

Article

PCB Defect Classification with Data Augmentation-Based Ensemble Method for Sustainable Smart Manufacturing

Jaeseok Jang ¹ , Qing Tang ¹  and Hail Jung ^{2,*} 

¹ Data Science Group, INTERX, Ulsan 44542, Republic of Korea; jay.jang@interxlab.com (J.J.); tangqing@interxlab.com (Q.T.)

² Department of Business Administration, Seoul National University of Science and Technology, Seoul 01811, Republic of Korea

* Correspondence: hail95@seoultech.ac.kr; Tel.: +82-10-2994-7527

Abstract: In the rapidly evolving field of printed circuit board (PCB) manufacturing, automated optical inspection (AOI) systems play a critical role but often face challenges such as computational inefficiencies, high costs, and limited defect data. To address these issues, we propose an ensemble methodology that combines lightweight models with custom data augmentation techniques to enhance defect classification accuracy in real-time production environments. Our approach mitigates overfitting in small datasets by generating diverse models through advanced data augmentation and employing feature-specific validation strategies. These models are integrated into an ensemble framework, achieving complementary results that improve classification accuracy while reducing computational overhead. We validate the proposed method using two datasets: the general classification dataset CIFAR-10 and an on-site real-world PCB dataset. With our approach, the average accuracy on CIFAR-10 improved from 97.6% to 98.2%, and the accuracy on the PCB dataset increased from 81% to 89%. These results demonstrate the method's effectiveness in addressing data scarcity and computational challenges in real-world manufacturing scenarios. By improving quality control and reducing waste, our method optimizes production processes and contributes to sustainability through cost savings and environmental benefits. The proposed methodology is versatile, scalable, and applicable to a range of defect classification tasks beyond PCB manufacturing, making it a robust solution for modern production systems.



Citation: Jang, J.; Tang, Q.; Jung, H. PCB Defect Classification with Data Augment-Based Ensemble Method for Sustainable Smart Manufacturing. *Sustainability* **2024**, *16*, 10417. <https://doi.org/10.3390/su162310417>

Academic Editor: Hossein Bonakdari

Received: 18 October 2024

Revised: 20 November 2024

Accepted: 22 November 2024

Published: 28 November 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: deep learning; ensemble; augmentation; printed circuit board; automated optical inspection; manufacturing

1. Introduction

In the rapidly evolving field of printed circuit board (PCB) manufacturing, ensuring product quality while minimizing production errors is crucial for maintaining competitiveness in an increasingly demanding market. The complexity of modern PCB designs, coupled with the need for high production volumes, presents significant challenges for manufacturers striving to meet strict quality standards while maintaining operational efficiency [1,2]. Additionally, there is growing pressure to adopt sustainable practices that optimize resource use, reduce operational costs, and minimize environmental impact.

Traditionally, PCB inspection has relied heavily on human intervention due to frequently changing inspection standards [3,4]. However, manual inspection has inherent drawbacks:

1. **Difficulty of Inspection:** The small size and intricate details of PCBs make human inspection challenging and error-prone.
2. **Human Fatigue:** Inspectors are susceptible to fatigue, leading to reduced accuracy over time.

3. Labor Intensity: High production volumes necessitate a large workforce, increasing labor costs.
4. Inconsistency: Variability in classification criteria among workers results in inconsistent defect categorization.

These limitations highlight the need for automated systems that can enhance consistency and accuracy in defect classification.

Automated Optical Inspection (AOI) systems have been introduced to address some of these challenges, leveraging computer vision techniques to identify defects [5–8]. While effective in improving detection and quality control, these systems still require substantial manual intervention, especially for edge cases or highly complex boards. Developing new algorithms for novel cases can be time-consuming, and reliance on human oversight increases labor costs and slows production, limiting scalability.

As the industry increasingly focuses on sustainability [9,10], integrating advanced technologies into AOI systems offers potential solutions to both operational and environmental challenges. Deep learning technologies in particular hold promise for enabling intelligent, real-time decision-making. By facilitating faster and more accurate inspections, deep learning models can enhance production efficiency while reducing the need for energy-intensive rework and material wastage. This aligns the manufacturing process with global sustainability goals, making it more resource-efficient and environmentally friendly [11,12].

However, several challenges remain in achieving fully automated defect detection [3,6,13,14]:

1. Accuracy and Reliability: Ensuring the automation system accurately detects and classifies defects.
2. Data Scarcity: Insufficient defect data hamper the development of models with high precision, especially for rare defect types.
3. Defect Diversity: The need to identify and classify a wide variety of defect types for complete automation.

Addressing these challenges requires a focus on accurate classification of specific defect types, which is essential for effective quality control within the production process. Effective quality control within the production process depends on understanding that different defects have varying tolerances based on characteristics such as size and shape. For example, minor surface scratches might be acceptable if they are within certain limits (e.g., less than 1 micron), whereas critical defects are unacceptable even at sizes as small as 0.1 microns [13,15,16]. Therefore, the ability to not only detect but also classify defects with high precision is essential.

Despite advances in automated inspection systems that can rapidly determine whether PCBs are acceptable or defective, identifying and classifying specific types of defects with high accuracy remains a significant challenge. This complexity arises from the wide variety of defect types, the scarcity of data for some defects, and the necessity for real-time processing. Moreover, deep learning models commonly used for defect classification face limitations such as extended inference times and the requirement for extensive datasets [17,18]. Large models may not be practical for real-time applications due to computational constraints, and collecting sufficient data for training can be difficult, particularly for rare defects.

To address these issues, we propose an ensemble methodology that employs models with fewer parameters while achieving performance comparable to or surpassing that of more complex models. Our approach leverages data augmentation techniques to supplement insufficient datasets, enhancing the model's ability to generalize from limited data. By generating additional training examples through augmentation, we effectively expand the diversity of the dataset. Additionally, when predicting real-time defective data, augmentation techniques can improve classification accuracy by simulating various defect scenarios. This methodology is particularly advantageous in scenarios where defective data are limited for training or where there is a high volume of real-time data production. By utilizing lightweight models and effectively augmenting the data, our approach

provides a practical solution for efficient and accurate defect classification in real-world manufacturing environments.

The main contributions of this paper are as follows:

1. **An Ensemble Methodology:** We introduce an ensemble approach using lightweight models to achieve high accuracy in defect classification, suitable for real-time applications with computational constraints.
2. **An Data Augmentation Techniques:** We develop tailored data augmentation methods for PCB defect datasets to enhance the diversity and quantity of training data, improving model generalization.
3. **Reduced GPU usage cost for same performance:** Since multiple lightweight models are used and predicted in parallel using the GPU threading method, there is a great advantage in GPU usage cost compared to using general heavy models. This can act as a factor in reducing environmental costs.
4. **Performance Improvement and Real-world Validation:** Our methodology demonstrates performance comparable to or better than more complex models, addressing challenges related to insufficient defect data and extended inference times. We validate our approach through experiments in real-world manufacturing scenarios, showing its effectiveness in environments with limited data and high production volumes.
5. **Easy to expand to all models and all fields:** This methodology is applicable to various models and shows high performance improvement depending on the type of dataset.

The paper is structured to provide a clear progression from foundational concepts through to experimental validation and future implications. Section 2 introduces the proposed methodology to address these issues, after discussing the evolution and limitations of PCB inspection over time. Section 3 briefly describes the overall methodology and then describes the techniques we used to optimize performance. Section 4 then applies this methodology to both the CIFAR-10 public dataset and actual PCB production data, comparing the results with those from existing models to assess improvements in classification accuracy and efficiency. Section 5 discusses the experimental results, analyzing the performance in both the CIFAR-10 and PCB cases and examining the strengths and areas for improvement in our approach. Finally, Section 6 addresses the broader contributions, scalability, and potential applications of this technology, discussing directions for future research to further enhance the methodology and extend its applicability across different fields and production scenarios. This structure ensures a logical flow, taking the reader from background and methodology to empirical findings and forward-looking insights.

2. Related Work

The inspection process for Printed Circuit Boards (PCBs) can be broadly categorized into two stages: Bare PCB (BPCB) inspection [3,5–7] and PCB Assembly (PCBA) [2] inspection [1]. BPCB involves evaluating a plain insulated material board that contains conductive tracks, pinholes, vias, and so on. BPCB inspection is crucial as defects at this stage can significantly impact the assembly and functionality of the PCB in later stages. PCBA focuses on inspecting the assembly process where components such as resistors, capacitors, inductors, and integrated circuits are soldered onto the board.

This paper focuses specifically on BPCB defect inspection, which plays a pivotal role in ensuring the quality and reliability of PCBs before they proceed to the assembly stage. This section provides a comprehensive review of BPCB inspection methods, categorized into two main approaches: traditional methods [3,5,7,13,14,19,20] and AI-based methods [21–24].

Automated Optical Inspection (AOI) systems have been widely studied and implemented for detecting and classifying PCB defects. Over the years, the research community has developed and refined various techniques to enhance both the detection accuracy and speed of AOI systems.

2.1. Traditional Methods

Traditional methods [3,5,14,19,20] for BPCB inspection rely primarily on image processing techniques, with template matching [19] being the most widely adopted approach. These methods aim to detect discrepancies between a reference (“golden”) image and the test image. Template matching involves pixel-wise comparison of the test image with a reference image to identify defects. Binary image subtraction and XOR-based image comparison [20] were early methods employed in this domain. However, these techniques face challenges in cases where slight mis-alignments in orientation, scale, or position occur between the images. Such issues often lead to inaccuracies, as minor deviations in PCB layout or lighting conditions can result in false positives or missed defects. To address the limitations of basic template matching, more advanced traditional methods have been developed, incorporating techniques like morphological processing and feature extraction. Morphological operations, such as dilation and erosion, enhance critical features like solder pads, conductive traces, and vias, thereby improving defect detection accuracy [14].

Additionally, Statistical Geometric Features (SGFs) [13] have been utilized to analyze the texture and geometric properties of PCB components. These approaches are particularly effective in detecting specific defect types, such as missing solder pads or track interruptions.

Despite their contributions, traditional methods face inherent limitations. (1) Sensitivity to Variations: Small differences in PCB layout or lighting can lead to inconsistent results. (2) Manual Fine-Tuning: Many methods require extensive manual tuning of parameters, which is time-consuming and reduces scalability.

Overall, while traditional methods laid the foundation for PCB inspection, their inability to handle complex designs and real-world variability limits their applicability in modern manufacturing environments. These challenges have driven the adoption of AI-based approaches, which are explored in the next section.

2.2. Deep Learning Approaches

The advent of machine learning [21–24], particularly deep learning, has significantly advanced the field of AOI for PCB defect detection. Unlike traditional methods, deep learning approaches eliminate the need for handcrafted feature extraction, offering a more robust and adaptable framework for handling complex PCB designs and noisy environments.

Convolutional Neural Networks (CNNs) [25] have emerged as the dominant architecture for PCB defect classification due to their ability to automatically extract hierarchical features from images. These models are designed to preprocess and denoise PCB images, improving the accuracy of defect detection in environments with significant noise [21,23]. By learning robust feature representations, these autoencoders address challenges related to lighting inconsistencies, image noise, and subtle defect patterns. Some approaches employ multi-input CNN architectures to combine information from different image modalities or views of the PCB. This has been shown to significantly improve the recognition of subtle defects, such as hairline cracks or microscopic mouse bites, which are difficult to identify using traditional techniques [22,24]. Deep learning approaches serve two advantages: (1) Noise Robustness: Denoising techniques incorporated into CNNs help mitigate the impact of noisy manufacturing environments. (2) Automation: The ability to learn from raw image data reduces the reliance on manual feature engineering, making the process more efficient and scalable. These two advantages lead the deep learning method to consistently outperforming traditional methods.

Despite their contributions, three challenges still remain. (1) Data Scarcity: The PCB industry frequently encounters limited labeled datasets for specific defect types. Deep learning models require large amounts of high-quality training data to achieve optimal performance, making data scarcity a major bottleneck [18]. This issue necessitates the development of data augmentation techniques or synthetic data generation to compensate for the lack of diverse training samples. (2) Model Deployment in Industry: Translating state-of-the-art deep learning methods into practical industrial applications often requires balancing accuracy with inference speed and hardware constraints. Efforts to compress

models or employ lightweight architectures are essential for real-world adoption. (3) High Computational Requirements: The complex architectures of deep learning models, such as CNNs, often require significant computational power. This poses a challenge for achieving real-time defect detection in fast-paced manufacturing environments, where throughput is critical [17].

2.3. The Ensemble Approach and Its Importance

Ensemble techniques are said to be a very efficient idea in various papers [26,27]. The main advantage is that it can balance computational efficiency and accuracy. It is suitable for fast-moving production environments because it can detect defects in real time without requiring high-end hardware by utilizing lightweight models [28]. Despite the low accuracy of individual models, the ensemble method combines the outputs to improve the overall performance and achieves detection accuracy similar to sophisticated deep learning models at a lower computational cost. In addition, the method is scalable and adaptable, and more powerful models can be integrated in the future to improve classification accuracy and maintain robustness against overfitting, especially in unbalanced datasets or rare defects. In addition, it has already been applied in various fields and its effectiveness has been proven [29].

In view of these points, the application of ensembles in the PCB field can be sufficiently utilized as a method to solve the limitations of imbalanced datasets and performance. However, in order to use the ensemble method efficiently, it is necessary to create models with various features.

2.4. Ensemble Approach with Augmented Data

Regarding image augmentation technology, it is said that various augmentation technologies can improve the performance of image classification models [30,31].

In addition, it is said that simple augmentation technologies have led to many performance improvements in models such as AlexNet [32]. In particular, it is said that there are many costs and difficulties in collecting image data, and augmented data can solve this [33].

In other words, PCB manufacturers are also collecting data based on images, and since it is very difficult to collect bad data in particular, by utilizing the augmentation technique, it is possible to efficiently generate less data and at the same time, expecting improved performance.

2.5. Our Approach

We attempted in various ways to solve the limitations of the past PCB prediction method.

In general, augmentation techniques are applied only to the train dataset and undergo the validation and test processes, but from a new perspective, we thought that if augmentation techniques are applied to the validation and test datasets, but each augmentation technique is applied once to the validation and test datasets, and each is used to optimize training, we can create multiple weak learners.

Through this, we propose a method of utilizing ensemble techniques together so that the results of models trained through data from various perspectives (datasets with different augmentations applied) can complement each other.

The expected effect is that by utilizing the augmentation technique, a small amount of data can be used more efficiently, and at the same time overfitting and generalization will be prevented. In addition, by utilizing the ensemble technique, it is possible to create a model with higher performance than the existing model by identifying the features from each learner and reinforcing the results.

In particular, this methodology seems to be applicable to most base models and is a method that can further improve the performance of the base model. Therefore, if the method is applied to a weak learner, it can show a significant advantage in the

existing prediction time part even with a similar result to that of a strong learner or with high accuracy.

3. Method

This section describes, step by step, how to apply the methodology and describes the application algorithms that make up the method.

3.1. Overall Step

The flowchart for creating augmentation technology-specific models from an existing dataset is illustrated in Figure 1.

Data Splitting for Ensemble Learning: The original dataset was divided into three subsets: training, validation, and test datasets. The training dataset was used to train the model, while the validation and test datasets were used to evaluate performance.

Augmentation [30] of Training Dataset: To enhance the generalization capability of the model, we applied a variety of data augmentation techniques to the training dataset. These augmentation techniques included methods such as flipping, rotation, scaling, and color adjustments. The augmented data were then combined with the original training data to create an enriched train dataset that captures a wider variety of patterns.

Augmentation of Validation and Test Datasets: The validation and test datasets were augmented by applying the same set of augmentation techniques used for the training dataset. This process resulted in multiple versions of both the validation and test datasets, each corresponding to a specific augmentation technique. This allowed us to evaluate model performance not only on the original data but also on augmented versions of the validation and test sets.

Model Training with Augmented Datasets: The augmented training dataset was used to train the model while the validation dataset was used to evaluate and optimize the performance of the model trained in real time. During this step, the training dataset was kept constant while the validation dataset changed depending on whether the original version or the augmented version was being evaluated. This allowed us to create an optimized model using the original data, and also create augmented versions of the optimized model for Horizontal Flip, Zoom, WidthShift, and shear. In other words, we created optimized models for each of the four augmented datasets, including the original data (a total of five models).

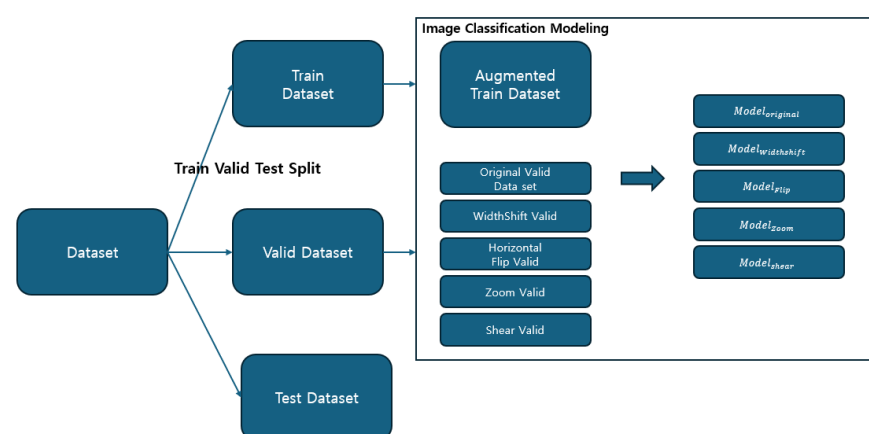


Figure 1. Flow chart for creating augmentation technology-specific models from existing datasets.

The flowchart for building a probability dataset using the created models is shown in Figure 2.

Probability Prediction on Validation and Test Datasets: Using the trained models, we predicted the probabilities on the original and augmented versions of the validation and

test datasets. This produced a set of probability distributions that were later used to inform the next steps in the model development process.

For each of the five types of validation datasets, an appropriate model was selected, and the probability results for each dataset were generated and merged into one dataset.

Retraining Using Validation Probabilities: The probability outputs from the validation datasets were used to create a new dataset. This newly constructed validation probability dataset was then split again into training and validation subsets and used for further model training. This step enabled us to incorporate the model's predictions directly into the training process, refining its predictive capability.

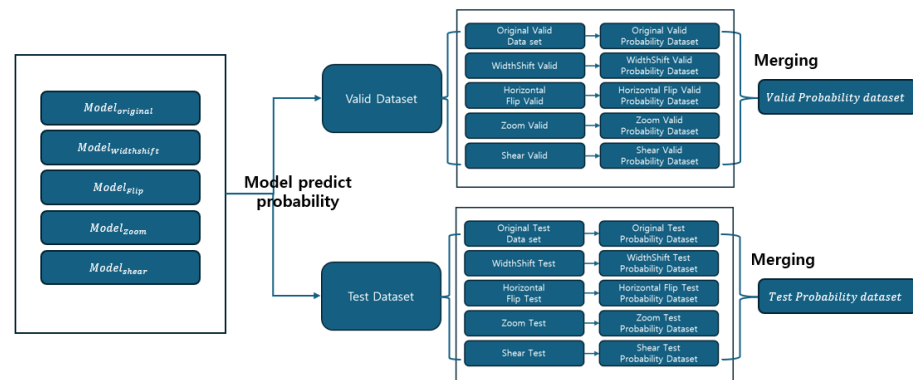


Figure 2. Flow chart for building a probability dataset using the created models.

The flowchart for the final ensemble model training and test results is shown in Figure 3.

Metalearner Model Optimization with Optuna: We employed Optuna, a state-of-the-art hyperparameter optimization framework, to create and tune a metalearner model. This metalearner was designed to aggregate the outputs from multiple base models, combining their predictions into a final, more accurate prediction. The metalearner model was trained on the split datasets created from the validation probability dataset.

Evaluation on Final Test Dataset: Finally, the trained metalearner model was evaluated on the test dataset using the probability predictions of the base model. For the test dataset, the test dataset was augmented into five test sets using the augmentation method from the initially divided test dataset, and the final test set was constructed using the validation dataset to obtain the predicted results for each of the five trained models. The performance of the metalearner was evaluated by calculating the accuracy on the test dataset that went through a series of processes, which provided the final measure of model performance.

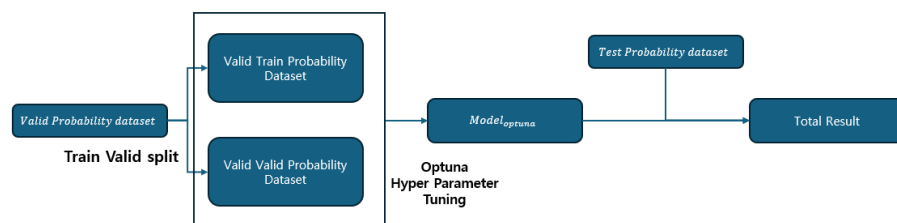


Figure 3. Flow chart for final ensemble model training and test results output.

3.2. Application Algorithm

The existing algorithms used in this method can be broadly divided into two categories: augmentation technique and Optuna technique.

3.2.1. Augmentation Method

In general, for PCB data, image data are collected in a precisely controlled environment [34]. Therefore, since the number of data on defects themselves is small, rather than

techniques such as noise and restoration, the following augmentation methodology was utilized for greater efficiency.

1. Horizontal Flip:

Mathematical Representation: A horizontal flip reflects an image across the vertical axis. In mathematical terms, for an image with pixel coordinates (x, y) , the flip transforms the horizontal coordinate x into $-x$, effectively reflecting the image.

Transformation Matrix: A horizontal flip can be expressed as

$$T_{flip} = \begin{bmatrix} -1 & 0 & w \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, (x', y') = (w - x, y) \quad (1)$$

where w is the width of the image. This matrix reflects the image across the vertical axis and translates it to maintain the image's position within the frame.

2. Zoom:

Mathematical Representation: Zooming is essentially a scaling operation on the image. When zooming in, the image is enlarged, and when zooming out, it is shrunk. The transformation for zoom is represented by a scaling factor.

- Zoom In: Increases the apparent size of objects by scaling the coordinates inward.
- Zoom Out: Decreases the size of objects by scaling the coordinates outward.

Transformation Matrix: For scaling by a factor of s (where $s > 1$ for zoom in and $0 < s < 1$ for zoom out), the transformation matrix is

$$T_{Zoom} = \begin{bmatrix} s & 0 & (1-s) \cdot w/2 \\ 0 & s & (1-s) \cdot h/2 \\ 0 & 0 & 1 \end{bmatrix}, (x', y') = (sx, sy) \quad (2)$$

3. Shear:

Mathematical Representation: Shearing shifts each pixel in a direction parallel to one of the axes, producing a slanted or skewed effect. Shearing can occur along the x-axis (horizontal shear) or the y-axis (vertical shear). The amount of shear is controlled by a shear factor m_x (for x-axis shear) or m_y (for y-axis shear).

Transformation Matrix: For shear along the x-axis by a factor of m_x , the transformation matrix is

$$T_{shear,x} = \begin{bmatrix} 1 & m_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, x' = x + m_x \cdot y, y' = y \quad (3)$$

For shear along the y-axis by a factor of m_y , the transformation matrix is

$$T_{shear,y} = \begin{bmatrix} 1 & 0 & 0 \\ m_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, x' = x, y' = y + m_y \cdot x \quad (4)$$

4. Width Shift (Translation):

Mathematical Representation: Width shift moves the image horizontally by a fixed amount, either to the left or the right, without altering its vertical coordinates. This is a translation operation.

A shift by a factor of T_x (usually in terms of pixels or a percentage of the image width) displaces all pixel coordinates along the x-axis.

Transformation Matrix: The transformation matrix for shifting the image along the x-axis by T_x units is

$$T_{shear} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, x' = x + t_x, y' = y \quad (5)$$

3.2.2. Optuna (Ensemble Meta Learner Modeling)

Optuna's [35] efficiency comes from combining Bayesian optimization with the TPE algorithm, which allows for a structured and probabilistic search of hyperparameters, leading to better performance in less time.

The goal of hyperparameter optimization is to find the optimal set of hyperparameters θ that minimize or maximize a certain objective function $f(\theta)$, which typically represents the model's performance (e.g., loss, accuracy).

These are the four key point of hyperparamter tuning of Optuna as below:

- Objective Function: Minimizes or maximizes $f(\theta)$ to find the best hyperparameters.
- TPE Algorithm: Models $l(\theta)$ and $g(\theta)$ to direct the search toward promising regions.
- Expected Improvement: Maximizes $l(\theta)/g(\theta)$ to balance exploration and exploitation.
- Pruning: Terminates trials that are unlikely to lead to improvement, saving.

Optuna mathematically combines probability modeling, density estimation, and Bayesian optimization principles to efficiently find optimal hyperparameters, leveraging TPE and pruning to reduce computational time and improve search efficiency.

- Good Hyperparameter Likelihood: set of observed hyperparameteres as $D_{good} = \{\theta_i | f(\theta_i)\}$. TPE splits D into two parts:
 $D_{good} = \{\theta_i | f(\theta_i) \leq y_{best}\}$ and $D_{bad} = \{\theta_i | f(\theta_i) > y_{best}\}$.
- Density Estimation: Estimates the probability distributions $l(\theta)$ and $g(\theta)$ using kernel density estimation (parzen windows).
 - $l(\theta)$: Probability density of hyperparameters that produce good results.
 - $g(\theta)$: Probability density of hyperparameters that produce worse results.
- Maximizaing Expected Improvement: Optuna chooses the next set of hyperparameters by maximizing the ratio $\frac{g(\theta)}{l(\theta)}$. This directs the search toward regions of the search space that are likely to improve the objective function.

3.2.3. Ensemble with Optuna

Ensemble algorithms in machine learning combine multiple models to produce a single predictive outcome, generally resulting in better performance than individual models.

Using multiple models' multi-class probabilities as input variables for an ensemble optimized with Optuna involves treating each model's prediction probabilities as new feature variables, then learning an optimal combination of these features to produce the final prediction. Here is how to define this approach mathematically.

- Define Probabilities as Feature Matrix X : Suppose there are M models (models made by each augmentation method), each of which produces a probability vector $h_i(x) = [p_{i,1}(x), p_{i,2}(x), \dots, p_{i,K}(x)]$ for K classes on input x . These probability vectors form a new feature matrix X , defined as follows:

$$X(x) = \begin{bmatrix} p_{original,1}(x) & p_{original,2}(x) & \cdots & p_{original,K}(x) \\ p_{zoom,1}(x) & p_{zoom,2}(x) & \cdots & p_{zoom,K}(x) \\ \vdots & \vdots & \ddots & \vdots \\ p_{M,1}(x) & p_{M,2}(x) & \cdots & p_{M,K}(x) \end{bmatrix}$$

Here, each column represents the predicted probabilities for class j , and each row corresponds to the probability vector of model h_i .

- Define Weights W to Optimize: Next, define a weight vector $W = [w_1, w_2, \dots, w_M]$, where each weight w_i is associated with model h_i . These weights, optimized with

Optuna, reflect the contribution of each model to the ensemble. To ensure the weights sum to 1, set the constraints

$$\sum_{i=1}^M w_i = 1 \quad \text{and} \quad w_i \geq 0 \quad (6)$$

- Calculate Final Class Probabilities: The ensemble probability for each class j , denoted $\hat{p}_j(x)$, is calculated as a weighted average of the individual model probabilities:

$$\hat{p}_j(x) = \sum_{i=1}^M w_i \cdot p_{i,j}(x) \quad (7)$$

- Final Prediction Decision: The final ensemble prediction $H(x)$ is the class with the highest weighted probability:

$$H(x) = \arg \max_j \hat{p}_j(x) = \arg \max_j \sum_{i=1}^M w_i \cdot p_{i,j}(x) \quad (8)$$

Using the optimized weights W^* found by Optuna, the final class prediction for each class j is given by

$$H(x) = \arg \max_j \sum_{i=1}^M w_i^* \cdot p_{i,j}(x) \quad (9)$$

4. Experimental Setup

Although the number of experimental data in each experiment is different, the number of data per class within each experiment is the same, and the purpose of the experiment is to show overall performance improvement. Therefore, accuracy, which reflects the probability of correctly predicting the outcomes across the entire dataset, was chosen as the evaluation metric.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$$

The meaning of each acronym is as follows: True Positives (TPs) are cases where the model correctly predicts a positive outcome, while True Negatives (TNs) are cases where it correctly predicts a negative outcome. False Positives (FPs) occur when the model wrongly predicts a positive outcome for a negative case, and False Negatives (FNs) happen when it fails to predict a positive outcome for a positive case.

4.1. CIFAR Data Application

In this paper, we observed changes in scores before and after applying an ensemble model using EfficientNetV2B0 [36] and the ResNet series (50, 101, 152) [37] on the CIFAR10 dataset.

Table 1 shows the callback method used was ReduceLROnPlateau, which reduces the learning rate based on the number of epochs. For the optimizer, we employed SGD [38] with a learning rate of 0.1 and momentum set to 0.9.

Table 1. Setting of parameter on cifar10 modeling.

Use Optimizer	Learning Rate	Momentum
SGD	0.1	0.9

After splitting the training and test datasets, we further divided the training set into training and validation sets for image modeling. After training, the validation dataset

was split again into training and validation subsets, which were used for ensemble model training and verification.

Table 2 shows the total number of training data points is 50,000, with 10,000 for the test set. The number of data is the same for each class, and the classes represent a total of 10 different everyday objects and animals. The training dataset was initially split into 40,000 training samples and 10,000 validation samples. The 40,000 training samples were augmented using the proposed augmentation methods, doubling the dataset to 80,000 samples.

Table 2. Definition on CIFAR10 dataset.

Train Data	Test Data	Class	Data Each Class
50,000	10,000	10	train—5000, test—1000

For the validation dataset, the 10,000 samples were augmented into five folders (including the original data) using each augmentation technique, maintaining the original 10,000 samples. The loss was measured for each augmented validation dataset, and models were created accordingly.

The test dataset was similarly augmented into five folders of 10,000 samples each, and the performance of the generated models was evaluated on these augmented test sets.

The augmentation techniques used included horizontal flip, width shift, shear and zoom (from the image generator), creating five models (including the original dataset). These models were then used to create an ensemble model, which was used to predict the test set.

The following table summarizes the classification accuracy of the augmented models and the ensemble models including lgbm, xgb, random forest, and explainable boosting machine (EBM) [39–42].

Table 3 shows the accuracy of each original model and augmented model for the applied EfficientV2B0, ResNet Series (50,101,152) models.

Table 4 shows the ensemble results improved accuracy by at least 0.7%, with most models seeing an increase of over 1% compared to the original models.

Table 3. Comparison of accuracy between the original model and each augmentation technique applied model on CIFAR10 dataset.

	Original	Horizontal	Shear (0.2)	Zoom (0.2)	WidthShift (0.2)
efficientNetV2B0	0.976	0.974	0.969	0.973	0.961
ResNet50	0.955	0.95	0.956	0.953	0.952
ResNet101	0.956	0.963	0.963	0.964	0.958
ResNet152	0.967	0.959	0.959	0.963	0.961

Table 4. Comparison of accuracy between single best model and final ensemble model on CIFAR10 dataset.

	Original	Ensemble (lgbm)	Ensemble (xgb)	Ensemble (ebm)	Ensemble (rf)
efficientNetV2B0	0.976	0.98	0.980	0.981	0.982
ResNet50	0.955	0.967	0.967	0.967	0.966
ResNet101	0.956	0.973	0.974	0.972	0.975
ResNet152	0.967	0.975	0.975	0.974	0.976

All test datasets were also augmented to match the augmented features of each model and the results were predicted. As shown in Figure 4, the original model and most augmented models predicted Frog, the Zoom model predicted Deer, and the ensemble model trained based on Probability rather than simple Prediction values was able to predict

Deer well. This part could also have a great impact on improving the performance of the data.

The following is a diagram showing the confusion matrix. The results show of the best model among single models on Figure 5 and the results of the ensemble model on Figure 6. Both comparison groups used EfficientNetV2B0 as the base model.

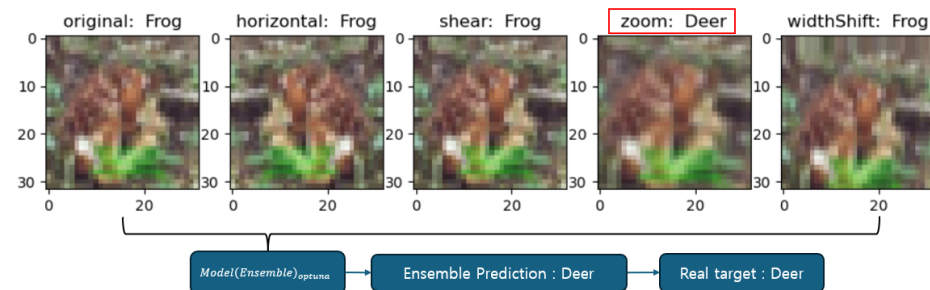


Figure 4. Comparison of results for each model and comparison of results for ensemble models on CIFAR10 dataset.

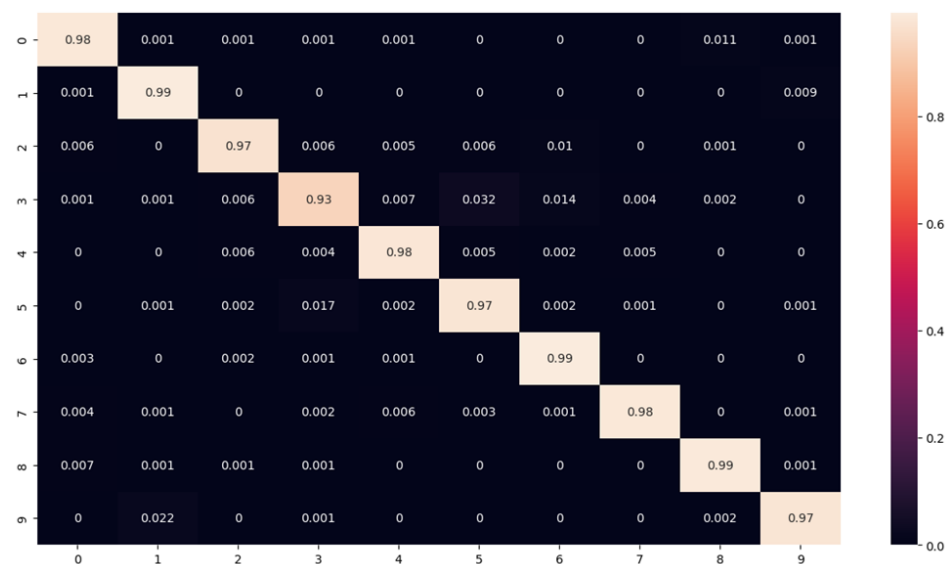


Figure 5. Confusion matrix efficientNetV2B0 of original model on CIFAR10 dataset.

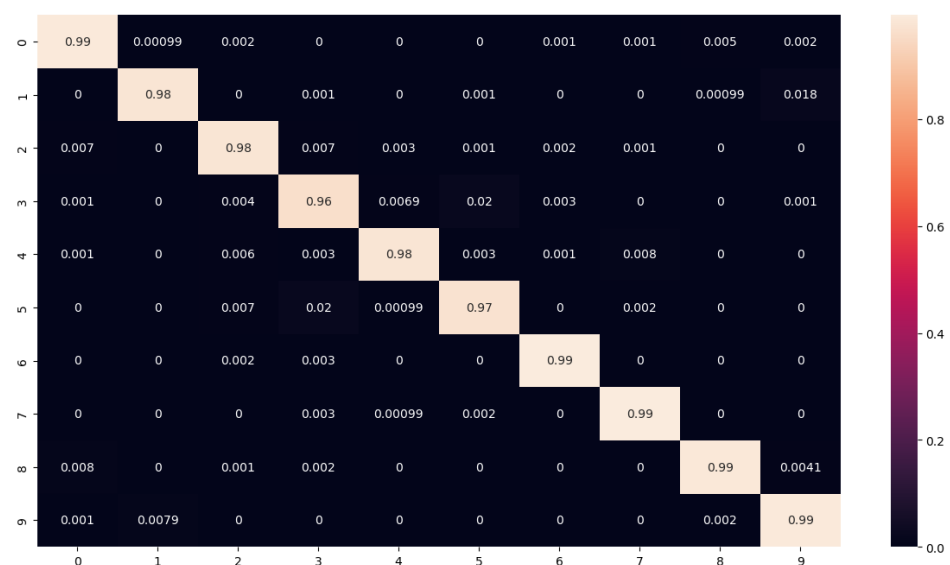


Figure 6. Confusion matrix efficientNetV2B0 of ensemble (rf) model (ours) on CIFAR10 dataset.

There has actually been a significant performance improvement in the confusion matrix of the original model and the confusion matrix of the ensemble model. We applied this ensemble method to PCB data as mentioned earlier in Section 3.

Table 5 shows the performance of the EfficientNetV2B0 model, indicated by the confusion matrix, is shown in terms of accuracy for each class. Since this work has the same number of data for each class and checks the overall model performance rather than improving the performance for a specific classification, the accuracy index is referred to more. Most of the class accuracies improved, and the overall average accuracy increased from 0.976 percent to 0.982 percent.

Table 5. Comparison accuracy of efficientNetV2B0 model of class-specific performance results between single and ensemble models to the CIFAR10 dataset.

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10
single best	0.98	0.99	0.97	0.93	0.98	0.97	0.99	0.98	0.99	0.97
ensemble	0.99	0.98	0.98	0.96	0.96	0.98	0.97	0.99	0.99	0.99

4.2. PCB Data Application

PCB boards are very small, so it is very difficult to detect fine defects among small boards. Therefore, we use a high-spec microscope or magnifying glass to image the product in high specs, and then use the image to determine the defects.

Due to the nature of PCB boards, a defect detection method with a defect rate close to 0 is required, and tic-time is very important for real-time prediction. However, there is a limit to achieving this tic-time while using a model with high accuracy.

That is why we do not use models with many parameters in practice, and we use models with small parameters and high performance such as efficientNet.

In addition, defects are rare in PCB manufacturing, and even if they occur, it is difficult to distinguish different types of defects. In reality, different operators often classify the same defect image into different defect categories. The lack of defect data and unreliability of such data result in low overall defect classification accuracy.

We applied image classification models to categorize these defects and used the ensemble method proposed in this paper to further improve classification accuracy.

Additionally, as discussed in many papers, the role of a lamp is very important in PCB prediction [43,44]. Therefore, in order to solve the problem of PCB defect detection related to light refraction and groove depth, PCB suppliers such as Company S use three lamps including metal illumination, SR illumination, and detection illumination, and utilize the results by utilizing the ensemble model as a whole.

Table 6 presents results from training the model on 1200 training samples and 400 validation samples across 8 labels. The model's accuracy was evaluated using 400 test samples. The number of data per class in train and test is the same. The meaning of each class is data that classify the type of defect.

Table 6. Definition on PCB dataset.

Type	Train Data	Test Data	Class	Data Each Class
lamp1	1200	400	8	train—150, test—50
lamp2	1200	400	8	train—150, test—50
lamp3	1200	400	8	train—150, test—50

Table 7 shows adamW [45] was used as the optimization function, and training was performed by starting with a learning rate of 0.1 and gradually reducing the learning rate using weight decay.

Table 7. Setting of parameter on PCB modeling.

Use Optimizer	Learning Rate	Weight Decay
AdamW	0.1	learning rate $\times$ 0.01

Table 8 presents results including both the performance of a single model (EfficientNetV2L [46]) under each lighting condition and the outcomes from the ensemble technique introduced in this paper. As a evaluation metric, we simply used accuracy.

Table 8. Comparison of augmented modeling each lamp results and final ensemble results applied to the PCB dataset.

Type	Test_TYPE	Accuracy	Ensemble (LGBM)	Total Ensemble
lamp1	original	0.795	0.84	0.89
	horizontal	0.8075		
	shear	0.8		
	widthShift	0.83		
	zoom	0.8175		
lamp2	original	0.7975	0.83	
	horizontal	0.8125		
	shear	0.805		
	widthShift	0.815		
	zoom	0.82		
lamp3	original	0.81	0.8475	
	horizontal	0.8275		
	shear	0.825		
	widthShift	0.83		
	zoom	0.8225		

In the table, there are three types of types, which refer to lighting. The data were collected from three types of lighting, and the results for the augmentation technique model for each lighting, the ensemble results for each lighting (Ensemble(LGBM)), and the total ensemble results using all lighting are organized (Total Ensemble in the table).

We applied Optuna to the LGBM model, creating ensemble results for each lighting condition. We then ensembled the results from each lighting condition to generate a final prediction. In this final ensemble approach, we used LGBM once more to obtain prediction by using each class's probability for the three lighting conditions.

In fact, the accuracy of the models was around 80% based on the quality of the dataset, but when the ensemble model was applied, it can be seen that the accuracy was improved by about 4% compared to the accuracy of the highest single model.

The following is a diagram showing the confusion matrix. The results show of the best model among single models on Figure 7 and the results of the ensemble model on Figure 8. Both comparison groups used EfficientNetV2L as the base model.

As shown in the results of the corresponding confusion matrix, the detection accuracy increased for all classes. In fact, for the first and last classes, the detection accuracy increased significantly from 38 percent to 45 percent and from 37 percent to 43 percent.

Table 9 compares the accuracy of a single model and an ensemble model for the efficientNetV2L model for each class. As can be seen in the table, the classification accuracy improved for all classes except Class4, and the detection accuracy for Class4 was the same.

Table 9. Comparison of class-specific performance results between single and ensemble models to the PCB dataset.

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
single_best	0.76	0.94	0.82	0.96	0.8	0.8	0.66	0.74
ensemble	0.9	0.98	0.88	0.96	0.96	0.82	0.76	0.86

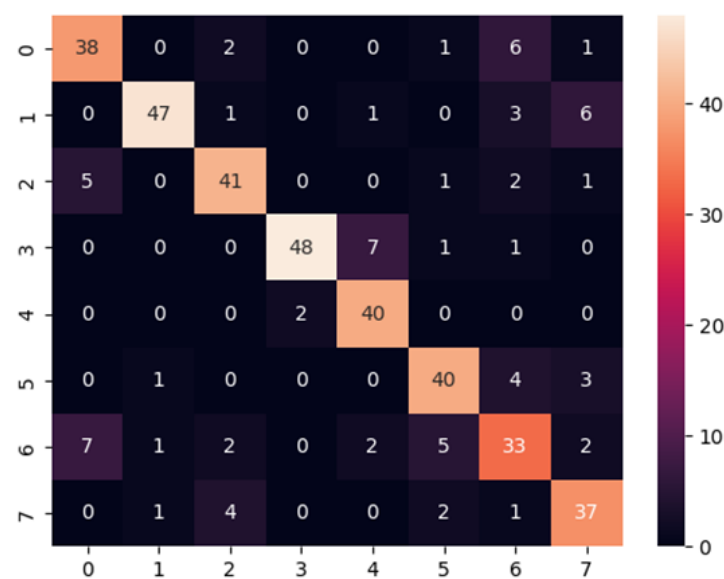


Figure 7. Confusion matrix efficientNetV2L of metal original model on PCB dataset.

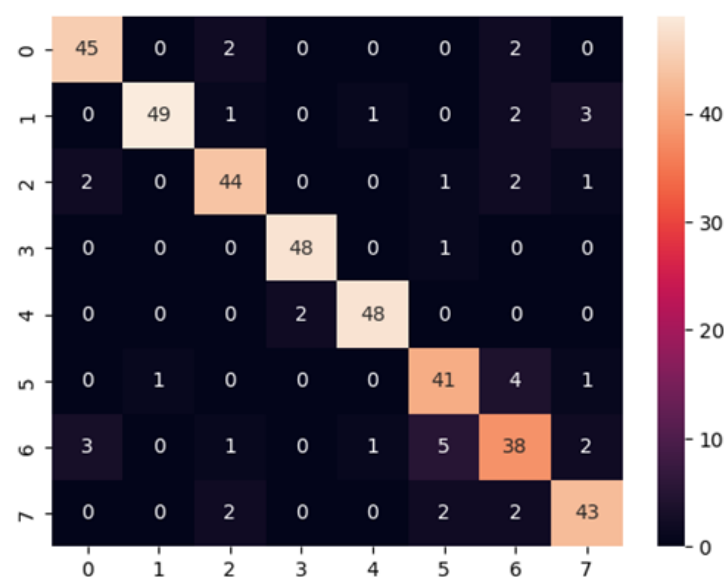


Figure 8. Confusion matrix efficientNetV2L of ensemble model (ours) on PCB dataset.

5. Discussion

This method presents a methodology that enhances the performance of image classification models by utilizing various augmentation techniques. The proposed approach not only improves performance on traditional training datasets, but also extends this improvement to test and validation sets. By tailoring datasets to specific augmentation methods, we achieved better classification accuracy, even with limited training data.

The results show that when the number of trained datasets per label was not large, the performance was improved by augmentation, and when a defect occurred due to the characteristics of the PCB board, the pattern of the defect was similar, but the location of occurrence, the background of the pattern, etc., were different, so the accuracy was greatly improved when the ensemble technique was applied.

In our experiments, we confirmed that applying various augmentation techniques significantly improved image classification performance. Specifically, it was revealed that each augmentation method enhanced the model's ability to distinguish different types of images. When these techniques were extended to the validation and test datasets, the

performance surpassed that of models using only the original data and even outperformed existing models with a single augmentation technique.

Our approach allows the creation of test and validation datasets optimized for each augmentation method, showcasing the unique strengths of each. The results were adjusted accordingly, leading to notable improvements. This demonstrates the versatility of the methodology, as it can be applied to different types of models, such as the ResNet Series (50, 101, 152) and EfficientNet-related models (B20—CIFAR, V2L—PCB), with consistent performance improvements.

In particular, when training data are limited, increasing the test set as well as the training dataset improves accuracy. This effect is most noticeable in environments where defect data are very scarce but defect detection accuracy is important, such as printed circuit board (PCB) manufacturing. For balanced datasets such as the CIFAR dataset, the accuracy increased by 1 percent on average, but for cases where the amount of data is insufficient, such as PCB data, the accuracy increased by as much as 5 percent. In addition, the PCB field has very high product quality inspection costs, especially because the defect rate of shipped products must be achieved close to zero. In this regard, applying the methodology can bring about many positive aspects by improving classification accuracy even in situations where data are limited.

Our research has strong implications for sustainable and smart manufacturing. Previous studies have focused on how the advent of deep learning and machine learning methods enhanced the sustainability of injection molding industry [47] and heat staking process [48]. In this regard, our study well highlights the importance of ensemble-based augmentation methods. The findings contribute to the PCB inspection industry by suggesting a method that significantly increases the inspection accuracy which, in the end, leads to sustainable growth of the industry.

6. Conclusions

The methodology has broad applicability and can be integrated into high-performance models for further improvement. As GPU technologies advance, enabling faster computations, the ensemble methodology discussed in this paper could unlock even higher performance levels. Particularly in industries such as PCB manufacturing, where defect detection is critical, this approach opens the door to more accurate, cost-effective inspections.

Despite the substantial benefits of our approach, several potential limitations remain. The effectiveness of this method may vary depending on the data type and specific industrial applications, and the use of more complex models could significantly increase computational costs. Although GPU threading techniques helped reduce prediction times, these techniques may not always be feasible in resource-constrained environments.

Nevertheless, this methodology is not only widely applicable to various industries, but can also be easily addressed by adjusting the number of ensemble model components to alleviate the limitations of resource constraints. Especially in fields such as PCB manufacturing, this approach has promising potential, and further improvement can contribute to the development of more efficient and autonomous smart manufacturing systems.

In addition, these computational cost reductions and proper quality control can also improve environmental impacts in terms of cost. In fact, the strength of this methodology is that it can be applied to most models and can bring many benefits. Therefore, if this methodology is applied to all fields, not just the simple PCB field, even small reductions can collectively bring significant environmental benefits. This change can establish a new paradigm where small but meaningful actions collectively solve the important problem of environmental waste.

Looking ahead, future research could focus on expanding the application of this methodology to different industries and further exploring its potential in data-scarce environments. Moreover, improving the quality of defect data and continuously refining the pipeline will contribute to the realization of autonomous factories with greater precision.

Author Contributions: Conceptualization, H.J.; Methodology, J.J. and H.J.; Validation, J.J. and Q.T.; Formal analysis, J.J.; Investigation, Q.T.; Resources, H.J.; Writing—original draft, J.J.; Writing—review & editing, Q.T. and H.J.; Supervision, Q.T. and H.J.; Project administration, H.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Korea Technology and Information Promotion Agency for SMEs (TIPA) funded by the Korean Government [Ministry of SMEs and Startups (MSS)] under Grant RS-2022-00140573.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: Authors Jaeseok Jang and Qing Tang were employed by the company Data Science Group, INTERX. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Vasilyev, F.; Isaev, V.; Korobkov, M. The influence of the PCB design and the process of their manufacturing on the possibility of a defect-free production. *Prz. Elektrotechniczny* **2021**, *97*, 91–96. [CrossRef]
2. Petkov, N.; Ivanova, M. Detection and Classification of Defects on Printed Circuit Board Assembly through Deep Learning. In Proceedings of the 2024 9th International Conference on Smart and Sustainable Technologies (SpliTech), Bol and Split, Croatia, 25–28 June 2024; IEEE: Piscataway, NJ, USA, 2024.
3. Malge, P.S.; Nadaf, R.S. A survey: Automated visual pcb inspection algorithm. *Int. J. Eng. Res. Technol.(IJERT)* **2014**, *3*, 223–229.
4. Houdek, C. *Inspection and Testing Methods for PCBs: An Overview*; Technical Report 401; Caltronics Design & Assembly, Inc.: Stacy, MN, USA, 2016. Available online: <https://www.caltronicsdesign.com/white-papers/pcb-inspection-and-testing-methods> (accessed on 24 November 2024).
5. Hara, Y.; Doi, H.; Karasaki, K.; Iida, T. A system for PCB automated inspection using fluorescent light. *IEEE Trans. Pattern Anal. Mach. Intell.* **1988**, *10*, 69–78. [CrossRef]
6. Moganti, M.; Ercal, F.; Dagli, C.H.; Tsunekawa, S. Automatic PCB inspection algorithms: A survey. *Comput. Vis. Image Underst.* **1996**, *63*, 287–313. [CrossRef]
7. Mar, N.S.S.; Yarlagadda, P.K.; Fookes, C. Design and development of automatic visual inspection system for PCB manufacturing. *Robot.-Comput.-Integr. Manuf.* **2011**, *27*, 949–962. [CrossRef]
8. Rehman, S.U.; Thang, K.F.; Lai, N.S. Automated PCB identification and defect-detection system (APIDS). *Int. J. Electr. Comput. Eng.* **2019**, *9*, 297. [CrossRef]
9. Oke, E.A.; Potgieter, H. Discarded e-waste/printed circuit boards: A review of their recent methods of disassembly, sorting and environmental implications. *J. Mater. Cycles Waste Manag.* **2024**, *26*, 1277–1293. [CrossRef]
10. Qiu, R.; Lin, M.; Ruan, J.; Fu, Y.; Hu, J.; Deng, M.; Qiu, R. Recovering full metallic resources from waste printed circuit boards: A refined review. *J. Clean. Prod.* **2020**, *244*, 118690. [CrossRef]
11. Wu, C.J.; Raghavendra, R.; Gupta, U.; Acun, B.; Ardalani, N.; Maeng, K.; Chang, G.; Aga, F.; Huang, J.; Bai, C.; et al. Sustainable ai: Environmental implications, challenges and opportunities. *Proc. Mach. Learn. Syst.* **2022**, *4*, 795–813.
12. Khakurel, J.; Penzenstadler, B.; Porras, J.; Knutas, A.; Zhang, W. The rise of artificial intelligence under the lens of sustainability. *Technologies* **2018**, *6*, 100. [CrossRef]
13. Fonseca, L.A.L.O.; Iano, Y.; Oliveira, G.G.D.; Vaz, G.C.; Carnielli, G.P.; Pereira, J.C.; Arthur, R. Automatic printed circuit board inspection: A comprehensible survey. *Discov. Artif. Intell.* **2024**, *4*, 10. [CrossRef]
14. Lakshmi, G.; Sankar, V.U.; Sankar, Y.S. A Survey of PCB Defect Detection Algorithms. *J. Electron. Test.* **2023**, *39*, 541–554. [CrossRef]
15. Melnyk, R.A.; Tushnytsky, R.B. Detection of defects in printed circuit boards by clustering the etalon and defected samples. In Proceedings of the 2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 25–29 February 2020; IEEE: Piscataway, NJ, USA, 2020.
16. Schubeck, J.; Koblah, D.; Botero, U.J.; Forte, D. A Comprehensive Taxonomy of PCB Defects; University of Florida: Gainesville, FL, USA, 2021.
17. Bhattacharya, A.; Cloutier, S.G. End-to-end deep learning framework for printed circuit board manufacturing defect classification. *Sci. Rep.* **2022**, *12*, 12559. [CrossRef]
18. Park, J.H.; Kim, Y.S.; Seo, H.; Cho, Y.J. Analysis of training deep learning models for pcb defect detection. *Sensors* **2023**, *23*, 2766. [CrossRef]

19. Putera, S.I.; Ibrahim, Z. Printed circuit board defect detection using mathematical morphology and MATLAB image processing tools. In Proceedings of the 2010 2nd International Conference on Education Technology and Computer, Shanghai, China, 22–24 June 2010; IEEE: Piscataway, NJ, USA, 2010; Volume 5, pp. 359–363.
20. Huang, W.; Wei, P. A PCB dataset for defects detection and classification. *arXiv* **2019**, arXiv:1901.08204.
21. Khalilian, S.; Hallaj, Y.; Balouchestani, A.; Karshenas, H.; Mohammadi, A. Pcb defect detection using denoising convolutional autoencoders. In Proceedings of the 2020 International Conference on Machine Vision and Image Processing (MVIP), Qom, Iran, 18–20 February 2020; IEEE: Piscataway, NJ, USA, 2020.
22. Shiina, T.; Iwahori, Y.; Kijisirikul, B. Defect Classification of Electronic Circuit Board Using Multi-Input Convolutional Neural Network. *Int. J. Comput. Softw. Eng.* **2018**, *3*, 137. [\[CrossRef\]](#)
23. Miao, Y.; Liu, Z.; Wu, X.; Gao, J. Cost-Sensitive Siamese Network for PCB Defect Classification. *Comput. Intell. Neurosci.* **2021**, *2021*, 7550670. [\[CrossRef\]](#)
24. Chen, I.-C.; Hwang, R.-C.; Huang, H.-C. Pcb defect detection based on deep learning algorithm. *Processes* **2023**, *11*, 775. [\[CrossRef\]](#)
25. Chua, L.O. CNN: A vision of complexity. *Int. J. Bifurc. Chaos* **1997**, *7*, 2219–2425. [\[CrossRef\]](#)
26. Zhou, Z.-H. *Ensemble Methods: Foundations and Algorithms*; CRC Press: Boca Raton, FL, USA, 2012.
27. Sagi, O.; Rokach, L. Ensemble learning: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1249. [\[CrossRef\]](#)
28. Li, X.; Zhu, M.; Yang, L.T.; Xu, M.; Ma, Z.; Zhong, C.; Li, H.; Xiang, Y. Sustainable ensemble learning driving intrusion detection model. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 1591–1604. [\[CrossRef\]](#)
29. Ullah, H.S.; Khushnood, R.A.; Farooq, F.; Ahmad, J.; Vatin, N.I.; Ewais, D.Y.Z. Prediction of compressive strength of sustainable foam concrete using individual and ensemble machine learning approaches. *Materials* **2022**, *15*, 3166. [\[CrossRef\]](#)
30. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 60. [\[CrossRef\]](#)
31. Wang, J.; Perez, L. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Netw. Vis. Recognit.* **2017**, *11*, 1–8.
32. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 5.
33. Xu, M.; Yoon, S.; Fuentes, A.; Park, D.S. A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognit.* **2023**, *137*, 109347. [\[CrossRef\]](#)
34. Lu, H.; Mehta, D.; Paradis, O.; Asadizanjani, N.; Tehranipoor, M.; Woodard, D.L. FICS-PCB: A Multi-Modal Image Dataset for Automated Printed Circuit Board Visual Inspection. IACR Cryptology ePrint Archive 2020, Report 2020/366. Available online: <https://eprint.iacr.org/2020/366> (accessed on 24 November 2024).
35. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019.
36. Tan, M. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv* **2019**, arXiv:1905.11946.
37. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
38. Robbins, H.; Monro, S. A stochastic approximation method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [\[CrossRef\]](#)
39. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A highly efficient gradient boosting decision tree. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017.
40. Chen, T.; Guestrin, C. XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.
41. Caruana, R.; Lou, Y.; Gehrke, J.; Koch, P.; Sturm, M.; Elhadad, N. Interpretable models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, Australia, 10–13 August 2015.
42. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [\[CrossRef\]](#)
43. Kim, J.; Ko, J.; Choi, H.; Kim, H. Printed circuit board defect detection using deep learning via a skip-connected convolutional autoencoder. *Sensors* **2021**, *21*, 4968. [\[CrossRef\]](#)
44. Liu, Y.; Wang, P.; Liu, J.; Liu, C. X-ray PCB defect automatic diagnosis algorithm based on deep learning and artificial intelligence. *Neural Comput. Appl.* **2023**, *35*, 25263–25273. [\[CrossRef\]](#)
45. Loshchilov, I. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101.
46. Tan, M.; Le Q. Efficientnetv2: Smaller models and faster training. In Proceedings of the International Conference on Machine Learning, PMLR, Online, 18–24 July 2021.
47. Jung, H.; Jeon, J.; Choi, D.; Park, J.Y. Application of machine learning techniques in injection molding quality prediction: Implications on sustainable manufacturing industry. *Sustainability* **2021**, *13*, 4120. [\[CrossRef\]](#)
48. Jung, H.; Rhee, J. Application of YOLO and ResNet in heat staking process inspection. *Sustainability* **2022**, *14*, 15892. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.