Distributed Anomaly Detection with Attention-guided Diffusion Models and Client-side Defect Generation

Pasquale Coscia, *Senior Member, IEEE,* Angelo Genovese, *Senior Member, IEEE,* Vincenzo Piuri, *Fellow, IEEE,* Konstantinos N. Plataniotis, *Fellow, IEEE,* and Fabio Scotti, *Senior Member, IEEE*

***Abstract*—Modern industrial systems are increasingly defined by geographically distributed production lines, stringent privacy constraints, particularly to protect intellectual property and manufacturing process details, and heterogeneous data pipelines. In such environments, centralized Anomaly Detection (AD) is often impractical due to data governance restrictions and limited computational resources at local sites. To address these challenges, we propose a modular and lightweight AD framework based on diffusion models, named D-ADDA (Distributed Anomaly Detection based on Data Augmentation), designed for distributed deployment. Unlike many state-of-the-art methods that depend on large pre-trained models or external datasets, our approach is trained entirely on defective data locally available, enhancing privacy and domain specificity. A novel Data Augmentation Module (DAM) generates diverse defective samples through a multi-stage pipeline, which are used to train an attention-based diffusion model for defect synthesis. This architecture supports dislocated components across multiple clients, enabling training and inference in resource-constrained or privacy-sensitive settings. Experimental results on the MVTec AD dataset confirm the effectiveness of our approach, achieving an average classification accuracy of 60.46% across 14 categories, outperforming state-of-the-art approaches, with competitive localization performance.**

***Index Terms*—Distributed anomaly detection, diffusion process, data augmentation, attention mechanism, defect synthesis.**

I. Introduction

INDUSTRIAL systems are rapidly evolving into complex, distributed infrastructures comprising geographically dispersed production sites, heterogeneous sensing systems, and tightly regulated data governance frameworks. Achieving reliable performance and high production quality in decentralized settings requires not only intelligent monitoring but also scalable and privacy-preserving solutions to protect details of the industrial process [1]. Localized intelligence at the point of data acquisition is therefore becoming increasingly critical, as it avoids transmitting sensitive information such as defect patterns or process information. In such contexts, practical constraints such as latency, limited network availability, and stringent requirements for preserving the confidentiality of sensitive information, particularly defective or anomalous samples protected by industrial secrecy or regulatory mandates, pose additional challenges to the design and deployment of Machine Learning (ML)-based solutions. These limitations undermine the viability of traditional centralized training and model deployment paradigms, which assume unrestricted access to comprehensive datasets.

Among industrial tasks, Anomaly Detection (AD) aims to detect deviations from expected patterns to identify system failures, operational inefficiencies, or defects [2]. AD is particularly challenging in manufacturing sites, especially for ML-based approaches, since operational data is abundantly available, while defective samples are scarce, highly sensitive, and often subject to strict confidentiality constraints [3], [4]. In such scenarios, collaborative strategies like distributed learning become essential, enabling different sites or devices to jointly train ML-based AD models without sharing raw data. However, when data types (*e.g.*, normal vs. defective) are unevenly distributed across nodes, this imbalance can lead to biased models, hinder convergence during training, and degrade overall detection performance, particularly for rare but critical anomalies. In fact, while deep learning models have shown promise in AD [5]–[7], they still face limitations related to data quality, training instability, domain shift, and generalization [8]–[10]. Alternative strategies like knowledge distillation, embedding-based methods [11], [12], and multimodal extensions [13] have attempted to address these issues but the reliance on scarce defective samples and pre-trained models often constrains performance.

In response to operational demands across modern industrial systems, we propose a modular architecture for image-based AD that supports deployment in environments with limited computational resources and strict privacy requirements, especially to protect confidential intellectual property and proprietary process information. Our method leverages generative diffusion models, trained locally on defective data augmented through a novel pipeline, and supports distributed execution across distinct clients or devices. Instead, a central server processes data that can be safely shared, keeping confidential and proprietary information local, thereby accelerating training and improving quality inspection (see Figure 1). More specifically, we present a three-stage approach built on a lightweight diffusion-based architecture (48.3 M parameters, compared to the 860 M parameters of Stable Diffusion v1.4 [14]), trained exclusively on collected data without relying on large-scale

P. Coscia, A. Genovese, V. Piuri, and F. Scotti are with the Department of Computer Science, Università degli Studi di Milano, 20133, Milan, Italy. (e-mail: pasquale.coscia@unimi.it, angelo.genovese@unimi.it, vincenzo.piuri@unimi.it, fabio.scotti@unimi.it).

K. N. Plataniotis is with the Department of Electrical and Computer Engineering, University of Toronto, M5S 3G4, Toronto, ON, Canada. (e-mail: kostas@ece.utoronto.ca).

This work was supported in part by the EC under projects EdgeAI (101097300) and GLACIATION (101070141), and by project SERICS (PE00000014) under the MUR NRRP funded by the EU - NGEU. Project EdgeAI is supported by the Chips Joint Undertaking and its members including top-up funding by Austria, Belgium, France, Greece, Italy, Latvia, Netherlands, and Norway under grant agreement No. 101097300. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union, the Chips Joint Undertaking, or the Italian MUR. Neither the European Union, nor the granting authority, nor Italian MUR can be held responsible for them.

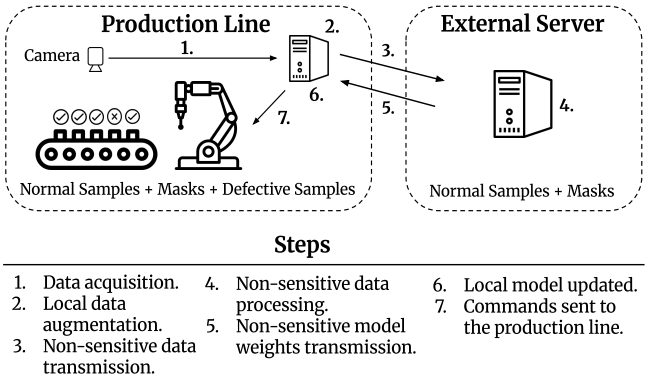


Fig. 1: System architecture considered in this work, showing local nodes (*e.g.*, production lines) connected to a central server for privacy-preserving anomaly detection. The numbered steps (1–7) indicate the operational workflow.

pre-trained models, thereby enhancing adaptability while addressing both privacy and resource constraints.

Our diffusion-based generative model, named D-ADDA (Distributed Anomaly Detection based on Data Augmentation), leverages a large set of defective samples created through a data augmentation pipeline, which simulates defects similar to those in the training data, thereby increasing the model's awareness of the defective domain. The defective samples are processed only at a local level, while the central server only handles non-sensitive data. Our framework prioritizes the protection of defective data, as these samples are most critical for AD and directly expose production failures. However, in industrial practice, even non-defective data may also be sensitive, since it can reveal proprietary or novel production processes. Thus, while the focus remains on defective data, sensitivity considerations can extend to both defective and non-defective samples depending on the confidentiality requirements of the application context. When local-only processing is possible and sufficient computational resources are available, running the full pipeline, including augmentation, generative synthesis, and classifier or detector training, for each domain or object type can still improve accuracy. Therefore, the system not only enables privacy-aware deployment, but also provides a performance-enhancing strategy.

Beyond its technical contributions to AD, D-ADDA also provides broader systems engineering benefits. By integrating diffusion-based generative modeling with attention-driven feature alignment in a distributed framework, D-ADDA provides a novel contribution to industrial image analysis and also directly supports industrial system-level objectives. In particular, its lightweight design and privacy-preserving operation enhance fault tolerance, reduce downtime, and improve adaptability of monitoring infrastructures under real-world constraints. Moreover, our approach demonstrates adaptability to different categories, making it suitable across diverse industrial contexts.

Our main contributions are as follows:

- A modular architecture that allows the core components of the AD pipeline, such as data augmentation, synthesis, detection, localization, and classification to operate independently across different locations. This supports incremental deployment across new production sites and facilitates scalability and maintainability in distributed environments.
- A decentralized anomaly detection approach that enables efficient, scalable, and privacy-preserving infrastructures by using centralized resources only for training on normal samples or masks, without privacy leaks.
- We outperform similar architectures on the MVTec AD dataset, demonstrating the effectiveness of the proposed solution.

The remainder of this paper is organized as follows. Section II reviews prior research on anomaly detection and generative approaches. Section III provides a detailed description of each stage of the proposed system architecture. Section IV reports extensive experimental evaluations, including ablation studies on key parameters of the framework. Finally, Section V summarizes the main contributions of the work and outlines potential directions for future research.

II. Related Work

Anomaly Detection. AD deals with discovering deviations from expected patterns and generally encompasses classification, detection, and localization tasks, with the main challenge being the scarce availability of data describing anomalous situations. To overcome this issue, unsupervised methods address detection and localization by relying solely on normal samples, using either features extracted from deep neural networks or reconstruction methods such as autoencoders, diffusion models, and image inpainting [15], [16].Within unsupervised learning, recent advancements have shown promising results using zero-shot [17], [18] and few-shot approaches [19], [20]. For example, MAEDAY [21] uses a reconstruction-based approach with a pre-trained masked autoencoder to identify anomalous regions as areas of failed reconstruction. CLIP [22] also describes an unsupervised approach that provides strong generalization capabilities through its joint alignment of textual and visual features. Building on this foundation, CLIP-based approaches aim to further enhance this alignment for anomaly detection [23], [24].

To deal with privacy concerns, Chen *et al*. [25] propose a decentralized approach to mitigate overgeneralization in unsupervised reconstruction; nevertheless, the method does not fully eliminate the necessity of exchanging sensitive information, including defect-related data. Federated learning techniques have also been proposed to mitigate data-sharing concerns [25], [26], but they introduce communication overhead and potential privacy risks [27]. In fact, although a number of approaches have advanced privacy-preserving anomaly detection, each presents inherent inefficiencies that limit their practical applicability to the industrial domain. For example, lattice-based frameworks [28] introduce considerable computational overhead arising from fine-grained feature-level policy negotiation, while edge-computing solutions [29] rely on lightweight encryption to protect data but still incur substantial latency and communication overhead, rendering them

unsuitable for latency-sensitive IoT environments. Similarly, the bi-level federated learning framework [30] enhances data privacy and scalability; however, it suffers from repeated transmission of encrypted model updates and continued dependence on manual labeling of anomalies in the second training stage, both of which hinder efficiency and scalability in industrial deployments. In this regard, our framework maintains privacy and removes the dependency on data encryption by focusing on the exchange of only non-sensitive data. In fact, sensitive samples are processed locally, achieving an efficient balance between collaboration and confidentiality.

Anomaly Generation. Given the scarcity of anomalous samples, generative models can help by creating additional data or capturing the distribution of normal (or abnormal) behavior. While model-free methods, such as cropping and pasting textures [31], can introduce synthetic anomalies, they often lack realism and fail to capture the subtle characteristics of real defects. In contrast, model-based approaches [6] produce more lifelike anomalies that better reflect real-world distributions. From an architectural standpoint, Generative Adversarial Networks (GANs) were among the first to be adopted for this task. For example, OCR-GAN [32] decomposes images into multiple frequency bands and reconstructs them in parallel, modeling the AD process as a form of omni-frequency restoration. However, despite their early success, GANs suffer from well-known issues such as training instability [9].

More recently, Denoising Diffusion Probabilistic Models (DDPMs) have emerged as a promising alternative, demonstrating impressive capabilities in generating high-quality images across diverse domains [10], [33]. Innovations like compressed latent spaces [14] and spatial conditioning mechanisms [34] have further enhanced their expressiveness and ability to model complex data distributions. In this regard, AnomalyDiffusion (AnomDiff) [7] leverages latent DDPMs to generate high-fidelity synthetic anomalies. By including dual encoders to disentangle defect appearance and location, along with an attention-based module to highlight subtle anomalies, AnomDiff improves the alignment between generated masks and image content, although challenges remain in fully replicating real defective images. Similarly, DRÆM [35] generates pseudo-anomalous images to train a U-Net-based [36] reconstruction framework, which learns to map abnormal inputs back to their normal counterparts. A discriminator further refines this process by distinguishing between pseudo-anomalous and reconstructed outputs, aiding in the precise localization of defects. UniAD [37], on the other hand, employs Transformers [38] with masked self-attention to reconstruct features while introducing a feature jittering strategy to prevent shortcut learning. To combine the advantages of generating both text and images, multi-modal approaches offer improved alignment between supervisory signals and enable better discrimination between visually similar defects, even if these methods often entail higher computational costs. For instance, Lee and Choi [13] propose a text-guided AD framework that aligns semantic priors from text with normal image representations, thereby enhancing the selection of discriminative samples. Nonetheless, such methods typically depend on carefully engineered prompts and pre-trained models, which can restrict adaptability and generalizability.

In contrast to existing approaches, our method does not rely on pre-trained architectures, thus allowing greater flexibility in the design of custom models. To mitigate data scarcity, we first apply an advanced data augmentation pipeline to enrich the available defective samples. We then employ a DDPM equipped with a cross-attention mechanism to synthesize novel image-mask pairs, effectively expanding the dataset and improving downstream AD performance.

III. System Architecture

The proposed D-ADDA framework is organized into three stages (see Figure 2), each comprising multiple interacting subsystems that collaboratively overcome the limitations of AD in industrial production lines, particularly in cases where defective training samples are both limited and sensitive. We structure the stages as follows:

- Stage 1: Data acquisition and augmentation system. This system collects data from the production lines on local nodes and performs data augmentation.
- Stage 2: Generative modeling system. This system trains two generative models and distributes non-sensitive data.
- Stage 3: Cross-conditional synthesis and detection system. This system performs defect synthesis on sensitive data and AD tasks.

In the following, we provide a detailed description of each stage and its constituent subsystems.

A. Stage 1: Data Acquisition and Augmentation System

Deployed at each local production line, this system uses low-cost imaging devices to acquire samples consisting of normal and defective products. To compensate for limited defective samples, it includes a Data Augmentation Module (DAM) that operates on the collected data. DAM uses image processing and superpixel-based techniques to transfer defect structures onto normal images, generating a large set of synthetic defective samples. The system involves the following steps:

- *Local Data Acquisition.* Each production node acquires N images using low-cost devices:
$$\mathcal{D}_{\text{local}} = \{(I_i, M_i, y_i)\}_{i=1}^{N}, \tag{1}$$
where $I_i \in \mathbb{R}^{H\times W\times 3}$ is the image, $M_i \in \mathbb{R}^{H\times W\times 1}$ is the corresponding mask, and $y_i \in \{0, 1\}$ is the defect label. H and W denote the image height and width, respectively.
- *Data Augmentation Module (DAM).* We apply a superpixel-based function $DAM(\cdot)$ to enrich the defective sample space:
$$\mathcal{D}_{\text{aug}} = \{(I'_i, M'_i, y_i)\}_{i=1}^{N}, \tag{2}$$
where $(I'_i, M'_i) = DAM(I_i, M_i)$.
- *Transmission of Non-Sensitive Data.* Only synthetic samples without sensitive content (*e.g.*, normal samples and masks) are transmitted to the central server:
$$\mathcal{D}_{\text{server}} = \{(I'_i, y_i) \mid y_i = 0\} \cup \{(M'_i, y_i) \mid y_i = 1\}. \tag{3}$$

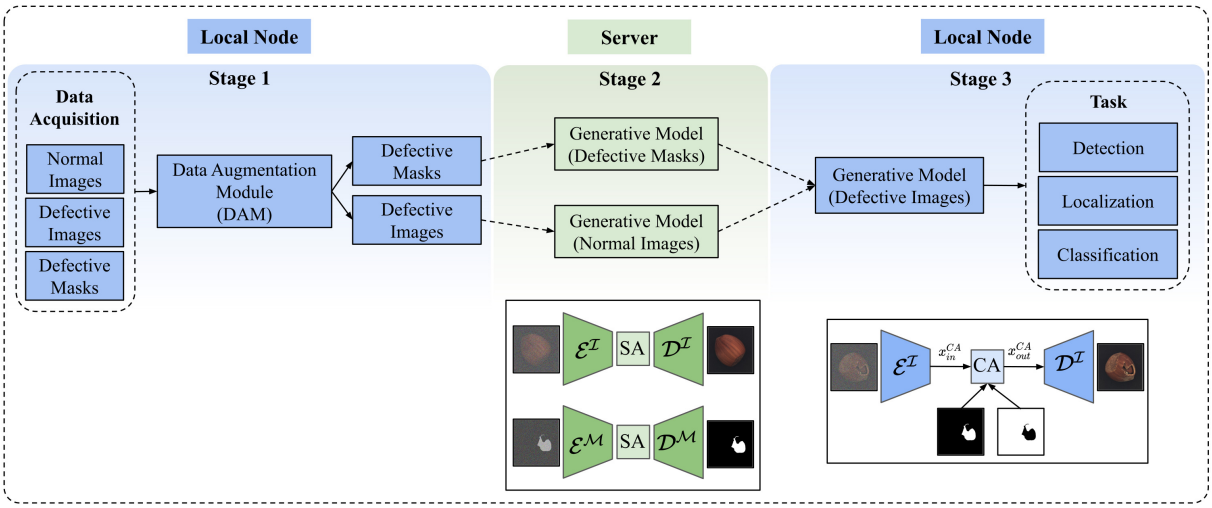


Fig. 2: Our D-ADDA framework integrates local augmentation, centralized diffusion modeling, and distributed model updates.

Algorithm 1 Data Augmentation Module (DAM) Pipeline

Require: n_0: Number of samples to generate

procedure DATAAUGMENTATIONMODULE($\mathrm{D^N}$, $\mathrm{D^D}$, $\mathrm{D^M}$) ▷ $\mathrm{D^N}$: Normal samples, $\mathrm{D^D}$: Defective samples, $\mathrm{D^M}$: Defective masks

  Randomly select a normal sample I_i^N, a defective sample I_j^D and a mask M_j^D.

  $SP_i \leftarrow \text{EXTRACTSUPERPIXELS}(I_i^N)$

  for $k = 1$ to n_0 **do** ▷ Iterate for each new sample

    $I_j^{D_{\text{aligned}}} \leftarrow \text{ALIGNIMAGES}(I_i^N, I_j^D)$ ▷ Using cross-correlation

    $M_j^{D_{\text{aligned}}} \leftarrow \text{ALIGNMASK}(M_j^D, I_j^{D_{\text{aligned}}})$

    $M_j^{D_A} \leftarrow \text{REPLACESUPERPIXELS}(M_j^{D_{\text{aligned}}}, SP_i)$

    $M_j^{D_A^*} \leftarrow \text{AUGMENTMASK}(M_j^{D_A})$

    $\text{MATCHPATCHHISTOGRAMS}(I_i^N, I_j^{D_{\text{aligned}}}, M_j^{D_A^*})$

    $I_j^{D^*} \leftarrow \text{REPLACEPATCH}(I_i^N, I_j^{D_{\text{aligned}}}, M_j^{D_A^*})$ ▷ Generate novel defect

    if $\text{QUALITYASSESSMENT}(I_j^{D^*})$ **then**

      Save $I_j^{D^*}$ and $M_j^{D_A^*}$ in $\mathcal{D}_{\text{aug}}$

    end if

  end for

  return $\mathcal{D}_{\text{aug}}$ ▷ Return novel defective dataset

end procedure

The DAM consists of a multi-step pipeline, outlined in Algorithm 1.

The key stages are detailed below, where I^D, M^D and I^N, M^N denote defective and normal images and masks, respectively:

- *Superpixel Extraction*. Initially, the quick shift algorithm [39] is employed to extract superpixels from a randomly selected normal sample I_i^N based on color space. These superpixels serve as potential patches for defect replacement. Quick Shift is a *mode-seeking* clustering approach in which each data point is linked to a local maximum or dense region in the underlying probability distribution. This method progressively moves each pixel toward its nearest neighbor with greater density, forming compact, consistent regions that conform to object contours and retain critical structural information.
- *Alignment Procedure*. A cross-correlation-based alignment procedure is used to align a defective training image I_j^D with the normal sample I_i^N using a center cropped patch obtained from the defective image.
- *Mask Creation and Augmentation*. Subsequently, the dot product between the defective mask M^D and the superpixels is computed to identify defective superpixels which are then augmented by incorporating adjacent superpixels, thereby defining a larger defective area. To introduce variability, a random target superpixel and a specified percentage of the nearest superpixels $p \sim \mathcal{U}(p_1, p_2)$ are selected to form a new defective mask, which is then subjected to smoothing. Patches corresponding to this new mask are extracted from both normal and defective samples and normalized using histogram equalization.
- *Defective Sample Generation*. Previous generated defective patch is superimposed onto the normal sample to create a new defective sample I_j^{D*}.
- *Data Quality Assessment (DQA)*. To assess the quality of the generated samples, we define a data quality assessment procedure. Specifically, we use a similarity function $S(I_A, I_B)$ to determine if two images are similar. In our work, the Structural Similarity Index (SSIM) score is employed as follows:

$$\frac{1}{m}\sum_{i=1}^{m} S(I_j^{D*}, I_i^D) \geq \frac{1}{n}\sum_{i=1}^{n} S(I_j^{D*}, I_i^N). \tag{4}$$

Here, m and n denote the number of defective and normal samples, for a category (or class) $\mathcal{C}$, respectively. In other words, if a novel defective image I_j^{D*} lies closer to the training defective images than to the normal images, we retain it along with its corresponding mask. Otherwise, we generate a new defective mask and repeat the process

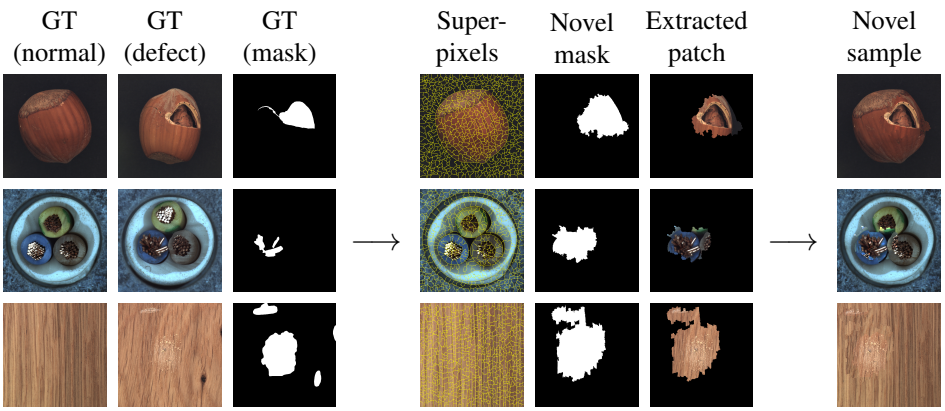


Fig. 3: Main steps of the local data augmentation pipeline for three categories: hazelnut (crack), cable (bent wire), and wood (scratch). GT stands for ground truth. Novel indicates newly generated synthetic samples that are distinct from the training set.

for a fixed number of steps, after which we include the image in the augmented set.

- *Textures*. Unlike objects, texture patches do not undergo the alignment phase, as almost each location achieves a maximum cross-correlation value without requiring alignment. For textures, we introduce additional randomness by randomly shifting the augmented mask in both the x and y directions.

A visualization of the main above steps and final results are shown in Figure 3.

B. Stage 2: Generative Modeling System

Housed on a central server, it is responsible for training generative diffusion models. It interacts with the local systems by receiving the augmented datasets without sensitive content generated in Stage 1. In this way, no sensitive information is exchanged between local and central nodes. In Stage 2, we train two generative models: an unconditional model $\mathcal{G}^{\mathcal{I}}$ trained on normal images I^N to capture the intrinsic characteristics of the category $\mathcal{C}$, and a conditional model $\mathcal{G}^{\mathcal{M}}$ trained on defective masks M^D to learn plausible defect structures. To train $\mathcal{G}^{\mathcal{M}}$, we use a classifier-free guidance to condition the architecture on specific defect classes [40]. The system involves the following steps:

- *Generative Model Training.* The central server trains two generative models on the received non-sensitive data:

  $\mathcal{G}^{\mathcal{I}}(\theta_I)$: unconditional model on normal images I^N;
  $\mathcal{G}^{\mathcal{M}}(\theta_M)$: conditional model on defective masks M^D.

  θ_I and θ_M represent the parameters of the models.
- *Distribution of Non-Sensitive Weights.* The learned model parameters, θ_I and θ_M, are transmitted back to the local nodes for downstream generation.

More in detail, $\mathcal{G}^{\mathcal{I}}$ and $\mathcal{G}^{\mathcal{M}}$ represent two generative models, which aim to learn the true distribution of observed samples by representing uncertainty through latent variables. The models synthesize novel normal samples and masks, with $\mathcal{G}^{\mathcal{I}}$ used in a pre-training phase to capture domain characteristics and $\mathcal{G}^{\mathcal{M}}$ used to generate novel masks. In particular, we consider DDPMs as the architecture for generative models, since diffusion-based models have shown excellent performance and flexibility across various applications [10].

In our work, we let $\mathcal{G} = (\mathcal{E}, \mathcal{D})$ represent a diffusion process [14] with an encoder-decoder structure following the U-Net architecture [36]. We use two separate models, $\mathcal{G}^{\mathcal{I}} = (\mathcal{E}^{\mathcal{I}}, \mathcal{D}^{\mathcal{I}})$ and $\mathcal{G}^{\mathcal{M}} = (\mathcal{E}^{\mathcal{M}}, \mathcal{D}^{\mathcal{M}})$, to generate normal images and masks, respectively.

Differently than the standard U-Net, we connect the encoder and decoder via a Self-Attention (SA) module, equipped with multiple multi-head attention mechanisms to focus on specific areas of the data, allowing the models to learn diverse representations and enhance the quality and variety of the generated images. Specifically, given a batch of image features at layer L with height H and width W, denoted as $\boldsymbol{x}_t^l \in \mathbb{R}^{B\times C\times H\times W}$, where B and C represent the batch size and number of channels, respectively, we define an attention function $\text{Attention}(X_t^k, X_t^q, X_t^v)$. Here, X_t^k, X_t^q, and X_t^v correspond to the key, query, and value matrices used in our attention mechanism to compute attention at time step t. The input X_t^l is projected into Q_t^h, K_t^h, and V_t^h for each attention head h, with dimensionality d_h, and passed to the attention function as follows:

$$Q_t^h = X_t^l W_Q^h, \quad K_t^h = X_t^l W_K^h, \tag{5}$$

$$A_t^h = \text{softmax}\left(\frac{Q_t^h (K_t^h)^\top}{\sqrt{d}}\right), \tag{6}$$

where W_Q^h and $W_K^h \in \mathbb{R}^{d_h\times d}$ for $h = 0, 1, \ldots, N-1$, and d is the projected dimension of our attention mechanism. Then, attention coefficients A_t^h are right multiplied by $V_t^h = X_t^l W_V^h$, where $W_V^h \in \mathbb{R}^{d_h\times d}$. We apply the SA module in both the intermediate and bottleneck layers of our U-Net while training $\mathcal{G}^{\mathcal{I}}$ and $\mathcal{G}^{\mathcal{M}}$.

After training, we obtain $\mathcal{G}^{\mathcal{I}}$ pre-trained on normal samples to capture the primary features of the category $\mathcal{C}$ and $\mathcal{G}^{\mathcal{M}}$ trained on augmented masks to generate novel defective masks.

C. Stage 3: Cross-Conditional Synthesis and Detection System

This system is local and integrates the outputs of the two previously defined diffusion models to synthesize realistic

defective images. Then, the generated images are used to train the classification and detection/localization model. Specifically, $\mathcal{G}^{\mathcal{I}}$ is modified to be conditioned on synthetic masks sampled from $\mathcal{G}^{\mathcal{M}}$, using self- and cross-attention mechanisms to produce novel defective samples that reflect the distribution of real-world anomalies. The system involves the following steps:

- *Local Defect Synthesis.* The local system fine tunes the generative model $\mathcal{G}^{\mathcal{I}}$, pretrained in Stage 2, to synthesize novel defects employing self- and cross-attention mechanisms:

$$z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_2),\ M_i = \mathcal{G}^{\mathcal{M}}(z),\ I_i = \mathcal{G}^{\mathcal{I}}(z \mid M_i), \quad (7)$$

- *Defect Classification and Detection/Localization.* We process the synthesized and acquired samples for downstream tasks:

$$\hat{y}_i = F_{\text{cls}}(I_i), \quad \hat{M}_i = F_{\text{det/loc}}(I_i, M_i), \quad (8)$$

where F_{cls} and $F_{\text{det/loc}}$ denote the classification and localization/detection networks, respectively.

Once the local nodes receive the weights θ_M and θ_I of both models trained on the central server, they first synthesize novel masks using the generative model $\mathcal{G}^{\mathcal{M}}$ and then fine-tune $\mathcal{G}^{\mathcal{I}}$ with a Cross-Attention (CA) mechanism to condition the network on the defective masks. In this stage, we first process the input features $\boldsymbol{x}_{in}^{CA}$, together with the conditioning mask M^D and its complement M^{1-D} representing normal regions, through two separate convolutional-based mask embeddings $ME(\cdot)$. We then extract queries, keys, and values for M^D and M^{1-D}, respectively. We adopt a gating-based strategy to condition our model on input masks using two cross-attention layers, one self-attention layer and a residual connection as follows:

$$\begin{aligned} \mathbf{G}_D &= \sigma(\text{GAP}(\text{Attention}(Q_t, K_t^D, V_t^D))), \\ \mathbf{G}_{1-D} &= \sigma(\text{GAP}(\text{Attention}(Q_t, K_t^{1-D}, V_t^{1-D}))), \\ \boldsymbol{x}_{SA} &= \text{Attention}(Q_t, K_t, V_t). \end{aligned} \quad (9)$$

Q_t, K_t, V_t are obtained as projections of $\boldsymbol{x}_{in}^{CA}$, K_t^D and V_t^D are projected from $ME(M^D)$ while K_t^{1-D} and V_t^{1-D} from $ME(M^{1-D})$. GAP denotes a global average pooling layer while σ a sigmoid activation function. Finally, our output $\boldsymbol{x}_{out}^{CA}$ is computed as a linear combination of the original image features $\boldsymbol{x}_{in}^{CA}$, the self-attention output $\boldsymbol{x}_{SA}$, and the gating terms $\mathbf{G}_D$ and $\mathbf{G}_{1-D}$:

$$\boldsymbol{x}_{out}^{CA} = \underbrace{\mathbf{G}_D \odot (\boldsymbol{x}_{SA} + \boldsymbol{x}_{in}^{CA})}_{\text{Defective Gate}} + \underbrace{\mathbf{G}_{1-D} \odot (\boldsymbol{x}_{SA} + \boldsymbol{x}_{in}^{CA})}_{\text{Normal Gate}}, \quad (10)$$

where $\odot$ represents the element-wise multiplication.

The proposed stage encourages the model to focus on both defective and non-defective regions while the gating mechanism separates their representations. A residual connection ensures better gradient flow. For added supervision, we project the defective mask into a similar space as the time projection and concatenate both embeddings into the U-Net's downsampling and upsampling blocks.

IV. Experimental Results

In the following, we present the details of the dataset used in our study, the model training parameters, and the evaluation protocols employed. We then describe the comparative methods, followed by the quantitative and qualitative results for both classification and localization/detection tasks. Finally, we report on distributed experiments and ablation studies that examine the impact of key hyperparameters in the proposed framework.

A. Experimental Setting

Dataset. For our experiments, we use a well-established AD dataset, namely MVTec AD [41], which comprises 5354 high-resolution images across 15 categories (10 objects and 5 textures). We adhere to the standard evaluation protocol, which involves using one-third of the anomalous data as the training set, with the remainder reserved for testing. Additionally, we generate 500 samples per defect using our data augmentation module, with an image resolution of 128×128 pixels, and train the diffusion processes to produce 500 image/mask pairs per defect, resulting in a total of $36\,500$ image/mask pairs. Furthermore, we perform a quality check on randomly generated images over a specific number of iterations, *i.e.*, 100. To prevent getting stuck in the quality inspection step, the last generated image is automatically added to the augmented dataset if none of the synthesized images pass the check.

Model Training. We use 1000 diffusion steps and train the denoising U-Net for $30\,000$ steps with a learning rate of 10^{-4}. As warm-up phase, we consider 5000 steps. Our mask embedding module comprises three Conv2D-ReLU-MaxPooling blocks, followed by an additional Conv2D layer to match the dimension of $\boldsymbol{x}_{in}^{CA}$. We use 4 heads for all the attention mechanisms and set d_h to 32. Additionally, we apply a weighted exponential moving average with a weight decay of 0.995. The entire framework is trained using the Adam optimizer on a single NVIDIA RTX-6000 GPU. In our data augmentation module, we set p_1 and p_2 to $0.35/0.65$ and $0.65/0.85$ for textures/objects, respectively.

Evaluation Protocol. To evaluate the quality of our synthesized images and masks, we focus on standard AD tasks: classification and detection/localization [7]. For classification, we consider each category in the MVTec AD dataset and perform a classification task among its defective classes. To assess the effectiveness of the generated images, we use a ResNet-based model to classify defective samples. Specifically, classification performance is measured as the proportion of correctly classified instances among all test instances across the N_d defect classes within a given category $\mathcal{C}$. For detection and localization, the objective is to identify defective regions at both the image level and the pixel level. Detection focuses on a global assessment, determining whether an image is anomalous from the anomaly mask, while localization evaluates how well the predicted anomaly mask aligns with the actual defective regions across all pixels. A single U-Net architecture is employed for both tasks, trained using a dataset comprising normal and defective samples.

Performance in detection and localization is evaluated using three standard metrics: AUROC (Area Under the Receiver

TABLE I: CLASSIFICATION PERFORMANCE (%)

Category	*U-Pro*									*F-Pro*		
	DiffAug	**CDC**	**Crop&Paste**	**SDGAN**	**Defect-GAN**	**DFMGAN**	**DAM**	**D-ADDA (NoP)**	**D-ADDA**	**DAM**	**D-ADDA (NoP)**	**D-ADDA**
bottle	48.84	38.76	52.71	48.84	53.49	56.59	72.09	79.07	**81.40**	67.44	65.12	**83.72**
cable	21.36	39.06	32.81	21.88	21.36	45.31	42.19	43.75	**46.88**	42.19	35.94	34.67
capsule	34.67	28.89	32.89	30.22	32.00	37.23	29.33	33.33	**44.00**	29.33	26.67	**31.25**
carpet	35.48	25.27	27.96	21.50	29.03	**47.31**	25.81	33.87	38.71	**35.48**	29.03	34.67
grid	28.33	35.83	28.33	30.83	27.50	**40.83**	20.00	27.50	32.50	12.50	17.50	**25.00**
hazelnut	65.28	54.86	59.03	43.75	61.11	81.94	62.50	95.83	**97.92**	64.58	**95.83**	93.75
leather	40.74	43.38	34.39	38.10	42.33	49.73	22.22	46.03	**71.43**	17.46	**58.73**	47.62
metal nut	58.85	48.44	59.89	44.27	56.77	64.58	**76.56**	67.19	73.44	59.38	64.06	**65.62**
pill	29.86	21.88	26.74	20.49	28.47	29.52	37.50	31.25	**40.62**	**38.54**	21.88	28.12
screw	25.10	32.92	28.81	26.75	28.81	37.45	35.80	40.74	**61.73**	23.46	23.46	**49.61**
tile	59.65	48.54	68.42	42.69	26.90	74.85	64.91	85.96	**94.74**	**75.44**	**75.44**	**75.44**
transistor	38.09	29.76	41.67	32.14	35.72	52.38	57.14	60.71	**64.29**	50.00	50.00	**57.14**
wood	41.27	28.57	47.62	30.95	24.60	49.21	47.62	38.10	**52.38**	**47.62**	30.95	37.71
zipper	22.76	14.63	26.42	21.54	18.70	27.64	25.61	**60.71**	46.34	31.71	56.10	**59.76**
Average	39.31	35.06	40.55	32.43	34.77	49.61	44.23	53.15	**60.46**	42.51	46.48	**51.72**

Notes. *U-Pro* = Unfair protocol; *F-Pro* = Fair protocol; D-ADDA (noP) = we test our framework also w/o the ImageNet pre-training.

Operating Characteristic curve), AP (Average Precision), and F_1-max. AUROC measures the trade-off between true positive rate and false positive rate across different thresholds. AP summarizes the precision-recall curve by computing a weighted average of precision at various recall levels. F_1-max refers to the maximum F_1 score achieved across all possible thresholds.

Since evaluations on the MVTec AD dataset are commonly performed using the test set [7], [35], we define two evaluation protocols to clarify the impact of tuning procedures. The first, named Unfair Protocol (*U-Pro*), uses the test set for fine-tuning model parameters, aligning with prior works. By contrast, we introduce a Fair Protocol (*F-Pro*), where performance is assessed using the model from the final training epoch, without test-time adaptation. We report results under these protocols for the classification task only, as detection and localization metrics showed limited sensitivity to the evaluation strategy.

Methods. For a comprehensive evaluation, we compare our D-ADDA framework with DiffAug [42], CDC [43], Crop&Paste [44], SDGAN [45], Defect-GAN [6], and DFM-GAN [5] on the classification task. For detection and localization, we evaluate D-ADDA against samples generated directly by our data augmentation module DAM, *i.e.*, without employing any generative architecture. For the qualitative comparison, we consider AnomDiff [7] and DRÆM [35].

B. Results

Classification Task. Table I reports the classification performance of our proposed framework under both unfair (*U-Pro*) and fair (*F-Pro*) evaluation protocols, including results without ImageNet pre-training (D-ADDA (NoP)). Under the *U-Pro* setting, D-ADDA achieves the highest accuracy in 10 out of 14 categories, reaching an overall average of 60.46%, which substantially outperforms both its unpretrained variant and all competing methods. The results highlight the strong contribution of ImageNet pre-training while also showing that the framework maintains competitive performance even without it. D-ADDA is also the best performing framework in particularly challenging categories such as hazelnut (97.92%), tile (94.74%), and bottle (81.40%), demonstrating robust generalization across diverse defect types and low-data regimes. Under the *F-Pro* protocol, which enforces fair evaluation conditions, D-ADDA again achieves the highest average accuracy, outperforming both DAM and D-ADDA (NoP). This indicates that our framework remains effective even when prior knowledge from large-scale datasets is removed, underscoring its ability to detect anomalies under realistic and unbiased evaluation settings. Additionally, D-ADDA maintains competitive results also in categories where prior methods had strong performance, such as metal nut, pill, and screw, reflecting the framework's versatility and robustness across a wide spectrum of defect types.

Detection/Localization Tasks. Table II presents a comparison between DAM and our D-ADDA framework on detection and localization AD tasks. Although it does not rely on any generative architecture, DAM outperforms most GAN-based methods and is therefore selected as the main competitor in this analysis. The evaluation metrics include AUC, AP, and F_1-max across 15 object categories. For detection, D-ADDA outperforms DAM across all average metrics: AUC improves from 76.71% to 82.05%, AP from 86.87% to 90.35%, and F_1-max from 83.59% to 84.93%. Additionally, significant improvements are observed in several categories. For example, in the wood category, AUC increases from 84.50% to 98.20%, AP from 93.40% to 99.30%, and F_1-max from 86.00% to 96.30%, showing a substantial enhancement in AD. Similarly, the grid category sees AUC rise from 84.90% to 91.20%, and F_1-max from 83.00% to 90.20%. The carpet category benefits from a jump in AUC from 66.60% to 77.40%, with AP also improving from 80.90% to 90.50%. In texture-dominated categories like leather, carpet, and wood, D-ADDA consistently scores higher, confirming improved generalization across both structured and unstructured object types.

Our framework also demonstrates clear advantages in accurately localizing defective regions. The average AUC increases from 79.83% to 81.63%, AP from 26.23% to 29.23%, and F_1-max from 28.75% to 31.91%. Particularly large improvements are observed in categories with subtle or small defects. For instance, in the carpet category, F_1-max improves from 18.10% to 30.20%, and in leather, it increases from 10.80% to 26.40%. The grid category, which poses challenges due to its regular repeating patterns, sees F_1-max increase from 3.30% to 21.50%, and AP from 2.00% to 11.30%. The wood

TABLE II: DETECTION/LOCALIZATION PERFORMANCE (%)

Category	Detection						Localization					
	DAM			D-ADDA			DAM			D-ADDA		
	AUC	AP	F_1-max	AUC	AP	F_1-max	AUC	AP	F_1-max	AUC	AP	F_1-max
bottle	75.50	87.90	**86.30**	**83.80**	**93.00**	84.80	57.00	13.60	**21.60**	**74.10**	**14.80**	15.00
cable	**70.20**	76.20	68.80	69.70	**77.60**	**69.20**	**88.20**	**30.90**	**35.50**	72.70	26.50	39.40
capsule	**66.60**	**85.40**	**87.20**	59.20	81.70	86.70	89.40	06.00	09.60	**91.90**	**06.30**	**09.80**
carpet	66.60	80.90	81.90	**77.40**	**90.50**	**82.10**	68.60	13.00	18.10	**88.50**	**28.20**	**30.20**
grid	84.90	92.20	83.00	**91.20**	**94.70**	**90.20**	55.20	02.00	03.30	**70.10**	**11.30**	**21.50**
hazelnut	**98.80**	**98.70**	**94.90**	92.00	95.10	87.60	**98.30**	**55.90**	**53.40**	93.10	41.90	44.90
leather	81.30	91.20	82.80	**85.80**	**93.40**	**85.30**	58.50	04.90	10.80	**77.50**	**19.30**	**26.40**
metal nut	61.90	85.40	85.90	**83.70**	**94.30**	**89.10**	92.30	76.20	**74.90**	**94.10**	**81.70**	69.20
pill	70.70	91.20	88.50	**76.60**	**92.80**	**89.60**	87.90	**58.20**	58.30	**88.60**	58.10	**60.00**
screw	80.00	**90.80**	82.40	**80.20**	90.60	**83.40**	**92.30**	**06.40**	**08.50**	77.20	01.70	04.40
tile	**96.10**	**97.70**	**91.90**	91.90	96.30	89.70	**80.50**	33.60	38.40	70.90	**52.20**	**54.00**
toothbrush	61.30	67.80	**83.30**	**70.80**	**81.80**	80.90	81.30	04.10	03.40	**86.10**	**06.30**	**09.30**
transistor	81.00	76.60	66.70	**87.80**	**80.20**	**73.80**	**93.30**	**51.60**	**50.70**	66.60	28.90	36.90
wood	84.50	93.40	86.00	**98.20**	**99.30**	**96.30**	70.90	22.50	22.10	**85.40**	**46.00**	**46.90**
zipper	71.20	87.70	84.20	**82.50**	**93.90**	**85.20**	83.80	14.60	**22.70**	**87.60**	**15.20**	10.70
Average	76.71	86.87	83.59	**82.05**	**90.35**	**84.93**	79.83	26.23	28.75	**81.63**	**29.23**	**31.91**

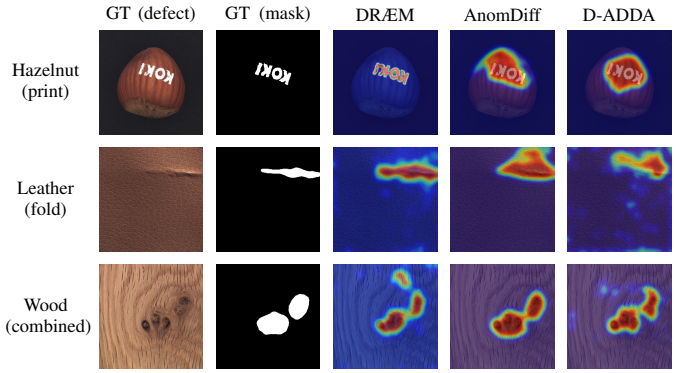


Fig. 4: Qualitative examples of localization performance for three categories. GT stands for ground-truth.

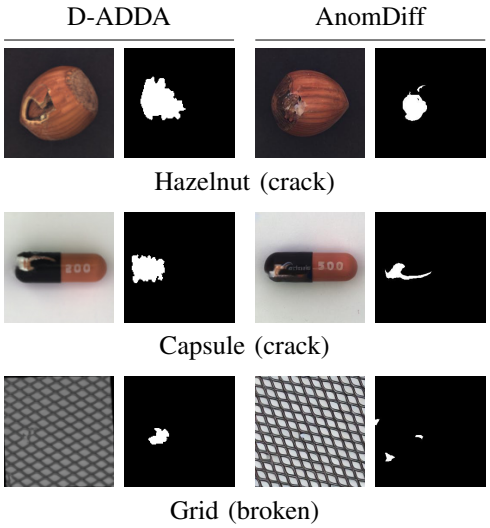


Fig. 5: Qualitative comparison between synthesized images by D-ADDA (left) and AnomDiff (right).

category also demonstrates strong gains: F_1-max improves from 22.10% to 46.90%, and AP from 22.50% to 46.00%. Even in low-performing cases for DAM such as toothbrush and screw, D-ADDA delivers consistent improvements across all three metrics. The results in Table II demonstrate that D-ADDA achieves robust improvements in both detection, which determines whether an object is anomalous, and localization, which identifies the specific anomalous regions. The gains are consistent across a wide variety of object geometries and textures, showing the method's effectiveness in both coarse and fine-grained AD scenarios.

Figure 4 provides qualitative examples for three categories considering centralized state-of-the-art generative methods. While DRÆM [35] excels at localizing the *print* defect on hazelnut objects, our approach competes closely with the pre-trained AnomDiff [7] model, accurately identifying the central defect area. Localization on leather is more difficult, with all methods struggling to achieve high precision. For wood texture, our model appears to outperform DRÆM, although AnomDiff shows better results, but this comes at the cost of an extensive pre-training.

Image Quality Inspection. Figure 5 shows a qualitative comparison between images generated by D-ADDA and a similar, yet centralized, diffusion-based one (*i.e.*, AnomDiff [7]). The synthesized images generated through our augmentation process enhance the diffusion-based learning, leading to more realistic defective images and masks in specific cases. For example, the hazelnut object shows defects that appear highly realistic, whereas reproducing defects for the capsule object proves more challenging. Nevertheless, we observe some misalignment in texture-based objects, as shown in Figure 6, which arises from the absence of an explicit alignment loss within our architecture. While this misalignment benefits certain classes, *e.g.*, wood or tile, due to the uniformity of their background, it negatively impacts localization metrics for grid patterns, where precision is more critical. Furthermore, certain generated images bear close resemblance to the training data, raising concerns of overfitting; this issue could be alleviated by diversifying the augmented samples, for example through the addition of local nodes.

Distributed Classification. To comprehensively evaluate

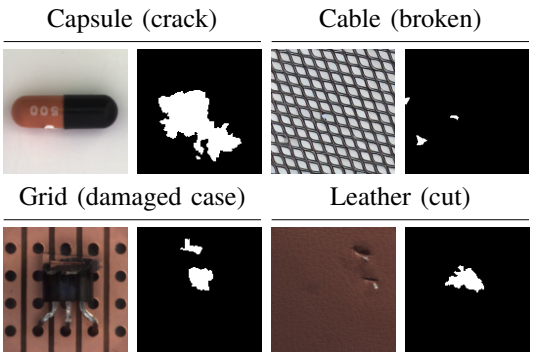


Fig. 6: Examples of misaligned image-mask pairs.

the performance of our proposed method in distributed learning scenarios, we conduct a series of experiments inspired by the FedAvg algorithm [46]. Distributed learning provides a well-established framework for simulating collaborative training across multiple clients, making it a suitable and widely adopted proxy for distributed industrial inspection systems. While decentralized learning approaches offer benefits such as improved fault tolerance and reduced dependence on a central server, the lack of publicly available implementations and standardized benchmarks limits fair and reproducible comparisons. Therefore, we assume that model weights are averaged across clients when performing the classification task.

As a baseline, we first consider the centralized scenario with a single client ($N_C = 1$). In this setup, all data are available to one client, representing an idealized reference where no inter-client variability exists. This allows us to isolate the effects of distributed training by comparison. Then, to extend the classification to a distributed learning context, we simulate multiple clients operating within each production line. For example, in a scenario where Manufacturer A and Manufacturer B produce distinct products, each manufacturer may include multiple inspection stations acting as clients. To simulate multiple clients, we consider two client data partitioning strategies:

- *Non-IID (Label-wise Partitioning)*. Client datasets are partitioned by defect type, so that each client has access only to samples of a single class. Given N_d defect types in a category $\mathcal{C}$, one client is assigned per defect. We optimize local models on class-specific data and aggregate their updates using the averaging algorithm. This setup simulates real-world inspection stations specialized for specific defect classes, resulting in highly skewed data distributions.
- *IID (Uniform Partitioning)*. The entire dataset is randomly shuffled and evenly divided across N_C clients, ensuring each client receives a representative sample of all classes. This serves as a baseline scenario, representing idealized homogeneous data distributions among clients.

For the non-IID configuration, the number of clients is set to N_d, corresponding to one client per defect class. For the IID configuration, experiments were conducted with $N_C = N_d$ and $N_C = 10$. We systematically vary the number of local training epochs ($E \in \{1, 5, 10\}$) and batch sizes ($B \in \{8, 16, 32\}$).

TABLE III: DISTRIBUTED CLASSIFICATION ACCURACY

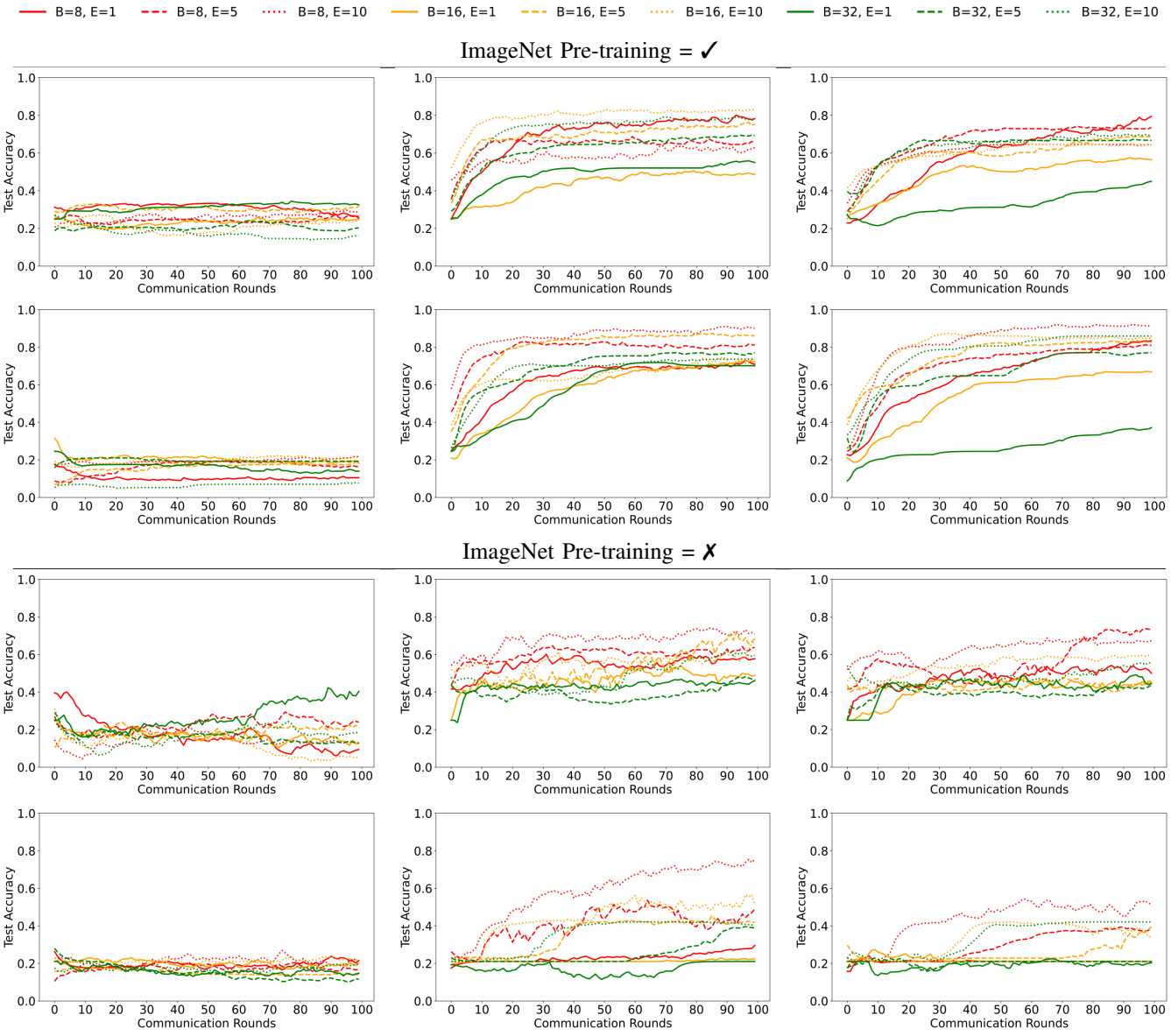
Distributed Parameters				Accuracy [%]			
				No Pre-training		Pre-training	
Num. Clients	B	E	IID	*U-Pro*	*F-Pro*	*U-Pro*	*F-Pro*
1	–	–	–	53.15	46.48	60.46	51.72
N_d	8	1	✗	**28.10**	18.21	26.12	21.04
N_d	16	1	✗	24.80	18.32	26.69	**22.60**
N_d	32	1	✗	25.76	21.79	**26.84**	20.17
N_d	8	5	✗	25.30	20.78	24.61	21.52
N_d	16	5	✗	25.68	20.94	24.38	22.11
N_d	32	5	✗	24.94	18.54	23.81	20.91
N_d	8	10	✗	27.49	21.35	23.84	20.25
N_d	16	10	✗	27.08	**22.10**	24.81	22.34
N_d	32	10	✗	25.47	20.45	24.33	20.12
N_d	8	1	✓	31.66	28.48	45.32	43.04
N_d	16	1	✓	27.64	24.74	40.24	38.61
N_d	32	1	✓	26.56	23.10	36.11	34.68
N_d	8	5	✓	37.62	33.27	48.20	44.68
N_d	16	5	✓	35.63	30.46	48.08	44.80
N_d	32	5	✓	32.38	29.72	42.72	40.63
N_d	8	10	✓	**39.79**	**35.30**	**49.99**	**47.21**
N_d	16	10	✓	35.94	31.59	47.00	44.23
N_d	32	10	✓	33.54	27.83	44.32	42.22
10	8	1	✓	27.43	25.13	41.07	38.84
10	16	1	✓	27.46	23.85	39.39	37.42
10	32	1	✓	25.70	21.31	31.73	28.28
10	8	5	✓	35.90	31.48	46.82	44.06
10	16	5	✓	32.41	28.79	**47.62**	**45.27**
10	32	5	✓	28.26	25.03	41.90	39.29
10	8	10	✓	**38.24**	**33.22**	45.69	43.55
10	16	10	✓	36.78	30.68	46.10	43.70
10	32	10	✓	31.31	27.85	45.35	42.92

Notes. B = local batch size; E = local number of epochs.

Table III reports the classification accuracy. In the centralized scenario ($N_C = 1$), the model has access to all data, yielding a reference point for performance without inter-client heterogeneity. In the non-IID regime, where each client receives samples from only a single defect class, accuracy remains consistently low (20–27%), indicating that this scenario struggles to reconcile highly divergent local updates. Adjustments of B or E produce minimal improvement: smaller batches fail to introduce sufficient gradient diversity, and additional local epochs increase divergence between client models.

In the IID regime, where each client receives a balanced subset of all defect classes, accuracy improves substantially. For $N_C = N_d$, $B = 8$, and $E = 10$, *U-Pro* reaches 49.99% and *F-Pro* reaches 47.21%. Performance generally benefits from more local epochs, particularly for smaller batch sizes, reflecting improved convergence when inter-client gradient variance is low. Larger batch sizes remain competitive but occasionally result in slightly lower accuracy due to reduced stochasticity. When N_C is fixed at 10, accuracy is slightly lower than the $N_C = N_d$ case, with the difference more pronounced at lower epoch counts.

The qualitative visualizations in Figure 7 for the hazelnut and tile categories confirm the quantitative trends observed in Table III. Under non-IID conditions, all batch sizes and epoch counts yield predictions with reduced confidence, reflecting the difficulty of generalizing from narrowly focused local models. Increasing the number of local epochs in this



a) $N_C = N_D$, IID = ✗ (b) $N_C = N_D$, IID = ✓ (c) $N_C = 10$, IID = ✓

Fig. 7: Distributed classification experiments on the hazelnut (1^{st} and 3^{rd} rows) and tile (2^{nd} and 4^{th} rows) categories.

regime does not enhance prediction quality; rather, it can increase misclassifications due to overfitting to local label distributions. In contrast, IID configurations produce more accurate results, with notable improvements as the number of local epochs increases. Smaller batch sizes in IID settings generally yield more reliable predictions, likely due to higher gradient diversity. When the number of clients is fixed at $N_C = 10$ in IID settings, outputs remain competitive but exhibit softer boundaries, particularly for larger batch sizes and lower numbers of local epochs, which aligns with the modest numerical decreases reported in Table III. The absence of ImageNet pre-training emphasizes the impact of data heterogeneity. Non-IID predictions show even lower confidence and less accurate defect localization compared to the pre-trained scenario. IID results without pre-training improve over the non-IID case but generally achieve lower test accuracies than their pre-trained counterparts. These observations indicate that classification pre-training can provide a beneficial initialization that stabilizes training across clients and enhances defect detection, especially under heterogeneous data distributions.

C. Ablation Studies

In the following, we present an analysis of different hyper-parameter settings within our framework.

Cross-Attention module. Table IV provides an ablation study of our Cross-Attention (CA) module applied to the hazelnut object across all tasks. The full module outperforms the other variants, achieving top accuracy and detection metrics. This confirms the efficacy of combining multiple gating-based attention mechanisms. Additionally, a partial CA model, based solely on the defective mask, shows better localization

TABLE IV: ABLATION STUDY ON THE CROSS-ATTENTION (CA) MODULE – HAZELNUT

CA Module	Class.	Detection			Localization		
	Acc. [%]	AUC	AP	F_1-max	AUC	AP	F_1-max
Norm. Gate	95.83	87.70	92.90	85.70	90.00	41.20	41.20
Def. Gate	95.83	88.40	92.50	85.10	92.60	**49.50**	**53.90**
Norm. Gate + Def. Gate	**97.92**	**92.00**	**95.10**	**87.60**	**93.10**	41.90	44.80

Notes. CA = Cross-Attention; Norm. Gate = partial CA model based solely on the normal image; Def. Gate = partial CA model based solely on the defective mask.

TABLE V: ABLATION STUDY OF DIFFUSION/TRAINING STEPS – HAZELNUT

Steps		Classification	Detection			Localization		
Diffusion	Training	Acc. (%)	AUC	AP	F_1-max	AUC	AP	F_1-max
250	10 000	75.00	91.60	94.70	88.70	92.00	39.20	44.20
1000	30 000	97.92	92.00	95.10	87.60	93.10	41.90	44.90

performance in this category. By contrast, its complementary version yields competitive but lower results across all metrics, suggesting that while attention on non-defective masks captures fine-grained details, it proves less effective when not complemented by complete information.

Diffusion/Training Steps. As reported in Table V, increasing the number of diffusion and training steps has a significant impact on classification, while the effect on detection and localization is more limited. Specifically, classification accuracy increases by 22% when moving from 250 diffusion steps with 10 000 training iterations to 1000 diffusion steps with 30 000 iterations. In contrast, detection performance remains relatively stable, whereas localization shows a slight but consistent improvement. These results indicate that longer diffusion sampling and training are beneficial for learning more discriminative global representations, whereas local anomaly cues for detection and localization tend to saturate earlier, yielding diminishing returns. Finally, Figure 8 illustrates the generative process of our diffusion-based model. Starting from almost pure noise (*i.e.*, $t = 900$), it progressively refines the image through a series of denoising steps. As time decreases, the model iteratively removes noise and adds structural detail, leading to a coherent and realistic image at the final step (*i.e.*, $t = 0$).

DAM Timing Analysis. Table VI presents the average computational time required to generate a single image using the Data Augmentation Module (DAM), computed over 50 samples. Superpixel extraction constitutes an initial and separate step. The subsequent alignment step is relatively lightweight, requiring less than one second in all cases. By contrast, mask creation and augmentation exhibit substantial variability, particularly for specific defects, suggesting that these cases introduce greater complexity during mask processing and frequently lead to the maximum number of iterations. Data quality assessment requires a moderate amount of time. Overall, the results indicate that although DAM is able to efficiently generate novel defective samples, more complex defects introduce significant variability and computational overhead during the generation process.

Latency/Memory Analysis. We report in Table VII per-image and per-object performance metrics to characterize both

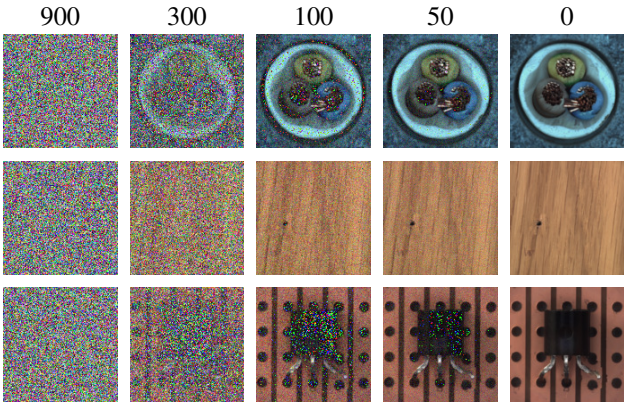


Fig. 8: Images sampled at multiple timesteps (indicated at the top) across different categories: cable (bent wire), wood (hole), and transistor (bent lead), from top to bottom.

TABLE VI: AVERAGE TIMINGS

Step	Defect			
	Bent lead	Cut lead	Damaged case	Misplaced
Superpixel extr.	14.38 ± 0.03			
Alignment	0.74 ± 0.01	0.73 ± 0.01	0.72 ± 0.01	0.72 ± 0.01
Mask creat. and augm.	0.09 ± 0.12	0.11 ± 0.03	23.54 ± 34.39	17.08 ± 22.49
Data qual. assess.	5.34 ± 2.88	5.48 ± 2.99	4.93 ± 2.24	3.80 ± 1.49
Total	6.09 ± 2.86	6.26 ± 3.00	29.14 ± 36.07	21.57 ± 22.65

Notes. Values represent average timings (in seconds) per generated image – Transistor.

the effectiveness and computational efficiency of the proposed D-ADDA framework. Latency refers to the total end-to-end inference time per image, including data loading, forward pass, and any preprocessing. GPU memory and CPU memory indicate the peak memory usage during inference on the respective hardware, reflecting the computational resource requirements of the model. For each metric, we report the mean and standard deviation across all images in a given category. The proposed framework demonstrates highly efficient per-image inference; however, the relatively large standard deviation indicates variability in processing time, which may arise from differences in image complexity or asynchronous GPU operations. GPU and CPU memory usage remain minimal on average, although occasional peaks are likely caused by dynamic data loading and preprocessing steps.

V. CONCLUSION

In this paper, we proposed D-ADDA (Distributed Anomaly Detection based on Data Augmentation), a novel anomaly generation model designed to synthesize paired anomalous images and masks in a distributed system. D-ADDA integrates data augmentation with self-attention and cross-attention mechanisms to enhance the learning of discriminative features and to condition the generation process on input masks. A further distinctive aspect of D-ADDA is its compact backbone, which requires an order of magnitude fewer parameters than widely adopted diffusion architectures. Our experiments on the MVTec AD dataset demonstrated that it achieves better performance than several pre-trained centralized models, while also enhancing accuracy in distributed classification tasks. Be-

TABLE VII: AD COMPUTATIONAL METRICS

Latency [ms]	GPU Memory [MB]	CPU Memory [MB]
Classification (21.29 M)		
9.04 ± 30.52	2.74 ± 13.91	5.65 ± 36.59
Detection/Localization (28.37 M)		
11.49 ± 20.87	13.30 ± 64.51	3.05 ± 23.97

Notes. The number of network parameters is indicated in parentheses.

yond enabling privacy-aware deployment, our pipeline can also be deployed locally, improving performance. Future works will focus on improving image resolution, developing more advanced generative architectures, and investigating different applicative domains.

REFERENCES

[1] W. Wei, J. C.-W. Lin, S. H. Shah, V. H. C. de Albuquerque, and W. Wang, "Introduction to the special section on artificial intelligence for next generation industrial cyber-physical systems," *IEEE Syst. J.*, vol. 17, no. 4, pp. 5070–5072, 2023.

[2] B. Bajic, A. Rikalovic, N. Suzic, and V. Piuri, "Toward a human-cyber-physical system for real-time anomaly detection," *IEEE Syst. J.*, vol. 18, no. 2, pp. 1308–1319, 2024.

[3] I. Ahmed, G. Jeon, and F. Piccialli, "From artificial intelligence to explainable artificial intelligence in industry 4.0: A survey on what, how, and where," *IEEE Trans Ind. Informat.*, vol. 18, no. 8, pp. 5031–5042, 2022.

[4] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of industrial equipment: A survey," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2213–2227, 2019.

[5] Y. Duan, Y. Hong, L. Niu, and L. Zhang, "Few-shot defect image generation via defect-aware feature manipulation," in *Proc. of AAAI*, 2023, p. 571–578.

[6] G. Zhang, K. Cui, T.-Y. Hung, and S. Lu, "Defect-gan: High-fidelity defect synthesis for automated defect inspection," in *Proc. of WACV*, 2021, pp. 2523–2533.

[7] T. Hu, J. Zhang, R. Yi, Y. Du, X. Chen, L. Liu, Y. Wang, and C. Wang, "AnomalyDiffusion: Few-shot anomaly image generation with diffusion model," in *Proc. of AAAI*, 2024.

[8] P. Coscia, A. Genovese, F. Scotti, and V. Piuri, "Adversarial defect synthesis for industrial products in low data regime," in *Proc. of ICIP*, 2023.

[9] ——, "Applications and limits of image-to-image translation models," in *Proc. of DSP*, 2023.

[10] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Proc. of NeurIPS*, 2021.

[11] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: A patch distribution modeling framework for anomaly detection and localization," in *Proc. of ICPRW*, 2021.

[12] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proc. of CVPR*, 2022.

[13] M. Lee and J. Choi, "Text-guided variational image generation for industrial anomaly detection and segmentation," in *Proc. of CVPR*, 2024.

[14] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. of CVPR*, 2022.

[15] X. Liu, J. Wang, B. Leng, and S. Zhang, "Dual-modeling decouple distillation for unsupervised anomaly detection," in *Proc. of ACM-MM*, 2024.

[16] J. Liu, K. Wu, Q. Nie, Y. Chen, B.-B. Gao, Y. Liu, J. Wang, C. Wang, and F. Zheng, "Unsupervised continual anomaly detection with contrastively-learned prompt," in *Proc. of AAAI*, 2024.

[17] X. Li, Z. Huang, F. Xue, and Y. Zhou, "MuSc: Zero-shot industrial anomaly classification and segmentation with mutual scoring of the unlabeled images," in *Proc. of ICLR*, 2024.

[18] W. Luo, Y. Cao, H. Yao, X. Zhang, J. Lou, Y. Cheng, W. Shen, and W. Yu, "Exploring intrinsic normal prototypes within a single image for universal anomaly detection," in *Proc. of CVPR*, 2025.

[19] Z. Pu, L. Yan, Y. Bai, D. Cabrera, and C. Li, "Incrementally generative adversarial diagnostics using few-shot enabled one-class learning," *IEEE Trans Ind. Informat.*, vol. 20, no. 10, pp. 12 189–12 199, 2024.

[20] Z. Fang, X. Wang, H. Li, J. Liu, Q. Hu, and J. Xiao, "FastRecon: Few-shot industrial anomaly detection via fast feature reconstruction," in *Proc. of ICCV*, 2023.

[21] E. Schwartz, A. Arbelle, L. Karlinsky, S. Harary, F. Scheidegger, S. Doveh, and R. Giryes, "MAEDAY: MAE for few- and zero-shot anomaly-detection," *Computer Vision and Image Understanding*, vol. 241, 2024.

[22] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. of ICML*, 2021.

[23] Q. Zhou, G. Pang, Y. Tian, S. He, and J. Chen, "AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection," in *Porc. of ICLR*, 2024.

[24] W. Ma, X. Zhang, Q. Yao, F. Tang, C. Wu, Y. Li, R. Yan, Z. Jiang, and S. Zhou, "AA-CLIP: Enhancing zero-shot anomaly detection via anomaly-aware CLIP," in *Proc. of CVPR*, 2025.

[25] P.-W. Chen, J. Chun-Wei, R. Cupek, and C.-C. Chen, "FedCali: Mitigating overgeneralization for anomaly detection in distributed sensor environments," in *Proc. of BigData*, 2024.

[26] J. Zhang, X. hua Qi, S. Pang, S. Pan, X. Tu, P. Wan, and B. Zhao, "Federated generative learning with foundation models," 2024.

[27] P. Coscia, S. Ferrari, V. Piuri, and A. Salman, "Synthetic and (Un)Secure: Evaluating generalized membership inference attacks on image data," in *Proc. of SECRYPT*, 2025.

[28] C. Leite, J. den Hartog, and P. Koster, "A framework for privacy-preserving white-box anomaly detection using a lattice-based access control," in *Proc. of SACMAT*, 2023.

[29] S. Mehnaz and E. Bertino, "Privacy-preserving real-time anomaly detection using edge computing," in *Proc. of ICDE*, 2020.

[30] W. Guo and P. Jiang, "Weakly supervised anomaly detection with privacy preservation under a bi-level federated learning framework," *Expert Syst. Appl.*, 2024.

[31] S. Lin, K. Wang, X. Zeng, and R. Zhao, "An effective crop-paste pipeline for few-shot object detection," in *Proc. of CVPRW*, 2023.

[32] Y. Liang, J. Zhang, S. Zhao, R. Wu, Y. Liu, and S. Pan, "Omni-frequency channel-selection representations for unsupervised anomaly detection," *IEEE Trans. Image Process.*, vol. 32, pp. 4327–4340, 2023.

[33] A. Verma, T. Badal, and A. Bansal, "Advancing image generation with denoising diffusion probabilistic model and ConvNeXt-V2: A novel approach for enhanced diversity and quality," *Computer Vision and Image Understanding*, vol. 247, p. 104077, 2024.

[34] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proc. of ICCV*, 2023.

[35] V. Zavrtanik, M. Kristan, and D. Skočaj, "DRAEM - a discriminatively trained reconstruction embedding for surface anomaly detection," in *Proc. of ICCV*, 2021.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. of MICCAI*, 2015.

[37] Z. You, L. Cui, Y. Shen, K. Yang, X. Lu, Y. Zheng, and X. Le, "A unified model for multi-class anomaly detection," in *Proc. of NeurIPS*, 2022.

[38] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. of ICLR*, 2021.

[39] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Proc. of ECCV*, 2008.

[40] J. Ho and T. Salimans, "Classifier-free diffusion guidance," in *Proc. of NeurIPSW*, 2021.

[41] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD — a comprehensive real-world dataset for unsupervised anomaly detection," in *Proc. of CVPR*, 2019.

[42] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han, "Differentiable augmentation for data-efficient GAN training," in *Proc. of NeurIPS*, 2020.

[43] U. Ojha, Y. Li, J. Lu, A. A. Efros, Y. Jae Lee, E. Shechtman, and R. Zhang, "Few-shot image generation via cross-domain correspondence," in *Proc. of CVPR*, 2021.

[44] D. Lin, Y. Cao, W. Zhu, and Y. Li, "Few-shot defect segmentation leveraging abundant defect-free training samples through normal background regularization and crop-and-paste operation," in *Proc. of ICME*, 2021.

[45] S. Niu, B. Li, X. Wang, and H. Lin, "Defect image sample generation with GAN for improving defect recognition," *IEEE Trans. Autom. Sci. Eng.*, pp. 1611–1622, 2020.

[46] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. of AISTATS*, 2017.