IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 March 2024, accepted 19 March 2024, date of publication 22 March 2024, date of current version 28 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3380618

RESEARCH ARTICLE

Enhancing Battery Exterior Defect Inspection Accuracy Through Defect-Background Separated GAN Development

DONGHUN KU AND HEUI JAE PAHK
School of Mechanical and Aerospace Engineering, Seoul National University, Seoul 08826, South Korea

Corresponding author: Heui Jae Pahk (hjpahk@snu.ac.kr)

This work was supported by the Institute of Advanced Machines and Design-Seoul National University (IAMD-SNU).

ABSTRACT This paper aims to develop a defect-background separated generative adversarial network (GAN) using deep learning and GAN to enhance the accuracy of battery exterior defect inspection. In actual battery production lines, the occurrence rates of defects vary by defect type, making it challenging to create a large, uniform defect dataset due to the time required for defect acquisition. This leads to a reduction in the accuracy of battery exterior defect inspection. To construct a large, uniform defect dataset, this paper proposes a defect-background separated GAN based on the principles of GANs. The defect-background separated GAN performs effective defect and background separation learning by referring to the segmentation labeling of defects. Through dataset augmentation using the defect-background separated GAN, the performance quality of newly generated synthetic defect images has improved, and accuracy in battery exterior defect inspection can be enhanced through extensive dataset training. Experimental results show the lowest Fréchet inception distance score among various other methods for the battery exterior defect dataset, and it generates clear synthetic defects perceptible to the human eye. By training defect segmentation on this large, uniform defect dataset, an accuracy of 96.1% and an intersection over union value of 0.71 were achieved. Ultimately, applying this defect inspection network to the actual production line demonstrated a 72% improvement in time efficiency. This demonstrates the stability and robustness of the large, uniform defect dataset generated through the defect-background separated GAN.

INDEX TERMS Battery exterior defect inspection, deep learning, defect segmentation, generative adversarial network, synthetic defect.

I. INTRODUCTION

The fourth industrial revolution is ushering in significant changes in areas applying innovative technologies such as deep learning. One of these transformative shifts is the increasing automation in the field of defect inspection. In the past, manual inspection and machine vision inspection were primarily employed for scrutinizing product defects. Manual inspection relies on human labor to visually identify product defects, depending on the senses and rich experience of skilled workers. While this method can provide accuracy and reliability, it is susceptible to subjective judgments and errors due to human factors and fatigue. Additionally, it may be challenging to swiftly process large quantities of products. Machine vision inspection involves analyzing images or video data using computer vision technology to detect defects [1], [2], [3]. This method enables automation and high-speed processing, delivering consistent results. However, in the early stages, it was limited in accuracy compared to manual inspection due to the constraints of image processing algorithms. Moreover, there were challenges in operating under various lighting and environmental conditions.

Deep learning is emerging as an innovative technology that harnesses the advantages of visual inspection and machine vision in the realm of defect detection [4], [5]. Deep learning models can assimilate knowledge from extensive datasets,

The associate editor coordinating the review of this manuscript and approving it for publication was Tai Fei.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

autonomously discerning patterns for accurate defect inspections. Furthermore, these models exhibit robust performance under diverse environmental conditions, ensuring consistency by eliminating subjective human judgments. As deep learning continues to evolve, it solidifies its position as a technology effectively applicable in real production environments. In this manner, the progress of the fourth industrial revolution is ushering in a deep learning-based inspection technology that enhances efficiency and accuracy through the collaboration between humans and machines in the field of defect detection. This is expected to contribute significantly to the improvement of product quality and increased productivity.

Defect inspection methods based on deep learning typically employ three main techniques: Classification, Segmentation, and Object Detection [6]. Classification involves assigning given input data to predefined categories. In the context of defect inspection, a pre-trained deep learning model is utilized to determine whether defects are present in products. The model, having learned from each defect class, identifies and classifies the type of defect. This method is primarily employed to confirm the presence of a single defect. Segmentation is a technique that divides individual objects or defect areas at the pixel level in images or videos. This approach precisely identifies the location and area of defects, providing detailed information about multiple defects or their shapes. In applications such as battery exterior defect inspection, segmentation proves beneficial for accurately identifying the position and size of defects. Object detection is a technology that identifies the location and bounding boxes of individual objects in images or videos. It is applied to locate defects or faulty areas. Object detection typically uses bounding boxes to indicate the location of defects, allowing for a precise understanding of the position of defective parts.

We utilize segmentation technology, which detects defect areas at the pixel level, for battery exterior defect inspection. Segmentation precisely identifies the shape and location of defects, enabling accurate judgments in automated defect inspection systems. In the context of battery exterior defect inspection, segmentation plays a crucial role in improving product quality and enhancing defect detection accuracy. One of the significant challenges in defect segmentation is the heavy reliance on the deep learning model's performance on the quantity and quality of training data. Deep learning models require extensive and diverse datasets for effective training, but in reality, data for defect inspection are often limited and unevenly distributed. This limited and uneven dataset is a major factor compromising the accuracy of deep learning-based defect inspection. Particularly, when deep learning models are trained on imbalanced data, the recognition rate for defects or faulty parts decreases. The model may fail to sufficiently learn specific defect patterns or features, leading to decreased accuracy and instances of missing defects. Overcoming such challenges necessitates the creation of additional data for segmentation training. Ideally, this additional data should encompass various defect types and originate from diverse environments. Through this approach, the model can grasp a wide range of defect patterns, thereby acquiring stronger defect inspection capabilities. Improving the quantity and quality of data is a crucial strategy for enhancing the performance of deep learning-based defect inspection, contributing to increased accuracy and rapid detection of faulty areas.

Various methods have been employed in the past to generate data. These methods are utilized to train deep learning models by either collecting real data or synthesizing data. Data synthesis involves using computer graphics software to simulate defects, generating virtual data through mock experiments. While this method allows for easy adjustment of the shape and size of defects, there may be differences from realistic defects. Data augmentation is a technique for creating new data by transforming existing data through operations like rotation, resizing, and adding noise, ensuring diversity in the dataset. The generative adversarial network (GAN) stands out as one of the most innovative and powerful methods for data generation [7]. GAN employs two neural networks, the generator and discriminator, to generate synthetic data. The generator strives to produce data similar to real data, while the discriminator aims to distinguish between generated and real data. Through this process, the generator gradually generates synthetic data that is challenging to differentiate from real data. GAN has the advantage of creating synthetic data that closely resembles real data, enhancing data quality. However, as GAN is based on unsupervised learning, it may be challenging to create defects of desired shapes and types. Additionally, GAN training can be unstable, requiring careful tuning of hyperparameters. Despite these challenges, GAN can contribute significantly to the training of deep learning models by generating high-quality synthetic data that is nearly indistinguishable from real data. GAN has become an innovative technology widely used in the field of data generation, and it is anticipated to play a crucial role in the realm of deep learning-based defect inspection.

We propose a defect-background separated GAN that utilizes GAN principles to construct a comprehensive and uniform defect dataset. The defect-background separated GAN, through separate training on defects and backgrounds, addresses the issues of defect generation failures commonly observed in conventional GANs. Furthermore, it incorporates segmentation labeling as input, allowing the generation of defects with desired shapes and types against specific backgrounds. Evaluation metrics are employed to assess the quality of the generated images, providing a quantitative measure of the similarity between synthesized and real data. The primary objective of this research is to enhance the performance of defect segmentation learning using the generated large and uniform defect dataset, thereby improving the accuracy of defect inspection. Training with synthetic data enhances the model's generalization capabilities and plays a vital role in improving the reliability of defect inspection systems. Ultimately, the defect inspection network developed in this study is applied to real production lines, verifying the stability and robustness of the generated large and uniform defect dataset. The experimental results validate the practical applicability of the developed system, demonstrating its potential impact in the fields of manufacturing and quality improvement. This contribution aims to enhance the accuracy and efficiency of defect inspection technology, making significant strides in the realm of manufacturing and quality improvement.

II. RELATED WORK

A. DEFECT INSPECTION IN INDUSTRY

Several techniques have been developed and employed for defect inspection, yet some researchers are investigating unique methods. Standard image-processing methods for fault detection in aluminum castings use a bank of filters to generate an error-free reference image [8]. A new method for autonomously evaluating aluminum castings is based on a series of radioscopic images obtained at different points of the casting. When the real radioscopic image is compared to this reference image, defects are identified in pixels where there is a significant discrepancy. However, the arrangement of each filter is heavily influenced by the dimensions and form of the casting structure being examined. Additionally, a newly developed deep neural network-based metal surface flaw inspection system is presented [9]. The goal of this data-driven technique is to automatically classify flaws in metal surfaces using a unique convolutional neural network (CNN) architecture. A substantial amount of learning data is required to achieve good performance in such a deep CNN model, and, in particular, the amount of data for each class should be comparable. However, due to the nature of the industrial sector, there are numerous instances in which the occurrence of each fault varies. This work proposes a variational autoencoder-based data augmentation technique to address the issue of unbalanced data. To enhance the performance of image production, a convolutional layer is incorporated into the network instead of a fully connected layer. Another novel method utilizes convolutional neural networks to identify microstructural flaws in Li-ion battery electrodes [10]. Demonstrating the capability of CNNs to yield satisfactory results without prior task understanding, the study showcases their ability to identify numerous flaws in battery micrographs without requiring manual feature engineering. Meaningful features, such as the presence of foreign particles or a distorted collector layer indicating the existence of faults, are characteristics that the system can learn on its own.

B. DEFECT INSPECTION USING DEEP LEARNING

To ensure product quality, production efficiency, and market share in industrial manufacturing, surface defect testing is essential. To reduce costs, minimize waste, and enhance factory reputation, research focuses on the quick and accurate identification of flaws. In the industry, manual inspection is the initial technique used to inspect surfaces for defects. Due to human energy and attention limitations, product surface defect inspection is not only inaccurate but also time-consuming, significantly increasing labor expenses and hindering the advancement of production efficiency. As artificial intelligence and machine learning have advanced, image-based autonomous surface defect assessment has become more commonplace in various industrial settings, taking over the role of manual inspection. The two primary categories of current surface defect inspection techniques are deep learning-based techniques with automated feature extraction and conventional techniques with manual feature extraction [11]. While deep learning-based approaches automatically extract defective features after learning from a given number of defect samples, traditional methods extract defect characteristics through image processing, image analysis, etc. There are three types of learning-based approaches: supervised, unsupervised, and semi-supervised [12]. Learning-based techniques operate optimally when extensive datasets are available, and supervised methods perform best when the dataset contains an adequate number of samples from each class.

Large datasets containing labeled defect-free and faulty samples in a training set are essential for supervised detection. The labeling of all training data enables extremely high detection rates. However, it should be noted that due to the unequal class distribution in the dataset, supervised detection might not always be the optimal strategy. Supervised learning techniques utilize various datasets, including the railroad dataset [13], the fabric dataset [14], and the factory fault dataset [15]. The structures of feature extraction and classification techniques in supervised methods, as well as deep neural networks, vary. For instance, literature suggests a two-layer neural network for identifying cross-category faults without the need for retraining. This method learns differential features based on structural similarities between image pairs, potentially resulting in structural similarities between various classification objects. Experiments conducted with real industrial datasets have demonstrated the efficacy of this strategy in detecting problems across different types of factories.

Researchers have begun exploring unsupervised approaches to address the limitations of supervised methods. In the absence of labeled information, machines can autonomously discern inherent characteristics and connections by assimilating the intrinsic qualities of the input training data. Consequently, the system automatically classifies the input training data based on the patterns identified within these unlabeled data [16]. Utilizing innate traits and relationships among the data, it automatically categorizes these unlabeled data. Among unsupervised learning techniques, reconstruction and embedding similarity-based methods are most commonly employed to identify surface flaws. The frequently utilized techniques include reconstruction-based methods such as autoencoders and

GANs. Several popular algorithms, such as PaDIM [17], SPADE [18], PatchCore [19], and others, have been developed. A study proposed an algorithm based on deep belief networks to detect flaws in solar cells [20]. The fine-tuning network of the backpropagation algorithm utilized supervision data from both training and reconstructed images.

Semi-supervised methods amalgamate the advantages of supervised and unsupervised approaches. In semi-supervised defect detection, only normal samples are used as training data. A defect-free boundary is learned and defined, with samples outside of it deemed abnormal. This approach proves valuable as obtaining defective samples is infrequent. However, in terms of defect identification, this approach is less precise than supervised approaches. Semi-supervised techniques can automatically generate unlabeled sample data without requiring human participation [21], [22], [23]. A methodology for detecting flaws in PCB solder joints has been presented in the literature [24]. It utilizes a sample query suggestion algorithm for classification and integrates active learning and self-training. It has been demonstrated that this framework enhances classification accuracy while reducing the need for human annotations.

C. DEFECT INSPECTION USING GAN

A class of generative models is called GAN, initially presented by Goodfellow. This implies that GANs can generate new material using their training data. A GAN consists of two neural networks, the generator, and the discriminator, which compete with each other [7]. While the discriminator assesses the legitimacy of each newly created data instance, the generator generates new ones from the latent space. Samples or data from the original dataset, as well as those produced by the generator, are fed into the discriminator. The discriminator then attempts to predict the origin of each sample. The generator assesses the quality of the data it creates by learning to map a latent space to the distribution of the original data it seeks to replicate, as evaluated by the discriminator.

GANs are popular in various image processing fields [25]. They have been applied to the development of new molecules for oncology, while other researchers have used them to generate high-resolution images. However, training GANs is known to pose difficulties, as the two-player objective function often leads to artifacts and mode collapse, especially when generating high-resolution images. An automated optical inspection system is also built on semi-supervised deep learning for the inspection of IC metal package surface defects [26]. In contrast to earlier inspection techniques, a fully multi-scale inspection framework is suggested to conduct assessments of flaws at various scales. To effectively capture the inherent patterns of eligible samples at various scales, a highly intricate multi-scale GAN with a transformer is created, integrating many innovative modules. To adequately extract features from the IC metal package image, a multi-scale CNN encoder with a new feature extraction strategy and a cross-scale feature fusion module is developed. Researchers present a novel CVAE-GAN model in the literature for the creation of finely detailed category images [27]. Its higher performance on three separate datasets demonstrates the ability to generate diverse sorts of objects. The suggested approach supports a wide range of applications, including picture generation, attribute morphing, image inpainting, and data augmentation for improving face recognition models. In a specific defect research, the author [28] generated faulty images using a GAN. A novel surface defect image-generation method, named SDGAN, has been developed based on features from an industrial defect dataset. SDGAN aims to generate defect samples solely from defect-free industrial images. Faulty datasets with high single-image quality and diversity were created using SDGAN. Anomaly detection and defect classification models were trained on the augmented dataset, incorporating SDGAN and other image-generation algorithms.

The defect-background separated GAN proposed in this paper demonstrates a remarkable improvement in accuracy in the field of battery exterior defect inspection and stands out in constructing extensive and uniform defect datasets. In contrast to previous research, this GAN adopts a method that separates and trains defects and backgrounds within the generator network, leading to the creation of a more effective defect generation model. Consequently, the quality of generated defects has increased, and the GAN's ability to generate defects has been enhanced. The ability for users to customize synthetic defects by setting desired shapes and backgrounds means they can efficiently tailor datasets according to their needs. This flexibility allows for the selective synthesis of data that was previously lacking in real industrial settings, leading to more significant benefits in the segmentation training of defects.

III. METHODOLOGY

A. DEFECT-BACKGROUND SEPARATED GAN

GAN is an effective artificial intelligence network capable of continuously generating synthetic defect images. The basic GAN consists of a generator and a discriminator. The defect-background separated GAN proposed in this paper aims to enhance the defect generation performance and create more precise synthetic defects by improving the structure of the GAN's generator. Figure 1 illustrates the structure of the defect-background separated GAN. While using the basic GAN structure, improvements in the generator network's structure and training approach were made to enhance the overall performance of the GAN. The latent matrix contains information about the defect to be generated. In cases where the features of defects and backgrounds are being learned, the latent matrix receives segmentation labeling information. When generating defects in the GAN using the learned features of defects and backgrounds, either random defect-background information or user-specified defect labeling is input. The encoder in the defect-background separated GAN processes the input labeling information for defects and

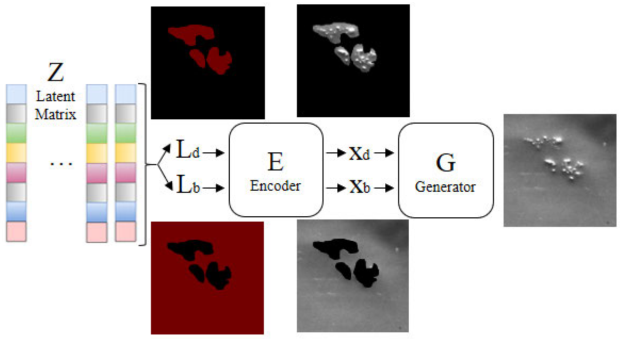


FIGURE 1. Structure of defect-background separated GAN.

backgrounds, producing individual defect and background images. Ultimately, the generator seamlessly combines these individual defect and background images to generate a final synthetic defect image.

The latent space used in GAN represents an abstract space that the generative model utilizes to generate data. This space acts as the input space for the generative model, taking random or latent vectors as input to generate data resembling real-world examples. The latent space is filled with latent vectors extracted from various probability distributions or random numbers, and the generative model uses these latent vectors to produce diverse data. GANs efficiently manipulate this latent space, allowing the synthesis of images with various characteristics by aligning them with the latent space. Typically, the latent space is initially composed of random values, making it challenging to predict what defects will be generated from these values. In this paper, we make the latent space more intuitive as input, enabling users to conveniently modify and create defect types and shapes directly. The latent matrix used as input in the defect-background separated GAN expands the latent space. It contains information about the features of defects to be generated for each pixel. Through pixel-wise information input, users can ultimately create defects of the desired type and shape. In the context of battery exterior defect inspection, there are two main backgrounds and six defect types. Backgrounds are classified as body and outside, while defects include groove, spore, folding, leakage, white dot, and black dot, as illustrated in Figures 2 and 3. The latent matrix allocates an 8-array for each pixel to represent background and defect types. Among the eight arrays, the first two represent the background, with values [1,0] for body and [0, 1] for outside. The subsequent six items indicate the presence of each defect type.

The latent matrix has a size of 8 x width x height for each image. In Figure 4, a scenario is depicted where a folding defect occurs in the body region. To designate the entire pixel space as the body region, the first two values for all pixels are represented as [1,0]. For pixels affected by the folding defect, the last six values are denoted as [0,0,1,0,0,0]. In other words, the latent matrix for pixels with defects is [1,0,0,0,1,0,0,0], while the latent matrix for pixels without defects is [1,0,0,0,0,0,0,0]. Since the latent matrix contains information about defect presence at the pixel level, it aligns with the labeling information for defects. Conversely, by segmenting the defect region through direct user labeling, this information can be converted into the latent matrix format, serving as the input for the GAN. Users can input their desired background and defect shapes into the latent matrix, enabling the GAN to generate defects according to user specifications.

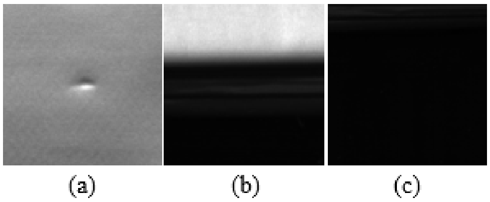


FIGURE 2. Battery background type: (a) Body, (b) Body and outside, (c) Outside.

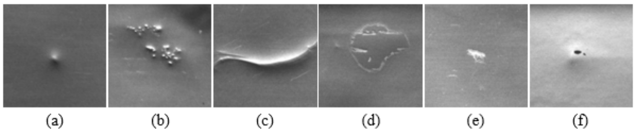


FIGURE 3. Defect type: (a) Groove, (b) Spore, (c) Protrusion, (d) Leakage, (e) White dot, (f) Black dot.

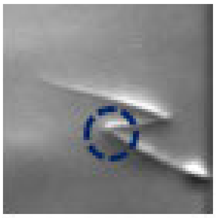


FIGURE 4. Class C defect on body region.

The encoder plays a crucial role as a component in the field of AI. Essentially, the encoder is responsible for transforming various forms of raw data into a format that the model can understand and process. For instance, data such as text, images, and sounds are converted through the encoder into vectors or other numerical forms, enabling subsequent machine learning algorithms or neural networks to handle them. Another essential function of the encoder is to extract the core features or patterns of the data. Through this process, the model captures important information in the data while ignoring unnecessary parts or noise. This is particularly crucial when dealing with complex datasets. For example, in image recognition, the encoder identifies key objects in a photo and filters out background noise. In summary, the encoder significantly enhances the performance and efficiency of the model through various functions within the AI system, including data transformation, extraction, reduction, and compression. The role of the encoder is essential for the model to effectively address complex problems.

In this paper, we introduce an encoder to enhance the probability of defect generation in GAN by isolating the defect and background during training. GANs possess the advantage of generating synthetic data that closely resembles real data, thereby improving data quality. However, as GANs operate on unsupervised learning, creating defects of the desired shape and type can be challenging. Moreover, training can be unstable, requiring appropriate hyperparameter settings and tuning. The proposed defect-background separated GAN addresses the shortcomings of conventional GANs by independently training the defect and background, thereby reducing instances of defect generation failures observed in traditional GANs. Defect generation failures refer to cases where defects and backgrounds are mixed in synthesis or, in some instances, defects are not visually synthesized at all. Additionally, utilizing segmentation labeling as input allows for the creation of defects of the desired shape and type on specific backgrounds.

The inserted encoder aims to independently learn defects and backgrounds, extracting features for each and generating images containing only defects and backgrounds, respectively. The encoder takes the segmentation labeling of defects and backgrounds, which is also the user-specified defect information in the form of a latent matrix, as input. It converts this information into labels for defects L_d and backgrounds L_b outputting individual images for defects X_d and backgrounds X_b. The latent matrix contains information about the type and shape of the defect to be generated, which can be set by the user or specified randomly. During the defect training of the GAN, the latent matrix contains segmentation labeling information provided by the operator, while during the defect generation of the GAN, it includes user-specified defect information as input. The GAN, operating as unsupervised learning, automatically learns the characteristics of defect images and consistently generates similar defect images. To enhance the success rate of defect image generation and extract the features of defects and backgrounds separately, a dataset with segmentation labeling for defects was modified and incorporated into the GAN using a semi-supervised approach. This extensive dataset of defect segmentation serves for both the GAN's defect synthesis learning and defect segmentation learning. In summary, the encoder learns the features of defects and backgrounds separately through the training of a dataset with segmentation labeling. It generates defects and backgrounds matching the user-specified latent matrix. The structure of this encoder is briefly depicted in Figure 5.

The input latent matrix contains information about the type of defects and background for each pixel. Due to this characteristic, the latent matrix can be represented in the format of an 8 x width x height array or a width x height segmentation labeling. The initial part of the encoder converts this input data format into segmentation labeling of width x height, separating it into defect labels and background labels. Subsequently, each label undergoes training to generate images specific to defects and backgrounds. This separation training of defects and backgrounds prevents blurring of the edges and enables sharp defect generation in the final image. Each defect and background is composed of convolution layers, as outlined in Table 1. An encoder with this configuration generates defects and backgrounds of size 512 × 512 × 1 from a label of size 512 × 512 × 1. It is fundamentally composed of convolution layers, ReLU, pooling layers, and transposed convolution layers. The convolution layer applies filters to the input data to extract features, ReLU acts as an activation function to maintain non-linearity and sparsity, pooling layers are used to reduce the data size, and transposed convolution layers perform the inverse operation of convolution layers, increasing the spatial size of the input data. The encoder ultimately transforms the 512 × 512 × 1 label image to a dimension of 128 × 128 × 512 and, through transposed convolution, amplifies the image size back to the final 512 × 512 × 1 defect or background image for training. Firstly, four convolution groups were employed to transform label images of size 512 × 512 × 1 into 128 × 128 × 512, while learning the characteristics of defects or backgrounds. The purpose of these convolution groups is to reduce image size while increasing depth to gradually learn the features of defects and backgrounds. Each convolution group consists of two convolution layers, two ReLU activations, and one pooling layer. Each convolution layer utilizes a 3 × 3 filter with a stride of 1 to maintain the size of the input and output data, while increasing depth. Zero padding was applied with the same setting to preserve input and output data sizes. ReLU activation functions play a crucial role in adjusting the distribution of activation values within the network by zeroing out negative values. Pooling layers were utilized with a 3 × 3 filter and a stride of 2 to maintain the depth of the input data while halving the image size. The convolution groups in this paper generally reduce input data size while increasing depth, and the size and depth can be adjusted based on hyperparameters such as filter type, padding type, and stride. This utilization of convolution groups aids in generalizing the model compared to using a single convolution layer, thereby helping to avoid overfitting. Through the four convolution groups, the input data size transitions from 512 × 512 × 1 to 128 × 128 × 512, with intermediate sizes of 256 × 256 × 64, 256 × 256 × 128, and 128 × 128 × 256. The output data of size 128 × 128 × 512 is further transformed into defect

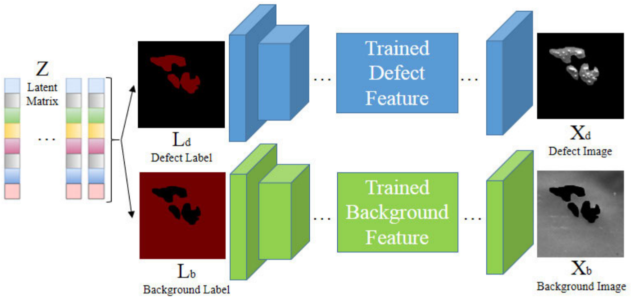


FIGURE 5. Encoder structure.

TABLE 1. **Configuration of the encoder.**

Layer Type	Filter Size	Stride	Output Size
Convolution	64 x 3 x 3	1	512 x 512 x 64
ReLU			
Convolution	64 x 3 x 3	1	512 x 512 x 64
ReLU			
Pooling	64 x 2 x 2	2	256 x 256 x 64
...			
Convolution	512 x 3 x 3	1	128 x 128 x 512
ReLU			
...			
Transposed Convolution	64 x 3 x 3	2	512 x 512 x 64
Convolution	64 x 3 x 3	1	512 x 512 x 64
ReLU			
Convolution	1 x 1 x 1	1	512 x 512 x 1

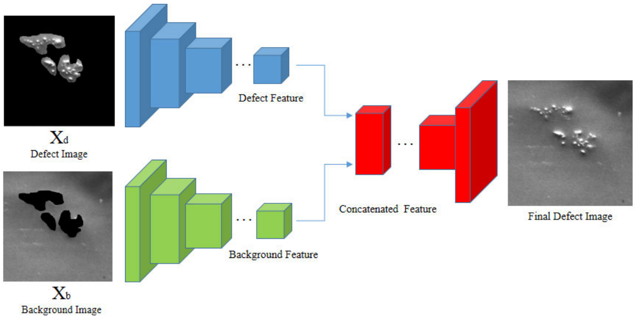


FIGURE 6. **Generator structure.**

and background images of size 512 × 512 × 1 through an additional set of four convolution groups. These convolution groups are similar in structure to the ones described earlier, but they utilize transpose convolution instead of pooling layers. While pooling layers halve the size of the data while maintaining its depth, transpose convolution doubles the size while preserving the depth. Transpose convolution employs a 3 × 3 filter with a stride of 2, allowing for an increase in output size without the use of zero padding. Through the four convolution groups, the input data size transitions from 128 × 128 × 512 to 512 × 512 × 1, with intermediate sizes of 128 × 128 × 256, 256 × 256 × 128, and 256 × 256 × 64. Finally, the output data is trained to be of size 512 × 512 × 1. The encoder utilizes combined defect and background label data as input, enabling training through size and depth adjustments, and ensuring that only the corresponding defects and backgrounds are outputted for each label.

The loss function employed in this process is the mean squared error (MSE) loss [29]. The MSE loss function is commonly utilized in tasks involving the generation or reconstruction of original images. By minimizing the difference in pixel values between the generated images during training and the ground truth images, the model for image generation can be effectively trained. The formula for the MSE loss is identical to Equation (1), where N represents the total number of data points or pixels, y_i is the actual ground truth image, and $\hat{y}_i$ is the image generated by the learning model. The MSE calculation involves squaring the difference between predicted values and actual values for each data point, summing these squared differences across all data points, and then dividing by the total number of data points to obtain the mean. A smaller MSE indicates that the model's predictions are closer to the actual values.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{1}$$

The generator of the defect-background separated GAN synthesizes the background and defect images output by the encoder, seamlessly creating defects within the background. The goal is to combine the two images, X_d and X_b, produced by the encoder representing the defect and background, respectively, to generate a final defect image. This generated image is then input into the GAN's discriminator for comparison with real defect images. The network structure of the generator is depicted in Figure 6.

The generator takes input defect and background images of size 512 × 512 × 1 and ultimately generates a single defect image of the same size. The generator is composed of convolution layers, ReLU activation functions, pooling layers, and transposed convolution layers as shown in Table 2. Since the encoder generates defects from the defect label and backgrounds from the background label, their labels or feature maps do not overlap during training. In contrast, the generator needs to create one image from two inputs, so a concatenate layer is added to combine the two feature maps. The 512 × 512 × 1 defect image X_d and background image X_b generated from the encoder are dimensioned through convolution layers to form 32 × 32 × 512 feature maps. Then, through concatenation, the two 32 × 32 × 512 feature maps are combined into a single 32 × 32 × 1024 feature map. The transposed convolution process of this combined feature map ultimately generates a natural combination of a 512 × 512 × 1 defect and background image. Similarly, the MSE loss is used as the loss function for synthesis, minimizing the difference between the ground truth defect image and the defect image generated through training in a progressive learning process. The detailed network structure of the generator, which synthesizes defect and background images, is as follows. The architectures for extracting features from defect images and background images are identical. Both networks transform the generated 512 × 512 × 1 defect and background images into 32 × 32 × 512 feature maps at the encoder. Each network consists of a total of 4 convolution groups. Each convolution group comprises one convolution layer and one ReLU function. The generator's convolution groups have lower complexity compared to the encoder, as the learning task is relatively simpler. Additionally, to expedite convergence, the filter size of the convolution layers was set to 4 × 4, and they were performed with a stride of 2. Applying zero padding in the convolution groups allowed for increasing the depth while simultaneously reducing the feature size. Through the four convolution groups, the input data size transitions from 512 × 512 × 1 to 32 × 32 × 512, with intermediate sizes of 256 × 256 × 64, 128 × 128 × 128, and 64 × 64 × 256. The defect and background images are concatenated into a single

TABLE 2. Configuration of the generator.

Layer Type	Filter Size	Stride	Output Size
Convolution	64 x 4 x 4	2	256 x 256 x 64
ReLU			
Convolution	128 x 4 x 4	2	128 x 128 x 128
ReLU			
...			
Convolution	512 x 4 x 4	2	32 x 32 x 512
Concatenate			32 x 32 x 1024
...			
Transposed Convolution	64 x 4 x 4	2	512 x 512 x 64
ReLU			
Convolution	1 x 4 x 4	1	512 x 512 x 1

32 × 32 × 1024 feature map through a concatenate layer, each generated through their respective networks, resulting in a 32 × 32 × 512 feature map. This combined feature map integrates the feature information of both defect and background images. Subsequently, the next network utilizes this feature map to generate defect images seamlessly onto the synthesized background. Similar to the previous network, this network consists of 4 convolution groups, with each convolution layer comprising transpose convolution and ReLU functions. The filter size of the transpose convolution is set to 4 × 4, with a stride of 2 and no zero padding, resulting in an increase in feature map size while reducing depth. Through the four convolution groups, the input data size transitions from 32 × 32 × 1024 to 512 × 512 × 64, with intermediate sizes of 64 × 64 × 512, 128 × 128 × 256, and 256 × 256 × 128. The final convolution layer generates the ultimate defect image from the feature map using a 4 × 4 filter and a stride of 1. Through this network, a final grayscale defect image of size 512 × 512 is ultimately generated.

In this paper, the aim is to augment the quantity of the entire battery dataset by generating 50,000 synthetic defect images from 10,000 battery defect images using the proposed defect-background separated GAN. Simultaneously, the quantity of each class is intended to be uniformly adjusted. As the segmentation performance of defects is proportional to the quantity and quality of the dataset, synthesizing the dataset becomes a crucial element in the segmentation training process. A single generator and discriminator are employed in the defect-background separated GAN. The generator of this GAN utilizes a total of 47 convolution layers, which is fewer than those found in open-source models like CycleGAN, StarGAN, and StyleGAN. Additionally, while high performance in other GANs is achieved through numerous generators, discriminators, residual blocks, or high-dimensional latent vectors, the defect-background separated GAN consists of relatively simpler networks. Consequently, it is expected that the number of training hyperparameters in the defect-background separated GAN is the lowest, leading to relatively faster training speeds.

The defect-background separated GAN is effectively trained using GPU (Graphics Processing Unit) for image processing. When not utilizing GPU, training relies on CPU memory, necessitating smooth performance coordination between the CPU and GPU. In this paper, GAN training was conducted on a workstation equipped with an i9 32GB CPU and an RTX 3080TI GPU. The defect-background separated GAN was developed programmatically using the Python programming language, which is widely used for building and training deep neural networks due to its rich AI libraries and various open-source resources. The proposed GAN in this paper defined network architecture and underwent training utilizing libraries such as TensorFlow, OpenCV, PyTorch, and scikit-learn. Furthermore, GPU acceleration libraries like NVIDIA CUDA and cuDNN were installed to leverage TensorFlow on GPU for parallel model training. Through such code optimization and parallel GPU setups, a development environment facilitating more convenient and efficient GAN training was established.

When training with the actual 10,000 defect images, additional image processing was conducted to augment the defect dataset, aiming to incorporate a wider range of defects into the training. Augmentation of the training dataset promotes performance enhancement and the generalization ability of the model. By applying various augmentation techniques, the diversity of data is increased, overfitting is mitigated, and the robustness of the model is improved. Moreover, effective learning from small datasets can enhance the model's performance. This allows for the development of models that can be more effectively applied in real-world environments. In the defect-background separated GAN, augmentation of defect images was performed by adding rotation, translation, scaling, inversion, and various noises, programmed and applied using the Python library OpenCV. Ultimately, to generate 50,000 synthetic defect images from the 10,000 defect images using the defect-background separated GAN, training was conducted for 1,000 epochs with a learning rate set at 0.005. Results from training various GANs on the same PC environment revealed that the net or ckpt file size from training with the defect-background separated GAN was the smallest, and the training speed was also the fastest. Detailed comparisons among GANs will be further discussed in the experimental chapter. The generated synthetic defect images are utilized for defect segmentation training. This synthetic defect dataset allows for defect generation with desired labels, enabling segmentation training without additional labeling work by workers, as labels and images are generated as a set. The final dataset of 60,000 defect images enhances defect detection performance and can be discussed with performance metrics.

B. TRAINING DEFECT SEGMENTATION

In the defect inspection process, segmentation is a crucial technique for accurately detecting the area, position, and size of defects. This technique proves particularly useful in image analysis tasks and offers significant advantages in applications such as battery exterior defect inspection. Segmentation is employed due to its ability to precisely extract object or defect, enabling accurate identification of their size and position. Unlike other image classification tasks, where determining the location and size of objects or defects can be challenging, segmentation processes each pixel individually, accurately classifying whether it belongs to a specific object or defect. Consequently, this approach provides precise information about the defect's location and size.

The basic structure of the segmentation network primarily relies on a CNN [30]. CNNs are particularly effective in image processing tasks, utilizing multiple convolutional layers and pooling layers to extract various features from images. Segmentation networks typically comprise two main stages: an encoder and a decoder. The encoder takes the image as input and extracts feature maps containing detailed information about the image. The decoder then utilizes the feature maps obtained from the encoder to perform pixel-wise segmentation of the image, assigning each pixel to its corresponding class or object. Segmentation training involves using labeled training data, which includes images along with pixel-wise class or defect information. The network is trained to minimize a loss function, enabling it to achieve accurate segmentation results.

We employed HarDNet, selected from a variety of open-source segmentation networks, for defect segmentation [31]. HarDNet has firmly established itself as a highly effective tool due to its unique features. One of the reasons for choosing this network is its rapid learning speed. Defect segmentation tasks are complex and computationally expensive, and a slow learning speed can prolong the time required for experiments and model development, posing challenges for application in actual battery production lines. HarDNet, short for harmonic DenseNet, is designed to optimize memory usage efficiently and make effective use of computational resources. This design enables the model to be effectively utilized in resource-constrained environments, providing a significant advantage by reducing the model size while maintaining result quality. Additionally, HarDNet helps reduce memory traffic while preserving defect segmentation performance. This effectively utilizes the network, lessening memory burden during both the training and inference phases while maintaining high accuracy. Figure 7 compares the performance and inference speed of various networks. The x-axis represents inference fps corresponding to training time, and the y-axis shows the average accuracy of segmentation. HarDNet, located in the top right corner of the chart, demonstrates the fastest speed among various networks while maintaining accuracy performance. HarDNet proves to be an ideal solution for computationally expensive tasks such as defect segmentation, and its combination of fast learning speed and outstanding performance makes it highly valuable for real-time tasks in both research and industrial settings.

Labeled training data is employed when conducting training for defect segmentation using HarDNet. This data comprises images along with pixel-wise class or defect information for each corresponding image. The labeling task is performed by operators before defect segmentation training, aiding in understanding and defining the data. This task helps determine the necessary information from each image in the dataset and precisely identifies which objects or areas need to be distinguished. Furthermore, labeling information is utilized for the training of the segmentation network. The segmentation network learns the relationship between the image and the labeled defect areas, enabling it to predict and locate defects in new images based on this acquired knowledge. Without labeling information, the model cannot comprehend which parts are defects, making it challenging to accurately segment defect areas or detect objects within the image. Labeling provides the model with precise indications of regions, enabling it to perform such tasks. Additionally, labeling information is crucial for evaluating and improving the model's performance. With accurate labeling information, the model's results can be compared to the actual defect locations to assess its accuracy.

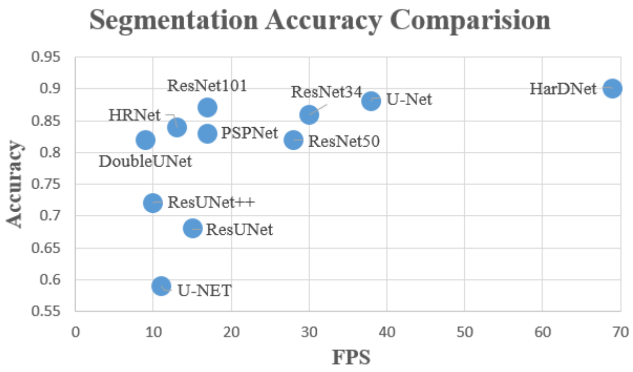
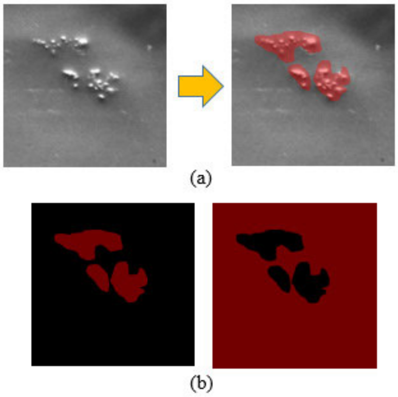


FIGURE 7. Segmentation performance comparison for various model.


(a)

(b)

FIGURE 8. (a) Labeling process, (b) Defect label and background label.

Figure 8 provides an example of the labeling performed for defect segmentation training. Workers use tools such as paint, Photoshop, or dedicated labeling programs to color the defect areas in the upper image set, as shown. This information is then utilized as labels for the defect and background parts, as illustrated in the lower image set. Typically, when conducting segmentation for defects, the training involves learning to output defect labels from images containing defects. The defect-background separated GAN, aiming for more precise feature learning by separating defects and backgrounds, undergoes training at the generator stage using both defect labels and background labels. The proposed GAN learns from a dataset labeled with defects in a semi-supervised format, storing the characteristics of defects and backgrounds separately. The generator refers to this information to synthesize defect and background components.

C. PERFORMANCE EVALUATION FOR GAN

Evaluating the performance of the defect-background separated GAN in generating synthetic defects poses a challenging task. Traditionally, the quality assessment of generated synthetic defects has relied on subjective evaluation by human assessors who visually inspect the output. However, this approach becomes impractical for large datasets. To overcome this limitation, objective performance metrics such as PSNR, SSIM, Fre̒chet inception distance (FID) score, and inception score are employed [32], [33]. PSNR and SSIM have traditionally been used in image and video compression and restoration processes. While PSNR and SSIM serve as metrics for measuring image quality, they may struggle to detect subtle improvements in image generation since they operate at the pixel level between images. Specifically, GAN-generated images may exhibit high resolution and intricate details that PSNR and SSIM might not effectively capture. For a comprehensive evaluation of GAN defect generation performance, the FID Score is considered the most widely used metric.

The FID score is a crucial metric for evaluating the quality of images generated using GANs. This metric quantifies the dissimilarity between real and generated images, providing a quantitative assessment of a GAN model's performance. The FID score captures the distinction between two key distributions: the distribution of real images and the distribution of generated images. To calculate the FID score, the process involves generating images using a GAN model tailored to produce images of a specific class. Subsequently, a pre-defined feature extractor, typically utilizing Inception v3, is employed to extract features from both the generated and real images. The FID score then calculates the Fre̒chet distance between these two feature distributions, offering an assessment of the quality of the generated images.

$$\mathrm{FID}\left(I_r, I_s\right) = ||\mu_r - \mu_s\| + TR\left(\Sigma_r + \Sigma_s - 2\left(\Sigma_r \Sigma_s\right)\right)^{1/2} \tag{2}$$

μ_r and μ_s represent the characteristics of the real and synthesized images, respectively, while Σ_r and Σ_s denote the covariance matrix values for the real and synthesized images, respectively. Equation (2) assesses the statistical difference between the two datasets by calculating variations in mean and covariance. These statistical characteristics reflect the diversity and quality of the datasets.

The evaluation methodology for GAN performance using FID score is as follows. Initially, real and synthetic images are collected, and crucial features are extracted from these images. Subsequently, the FID score is computed using the extracted features. This score is generally better when lower, indicating that a lower FID score corresponds to a smaller difference between real and generated images. This implies an enhancement in the GAN model's performance. FID scores serve as a valuable tool for quantifying and comparing the performance of GAN models. A high FID score may signal the need for further improvements in the GAN model, providing insights for directions to enhance and refine the model.

Next, to assess the effectiveness of the augmented dataset generated using GANs in real-world processes, it is crucial to quantitatively evaluate the segmentation performance of defects. In defect inspection through deep learning, classification or segmentation is commonly employed. In our battery production process, we conducted exterior defect inspection using segmentation, enabling the extraction of information about the presence and location of defects. Therefore, accuracy and intersection over union (IoU) become important metrics in the defect inspection process [34]. Accuracy measures how overall accurate the model's predictions are, providing an assessment of the overall performance of the model. However, in cases of class imbalance or specific scenarios, accuracy alone may not precisely reflect the actual performance of the model. Hence, in defect segmentation tasks, IoU plays a more crucial role. IoU quantitatively measures how well the model's predictions align with the actual object's location, evaluating how successfully the model delineates the boundaries of the objects.

Accuracy is a fundamental metric for evaluating the performance of a machine learning model, representing the ratio of correctly predicted results to the total number of predictions made by the model. It is expressed by Equation (3), where the numerator denotes the number of samples accurately predicted by the model, and the denominator represents the total number of samples predicted by the model. While accuracy is easy to comprehend and interpret, it may present challenges when dealing with datasets exhibiting imbalanced class distributions. For example, when one class contains significantly more samples than others, the model may exhibit a tendency to favor the majority class, leading to inflated accuracy. In such cases, relying solely on accuracy to evaluate model performance might not be appropriate, and other evaluation metrics should be considered. Therefore, while accuracy is valuable for assessing model performance, it should be considered alongside other metrics, especially in the presence of class imbalances or specific scenarios.

$$Accuracy = \frac{Number\ of\ Correct\ Prediction}{Total\ Number\ of\ Prediction} \tag{3}$$

IoU is a crucial evaluation metric for assessing the performance of models in image processing tasks, such as object segmentation. This metric measures how well the model's predicted results align with the actual location of objects. IoU is calculated by dividing the intersection of the predicted and ground truth areas by their union, as defined by Equation (4). Here, $Area_{Pred}$ represents the model's predicted segmentation area, and $Area_{GT}$ denotes the actual segmentation area or ground truth. The IoU value falls within the range of 0 to 1, where a higher IoU indicates a closer match between the model's prediction and the actual object location. Typically, an IoU value exceeding 0.5 is considered indicative of an accurate prediction by the model. A lower IoU implies that the model fails to precisely delineate the object's boundaries, indicating a lower quality of segmentation. IoU is widely used in various image processing tasks, including object detection, segmentation, and tracking, providing a useful measure of a model's accuracy and precision. In particular, in object segmentation tasks, IoU serves as a crucial metric for assessing how successfully a model separates the boundaries of objects.

$$\mathrm{IoU} = \frac{Area_{Pred} \cap Area_{GT}}{Area_{Pred} \cup Area_{GT}} \tag{4}$$

By concurrently employing the two metrics mentioned earlier, it becomes possible to comprehensively evaluate the model's performance. Accuracy assesses overall correctness, while IoU measures the accuracy of object segmentation. The use of both these metrics enhances the model's performance, enabling the achievement of high accuracy and precision in defect inspection tasks.

IV. EXPERIMENT

A. DEFECT INSPECTION SYSTEM

In this paper, a defect-background separated GAN was applied to conduct defect inspection using the structure depicted in Figure 9 on an actual battery production line. The process of deep learning-based defect inspection can be broadly divided into offline and inline components. The offline process involves initially training the segmentation network for defect inspection. Images of battery objects, which are the inspection targets, are captured from line-scan cameras, and a labeling process is conducted to identify the defects present in these images. Subsequently, the object images and labeling information are input into HarDNet to train the segmentation network for defect inspection. This model performs segmentation by taking images as input and outputting defect information. The defect-background separated GAN is incorporated to enhance the performance of defect segmentation and generate a large, uniform dataset for additional training. The optimized and trained model is then utilized in the inline process, which involves inspecting defects in the actual objects during the real production process. Images of the inspection targets are acquired from line-scan cameras, and the previously trained segmentation network is applied to inspect defects in these images. Finally, the type, size, and location of defects are output as inspection results.

Various metrics were utilized to assess the performance of the defect inspection system in the described process and the proposed defect-background separated GAN. The quality of the constructed large and uniform dataset in the offline process was scrutinized, and the performance of the segmentation network was enhanced through additional training on this dataset. The accuracy and IoU of the segmentation network were analyzed to benchmark its performance against alternative approaches. Ultimately, the outcomes of deploying deep learning-based defect inspection with the defect-background separated GAN on an actual battery production line were summarized, affirming the effectiveness and robustness of this approach.

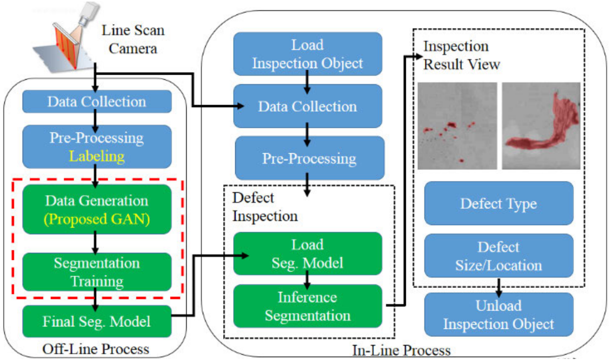


FIGURE 9. Defect inspection system.

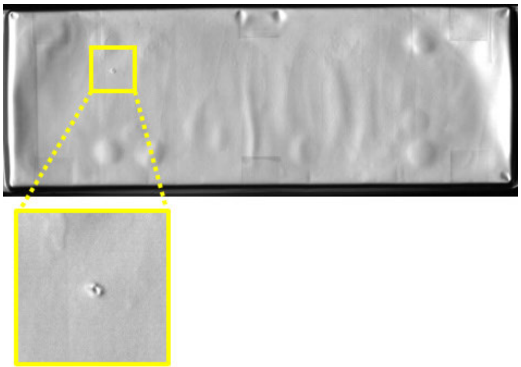


FIGURE 10. Pouch-type battery image and grid.

B. DEFECT DATASETS

The product targeted for defect inspection in this paper is pouch-type batteries. Pouch-type batteries are high-performance batteries primarily utilizing lithium-ion technology, characterized by their flat and lightweight design. Due to this design, these batteries effectively utilize space and offer high capacity with a high energy density, making them widely used in various fields. In recent years, the electric vehicle industry has experienced rapid growth, primarily driven by efforts to provide an environmentally friendly driving experience, with high-performance batteries being a key component of these efforts. Pouch-type batteries are crucial components in the battery packs of electric vehicles, and as the market share of electric vehicles continues to rise, there is a growing demand for improved efficiency in battery production processes. Enhancing the efficiency of these battery production processes is considered a key element supporting the sustained growth and development of the electric vehicle industry.

The pouch-type batteries used for inspection have dimensions of 2700 × 1000 mm, and the images captured through the optical system have a pixel size of 5400 × 2000, as illustrated in Figure 10. Inputting the entire 5400 × 2000-pixel image into the segmentation network at once poses challenges in terms of memory and processing time. Additionally, the characteristics of deep neural networks, which involve downsizing large images while extracting features, can result in significant resolution loss, proving disadvantageous for training. To mitigate these issues, images were acquired by moving a 512 × 512 size grid throughout the entire inspection area, and training and testing were performed based on the generated 512 × 512 size images. In the training of defect segmentation, only defects with sizes larger than 10 mm (20 pixels) were considered for learning and testing, as it is challenging to confidently determine smaller defects as definite faults.

The major defects addressed in the inspection of battery appearance, as illustrated in Figure 3, can be broadly categorized into six types. These appearance defects on battery exteriors may indicate the potential for battery malfunctions. Consequently, an initial visual inspection for exterior defects is conducted to classify battery cells with potential defects, followed by a subsequent conductivity test to further assess battery malfunctions. The segmentation network used for defect inspection plays a crucial role in accurately identifying the defective parts of the battery by analyzing its image. To maximize the performance of this model, a large and uniform dataset is essential. However, practical defects occurring in battery production lines can vary widely depending on the type and may arise in uneven proportions. This non-uniform distribution is a significant challenge that can compromise the accuracy of deep learning-based defect inspection. To address this issue, we developed and applied a defect-background separated GAN to generate a large and uniform battery dataset. This dataset, constructed in this manner, considers various defect types while maintaining a uniform distribution of defect types to ensure that the trained model maintains high accuracy in diverse situations. The task of distinguishing and classifying defects is a crucial step in training deep learning models, and the composition of the dataset for this purpose plays a vital role in enhancing the model's performance.

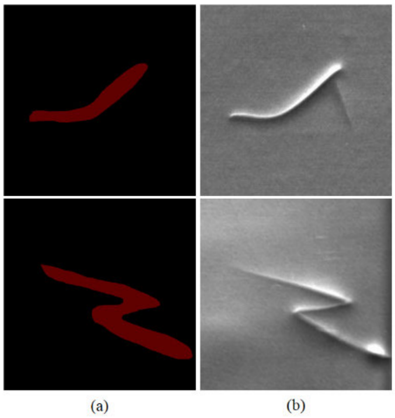


FIGURE 11. **(a) Custom defect label, (b) Synthetic defect.**

Another functionality of the defect-background separated GAN is its capacity to enable users to generate defects with the desired shape and background. The proposed GAN is trained in a semi-supervised format using a labeled dataset of defects that captures the characteristics of both defects and backgrounds. This information is referenced during the generation of synthetic images by the generator. An additional feature of the defect-background separated GAN is its ability to seamlessly synthesize defects and backgrounds corresponding to the user's desired shape and background. Users can input the desired background and shape of the defect into the latent matrix, which serves as the input to the GAN. This results in the generation of defects that naturally integrate with the specified background. Figure 11 depicts an example image generated by the model based on the user's desired defect shape.

Utilizing the defect-background separated GAN, we generated a dataset in the quantities outlined in Table 3. In the existing battery production line, datasets tend to be biased toward more common defects. Specifically, defects like B, E, and F have significantly low occurrences, making it challenging to ensure segmentation performance even with their inclusion in the training set. To address this issue, we synthetically generated additional instances of these less frequent defects to align both the absolute quantity and relative proportions of all defects. In this study, we augmented the dataset by generating an additional 50,000 synthetic defect images, supplementing the initial set of 10,000 real defect images, resulting in a total of 60,000 defect images for training and testing. For training, a learning rate of 0.005 was set, and the model underwent 1,000 epochs. Ultimately, by applying the defect-background separated GAN to a limited and imbalanced dataset, we aimed to create a large and balanced dataset, thereby enhancing the performance of battery exterior defect inspection.

C. PERFORMANCE MEASURES

Synthetic battery defect images generated using various GANs are illustrated in Figure 12. When utilizing the defect-background separated GAN, it demonstrates a more efficient defect generation capability, allowing users to create defects according to their preferences. Typically, images generated from GANs are created using random latent vectors, making it challenging for users to directly produce the desired defects, and there is a possibility of generation failure. In contrast, the defect-background separated GAN can generate sharper defects without failure, as it is based on the input latent matrix provided by the user.

The FID scores are applied to evaluate the defect images generated using the defect-background separated GAN. Table 4 presents the analysis results of the FID score for various GANs [35], [36], [37]. The defect-background separated GAN achieved the lowest FID scores for all defects

TABLE 3. Composition of train data.

Defect Amount	Class A	Class B	Class C	Class D	Class E	Class F
Dataset (Before)	2,200 (22%)	200 (2%)	2,300 (23%)	4,600 (46%)	400 (4%)	300 (3%)
Dataset (After)	10,000 (16.67%)	10,000 (16.67%)	10,000 (16.67%)	10,000 (16.67%)	10,000 (16.67%)	10,000 (16.67%)

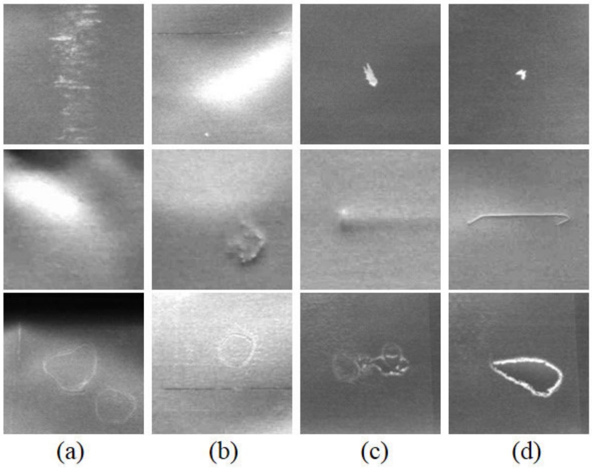


FIGURE 12. Synthetic image using (a) CycleGAN, (b) StarGAN, (c) StyleGAN, (d) Proposed GAN.

except class F. This indicates that, overall, it provides the best quality and diversity of defects for most types. Examining the FID scores for each class demonstrates superior performance improvement, particularly for defects in classes B, C, and D, which had high FID scores in other GANs. As classes B, C, and D involve complex shapes among defects, they generally yield high FID scores. Through the defect-background separated GAN, the generation of the most precise synthetic defects is achieved. For the remaining defects in classes A, E, and F, which have relatively simple shapes, the FID scores are generally lower, and the differences in FID scores between each GAN are minimal. Using the defect-background separated GAN to separately learn the characteristics of defects and background, and generating defects matching the specified latent matrix by the user has effectively enhanced the defect generation performance.

The training speed of GANs for generating synthetic defect images is also a crucial factor to consider. As the available time for training and the resources of PCs or servers are limited in industrial settings, a GAN that can maintain defect generation performance while achieving fast training speed is considered practical for real-world use. Table 5 compares various open-source GANs trained in the same environment. To facilitate diverse GAN training, an equal number of 10,000 defect images were prepared, and for CycleGAN, an additional set of 10,000 normal images was prepared since it requires both defect and normal images as a set. When comparing the model sizes, it is evident that the defect-background separated GAN has the smallest capacity. Generally, a larger model size indicates that the model is deep and complex, with many hyperparameters to train. StarGAN and StyleGAN, having various domains and conditions for generators and discriminators, and incorporating additional CNNs for generating high-quality images, are heavier models compared to other GANs. These GANs with such characteristics were trained in an i9 32GB CPU, RTX 3080TI GPU environment. The training time tends to be influenced by the complexity of the model and the quantity of training data. In comparison to other GANs, the defect-background separated GAN demonstrates a fast training speed of 26.8 min/epoch. This can be attributed to the proposed GAN using a simple single generator and discriminator and having a relatively small amount of defect data required for training. Moreover, employing a semi-supervised GAN format, which provides additional input labeling for the input defect during GAN training, facilitated faster and more stable training. This approach eliminated the need for the GAN to generate all the information for each defect.

To assess the effectiveness of the augmented dataset using the defect-background separated GAN in actual processes, it is crucial to quantitatively understand the segmentation performance of defects. In this paper, to evaluate the efficiency of the defect-background separated GAN, we analyzed the accuracy of defect segmentation before and after dataset augmentation. Table 3 presents the dataset composition ratios before and after dataset augmentation through GAN. Through the defect-background separated GAN, the overall quantity of defects was augmented, aiming to generate a uniformly distributed number of defects for each class and avoid biasing the segmentation network towards one class of defects. Each GAN was utilized to train the segmentation network, HarDNet, with a total of 60,000 images. For the HarDNet segmentation training, the settings included a learning rate of 0.001, a batch size of 16, and 1,000 epochs.

Figure 13 illustrates the accuracy performance of defect segmentation when trained on defects generated using various GANs. When trained on the uneven and limited dataset before dataset augmentation, the overall defect detection accuracy across classes is low, with particularly lower accuracy for Classes B, E, and F, which had a smaller training amount. This demonstrates, as mentioned earlier, that an uneven and limited dataset degrades the performance of deep learning-based defect inspection. To enhance defect inspection accuracy, synthetic defects were

TABLE 4. Comparison of FID score.

FID Score	Class A	Class B	Class C	Class D	Class E	Class F
CycleGAN	84.6	200.1	206.8	100.6	64.8	63.8
StarGAN	78.3	151.5	214.9	105.7	57.4	59.4
StyleGAN	69.4	142.2	178.8	97.1	58.1	57.1
Proposed GAN	66.4	92.5	121.3	79.5	55.4	58.7

TABLE 5. Comparison of training time.

	CycleGAN	StarGAN	StyleGAN	Proposed GAN
Model Size (MB)	43.4	170.8	288.5	38.7
Train Dataset Amount	2N	N	N	N
Training Time per Epoch (Min/Epoch)	38.4	58.2	71.9	26.8

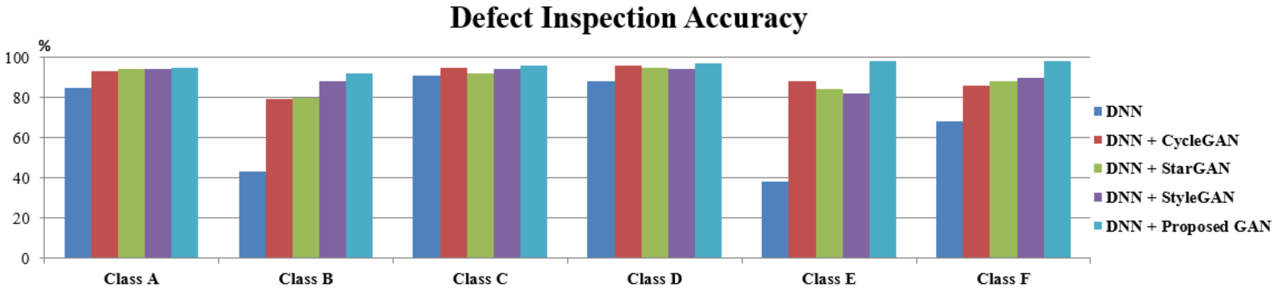


FIGURE 13. Comparison of defect inspection accuracy.

generated through CycleGAN, StarGAN, StyleGAN, and the defect-background separated GAN, forming a uniformly large dataset for defect segmentation training. As a result, there was an overall increase in accuracy for battery defect inspection across all GANs. The accuracy improvement was particularly notable for Classes B, E, and F defects, which initially had low accuracy values. Constructing a large dataset and optimizing its composition ratio is a fundamental approach to maximizing the performance of defect inspection in deep learning-based systems. Specifically, in cases where images were trained from defects generated by the defect-background separated GAN, which exhibited the highest quality of synthetic defects, the accuracy was the highest.

IoU measures how accurately the model represents the defect area on a pixel-by-pixel basis. The IoU value ranges from 0 to 1, with a generally positive assessment of the model's performance when the IoU value is above 0.5. A higher IoU implies that the segmentation network more accurately predicts the defect area on a pixel-by-pixel basis. A lower IoU

indicates that the segmentation results do not precisely delineate the boundaries of the defect, signifying a decrease in the quality of the segmentation task. Table 6 presents the average IoU values when defects generated by various GANs were used for segmentation training. The IoU values were lowest when segmentation training was performed using a small and uneven dataset. Augmenting the defect dataset using GANs showed an overall increase in IoU values. In particular, defects amplified using the proposed defect-background separated GAN exhibited higher FID scores and sharpness, resulting in even higher IoU values. The defect-background separated GAN, which references and separates defect and background labeling during training, demonstrates a more precise pixel-wise segmentation analysis, showcasing its ability to achieve a higher level of accuracy.

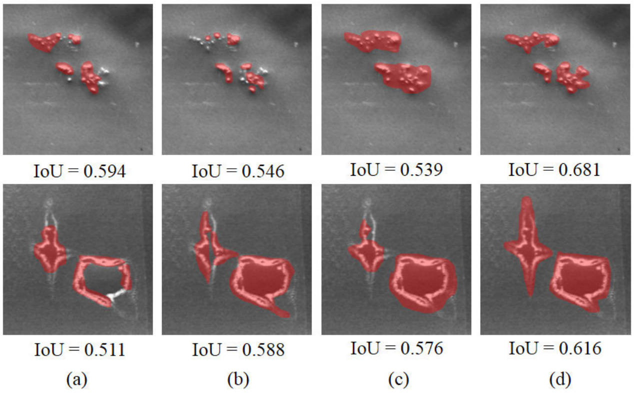


FIGURE 14. IoU result using (a) CycleGAN, (b) StarGAN, (c) StyleGAN, (d) Proposed GAN.

Figure 14 illustrates the defect detection results for the same image when segmentation training was conducted using various GANs. This particular image was chosen to highlight

TABLE 6. Comparison of IoU.

	HarDNet	HarDNet +CycleGAN	HarDNet +StarGAN	HarDNet +StyleGAN	HarDNet +Proposed GAN
IoU	0.5918	0.6506	0.6753	0.6819	0.7061

the most significant variations in segmentation detection performance across different GANs. Defects with simpler shapes in classes A, E, and F exhibited minimal differences in IoU values regardless of the GAN employed. The segmentation analysis was primarily focused on defects in classes B, C, and D, where the complex shapes allowed for the observation of changes in IoU values. Among the diverse GANs, those generated by the defect-background separated GAN most accurately captured the synthetic defect images, visually showcasing the highest IoU values in segmentation results.

Table 7 below summarizes the IoU analysis results for each class. When employing traditional fixed-parameter machine vision analysis programs, they often exhibit vulnerability in detecting defects with varying brightness, shape, and color. In the testing of various defects, IoU values generally fall below 0.5, indicating challenges in detecting defects with diverse characteristics. Performing defect inspection using the small and imbalanced dataset before augmentation results in an overall increase in IoU values compared to machine vision programs. However, this phenomenon is more pronounced for classes A, C, and D, where there is sufficient training data, while the increase in IoU values is less prominent for classes B, E, and F, which have relatively fewer training examples. Ultimately, employing the defect-background separated GAN to generate a large and uniform defect dataset for segmentation leads to an increase in IoU values for all defects. Particularly, for defects in classes B, E, and F, where the quantity of defects was limited, successful IoU values are achieved. This suggests that augmenting datasets through the defect-background separated GAN has a positive impact on the efficiency and performance enhancement of deep learning-based defect inspection.

In this study, a defect-background separated GAN was proposed to enhance battery exterior defect inspection. Similarly, various GANs and CNNs are being researched to improve defect detection performance across diverse industries such as construction, agriculture, apparel, and more. Recent studies include the development of various GANs and CNNs for defect detection, including widely compared models like CycleGAN, StarGAN, and StyleGAN. This chapter compares and analyzes the performance of the proposed GAN model with the latest defect detection GANs and CNNs. Specifically, we investigated cases where defects occur on backgrounds using datasets resembling characteristics similar to battery defects, such as apparel, leather, and road datasets. Such research aims to provide practical solutions for real-world industrial challenges. This experiment primarily focused on evaluating GAN models for defect detection, with an emphasis on GAN models with code implementations.

First, Defect-GAN introduces a strategy based on layer composition to generate real defects with various textures and appearances in diverse image backgrounds [38]. This defect generation capability is useful for mimicking various defects found in actual industries. Moreover, it can simulate random variations of defects and flexibly control the location and category of defects in the image background, aiding in the inspection of various defects occurring in real-world environments. Next, GAN and CNN training methods for detecting pavement cracks from small samples were investigated [39]. This approach involves using images generated by the GAN model to expand the original small sample dataset and simultaneously constructing a CNN model for detecting defects. Transfer learning methods were employed to train the defect dataset for defect inspection. Such methods are effective in detecting defects even with limited datasets and have garnered significant interest in the manufacturing industry.

Table 8 presents the results of performing battery exterior defect inspection using various GANs and evaluating their performance. For the segmentation learning of battery exterior defects, HarDNet was used as the base model and experiments were conducted accordingly. Additionally, the defect dataset was augmented using Defect-GAN, pavement crack GAN, and background-defect separated GAN, followed by segmentation learning of defects using HarDNet. In cases where only HarDNet was used for defect learning, without the use of GANs for synthesis, FID scores and GAN training time were not considered. The performance of defect inspection for four methods was further evaluated. It was observed that applying GAN to augment the defect dataset significantly increased accuracy compared to HarDNet learning without GAN. It was evident that both dataset quantity and quality were crucial factors for defect segmentation learning, with GAN application showing higher IoU values. Augmenting the dataset ultimately increased the quantity of trainable defect images and their corresponding labeling, thereby enhancing both defect classification accuracy and defect area detection performance. The proposed defect-background separated GAN exhibited the most significant improvements in both accuracy and IoU, indicating its ability to generate the most realistic synthetic defect images by separately learning defects and backgrounds. This performance evaluation was also reflected in the FID scores, where a lower score indicates superior performance in GAN's synthetic defect generation. Although defect-background separated GAN showed the lowest FID score, it required more training time compared to other Defect-GAN and pavement crack GANs. This implies that while the networks of Defect-GAN and pavement crack GAN are relatively simple,

TABLE 7. Comparison of IoU by defect.

	Class A	Class B	Class C	Class D	Class E	Class F
Machine Vision	0.2618	0.2815	0.3097	0.2695	0.3504	0.3118
HarDNet	0.6862	0.4916	0.6924	0.6948	0.4917	0.4941
HarDNet+Proposed GAN	0.6948	0.7211	0.7119	0.7043	0.6925	0.7121

requiring less time for feature learning, their defect generation performance is inferior. Overall, when considering the results of comparing CycleGAN, StarGAN, and StyleGAN experiments, defect-background separated GAN proved to be the most efficient in terms of performance and reasonable in terms of time. Due to its characteristic of separately learning backgrounds and defects, defect-background separated GAN is effective for defects such as those in battery datasets where defects occur in specific backgrounds. Additionally, the ability for users to select the shape and type of defects is advantageous not only in research but also in industrial applications.

Using the defect-background separated GAN to generate additional synthetic defects can be seen as enhancing defect images to create defects under more diverse conditions. Since the quality and quantity of defect images directly influence defect detection training, this paper ensures diversity in images through two main approaches. Firstly, by leveraging the proposed GAN to generate defect images anew. In actual battery production processes, the occurrence frequency and quantity of defects may vary depending on the defect type. The GAN compensates for such variations, efficiently balancing the quantity and distribution of defects. Secondly, augmentation through image processing during HarDNet segmentation training. This method involves rotations, translations, flips, resizing, brightness adjustments, and noise addition, among others. Such augmentations alter defect image characteristics through real defect image processing, enabling rapid generation of diverse defect images. However, as the algorithm does not consider defect characteristics during image processing, synthetic defects may not resemble actual defects. Therefore, this paper emphasizes the use of defect-background separated GAN to mass produce synthetic defects resembling real defects, and applies various image processing techniques to maximize dataset augmentation performance in segmentation based defect detection. Through this approach, dataset quantity and quality are improved, leading to enhanced performance of defect detection models.

To generalize the application of defect-background separated GAN and image processing, it is crucial to assess how well the initially occurring defects can be detected. Separating the defects used for training from those unused, and evaluating the defect detection performance on various unused defects, is essential to verify their applicability in real industrial processes. Figure 15 demonstrates the results of applying various image processing techniques to unused battery defects for inspection. By applying various image processing techniques to unused defects, the aim is to simulate new measurement environments or new defects. This visually confirms how robust defect detection is for newly occurring defects in diverse environments. These images were selected from the test set used in performance measurement experiments, and defect inspection tests were performed by applying image processing such as rotation, flipping, resizing, and brightness adjustment to actual defects. The defect segmentation areas in Figure 15 show accurate IoU for all defects regardless of the image processing technique. Particularly, with an average IoU of over 0.7 on the test set, considerable overlap with the ground truth of actual defects is confirmed. This verifies the reliable defect inspection capability of the proposed method even for battery defects with different characteristics measured in diverse environments. Additionally, datasets with entirely different characteristics from battery defects were also trained and evaluated, the details of which can be found in Section E.

D. PRODUCTION LINE APPLICATION

The segmentation network, trained on the extensively augmented and uniformly labeled defect dataset generated through a defect-background separated GAN, was applied to the actual battery production line. Figure 16 illustrates the structure of the defect inspection system before and after implementing the deep learning-based defect inspection system in the battery production line. The provided values represent compiled results from the defect determination outcomes over one month on two battery production lines. Before introducing the deep learning-based defect inspection, eight workers spent 12 hours a day visually inspecting exterior defects on batteries. Initially, out of the total 100% of battery cells, 81% were classified as acceptable, and 19% were identified as defective for subsequent testing. Upon the introduction of deep learning-based defect inspection, a two-step process was adopted: the primary stage involved the application of deep learning-based defect inspection, followed by a secondary visual inspection conducted by workers for verification. Examining the defect inspection results, it can be observed that out of 100% of battery cells inspected, 72% were classified as acceptable, and 28% were deemed defective. Considering that falsely classifying acceptable products as defective is a more significant issue in deep learning-based defect inspection, the threshold score for defects during segmentation network training and testing was set low to increase the detection of defects. In the second visual inspection stage, workers conducted visual inspections on the 28% of products identified as defective in the first stage of deep learning-based defect inspection. This resulted in a final determination of an additional 9% as acceptable, maintaining the original 19% as defective. Consequently,

TABLE 8. Comparison of GANs.

	Accuracy (%)	IoU	FID Score	Training Time/Epoch (Min/Epoch)
HarDNet (Baseline)	68.3	0.5918	-	-
Defect-GAN	91.6	0.6887	115.6	22.9
Pavement Crack GAN	85.9	0.6408	127.1	18.3
Proposed GAN	96.1	0.7061	78.9	26.8

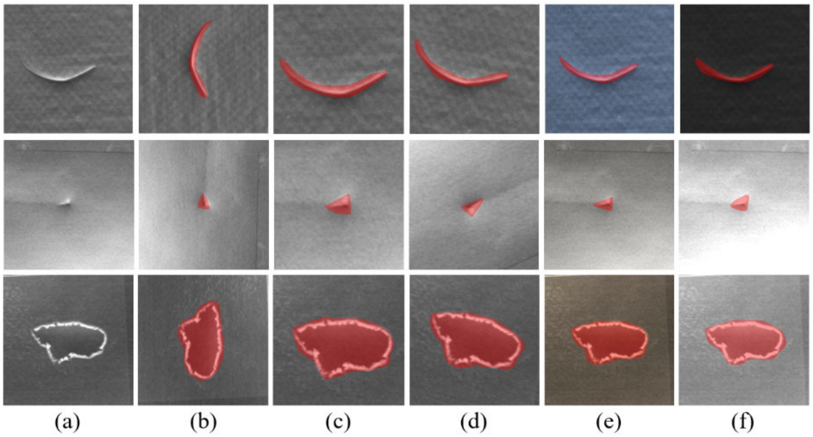


FIGURE 15. Defect segmentation result for image processing: (a) Original, (b) Rotation, (c) Scaling, (d) Shearing, (e) Coloring, (f) Brightening.

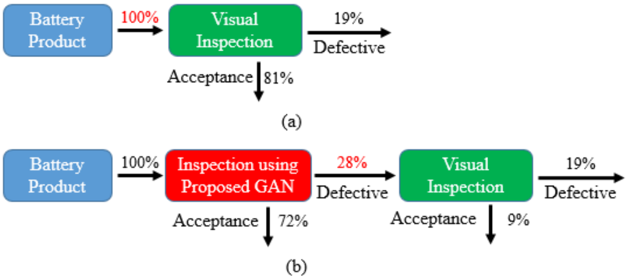


FIGURE 16. Inspection flowchart: (a) Before applying proposed GAN, (b) After applying proposed GAN.

the workload for visual inspectors reduced, and in practical terms, the line was operated with two workers, each working 12 hours a day. The defect inspection system proved to be efficient in long-term personnel management, providing consistent inspection results. This highlights the stability and robustness of the extensively augmented and uniformly labeled defect dataset generated through a defect-background separated GAN. Furthermore, it emphasizes the crucial role of such datasets in enhancing the performance of defect inspection.

E. OTHER DATASETS

The proposed defect-background separated GAN is trained using various open-source datasets other than the battery defect dataset to evaluate scalability. This allows evaluation of the model's inspection accuracy and performance. In this experiment, open datasets such as BTAD, ELPV, VISA, MVTec AD, MVTec LOCO AD, KolektorSDD2, NEU_Cluster, and CODEBRIM are considered. Additional experiments are conducted using datasets providing a sufficient amount of data for training and including segmentation labeling of surface defects among various datasets. In this chapter, the focus is on evaluating the scalability of the defect-background separated GAN using the MVTEC AD and KolektorSDD2 datasets. MVTEC AD contains high-resolution images captured in real manufacturing environments, capturing various types of anomalies occurring in diverse manufacturing processes. Additionally, the KolektorSDD2 dataset, provided by the Kolektor Group, is a surface defect detection dataset containing various types of defects. Both datasets provide labeling of defect images and defect areas as shown in Figure 17. Through these experiments, assessment of the generalization ability of the proposed model and its applicability in real manufacturing environments is possible, further substantiating the research findings.

The defect-background separated GAN was trained using the defect images and labeling information from the MVTec AD and KolektorSDD2 datasets. Similar to the training method for the battery dataset, the defect-background separated GAN was applied to separate the background and defects for training and aimed to generate new synthetic defects. For training, a learning rate of 0.005 was set, and training was conducted for 1,000 epochs. The training of the proposed GAN increases the quantity of the entire defect images and adjusts the relative balance between dataset classes. As shown in Figure 18 below, training on the MVTec AD and KolektorSDD2 datasets effectively generates new

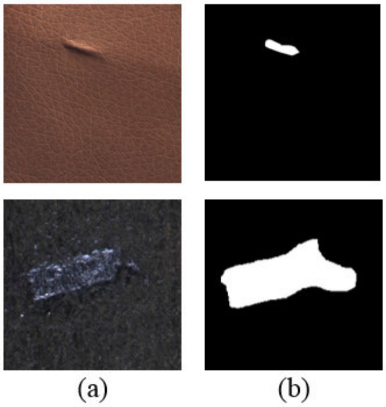


FIGURE 17. MVTec AD and Kolektor dataset (a) Defect, (b) Labeling.

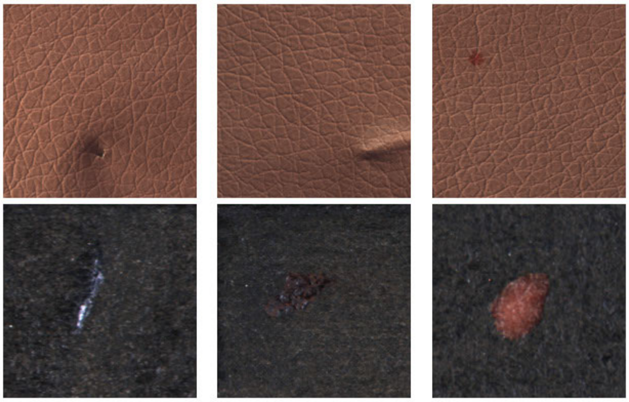


FIGURE 18. Synthetic defect for MVTec AD and Kolektor dataset.

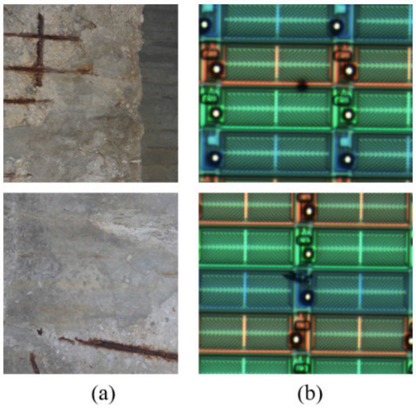


FIGURE 19. Synthetic defect images (a) CODEBRIM, (b) Display panel.

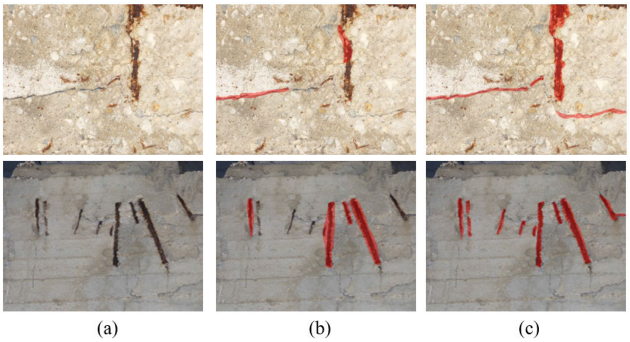


FIGURE 20. CODEBRIM dataset segmentation result: (a) Original, (b) Before applying GAN, (c) After applying GAN.

synthetic defects. To confirm the general enhancement of defect detection ability by the proposed GAN, amplified defect datasets were trained using the defect-background separated GAN, and the defect detection performance before and after amplification were analyzed. For defect segmentation training, HarDNet was used with a learning rate of 0.001, batch size of 16, and 1,000 epochs, similar to the battery dataset. Table 9 shows the results for defect detection accuracy and average IoU before and after dataset amplification. It can be observed that the defect detection accuracy increased in both datasets. Particularly, for the KolektorSDD2 dataset, where the characteristics of background and defects were relatively ambiguous, the increase in IoU is more pronounced. Since defect segmentation training is proportional to the quantity and quality of the training dataset, the dataset amplification method using the defect-background separated GAN positively impacts defect detection performance. Moreover, the proposed GAN can be generalized to apply to surface defects with defects in the background, and it can be robustly utilized across various datasets.

The investigation into the extended applicability of the proposed defect-background separated GAN involved utilizing datasets beyond battery datasets with characteristics similar to MVTecAD and Kolektor datasets. These datasets typically feature defects occurring in relatively simple backgrounds. For this evaluation, datasets other than the battery dataset were employed to assess defect detection performance. Firstly, the CODEBRIM dataset, a multi-target multi-class concrete defect dataset, was utilized. It classifies defects into five categories: crack, spallation, efflorescence, exposed bars, and corrosion. This dataset offers both image patches for multi-label classification and full-resolution images with cropped patches. Moreover, it presents more diverse backgrounds compared to the battery dataset, reflecting complexity in industrial environments. Additionally, the display panel dataset, comprising defect images from actual display panel production, was utilized. The majority of defects in this dataset appear as black foreign substances due to the nature of defect measurement methods. Given its complex backgrounds where defects occur, this dataset was deemed suitable for further comparative analysis. To train these datasets, types of defects and backgrounds were classified and labeled using the commercial labeling program Labelme. Through this exploration, we aimed to assess the diverse potential applications of the proposed GAN model and its relevance in real-world industrial settings.

Like other datasets, each labeled dataset was utilized to generate additional synthetic defect images using the defect-background separated GAN. The learning rate was

TABLE 9. Performances for other datasets.

Datasets	MVTec AD	MVTec AD +Proposed GAN	KolektorSSD2	KolektorSSD2 +Proposed GAN
Accuracy (%)	92.4	95.1	73.7	87.2
IoU	0.6489	0.7018	0.5196	0.6804

set to 0.005, and training was conducted for 1,000 epochs. The training of the proposed GAN increases the quantity of overall defect images and adjusts the relative balance between dataset classes. The newly generated defect images are depicted in Figure 19. It's evident that the synthetically generated defects closely resemble actual defects visually. The amplified defect images via GAN are further used for defect segmentation training through HarDNet. Figure 20 illustrates the defect detection regions before and after applying the defect-background separated GAN for comparison of defect detection performance. Images showing significant differences in results among the test images of the CODEBRIM dataset were selected. After applying the defect-background separated GAN, it's observed that the defect regions are more accurately delineated for all defects. Similar to other datasets, defect segmentation training for defect detection is proportional to the quantity and quality of the training dataset. Therefore, the dataset augmentation method using the defect-background separated GAN positively impacts defect detection capabilities. Through application and validation experiments on various datasets, it's confirmed that the defect-background separated GAN can effectively be applied not only to limited datasets but also to complex cases in industrial settings. In conclusion, this research demonstrates that the defect-background separated GAN operates reliably in industrial environments and has a broad applicability range. This model can be effectively utilized across diverse datasets and contribute to solving complex defect detection problems in industrial settings. Thus, these results underscore the efficacy of the defect-background separated GAN as a powerful tool applicable to defect detection and segmentation in engineering and industrial fields.

V. CONCLUSION

In this paper, we propose a defect-background separated GAN to enhance the performance of battery exterior defect inspection. Typically, the performance of the segmentation network used for battery exterior defect inspection is directly proportional to the quantity, uniformity, and quality of the training data. The defect-background separated GAN contributes to constructing a balanced dataset by increasing the overall quantity of the defect segmentation dataset, addressing the shortage of defect instances. Notably, a significant advantage of the defect-background separated GAN is its ability to allow users to customize the types and shapes of defects and backgrounds. The quality of generated synthetic defect images is analyzed through a comparison of FID scores. The results of defect inspection, augmented by incorporating synthetic defect images into the training, are further analyzed using accuracy and IoU. Experimental evaluations were conducted by generating synthetic defects using various GANs, aiming to replace conventional methods currently utilized in real battery production lines. The generated images exhibited the lowest FID score, and the performance of the trained defect inspection model demonstrated an accuracy of 96.1% and an IoU of 0.71. The ability to selectively synthesize additional data that was lacking in actual industrial settings proved to be particularly effective in defect segmentation training. Through on-site application experiments, the defect-background separated GAN demonstrated the stability and robustness of generating a massive and uniform defect dataset. These experimental results validate the practical applicability of the developed system, highlighting its potential impact on manufacturing and quality improvement.

REFERENCES

[1] K. K. Patel, A. Kar, S. N. Jha, and M. A. Khan, "Machine vision system: A tool for quality inspection of food and agricultural products," *J. Food Sci. Technol.*, vol. 49, no. 2, pp. 123–141, Apr. 2012, doi: 10.1007/s13197-011-0321-4.

[2] J. I. Umar Shahbaz Khan, "Automatic inspection system using machine vision," in *Proc. 34th Appl. Imag. Pattern Recognit. Workshop (AIPR)*, Oct. 2005, p. 217, doi: 10.1109/aipr.2005.20.

[3] Q. Zhou, R. Chen, B. Huang, C. Liu, J. Yu, and X. Yu, "An automatic surface defect inspection system for automobiles using machine vision methods," *Sensors*, vol. 19, no. 3, p. 644, Feb. 2019, doi: 10.3390/s19030644.

[4] R. Ren, T. Hung, and K. C. Tan, "A generic deep-learning-based approach for automated surface inspection," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 929–940, Mar. 2018, doi: 10.1109/TCYB.2017.2668395.

[5] X. Zheng, S. Zheng, Y. Kong, and J. Chen, "Recent advances in surface defect inspection of industrial products using deep learning techniques," *Int. J. Adv. Manuf. Technol.*, vol. 113, nos. 1–2, pp. 35–58, Mar. 2021, doi: 10.1007/s00170-021-06592-8.

[6] G. Yang, K. Liu, J. Zhang, B. Zhao, Z. Zhao, X. Chen, and B. M. Chen, "Datasets and processing methods for boosting visual inspection of civil infrastructure: A comprehensive review and algorithm comparison for crack classification, segmentation, and detection," *Construction Building Mater.*, vol. 356, Nov. 2022, Art. no. 129226, doi: 10.1016/j.conbuildmat.2022.129226.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.

[8] D. Mery and D. Filbert, "Automated flaw detection in aluminum castings based on the tracking of potential defects in a radioscopic image sequence," *IEEE Trans. Robot. Autom.*, vol. 18, no. 6, pp. 890–901, Dec. 2002, doi: 10.1109/TRA.2002.805646.

[9] J. P. Yun, W. C. Shin, G. Koo, M. S. Kim, C. Lee, and S. J. Lee, "Automated defect inspection system for metal surfaces based on deep learning and data augmentation," *J. Manuf. Syst.*, vol. 55, pp. 317–324, Apr. 2020, doi: 10.1016/j.jmsy.2020.03.009.

[10] O. Badmos, A. Kopp, T. Bernthaler, and G. Schneider, "Image-based defect detection in lithium-ion battery electrode using convolutional neural networks," *J. Intell. Manuf.*, vol. 31, no. 4, pp. 885–897, Apr. 2020, doi: 10.1007/s10845-019-01484-x.

[11] T. Czimmermann, G. Ciuti, M. Milazzo, M. Chiurazzi, S. Roccella, C. M. Oddo, and P. Dario, "Visual-based defect detection and classification approaches for industrial applications—A SURVEY," *Sensors*, vol. 20, no. 5, p. 1459, Mar. 2020, doi: 10.3390/s20051459.

[12] A. Saberironaghi, J. Ren, and M. El-Gindy, ''Defect detection methods for industrial products using deep learning techniques: A review,'' *Algorithms*, vol. 16, no. 2, p. 95, Feb. 2023, doi: 10.3390/a16020095.

[13] S. Faghih-Roohi, S. Hajizadeh, A. Núñez, R. Babuska, and B. De Schutter, ''Deep convolutional neural networks for detection of rail surface defects,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 2584–2589, doi: 10.1109/IJCNN.2016.7727522.

[14] J. Silvestre-Blanes, T. Albero-Albero, I. Miralles, R. Pérez-Llorens, and J. Moreno, ''A public fabric database for defect detection methods and results,'' *Autex Res. J.*, vol. 19, no. 4, pp. 363–374, Dec. 2019, doi: 10.2478/aut-2019-0035.

[15] C. Luan, R. Cui, L. Sun, and Z. Lin, ''A Siamese network utilizing image structural differences for cross-category defect detection,'' in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 778–782, doi: 10.1109/ICIP40778.2020.9191128.

[16] H. U. Dike, Y. Zhou, K. K. Deveerasetty, and Q. Wu, ''Unsupervised learning based on artificial neural network: A review,'' in *Proc. IEEE Int. Conf. Cyborg Bionic Syst. (CBS)*, Oct. 2018, pp. 322–327, doi: 10.1109/CBS.2018.8612259.

[17] T. Defard, A. Setkov, A. Loesch, and R. Audigier, ''PaDiM: A patch distribution modeling framework for anomaly detection and localization,'' in *Proc. Int. Conf. Pattern Recognit.*, Mar. 2021, pp. 475–489, doi: 10.1007/978-3-030-68799-1_35.

[18] N. Cohen and Y. Hoshen, ''Sub-image anomaly detection with deep pyramid correspondences,'' 2020, *arXiv:2005.02357*.

[19] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, ''Towards total recall in industrial anomaly detection,'' in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2022, pp. 14318–14328.

[20] H. Chen, Y. Pang, Q. Hu, and K. Liu, ''Solar cell surface defect inspection based on multispectral convolutional neural network,'' *J. Intell. Manuf.*, vol. 31, no. 2, pp. 453–468, Feb. 2020, doi: 10.1007/s10845-018-1458-z.

[21] H. Di, X. Ke, Z. Peng, and Z. Dongdong, ''Surface defect classification of steels with a new semi-supervised learning method,'' *Opt. Lasers Eng.*, vol. 117, pp. 40–48, Jun. 2019, doi: 10.1016/j.optlaseng.2019.01.011.

[22] L. Xu, S. Lv, Y. Deng, and X. Li, ''A weakly supervised surface defect detection based on convolutional neural network,'' *IEEE Access*, vol. 8, pp. 42285–42296, 2020, doi: 10.1109/ACCESS.2020.2977821.

[23] J. Kim, J. Ko, H. Choi, and H. Kim, ''Printed circuit board defect detection using deep learning via a skip-connected convolutional autoencoder,'' *Sensors*, vol. 21, no. 15, p. 4968, Jul. 2021, doi: 10.3390/s21154968.

[24] W. Dai, A. Mujeeb, M. Erdt, and A. Sourin, ''Soldering defect detection in automatic optical inspection,'' *Adv. Eng. Informat.*, vol. 43, Jan. 2020, Art. no. 101004, doi: 10.1016/j.aei.2019.101004.

[25] X. Yi, E. Walia, and P. Babyn, ''Generative adversarial network in medical imaging: A review,'' *Med. Image Anal.*, vol. 58, Dec. 2019, Art. no. 101552, doi: 10.1016/j.media.2019.101552.

[26] K. Chen, N. Cai, Z. Wu, H. Xia, S. Zhou, and H. Wang, ''Multi-scale GAN with transformer for surface defect inspection of IC metal packages,'' *Expert Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118788, doi: 10.1016/j.eswa.2022.118788.

[27] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua, ''CVAE-GAN: Fine-grained image generation through asymmetric training,'' in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2745–2754.

[28] S. Niu, B. Li, X. Wang, and H. Lin, ''Defect image sample generation with GAN for improving defect recognition,'' *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1611–1622, Jul. 2020, doi: 10.1109/TASE.2020.2967415.

[29] W. James and C. Stein, ''Estimation with quadratic loss,'' in *Breakthroughs in Statistics: Foundations and Basic Theory*. New York, NY, USA: Springer, 1992, pp. 443–460, doi: 10.1007/978-1-4612-0919-5_30.

[30] D. Tabernik, S. Šela, J. Skvarč, and D. Skoč aj, ''Segmentation-based deep-learning approach for surface-defect detection,'' *J. Intell. Manuf.*, vol. 31, no. 3, pp. 759–776, Mar. 2020, doi: 10.1007/s10845-019-01476-x.

[31] P. Chao, C.-Y. Kao, Y. Ruan, C.-H. Huang, and Y.-L. Lin, ''HarDNet: A low memory traffic network,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3551–3560.

[32] A. Horé and D. Ziou, ''Image quality metrics: PSNR vs. SSIM,'' in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 2366–2369, doi: 10.1109/ICPR.2010.579.

[33] M. J. Chong and D. Forsyth, ''Effectively unbiased FID and inception score and where to find them,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6069–6078.

[34] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, ''Generalized intersection over union: A metric and a loss for bounding box regression,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 658–666.

[35] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, ''Unpaired image-to-image translation using cycle-consistent adversarial networks,'' in *Proc. ICCV*, Oct. 2017, pp. 2223–2232.

[36] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, ''StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8789–8797.

[37] T. Karras, S. Laine, and T. Aila, ''A style-based generator architecture for generative adversarial networks,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4396–4405.

[38] G. Zhang, K. Cui, T.-Y. Hung, and S. Lu, ''Defect-GAN: High-fidelity defect synthesis for automated defect inspection,'' in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 2523–2533.

[39] B. Xu and C. Liu, ''Pavement crack detection algorithm based on generative adversarial network and convolutional neural network under small samples,'' *Measurement*, vol. 196, Jun. 2022, Art. no. 111219, doi: 10.1016/j.measurement.2022.111219.

DONGHUN KU received the B.S. degree in mechanical engineering from Korea Advanced Institute of Science and Technology, in 2014. He is currently pursuing the Ph.D. degree with the Metrology and Thin Film Process Laboratory, Seoul National University. His research interests include metrology, image processing, and deep learning.

HEUI JAE PAHK received the Ph.D. degree in mechanical engineering from The University of Manchester, Manchester, U.K., in 1990. He is currently a Professor with the School of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Republic of Korea. His research interests are in thin-film processing and metrology including thin-film deposition, thickness and component measurement, and process control.