The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

Few-Shot Defect Image Generation via Defect-Aware Feature Manipulation

Yuxuan Duan, Yan Hong, Li Niu*, Liqing Zhang*

MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University
sjtudyx2016@sjtu.edu.cn, yanhong.sjtu@gmail.com, ustcnewly@sjtu.edu.cn, zhang-lq@cs.sjtu.edu.cn

Abstract

The performances of defect inspection have been severely hindered by insufficient defect images in industries, which can be alleviated by generating more samples as data augmentation. We propose the first defect image generation method in the challenging few-shot cases. Given just a handful of defect images and relatively more defect-free ones, our goal is to augment the dataset with new defect images. Our method consists of two training stages. First, we train a data-efficient StyleGAN2 on defect-free images as the backbone. Second, we attach defect-aware residual blocks to the backbone, which learn to produce reasonable defect masks and accordingly manipulate the features within the masked regions by training the added modules on limited defect images. Extensive experiments on MVTec AD dataset not only validate the effectiveness of our method in generating realistic and diverse defect images, but also manifest the benefits it brings to downstream defect inspection tasks. Codes are available at https://github.com/Ldhlwh/DFMGAN.

1 Introduction

Defect inspection, whose typical tasks include defect detection, classification, and localization, plays an important role in industrial manufacture. So far, many research efforts have been paid to design automated defect inspection systems to ensure the qualification rate without manual participation (Pang et al. 2021). However, it is challenging to adequately obtain diverse defect images due to the scarcity of real defect images in production lines and the high collection cost, also known as the *data insufficiency* issue. Therefore, nowadays deep learning-based defect inspection methods (Schlegl et al. 2017; Bergmann et al. 2020; Li et al. 2021) usually adopt an unsupervised paradigm, that is, training one-class classifiers with defect-free data only. Without the supervision of defect images, those models cannot distinguish different defect categories and thus inapplicable to certain tasks such as defect classification.

Aiming to solve the data insufficiency problem, an intuitive idea is to generate more defect images. Previous methods try to render simple yet fake defect images by manually adding artifacts (DeVries and Taylor 2017), cutting/pasting patches of defect-free images (seen as defects) (Li et al. 2021), or copying the defect region from one image to another (Lin et al. 2021). Nevertheless, defect images generated by these methods are far from being realistic and diverse. On the other hand, though Generative Adversarial Network (GAN) (Goodfellow et al. 2014) and its variants are widely used in many image generation tasks, they are scarcely used for defect image generation because GANs are susceptible to data shortage. Previously, quite few GAN-based defect image generation works (Niu et al. 2020; Zhang et al. 2021) are designed. They either rely on hundreds or thousands of defect images and even more defect-free ones, or merely focus on a single category of texture (*e.g.*, marble, metal, concrete). However, in real industrial manufacture, usually only a few defect images are available because of the rarity of real defect images in production lines and the difficulty in collection. Moreover, comparing with textures, objects (*e.g.*, nut, medicine, gadget) have richer structural information and fewer regular patterns, which further escalates the difficulty in defect image generation for objects.

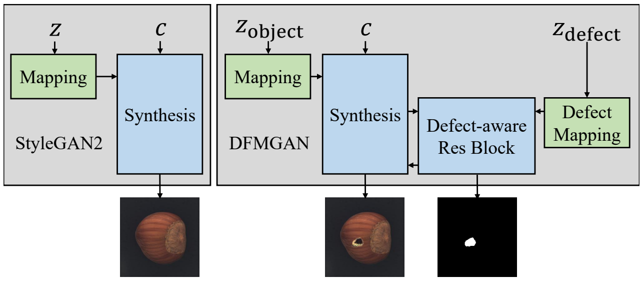

Figure 1: An overview of our DFMGAN and its two-stage training strategy. Left: First, a StyleGAN2 is pretrained on defect-free data. Right: Then, defect-aware residual blocks are attached to the backbone to produce defect masks and manipulate the features within defect regions.

To deal with such cases, we propose a novel Defect-aware Feature Manipulation GAN (DFMGAN) to generate realistic and diverse defect images using limited defect images. DFMGAN is inspired by few-shot image generation methods (Wang et al. 2020; Zhao, Cong, and Carin 2020; Robb et al. 2020) which adapt pretrained models learned on large domains to smaller domains. However, these methods focus on transferring whole images without particular design on

[*] Corresponding authors.

specific regions (*e.g.*, defect areas in defect images). Based on the fact that a defective object has completely defect-free appearance except the defect regions, an intuitive idea could be adaptively adding defects to the generated defect-free images. In this work, with a backbone generator trained on hundreds of defect-free object images[1], we propose defect-aware residual blocks to produce plausible defect regions and manipulate the features within these regions, to render diverse and realistic defect images. An overview of the training process is shown in Figure 1. Extensive experiments on *MVTec Anomaly Detection* (MVTec AD) (Bergmann et al. 2019) prove that DFMGAN can not only generate various defect images with high fidelity, but also enhance the performance of defect inspection tasks as non-traditional augmentation.

Our contributions can be summarized as follows: (1) we make the first attempt at the challenging few-shot defect image generation task using a modern dataset MVTec AD with multi-class objects/textures and defects; (2) we provide a new idea of transferring critical regions rather than whole images which may inspire future works in many few-shot image generation applications; (3) we propose a novel model DFMGAN to generate realistic and diverse defect images associated with defect masks, via feature manipulation using defect-aware residual blocks; (4) experiments on MVTec AD dataset validate the effectiveness of DFMGAN on defect image generation and the benefits it brings to the downstream defect inspection tasks.

2 Related Work

Defect Inspection Due to the data insufficiency issue, defect inspection methods cannot adopt a fully supervised paradigm. With the reconstruction and comparison strategy, AnoVAEGAN (Baur et al. 2019) and AnoGAN (Schlegl et al. 2017) utilized autoencoders (Goodfellow, Bengio, and Courville 2016) and GANs respectively. Besides these generative methods, Li et al. (2021) used Grad-CAM (Selvaraju et al. 2017) to show defect regions when identifying pseudo-defect images constructed from defect-free ones. Bergmann et al. (2020) trained student networks imitating the output of a teacher network on defect-free data, and inferred a defect when obvious distinction between the students and the teacher occurs.

Image Generation on Limited Data Since proposed, Generative Adversarial Network (GAN) and its variants (Goodfellow et al. 2014; Zhu et al. 2017; Choi et al. 2020; Karras et al. 2020b) are renowned for the enormous data required to ensure the quality and diversity of generated images. There are some works focusing on training data-efficient GANs on small datasets. For instance, Zhao et al. (2020) proposed differentiable augmentation as a plugin to StyleGAN2 (Karras et al. 2020b). Nevertheless, these works still generally required at least hundreds of images, leaving directly training on just several or tens of images unsolved. Some other works tried to transfer the model pretrained on larger datasets to boost its performance on small datasets. For example, Noguchi and Harada (2019); Zhao, Cong, and Carin (2020); Robb et al. (2020) eased the transfer process by limiting the number of trained parameters. Wang et al. (2020) explored the transferable latent space regions of the generator. Ojha et al. (2021) preserved a one-to-one correspondence with cross-domain consistency loss. These methods transferred the distribution of the whole images, while we suppose transferring only specific regions (*i.e.* defect regions) may be beneficial to our task.

Defect Image Generation The rarity of defect samples has motivated research efforts on defect image generation as data augmentation for defect inspection applications. DeVries and Taylor (2017) added random cutouts on normal images as artificial defects. Li et al. (2021) copied a patch from a defect-free sample and pasted it to another location, rendering a pseudo-defect. Lin et al. (2021) cropped the defect regions of a defect image and pasted it to another defect-free one. Among these non-generative methods, the first two utilized defect-free images only, whose generated samples are not category-specific thus not applicable to inspection tasks such as defect classification. Crop&Paste (Lin et al. 2021) was only able to yield limited number of defect samples depending on the size of datasets, and actually it could not generate new defects, but moved in-dataset defects onto different objects. Also, traditional data augmentation can be hardly used on defect images because few transformations (*e.g.*, flipping, rotation) keep intact defects without affecting color, pattern, position and other characteristics.

To the best of our knowledge, only two previous works (Niu et al. 2020; Zhang et al. 2021) designed generative augmenting methods. Niu et al. (2020) proposed SDGAN, translating defect-free and defect images interchangeably through two generators. Similarly, Zhang et al. (2021) simulated defacement and restoration processes by adding and removing defect foregrounds using Defect-GAN. However, these two works had certain limitations: (1) **Large texture datasets**: They had access to hundreds or thousands of defect samples of a single category, which are not always accessible. Also, the datasets they used are on highly specific textures (cylinder surfaces of commutators or concrete surfaces), which had much less structural information than objects (*e.g.*, hazelnuts as we use for the experiments). (2) **Merely generate defects**: Both works needed defect-free samples as their input while rendering defects via image-to-image paradigm. This strategy limited the diversity of the object/texture backgrounds, especially in cases that defect-free images were not abundant either. (3) **Lack randomness**: SDGAN did not involve randomly sampled codes or noises as GANs usually do, which further limited the diversity. (4) **No masks**: Neither of these works produced defect masks with clear boundaries, restricting their usage in certain inspection tasks (*e.g.*, defect localization) requiring ground-truth defect masks.

To tackle the aforementioned limitations of previous works, in the following sections, we will introduce DFMGAN, which is the first few-shot defect image generation method capable of rendering realistic images with high di-

[1] For simplicity, we collectively call object and texture as *object* when describing our model.

versity on both objects and defects.

3 Method

As shown in Figure 2, DFMGAN adopts a two-stage training strategy. First, we train a data-efficient StyleGAN2 as the backbone on hundreds of defect-free images, which maps a random code $\boldsymbol{z}_{\text{object}}$ to an image without defect (Section 3.1). Second, we attach defect-aware residual blocks along with defect mapping network to the backbone, and train these added modules on a few defect images. The entire generator maps $\boldsymbol{z}_{\text{object}}$ and a defect code $\boldsymbol{z}_{\text{defect}}$ to defect images with controllable defect regions (Section 3.2).

3.1 Pretraining on Defect-free Images

In the first training stage, we aim to train a generator as the backbone of DFMGAN to produce diverse defect-free images by randomly sampling object codes $\boldsymbol{z}_{\text{object}}$. Considering the superiority of generation ability of StyleGANs, we adopt StyleGAN2 with Adaptive Differentiable Augmentation (Karras et al. 2020a) as the backbone, which consists of a mapping network and a synthesis network. The synthesis network, taking a learned constant feature map c, is composed of convolutional synthesis blocks with the *skip* architecture accumulating RGB appearances through *ToRGB* modules to the final generated images. The mapping network takes a random $\boldsymbol{z}_{\text{object}}$ and maps it to $\boldsymbol{w}_{\text{object}}$ which modulates the convolution weights of the synthesis network (green arrows in Figure 2), importing variations to the generated images. Besides the generator, a discriminator is also trained to provide supervision. Refer to Karras et al. (2020b) for detailed designs of StyleGAN2. After this stage, the backbone generator encodes rich object features in its network. In the next stage, we will attach defect-aware residual blocks, which can adapt the model from defect-free images to defect ones.

3.2 Transferring to Defect Images

Considering the fact that a defect image is composed of defect regions and defect-free regions, we conjecture that by properly manipulating the potential defect regions of the object feature maps from the backbone, the whole model can be extended to produce defect images while maintaining the generation ability of defect-free images. Motivated by this idea, in the second training stage, we propose the defect-aware residual blocks attached to the backbone, rendering plausible defect masks delimiting defect regions and the corresponding defect features. The masks and the defect features are then used to manipulate the object feature maps to add defects to defect-free images. To ensure the fidelity and variation of the defects, we further employ an extra defect matching discriminator and a modified mode seeking loss respectively during this stage.

Defect-aware Residual Blocks Our proposed defect-aware residual blocks share similar structure with the synthesis blocks of the backbone. At resolution 64 where the first residual block is attached, the synthesis block S and the residual block R both take the feature $\boldsymbol{F}^{32}$ from the last synthesis block at resolution 32 and output object feature map $\boldsymbol{F}^{64}_{\text{object}} = S(\boldsymbol{F}^{32})$ and defect residual feature map $\boldsymbol{F}^{64}_{\text{defect}} = R(\boldsymbol{F}^{32})$ respectively, where $\boldsymbol{F}^{64}_{\text{object}}, \boldsymbol{F}^{64}_{\text{defect}} \in \mathbb{R}^{N\times 64\times 64}$ and N is the number of channels. Then the extra *ToMask* module, like its counterpart *ToRGB* modules of the backbone, determines the defect region of the current image by generating a mask $\boldsymbol{M} = \textit{ToMask}(\boldsymbol{F}^{64}_{\text{defect}}) \in \mathbb{R}^{64\times 64}$. Only the residual features corresponding to the defect pixels (non-negative values on the mask $\boldsymbol{M}$) are added to the object feature map, leading to the manipulated feature map $\boldsymbol{F}^{64}$:

$$\boldsymbol{F}^{64}(i,j) = \begin{cases} \boldsymbol{F}^{64}_{\text{object}}(i,j) + \boldsymbol{F}^{64}_{\text{defect}}(i,j), & \boldsymbol{M}(i,j) \geq 0, \\ \boldsymbol{F}^{64}_{\text{object}}(i,j), & \boldsymbol{M}(i,j) < 0, \end{cases} \tag{1}$$

where (i,j) represents any pixel on the feature map or the mask. In this way, the residual blocks only manipulate the object features within the defect regions, and those in non-defect areas remain unchanged. The manipulated feature map $\boldsymbol{F}^{64}$ then takes the place of $\boldsymbol{F}^{64}_{\text{object}}$ to be the input of the following blocks. Later at resolution 128 and 256, the mask $\boldsymbol{M}$ is upsampled to the corresponding resolution to control the defect residual features, which further manipulate the object feature maps within the defect regions in a similar way to resolution 64. We leave the synthesis blocks at resolution 32 or lower untouched because the high-level layers (with lower resolution) of the networks decide the coarse structure of the images, while low-level layers generate detailed appearances including the defects (Zhao, Cong, and Carin 2020). To ensure the diversity of the generated defect images, instead of being solely determined by the object feature map from the backbone, we introduce an extra defect mapping network to control the variation of defects. The defect mapping network takes in a randomly sampled defect code $\boldsymbol{z}_{\text{defect}}$ and outputs the modulation weights $\boldsymbol{w}_{\text{defect}}$, which is used to modulate the residual blocks (green arrows in Figure 2) similar to the backbone. The two mapping networks share the same network structure.

During the second training stage, DFMGAN fixes its backbone and trains the defect mapping network along with our proposed defect-aware residual blocks on defect images in order to generate more defect samples with high fidelity and diversity. We control the number of trainable parameters in this stage to 3.7M. Compared with the fixed backbone of 23.2M trainable parameters in the previous stage on defect-free images, it will be much easier to train on just a handful of defect images in the second stage. Another advantage of DFMGAN is that, by fixing the parameters of the backbone, it retains the ability of generating defect-free images as long as we cut off the defect-aware feature manipulation by ignoring the defect residual features from the residual blocks.

Two Discriminators Due to the content similarity between the defect-free images and the defect images, we can easily transfer the pretrained discriminator from defect-free images to defect ones by finetuning. Yet, this discriminator can only provide supervision to the images, not the masks. To guarantee that the generated defect masks precisely delimit the defect regions of the images, we use an extra defect matching discriminator D_{match} to bridge the gap between real pairs of defect image and mask and generated pairs.

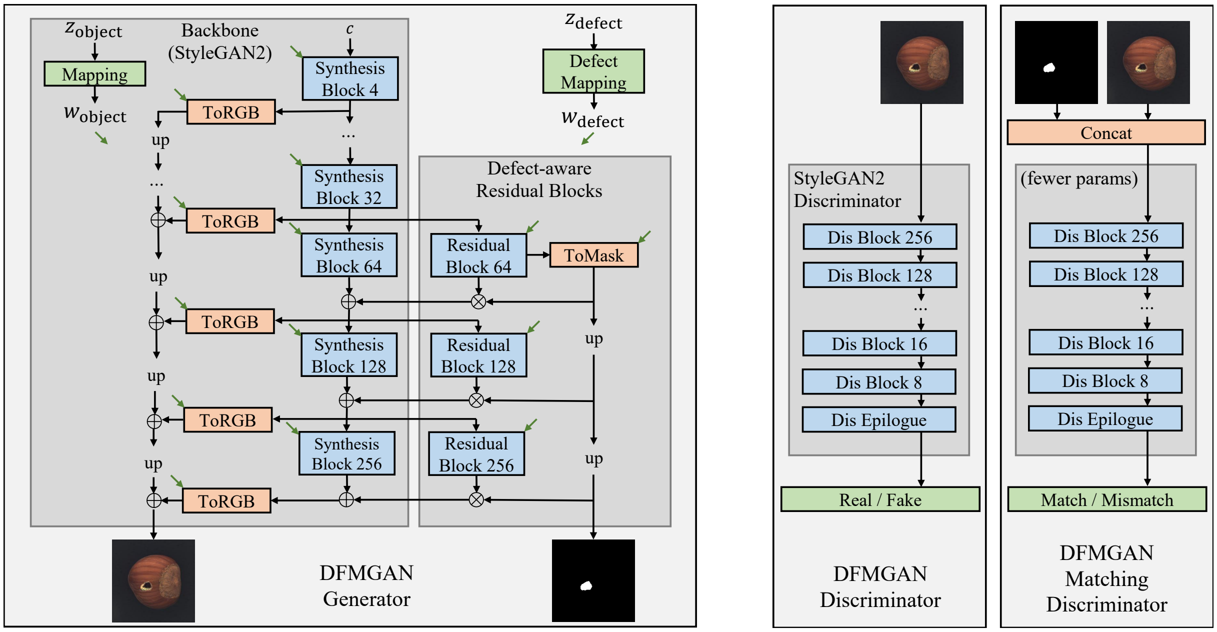

Figure 2: The architecture of DFMGAN. Left: The generator mainly consists of the backbone and the defect-aware residual blocks. The backbone adopts the original structure of StyleGAN2 with *ToRGB* modules accumulating RGB appearances in the *skip* manner. The defect-aware residual blocks manipulate the features starting from resolution 64×64, with masks from *ToMask* module controlling the manipulating areas. Both parts have weights modulated by their corresponding mapping networks shown as green arrows. Right: The two discriminators, respectively in charge of judging the realism of images and whether the defects in images match the masks. We generally remain the original structure of StyleGAN2 discriminator.

D_{match} has almost the same architecture with the original discriminator D, but we reduce the number of channels in each layer based on the intuition that judging whether a defect image matches a mask is easier than judging its realism. With much fewer parameters (1.5M comparing with 24M of D), D_{match} is suitable for few-shot defect image generation. Pairs of image and mask are concatenated before being fed into D_{match}. The output scores judge whether these defect images match their corresponding defect masks. D_{match} provides supervision by optimizing Wasserstein adversarial loss (Gulrajani et al. 2017) with R1 regularization as D does in StyleGAN (Karras, Laine, and Aila 2019). The two discriminators cooperate with each other in the process of defect image generation.

Mode Seeking Loss In our model, the generated defect images depend on the object features from the backbone and the defect features from the defect-aware residual blocks. This design matches the fact that the defect on an object depends on both the object itself and the external factors. However, preliminary experiments with DFMGAN, where we vary $\boldsymbol{z}_{\text{defect}}$ yet fix $\boldsymbol{z}_{\text{object}}$ (thus also fix the object features), have shown that the defects are almost merely determined by the object features, with hardly noticeable changes when using different $\boldsymbol{z}_{\text{defect}}$. In this way, it can be foreseen that similar objects will always be accompanied by resembling defects, which substantially harms the diversity.

To mitigate this problem, we employ a variant of the mode seeking loss (Mao et al. 2019) in the second training stage. With two random defect codes $\boldsymbol{z}^1_{\text{defect}}$ and $\boldsymbol{z}^2_{\text{defect}}$, the defect mapping network outputs two corresponding modulation weights $\boldsymbol{w}^1_{\text{defect}}$ and $\boldsymbol{w}^2_{\text{defect}}$. The whole model produces defect masks $\boldsymbol{M}^1$ and $\boldsymbol{M}^2$ respectively using $\boldsymbol{w}^1_{\text{defect}}$ and $\boldsymbol{w}^2_{\text{defect}}$ along with the same $\boldsymbol{w}_{\text{object}}$. Then, DFMGAN minimizes the mode seeking loss

$$L_{\text{ms}} = \frac{\|\boldsymbol{w}^1_{\text{defect}} - \boldsymbol{w}^2_{\text{defect}}\|_1}{\|\boldsymbol{M}^1 - \boldsymbol{M}^2\|_1}. \tag{2}$$

In other words, when using different $\boldsymbol{w}_{\text{defect}}$, we hope that the difference between the defect masks is maximized.

Practically, we use $\boldsymbol{w}_{\text{defect}}$ instead of $\boldsymbol{z}_{\text{defect}}$ because Karras, Laine, and Aila (2019) states that the latent space of $\boldsymbol{w}$ is less entangled and hence better to represent the input space of the generator than a fixed distribution of $\boldsymbol{z}$. Also, due to the unexpected artifacts on the defect appearances, we use the differences between masks instead of images. See the ablation study for details.

Objective Given the original loss function L_{StyleGAN} used by StyleGAN2 (including adversarial loss, path length regularization and R1 regularization, refer to Karras, Laine, and Aila (2019)), the loss function L_{match} of D_{match} and the mode seeking loss L_{ms}, our DFMGAN alternatively optimizes the generator G and the discriminators D, D_{match} according to the overall objective function

$$\begin{aligned} L(G, D, D_{\text{match}}) = & L_{\text{StyleGAN}}(G, D) + \\ & L_{\text{match}}(G, D_{\text{match}}) + \lambda L_{\text{ms}}(G), \end{aligned} \tag{3}$$

where setting $\lambda = 0.1$ generally renders good results.

Defect	Crack		Cut		Hole		Print	
Method	KID↓	LPIPS↑	KID↓	LPIPS↑	KID↓	LPIPS↑	KID↓	LPIPS↑
Finetune	41.64	0.1541	21.80	0.1192	30.54	0.1263	28.75	0.1526
DiffAug	24.69	0.0570	19.84	0.0456	22.43	0.0466	39.03	0.0604
CDC	206.14	0.0437	213.98	0.0390	271.72	0.0566	355.37	0.0500
Crop&Paste	-	0.1894	-	0.2045	-	0.2108	-	0.2185
SDGAN	148.86	0.1607	161.16	0.1474	152.86	0.1689	176.09	0.1748
Defect-GAN	30.98	0.1905	32.69	0.1734	36.30	0.2007	33.35	0.2007
DFMGAN	**19.73**	**0.2600**	**16.88**	**0.2073**	**20.78**	**0.2391**	**27.25**	**0.2649**

Table 1: The results of the few-shot defect image generation on the object category *hazelnut*, where we report KID$\times 10^3$@5k and clustered LPIPS@1k for each setting. The three groups of methods are respectively generic few-shot image generation methods, non-generative and generative defect image generation methods. DFMGAN outperforms in all four defect categories *crack*, *cut*, *hole*, *print*. For other object/texture categories, see *supplementary material*.

4 Experiment

To verify the effectiveness of DFMGAN, we conduct experiments on the dataset MVTec AD (Section 4.1), including the defect image generation task (Section 4.2) and the downstream defect classification task (Section 4.3). See *supplementary material* for implementation details and the ablation study validating several choices in designing our model.

4.1 Dataset: MVTec AD

MVTec Anomaly Detection[2] (MVTec AD) (Bergmann et al. 2019) is an open dataset containing ten object categories and five texture categories commonly seen, with up to eight defect categories for each object/texture category. All the images are accompanied with pixel-level masks showing the defect regions. Although originally designed for defect localization, MVTec AD fits the experimental setting for few-shot defect image generation since most object/texture categories have 200–400 defect-free samples, and most defect categories have 10–25 defect images. In the first stage, the backbone of DFMGAN is trained on the defect-free images of an object/texture category in the training set. In the second stage, an individual DFMGAN is trained for each defect category associated with this object/texture category. All images are resized to a moderate resolution of 256×256.

In the main paper, we mainly focus on the object category *hazelnut*, which is a highly challenging category in MVTec AD due to its naturally high variation and complex appearance compared to the other manufactured objects. It has four defect categories *crack*, *cut*, *hole*, and *print*. We leave the results on the other categories to *supplementary material*.

4.2 Defect Image Generation

Metric Following Karras et al. (2020a), we use Kernel Inception Distance (KID) (Bińkowski et al. 2021) as our main metric. KID resembles the conventionally used metric Fréchet Inception Distance (FID) (Heusel et al. 2017) in image generation tasks, yet is designed without bias and thus a more descriptive metric on small datasets. Similar to FID, KID evaluates both the reality and the diversity of the generated images where lower values indicate better performance. We report KID between 5,000 generated defect images and the defect images of the corresponding defect category in the dataset.

However, KID (as well as FID) is commonly observed to prefer reality to diversity. Therefore, to supplement the experiment results with a standalone diversity metric, we also report a clustered version of Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al. 2018) used by recent few-shot image generation works. Suppose the dataset of a defect category contains N images. First 1,000 generated images are grouped into N clusters by finding the closest (lowest LPIPS) dataset image, then we compute the mean pairwise LPIPS within each cluster and finally compute the average of them. Such clustered LPIPS suits few-shot image generation tasks because overfitted models will receive nearly zero scores, and higher scores indicate better diversity.

Baseline We compare our DFMGAN with six other methods. *Finetune* pretrains and finetunes both using the original StyleGAN2. *DiffAug* (Zhao et al. 2020) uses differentiable data augmentation to prevent overfitting when directly training on small datasets. *CDC* (Ojha et al. 2021) preserves the cross-domain correspondence among the source/target images using consistency loss. These three are generic few-shot image generation methods not specialized in generating defects. *SDGAN* (Niu et al. 2020) and *Defect-GAN* (Zhang et al. 2021) are the only two generative methods for defect image generation tasks prior to our work, whose details are discussed in Section 2. Finally, though not a generative model, we include *Crop&Paste* (Lin et al. 2021) as the representative of non-generative methods to make the experiments comprehensive.

Quantitative Result The KID and clustered LPIPS results of defect image generation are shown in Table 1. Note that since the produced images of Crop&Paste have almost the same distribution w.r.t. appearance with the datasets, they always receive nearly zero KID scores which are omitted. For all the defect categories of hazelnut, our DFMGAN outperforms all the other methods on both KID and clustered LPIPS, showing its strong ability in rendering defect images with high quality and diversity despite of the severely insufficient data it is trained on.

[2] https://www.mvtec.com/company/research/datasets/mvtec-ad, released under CC BY-NC-SA 4.0.

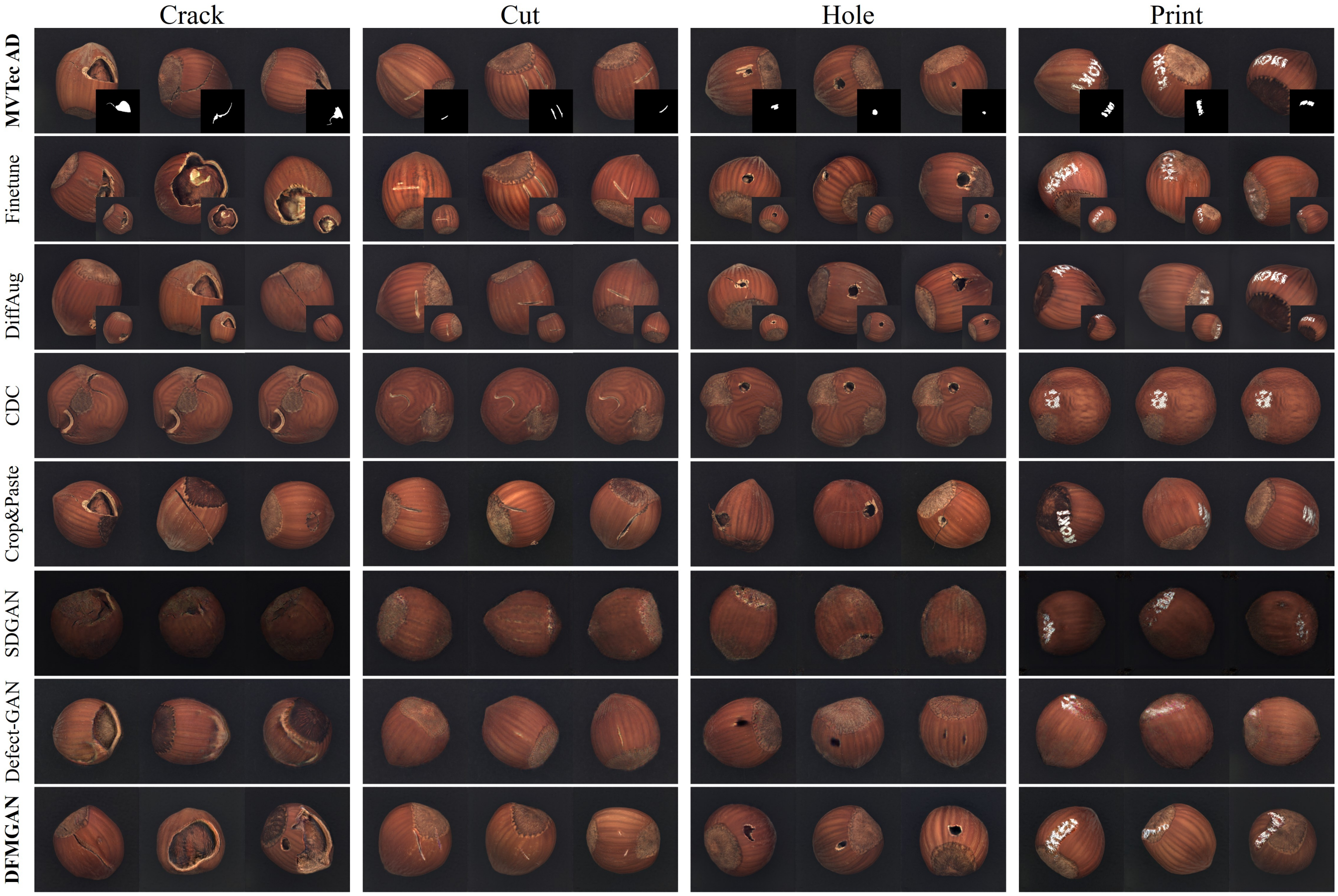

Figure 3: Examples of datasets (with defect masks) and generated defect images. Finetune and DiffAug are prone to overfit the dataset (we show their closest images in the dataset), while CDC generates unreal images without much differences. Crop&Paste cannot produce new defects (*e.g.*, the first *crack* image has the defect appearance of the first one in MVTec AD) and sometimes the defects go beyond the boundaries of the objects (*e.g.*, the first *hole* image). SDGAN and Defect-GAN fail to render realistic samples. DFMGAN has the most satisfying performance balancing quality and diversity. See *supplementary material* for other categories.

Qualitative Result To provide a visual comparison among DFMGAN and the other methods, we show examples of the generated defect images in Figure 3. Although the images yielded by Finetune and DiffAug have good quality, they are actually overfitting the training images, contributing marginal extra diversity. In contrast, CDC, SDGAN and Defect-GAN cannot produce realistic samples in these few-shot cases on the challenging object category. Crop&Paste only borrows the defect appearances from the datasets. Finally, our DFMGAN achieves a good balance between reality and diversity, generating satisfying images.

For our method, we also show groups of generated defect-free image, defect image and its mask of defect category *hole* in Figure 4, where the masks precisely show the defect regions. Also, by fixing the parameters of the backbone in the second training stage, DFMGAN is still able to generate defect-free samples if the model is forced to ignore the defect-aware residual features. Paired defect-free and defect images can be utilized in defect-related tasks such as defect restoration.

Discussion Comparing the few-shot image generation methods (Finetune, DiffAug, CDC) and the previous generative defect generation methods (SDGAN, Defect-GAN), we have found that the formers, without special designs for generating defects, are prone to overfit. Most images yielded by Finetune or DiffAug are roughly identical to one of the defect images from the dataset, thus generally achieving good KID scores but relatively low clustered LPIPS. On the other hand, CDC, originally tested on human face datasets, suffers from the close appearances of the defect-free images, hence it cannot render defect images with much variations either. Without much improvement to the diversity of the augmented dataset, these methods fail to provide helpful information for the downstream task in the next section.

On the contrary, SDGAN and Defect-GAN are able to ensure their diversity to a certain degree by generating defect images based on the relatively more defect-free data.

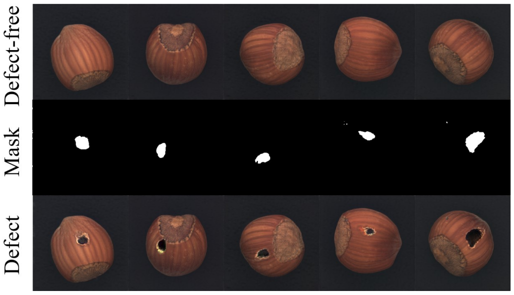

Figure 4: Examples of the triplets of generated defect-free image, mask and defect image. Our DFMGAN can render paired defect/defect-free images with difference only in the defect regions precisely delimited by the pixel-level masks.

However, we suppose that the critical flaw in the designs of these two methods is their image-to-image paradigm. They still have to guarantee the quality of the non-defect areas while rendering defects simultaneously, which is a harder task than focusing only on the defects explicitly delimited by the generated defect masks. Hence, the generated samples by SDGAN or Defect-GAN in Figure 3 expose quality deterioration to the objects. As Table 1 shows, these two methods receive worse KID, and the generated images are far from being realistic to help the downstream tasks.

For the non-generative method Crop&Paste, it can only render a finite number of defect images. For a dataset including $\boldsymbol{N}_g$ defect-free images and $\boldsymbol{N}_d$ defect images, Crop&Paste is unable to generate more than $\boldsymbol{N} = \boldsymbol{N}_g \times \boldsymbol{N}_d$ samples. Therefore, though it achieves rather high clustered LPIPS scores in Table 1 since $\boldsymbol{N} > 1000$ for the defect categories of hazelnut, it can be foreseen that the diversity will worsen when we require more than $\boldsymbol{N}$ images. Besides, as in Figure 3, defects produced by Crop&Paste sometimes (partially) locate outside the object because the generation process of this method violates the fact that defects are *co-decided* by the objects and external factors.

We have designed our DFMGAN aiming to settle the aforementioned issues. We train our model on defect-free images in the first training stage to capture the distribution of the object category, which is beneficial to be transferred to the defect categories. In the second stage, our model is forced to learn to add realistic defects on the features of various defect-free samples learned in the first stage, preventing overfitting and enhancing the diversity of the generated defect images. In addition, DFMGAN can merely focus on the defect regions since the non-defect areas are determined by z_{object} and the fixed backbone, which keeps the overall reality of our generated defect samples. In conclusion, with our specially designed architecture, DFMGAN can handle the challenging few-shot defect image generation cases even on objects with complex structural information and high variation, and outperform previous methods by a large margin.

Few-shot Generation To check the performance of DFMGAN in extreme cases with even fewer defect samples, we further challenge our model on 5-shot/1-shot defect image generation. We leave this part to *supplementary material.*

Method	P1 Acc↑	P2 Acc↑	P3 Acc↑
Finetune	70.83	72.91	70.83
DiffAug	64.58	62.50	68.75
CDC	58.33	64.58	41.67
Crop&Paste	66.67	52.08	58.33
SDGAN	56.25	31.25	43.75
Defect-GAN	60.42	68.75	54.17
DFMGAN	**83.33**	**81.25**	**81.25**

Table 2: The results of the defect classification experiments for the object category *hazelnut*. The training images generated by DFMGAN achieve the best accuracies on classifying unseen defect samples in all three partitions P1–3.

4.3 Data Augmentation for Defect Classification

Defect classification is a fundamental defect inspection task recognizing different types of defects of one object category. Since none of the baseline methods (except Crop&Paste) renders clear masks showing the defect regions, comparison on mask-requiring tasks such as defect localization is impossible. Thus we choose to test DFMGAN on defect classification which does not require masks. Nevertheless, DFMGAN can serve as a baseline in other tasks for future works.

For these experiments, we randomly choose 1/3 of the dataset images from each defect category as the base sets, and the other 2/3 from each category are combined as the test set. As for the hazelnut category, each base set has five or six images, and the test set consists of 12 images from each defect category, 48 in total. We train the methods on the four base sets corresponding to the four defect categories. Each method generates 1,000 images for each defect category and combines them as a whole training set with 4,000 images. Finally, for each method, we train a ResNet-34 (He et al. 2016) on its own training set and evaluate on the shared test set. We repeat these experiments three times with different partitions of base sets and test sets to avoid bias.

The accuracies on the test set are shown in Table 2. In this classification task simulating real-world defect inspection in industries, DFMGAN achieves the highest scores on all the partitions, with generally 10% improvement than the runner-up. It means that our method serves as the most informative data augmentation technique for the downstream task.

5 Conclusion

In this work, we propose the first few-shot defect image generation method DFMGAN which is capable of generating diverse defect images with high quality based on just a handful of defect samples. DFMGAN features its defect-aware residual blocks, which learn to produce reasonable defect masks and accordingly manipulate the object features. The highlight advantage is that it eases the transfer process from defect-free data to defect ones by delimiting the manipulation within the defect regions to focus solely on generating defects. Experiments on MVTec AD have proved the strong generation abilities of our method, as well as its benefits for downstream defect inspection tasks. We will discuss possible future works in *supplementary material.*

Acknowledgements

The work was supported by the National Science Foundation of China (62076162), and the Shanghai Municipal Science and Technology Major/Key Project, China (2021SHZDZX0102, 20511100300).

References

Baur, C.; Wiestler, B.; Albarqouni, S.; and Navab, N. 2019. Deep Autoencoding Models for Unsupervised Anomaly Segmentation in Brain MR Images. In Crimi, A.; Bakas, S.; Kuijf, H.; Keyvan, F.; Reyes, M.; and van Walsum, T., eds., *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, 161–169. Cham: Springer International Publishing. ISBN 978-3-030-11723-8.

Bergmann, P.; Fauser, M.; Sattlegger, D.; and Steger, C. 2019. MVTec AD – A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *CVPR*.

Bergmann, P.; Fauser, M.; Sattlegger, D.; and Steger, C. 2020. Uninformed Students: Student-Teacher Anomaly Detection With Discriminative Latent Embeddings. In *CVPR*.

Bińkowski, M.; Sutherland, D. J.; Arbel, M.; and Gretton, A. 2021. Demystifying MMD GANs. In *ICLR*.

Choi, Y.; Uh, Y.; Yoo, J.; and Ha, J.-W. 2020. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *CVPR*.

DeVries, T.; and Taylor, G. W. 2017. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv preprint arXiv:1708.04552*.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. Deep Learning. http://www.deeplearningbook.org. Accessed: 2023-03-10.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. In *NeurIPS*.

Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; and Courville, A. 2017. Improved Training of Wasserstein GANs. *arXiv preprint arXiv:1704.00028*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

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 *NeurIPS*.

Karras, T.; Aittala, M.; Hellsten, J.; Laine, S.; Lehtinen, J.; and Aila, T. 2020a. Training Generative Adversarial Networks with Limited Data. In *NeurIPS*.

Karras, T.; Laine, S.; and Aila, T. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020b. Analyzing and Improving the Image Quality of StyleGAN. In *CVPR*.

Li, C.-L.; Sohn, K.; Yoon, J.; and Pfister, T. 2021. CutPaste: Self-Supervised Learning for Anomaly Detection and Localization. In *CVPR*.

Lin, D.; Cao, Y.; Zhu, W.; and Li, Y. 2021. Few-Shot Defect Segmentation Leveraging Abundant Defect-Free Training Samples Through Normal Background Regularization And Crop-And-Paste Operation. In *ICME*.

Mao, Q.; Lee, H.-Y.; Tseng, H.-Y.; Ma, S.; and Yang, M.-H. 2019. Mode Seeking Generative Adversarial Networks for Diverse Image Synthesis. *arXiv preprint arXiv:1903.05628*.

Niu, S.; Li, B.; Wang, X.; and Lin, H. 2020. Defect Image Sample Generation With GAN for Improving Defect Recognition. *IEEE Transactions on Automation Science and Engineering*, 17(3): 1611–1622.

Noguchi, A.; and Harada, T. 2019. Image Generation From Small Datasets via Batch Statistics Adaptation. In *ICCV*.

Ojha, U.; Li, Y.; Lu, J.; Efros, A. A.; Lee, Y. J.; Shechtman, E.; and Zhang, R. 2021. Few-Shot Image Generation via Cross-Domain Correspondence. In *CVPR*.

Pang, G.; Shen, C.; Cao, L.; and Hengel, A. V. D. 2021. Deep Learning for Anomaly Detection: A Review. *ACM Computing Surveys*, 54(2).

Robb, E.; Chu, W.-S.; Kumar, A.; and Huang, J.-B. 2020. Few-Shot Adaptation of Generative Adversarial Networks. *arXiv preprint arXiv:2010.11943*.

Schlegl, T.; Seeböck, P.; Waldstein, S. M.; Schmidt-Erfurth, U.; and Langs, G. 2017. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In Niethammer, M.; Styner, M.; Aylward, S.; Zhu, H.; Oguz, I.; Yap, P.-T.; and Shen, D., eds., *Information Processing in Medical Imaging*, 146–157. Cham: Springer International Publishing. ISBN 978-3-319-59050-9.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *ICCV*.

Wang, Y.; Gonzalez-Garcia, A.; Berga, D.; Herranz, L.; Khan, F. S.; and Weijer, J. v. d. 2020. MineGAN: Effective Knowledge Transfer From GANs to Target Domains With Few Images. In *CVPR*.

Zhang, G.; Cui, K.; Hung, T.-Y.; and Lu, S. 2021. Defect-GAN: High-Fidelity Defect Synthesis for Automated Defect Inspection. In *WACV*.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*.

Zhao, M.; Cong, Y.; and Carin, L. 2020. On Leveraging Pretrained GANs for Limited-Data Generation. In *ICML*.

Zhao, S.; Liu, Z.; Lin, J.; Zhu, J.-Y.; and Han, S. 2020. Differentiable Augmentation for Data-Efficient GAN Training. In *NeurIPS*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networkss. In *ICCV*.