in which the generator produces images from random noises or given labels. We adopt the original DCGAN to do data augmentation and to produce a higher resolution, two more upsampling layers and convolution layers are added to the original DCGAN.

- *LeafGAN* (Cap et al., 2020). LeafGAN aims to keep the background, one of the challenges of CycleGAN, and introduces an attention module to distinguish the foreground and background. Performance in the cucumber dataset shows its superiority over than original CycleGAN.
- *CycleGAN\**. To get a stronger comparison, we updated CycleGAN. CycleGAN\* directly reuses the background from the input image with a mask as input. We emphasize that CycleGAN\* is the same as the LeafGAN when the mask is given. Simultaneously, our algorithm degrades to CycleGAN\* without keeping the style of the input image during the image translation process.

**Quantitive comparisons.** For each method, five independent training processes are performed. Except for the baseline, all methods are utilized to generate images with resolution 256 in width and height. The number of generated images is also the same except for the baseline. **Table 6** shows the performances of image classification for tomato leaves with different deep-learning models. From the table, CycleGAN\* and our model can significantly improve the classification accuracy. Simultaneously, our data augmentation method achieves the best accuracy and F1 score overall models, which suggests that choosing a good data augmentation method is the way to obtain a better result. In contrast,





**FIGURE 7** | Qualitative results of different algorithms to do image translation for image classification.

**TABLE 7** | Performance of object detection and instance segmentation for tomato leaves in multiple deep learning-based models.

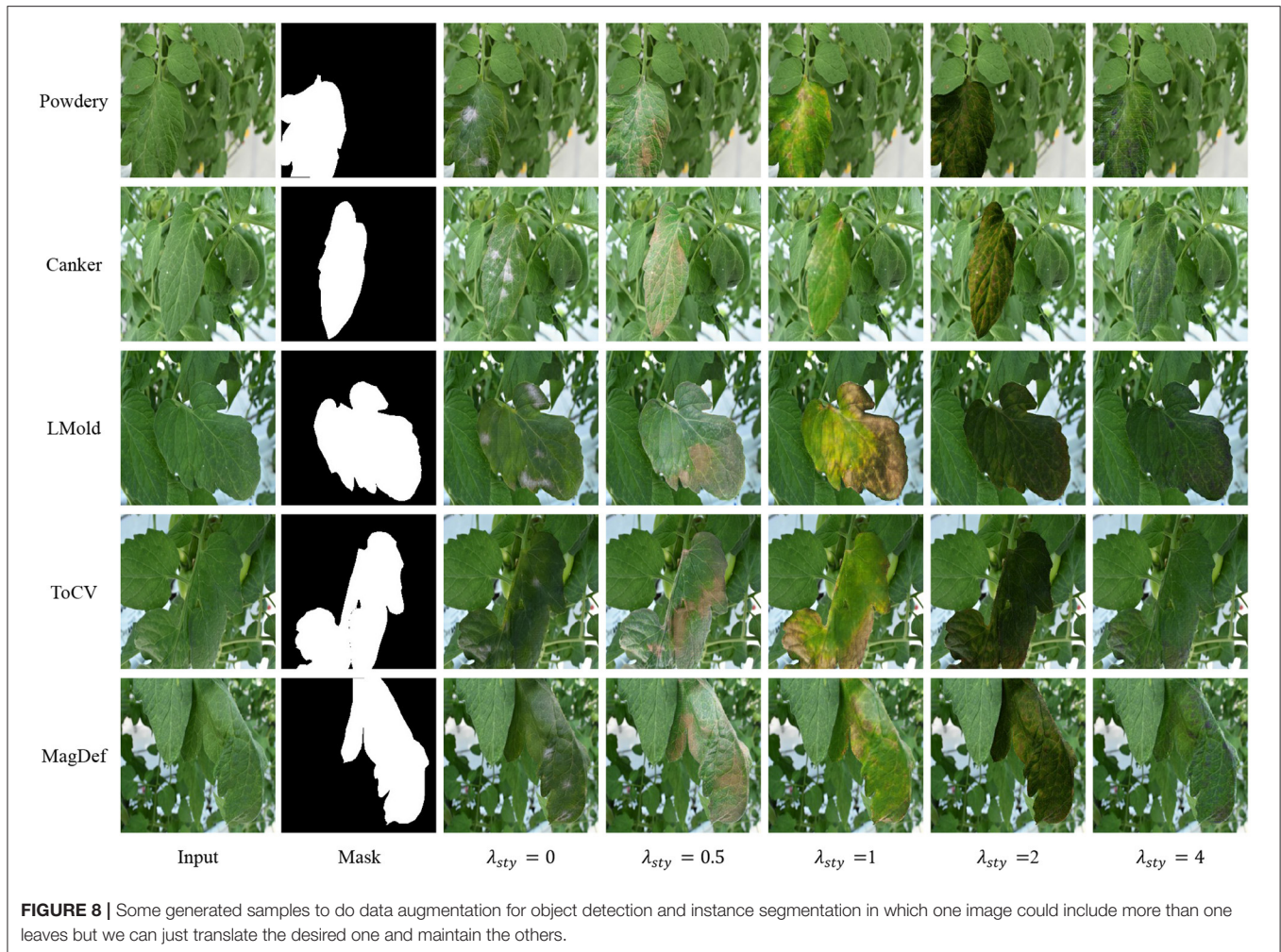
	Baseline		CycleGAN*		Ours	
	mAP	AP50	mAP	AP50	mAP	AP50
FasterRCNN	49.5 ± 0.007	76.8 ± 0.010	50.7 ± 0.015	77.2 ± 0.012	51.5 ± 0.019	77.4 ± 0.015
MaskRCNN	52.4 ± 0.014	79.6 ± 0.015	55.6 ± 0.010	80.4 ± 0.011	56.6 ± 0.005	80.5 ± 0.007
PointRend	51.7 ± 0.009	79.4 ± 0.006	52.8 ± 0.007	80.9 ± 0.011	53.4 ± 0.007	81.1 ± 0.011
YOLO-v3	29.5 ± 0.007	58.7 ± 0.013	31.2 ± 0.015	63.2 ± 0.025	32.6 ± 0.013	65.6 ± 0.024
MaskRCNN	62.6 ± 0.015	80.1 ± 0.009	66.6 ± 0.006	80.0 ± 0.012	67.1 ± 0.010	79.9 ± 0.007
PointRend	56.1 ± 0.023	80.6 ± 0.007	67.6 ± 0.007	81.0 ± 0.010	68.3 ± 0.006	81.3 ± 0.008

Four models have been used to do object detection, while two to do instance segmentation. Red font is utilized to show the best average mAP for each model while blue is adopted to show the best average AP50. In general, higher mAP and higher AP50 are better.

DCGAN and LeafGAN can not always boost performance. Moreover, each data augmentation method shows the best performance with DenseNet and the worst with ShuffleNet, which suggests that the classification model is

also essential and we should choose a better model in our own applications.

**Qualitative results.** Figure 7 shows several generated samples from each algorithm. First, DCGAN can learn similar patterns



such as the white part for the powdery and yellow part for LMold, but produces poor images and, in ToCV, fails. The visual results verify its impact in **Table 6**. Moreover, LeafGAN gives plausible images but its attention module is hard to find decent objects to be translated. Hence, it tends to change the background and fails to do image translation such as the canker image. In contrast, CycleGAN\*, an advanced LeafGAN with a perfect attention module, achieves much better results and hence boosts the classification performances. Furthermore, our method, adopting a style loss to maintain the style during the image translation and hence taking the variations from the source domain to the target domain, obtains decent visual images.

#### 4.4. Object Detection and Instance Segmentation

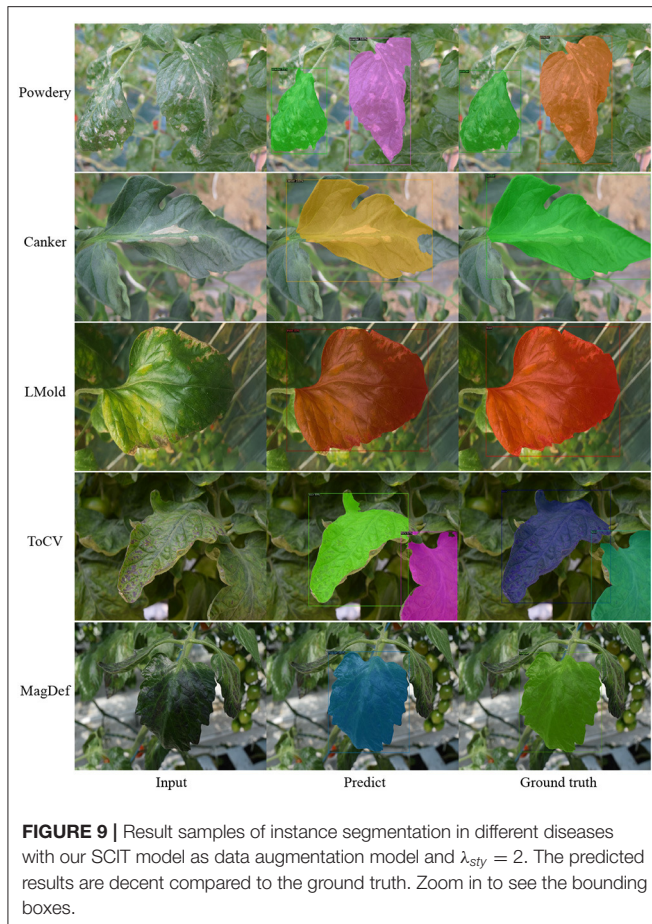
As discussed before, our algorithm reuses the annotation for object detection and instance segmentation. As DCGAN is not image translation-based and LeafGAN can not maintain the annotations, we compare our algorithm to CycleGAN\* in this subsection. To do object detection and instance segmentation,

mmDetection<sup>5</sup> is borrowed as it supports many models. For object detection, we leveraged FasterRCNN, MaskRCNN, and PointRend. In a different paradigm, YOLO aiming to achieve high-speed object detection is also used. Except for object detection, MaskRCNN and PointRend are deployed to do instance segmentation. As  $\lambda_{sty} = 2$  shows its superiority in classification, we use it as the default value and compare it to baseline and CycleGAN\*. For each model, we also execute it five times separately and compute the average performance.

**Table 7** displays the comparison. We observe that while the CycleGAN\* improves the performance, our method gives more improvements. Except for AP50 in YOLO-v3, our model achieves the best mAP and AP50. Besides, we find that FasterRCNN, MaskRCNN, and PointRend obtain similar results and better results than YOLO-v3 to do object detection. **Figure 8** displays some generated samples as data augmentation for object detection and instance segmentation, in which one image includes more than one healthy leaf but we choose one of them to do data augmentation with necessary annotations as output.

<sup>5</sup><https://github.com/open-mmlab/mmdetection>





**Figure 9** illustrates several samples of instance segmentation results using our SCIT model as a data augmentation method. As shown in the figure, the predicted results are highly competent to the ground truth.

## 5. CONCLUSION AND FUTURE WORK

In this study, we introduced a new data augmentation method to improve the abnormality recognition for tomato leaves, termed SCIT which aims to keep the image style from the source image when doing image translation. Armed with this data augmentation paradigm, the data variation in the variation-minority classes is enlarged by the variation-majority class. Simultaneously, we extended the data augmentation from image classification to object detection and instance segmentation which is more competing to do downstream applications. Experimental results validated that the proposed data augmentation method outperforms the baseline and popular methods, in image classification, object detection, and instance

segmentation. Although our algorithm was verified to be useful, our future study includes how to integrate different types of data augmentation methods to facilitate the data-hungry deep learning methods, such as advanced image manipulations (mixup Zhang et al., 2018 and cut and paste Dwibedi et al., 2017). We hope that our study can stimulate the community to use a more powerful data augmentation method to improve the recognition performance for diseases or other abnormalities in the agricultural field where data are hard or expensive to collect.

## DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding authors.

## AUTHOR CONTRIBUTIONS

MX conceived the idea and designed the algorithm, conducted the experiments, and wrote the manuscript. SY supervised the project, analyzed the algorithm, and contributed to part of the writing and overall improvement of the manuscript. AF collected the original images and performed the preliminary experiment on object detection and revised the manuscript. JY enriched the idea, contributed to data annotation, and improved the manuscript. DP conceptualized the paper, supervised the project, and got funding. All authors read and approved the manuscript.

## FUNDING

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2019R1A6A1A09031717); by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2021R1A2C1012174); and by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) and Korea Smart Farm R&D Foundation (KosFarm) through Smart Farm Innovation Technology Development Program, funded by Ministry of Agriculture, Food and Rural Affairs (MAFRA) and Ministry of Science and ICT (MSIT), Rural Development Administration (RDA) (421027-04).

## ACKNOWLEDGMENTS

We thank Yao Meng for the discussion on instance segmentation and preliminary results about instance segmentation. We also appreciate Ruihan Ma for her suggestions and image annotations in healthy tomato leaves.



## REFERENCES

- Abbas, A., Jain, S., Gour, M., and Vankudothu, S. (2021). Tomato plant disease detection using transfer learning with c-gan synthetic images. *Comput. Electron. Agric.* 187:106279. doi: 10.1016/j.compag.2021.106279
- Bi, L., and Hu, G. (2020). Improving image-based plant disease classification with generative adversarial network under limited training set. *Front. Plant Sci.* 11:583438. doi: 10.3389/fpls.2020.583438
- Brock, A., Donahue, J., and Simonyan, K. (2018). “Large scale gan training for high fidelity natural image synthesis,” in *International Conference on Learning Representations* (Stockholm).
- Cap, Q. H., Uga, H., Kagiwada, S., and Iyatomi, H. (2020). Leafgan: an effective data augmentation method for practical plant disease diagnosis. *IEEE Trans. Autom. Sci. Eng.* 1–10. doi: 10.1109/TASE.2020.3041499
- Douarre, C., Crispim-Junior, C. F., Gelibert, A., Tougne, L., and Rousseau, D. (2019). Novel data augmentation strategies to boost supervised segmentation of plant disease. *Comput. Electron. Agric.* 165:104967. doi: 10.1016/j.compag.2019.104967
- Dwivedi, D., Misra, I., and Hebert, M. (2017). “Cut, paste and learn: Surprisingly easy synthesis for instance detection,” in *Proceedings of the IEEE International Conference on Computer Vision* (Venice: IEEE), 1301–1310.
- Fuentes, A., Yoon, S., Kim, S. C., and Park, D. S. (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors* 17, 2022. doi: 10.3390/s17092022
- Gao, J., French, A. P., Pound, M. P., He, Y., Pridmore, T. P., and Pieters, J. G. (2020). Deep convolutional neural networks for image-based convolvulus sepium detection in sugar beet fields. *Plant Methods* 16, 1–12. doi: 10.1186/s13007-020-00570-z
- Gao, J., Westergaard, J. C., Sundmark, E. H. R., Bagge, M., Liljeroth, E., and Alexandersson, E. (2021). Automatic late blight lesion recognition and severity quantification based on field imagery of diverse potato genotypes by deep learning. *Knowl Based Syst.* 214:106723. doi: 10.1016/j.knsys.2020.106723
- Gonzalez-Garcia, A., Weijer, J. v. D., and Bengio, Y. (2018). “Image-to-image translation for cross-domain disentanglement,” in *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal Canada), 1294–1305.
- Gorad, B., and Kotrappa, S. (2021). “Novel dataset generation for indian brinjal plant using image data augmentation,” in *IOP Conference Series: Materials Science and Engineering, Vol. 1065* (Bristol: IOP Publishing), 012041.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Las Vegas, NV: IEEE), 770–778.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). “GANs trained by a two time-scale update rule converge to a local nash equilibrium,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, CA: NIPS), 6629–6640.
- Hu, R., Zhang, S., Wang, P., Xu, G., Wang, D., and Qian, Y. (2020). “The identification of corn leaf diseases based on transfer learning and data augmentation,” in *Proceedings of the 2020 3rd International Conference on Computer Science and Software Engineering* (New York, NY), 58–65.
- Huang, H., Yang, A., Tang, Y., Zhuang, J., Hou, C., Tan, Z., et al. (2021). Deep color calibration for uav imagery in crop monitoring using semantic style transfer with local to global attention. *Int. J. Appl. Earth Observat. Geoinform.* 104:102590. doi: 10.1016/j.jag.2021.10.2590
- Huang, X., and Belongie, S. (2017). “Arbitrary style transfer in real-time with adaptive instance normalization,” in *Proceedings of the IEEE International Conference on Computer Vision* (Venice: IEEE), 1501–1510.
- Kuznichov, D., Zvirin, A., Honen, Y., and Kimmel, R. (2019). “Data augmentation for leaf segmentation and counting tasks in rosette plants,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (Long Beach, CA).
- Lee, H.-Y., Tseng, H.-Y., Huang, J.-B., Singh, M., and Yang, M.-H. (2018). “Diverse image-to-image translation via disentangled representations,” in *Proceedings of the European Conference on Computer Vision* (ECCV) (Munich), 35–51.
- Li, Y., Wang, N., Liu, J., and Hou, X. (2017). “Demystifying neural style transfer,” in *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne), 2230–2236.
- Lin, H., Zhou, G., Chen, A., Li, J., Li, M., Zhang, W., et al. (2021). Em-ernet for image-based banana disease recognition. *J. Food Measur. Characterizat.* 15, 4696–4710. doi: 10.1007/s11694-021-01043-0
- Liu, J., and Wang, X. (2020). Tomato diseases and pests detection based on improved yolo v3 convolutional neural network. *Front. Plant Sci.* 11:898. doi: 10.3389/fpls.2020.00898
- Liu, J., and Wang, X. (2021). Plant diseases and pests detection based on deep learning: a review. *Plant Methods* 17, 1–18. doi: 10.1186/s13007-021-00722-9
- Martineau, M., Conte, D., Raveaux, R., Arnault, I., Munier, D., and Venturini, G. (2017). A survey on image-based insect classification. *Pattern. Recognit.* 65, 273–284. doi: 10.1016/j.patcog.2016.12.020
- Nazki, H., Yoon, S., Fuentes, A., and Park, D. S. (2020). Unsupervised image translation using adversarial networks for improved plant disease recognition. *Comput. Electron. Agric.* 168:105–117. doi: 10.1016/j.compag.2019.105117
- Pandian, J. A., Geetharamani, G., and Annette, B. (2019). “Data augmentation on plant leaf disease image dataset using image manipulation and deep learning techniques,” in *2019 IEEE 9th International Conference on Advanced Computing (IACC)* (Tiruchirappalli: IEEE), 199–204.
- Pawara, P., Okafor, E., Schomaker, L., and Wiering, M. (2017). “Data augmentation for plant classification,” in *International Conference on Advanced Concepts for Intelligent Vision Systems* (Antwerp: Springer), 615–626.
- Pinto Sampaio Gomes, D., and Zheng, L. (2020). “Recent data augmentation strategies for deep learning in plant phenotyping and their significance,” in *2020 Digital Image Computing: Techniques and Applications (DICTA)* (Melbourne), 1–8.
- Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- Saranya, S. M., Rajalaxmi, R., Prabavathi, R., Suganya, T., Mohanapriya, S., and Tamilselvi, T. (2021). Deep learning techniques in tomato plant-a review. *J. Phys.* 1767, 012010. doi: 10.1088/1742-6596/1767/1/012010
- Shen, Z., Huang, M., Shi, J., Liu, Z., Maheshwari, H., Zheng, Y., et al. (2021). Cddt: A large-scale cross-domain benchmark for instance-level image-to-image translation and domain adaptive object detection. *Int. J. Comput. Vis.* 129, 761–780. doi: 10.1007/s11263-020-01394-z
- Shorten, C., and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *J. Big Data* 6, 1–48. doi: 10.1186/s40537-019-0197-0
- Simonyan, K., and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv[Preprint]*. arXiv:1409.1556.
- Valerio Giuffrida, M., Scharr, H., and Tsafaris, S. A. (2017). “Arigan: Synthetic arabidopsis plants using generative adversarial network,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops* (Venice: IEEE), 2064–2071.
- Wang, X., Liu, J., and Zhu, X. (2021). Early real-time detection algorithm of tomato diseases and pests in the natural environment. *Plant Methods* 17, 1–17. doi: 10.1186/s13007-021-00745-2
- Xu, M., Lee, J., Fuentes, A., Park, D. S., Yang, J., and Yoon, S. (2021). Instance-level image translation with a local discriminator. *IEEE Access.* 9, 111802–111813. doi: 10.1109/ACCESS.2021.3102263
- Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). “mixup: Beyond empirical risk minimization,” in *International Conference on Learning Representations* (Stockholm).
- Zhong, Y., and Zhao, M. (2020). Research on deep learning in apple leaf disease recognition. *Comput. Electron. Agric.* 168:105146. doi: 10.1016/j.compag.2019.105146
- Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE International Conference on Computer Vision* (Venice: IEEE), 2223–2232.
- Zhu, Y., Aoun, M., Krijn, M., Vanschoren, J., and Campus, H. T. (2018). “Data augmentation using conditional generative adversarial networks for leaf counting in arabidopsis plants,” in *British Machine Vision Conference*, 324.

**Conflict of Interest:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

**Publisher's Note:** All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may

be made by its manufacturer, is not guaranteed or endorsed by the publisher.

*Copyright © 2022 Xu, Yoon, Fuentes, Yang and Park. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.*