IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 September 2023, accepted 15 October 2023, date of publication 18 October 2023,
date of current version 10 November 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3325487*

APPLIED RESEARCH

# LED-Display Defect Detection Based on YOLOv5 and Transformer

**JINWOO PARK[1], JIHUN BAE[2], JONGEON LIM[3], BYEONGCHAN KIM[2], AND JONGPIL JEONG[1]**
[1]Department of Smart Factory Convergence, Sungkyunkwan University, Jangan, Suwon, Gyeonggi 16419, Republic of Korea
[2]Department of Mechanical Engineering, Sungkyunkwan University, Jangan, Suwon, Gyeonggi 16419, Republic of Korea
[3]Department of Advanced Material Science and Engineering, Sungkyunkwan University, Jangan, Suwon, Gyeonggi 16419, Republic of Korea

Corresponding author: Jongpil Jeong (jpjeong@skku.edu)

This work was supported in part by the Sungkyunkwan University, in part by the BK21 FOUR (Graduate School Innovation) funded by the Ministry of Education (MOE), Korea, and in part by the National Research Foundation of Korea (NRF).

**ABSTRACT** In the case of small and medium-sized enterprises (SMEs), it is not easy to find a solution for predictive maintenance or product failure diagnosis because they use fewer infrastructures than large companies and use a low-volume production method. In SME manufacturing facilities, people inspect products directly, what leads to oversights. Defective products are delivered to customers leading to causes and remedies for defects, apologies and compensation to customers Automatic detection of product defects in SMEs, especially in manufacturing, is a very important process and reduce costs as well as optimize corporate management. In the development of this technology, fault detection of LEDs on Display boards such as air conditioners and air purifiers was done by visual inspection in the past, but the image was obtained with a machine vision camera and the YOLOv5 algorithm was used to automatically detect defects. Adding a transformer to YOLOv5 make it is easy to understand the relationship between objects or the global context relating to the whole image, recognize complex patterns or structures, and reduce training time by using a framework that is well suited for parallel processing. Replacing the current module with a transformer in the YOLOv5 network helped developing an efficient algorithm, and the LED fault detection technology was tested in the production line. We propose an improved model with greater accuracy and speed than the existing YOLOv5 model. Semiconductor part LED-Display is made, and after training with our improved proposal, excellent performance is obtained. For the advancement of MES, it is applied to the control process in the surface mount technology industry with many small devices according to the production. The mAP performance of our proposed model was 0.994, a significant improvement over the 0.889 of YOLOv5 model. The Precision of the upgraded MODEL was 0.988, which largely improved the result compared the YOLOv5 model score (0.889), and allowed the detection of objects in Reel units to be applied in the field during object inference.

**INDEX TERMS** LED-display, transformer, AI, small object detection,YOLOv5.

## I. INTRODUCTION

Small object detection is a technology used in images and an active field of computer vision. The development of methods for detecting small objects dates back to the beginning of computer vision research, and has enhanced significantly over time. From now on, we will be watching the development of small object detection.

Early machine learning methods predominantly utilized hand-crafted features and classifiers such as Haar-like features, Histograms of Oriented Gradients (HOG), and Support Vector Machines (SVM). Although these methods were somewhat effective, they encountered limitations in handling complex features of images and large data sets. Specifically, these methods struggled with processing intricate characteristics of images and required substantial computational resources to manage large-scale data sets. Convolutional Neural Networks (CNNs) employ a series of convolutional layers to extract deep and complex features from images. These layers are capable of capturing both global and local patterns in images, making them exceedingly useful for intricate tasks such as object detection. CNNs

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

rapidly learn and predict through parallel processing, making them efficient for handling large data sets. CNNs are widely accepted as one of the extensively used deep learning techniques in image processing tasks.

CNN uses a small filter (Kernel) in the convolution layer to extract image features. This filter moves the input image at regular intervals and performs operations locally at each location to create a feature map. Then the feature maps are down-sampled in the pooling layer to reduce their size. Reducing the size by pooling reduces also the amount of computation while preserving important features. During repeated application of the convolutional layer and the pooling layer, image features are extracted, and finally, small object detection is performed through tasks such as classification and object detection in the output layer.

CNN is known to show excellent performance in the field of image recognition. R-CNN (Region-based Convolutional Neural Network) [3] uses selective search algorithm to find a region where the object in the input image is likely to be located. Region proposals are extracted. All extracted candidate regions are resized to the same size and used as CNN input. Next, the CNN independently processes the extracted candidate regions to predict the presence or absence of objects in each region and estimate the location of the object. The predicted location is calibrated through bounding box regression and finally the class of the object is classified. Despite excellent performance, slowness caused by a bottleneck in extracting candidate regions is a huge disadvantage of R-CNN. Faster R-CNN [3] is a model of the R-CNN family with high accuracy and high speed due to improving the extraction speed of R-CNN candidate regions. Faster R-CNN extracts candidate regions using the Region Proposal Network (RPN). RPN applies anchor boxes to several locations in the input image and extracts candidate regions by predicting the presence and location of objects for each anchor box. Then, the extracted candidate region is converted into a fixes-size feature map by RoI (Region of Interest) Pooling. This feature map is used as an input for CNN to predict the location and class of an object. The SSD is used in the field of object detection. It is one of the representative models based on deep learning. SSD uses a network consisting of a convolutional layer and a pooling layer to process input images at once, extract feature maps of various sizes from each location, and simultaneously predict the location and class of an object. Therefore, it has the advantage of having both high accuracy and high speed. You Only Look Once (YOLO): YOLO works by dividing the input image into a grid and predicting the probable location and class of the object using a CNN structure in each grid cell. Object detection is performed by making a final prediction through the position and class probability of the object and the confidence score that indicates how accurate the proposed value is. The YOLO model shows exceptional results in detecting small objects because it detects objects while viewing the entire image at once. RetinaNet is an object detection model that applies a new loss function called Focal Loss that solves the problem of class imbalance for small object detection.

RetinaNet uses a method similar to Faster R-CNN. RetinaNet performs object detection by transforming input images into pyramid-shaped feature maps of various scales

**TABLE 1. Abbreviations & Description.**

| Abbr. | Name | Description |
|---|---|---|
| AI | Artificial Intelligence | A branch of computer science that focuses on creating machines that learn from and make decisions based on data. |
| GPU | Graphics Processing Unit | A specialized electronic circuit designed to accelerate the processing of images and videos to be Displayed. |
| GRU | Gated Recurrent Unit | A type of recurrent neural network that is capable of learning long-term dependencies, which is a variation of LSTM. |
| LSTM | Long Short-Term Memory | A special kind of RNN, capable of learning long-term dependencies, which is useful in time series prediction. |
| NLP | Natural Language Processing | A subfield of AI that focuses on the interaction between computers and humans using natural language. |
| RNN | Recurrent Neural Network | A class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. |
| SGD | Stochastic Gradient Descent | An iterative method for optimizing an objective function with suitable smoothness properties. |
| HOG | Histogram of Oriented Gradients | A deep learning model designed for image processing.Extract local features of an image through convolutional layers. |
| SVM | Support Vector Machine | A machine learning model that classifies data by maximizing the margin between classes. |
| CNN | Convolutional Neural Network | A deep learning model designed for image processing. Extract local features of an image through convolutional layers. |
| Kernel | Kernel/Filter in Convolution | Filter used to extract image features from CNN. This filter is used to pass over an image or the output of a previous layer and extract local features. |
| Conv | Convolution | A mathematical operation used to extract image features from CNNs. |
| R-CNN | Region with Convolutional Neural Networks | This is a method to detect objects in a specific region of an image using CNN. |
| SSD | Single Shot MultiBox Detector | An algorithm used to detect multiple objects within an image at once. |
| ROI | Region of Interest | Refers to a part of an image or video that is important for a particular task. |
| RetinaNet | RetinaNet | It is a CNN-based object detection algorithm designed to solve the object detection problem for imbalanced data sets. |
| SAFD | Scale-Adaptive Anchor Mechanism | A mechanism that dynamically resizes anchor boxes to better detect objects of different sizes. |
| SANet | Self-Attention Network | A neural network that uses a self-attention mechanism to learn dependencies between input data. |

using Feature Pyramid Network (FPN) and creating anchor boxes from the converted feature maps. The class imbalance problem is a problem in which accuracy is very low during the learning process when the number of some classes is very small or very large. RetinaNet achieves high precision even with a small number of classes by adjusting the weight of the loss function through focal loss. One-stage Detector with Scale-Adaptive Anchor Mechanism (SAFD) [4]: Since the existing One-stage Detector detects objects in the entire image, detection accuracy of small objects is low. SAFD introduced the Scale-Adaptive Anchor Mechanism and improved the exactness by using a method of dynamically adjusting the size and ratio of the anchor box. The Transformer model, which has been mainly used in the recent NLP field, has been refined and is now applied to object detection (Object Detection with Transformer). After converting the entire image into a sequence, it detects objects in the image using the self-attention mechanism. As a result, it has greater accuracy and speed than CNN-based methods. Scale-aware Alignment Network (SANet) [5] introduces a Scale-aware Alignment Module that scales feature maps in images to various sizes in order to detect objects of various sizes. This method correctly determine the size and location of an object, guaranteeing high exactitude for objects of different sizes. Dynamic R-CNN [6] detects objects using RPN and RoI pooling layer. However, this method has low accuracy for small objects because it does not take into account the size and shape of objects in the image. Dynamic R-CNN uses a RoI Transformer instead of the existing RoI Align to precisely detect objects in the image. This research introduced a transformer into the existing YOLOv5 network enhancing it. In the thesis, the improved model will be utilized in industrial facilities to support AI technology for smart factories. The contributions of this study include:

1) We propose a specialized model that contributes to computer vision as an efficient object recognition method by implementing upgrades to various existing methods. Transformers are suitable for capturing long-term dependencies, so that relationships between distant objects is sequenced, which enables recognition of connections between objects and specific patterns. YOLOv5 effectively detects objects, and when combined with Transformer's long-term dependency capture capability, more accurate object detection is possible. Transformers model semantic relationships between objects within an input sequence, which contributes to better understanding interactions between objects. Combining YOLOv5 and Transformers provides a deeper insight of semantic relationships and patterns between objects.
2) It has an excellent ability to capture long-term dependence, so it is possible to comprehend the relationship between distant objects. It increases efficiency by combining the advantages of effective object detection of YOLOv5 with a transformer's ability to recognize specific patterns. YOLOv5 detect and handle objects of different sizes. Transformers utilize full information from the input sequence, as well as capture long-term dependencies. The combination of these elements allows for more proficient detection and interaction modeling of objects of different sizes.
3) Efficient learning and inference: Both YOLOv5 and Transformer utilize the latest deep learning technology and have a framework optimized for effective learning and inference. Together, they improve the training speed of the model, enabling more accurate inference at a faster rate. It is possible to achieve smart factory advancement by applying AI technology throughout the production process by extending horizontal application to production sites such as defective LED- Display.


**FIGURE 1.** LED-Display Appearance.

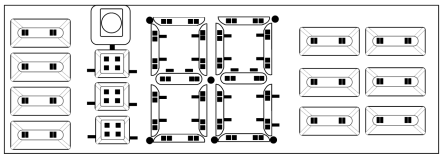

**FIGURE 2.** LED position on the LED-Display PCB board.

The structure of this paper is as follows. Section II of this paper explains LED-Display, experimental equipment, camera, transformer, and YOLOv5 before presenting the proposed system. Section III describes the proposed model of YOLOv5 as well as the composition and role of the process architecture. Section IV covers implementation, training using LED-Display data collected from actual factories, calculation of results and evaluation of the model through comparison with other ones. Finally, Section V summarizes the architecture, implementation and test results proposed in this paper and outlines future research directions.

## II. RELATED WORK

### A. LED-DISPLAY

The LED-Display is a device located outside the air conditioner and assembled as shown in Figure1 and attached to the inside of the air conditioner. After mounting the LED part on the PCB using SMT method, the reflector should be attached to it. After placing the LED on the PCB, the reflector is connected to the PCB to form the shape, and the foil screen is fixated on top of the reflector to form the outer shape. The sponge on both sides is a touch device. The LED Display consists of a total of 15 verification units. It is configured. A total of 15 points are defined as criteria for distinguishing between good and bad products. When testing the power supply, you check the good product if the power is on, and a bad product if it does not turn on.

Figure 2 shows the location of the LED on the PCB and the surface mounting of the LED on the SMT line. During this time, the flux facilitates the flow of the solder and prevents oxidation, but the wrong flux cause problems at the soldering point. Also, using too much or too little flux prevent the solder from adhering properly to the component. If the solder does not bond properly to the LED pads, a problem known as ''cold solder''. This causes reliability issues, and the connection weaken over time. In addition, if the LEDs are not placed correctly on the PCB, the LEDs may not work properly. This lead to reduced light output, mismatch, or total non-functionality of LEDs. LEDs generate heat during operation, which stress the PCB. If heat is not properly managed, the life span of the LEDs may be shortened or their performance may be degraded. Moreover, LEDs are very sensitive to ESD (Electrostatic Discharge), which even damage them. If ESD protection measures are not taken carefully, the LED may be damaged, which may lead to failure of the final product. This problem causes problems of reverse insertion, incorrect insertion, and cold soldering.

In Figure 3, after assembling the LED and connecting it to the test jig, the operation test is carried out directly by a person. This increases inspector's fatigue and deteriorates the quality over time.

**FIGURE 3.** Power on status for lighting test.

**TABLE 2.** Carmera specification.

| Parameter | Specification |
|---|---|
| Interface | USB3 Vision |
| Resolution | 2464 (H) × 2056 (V) |
| Sensor | IMX264 |
| Sensor type | CMOS |
| Sensor size | Type 2/3 |
| Pixel size | 3.45 µm × 3.45 µm |
| Shutter mode | GS (Global shutter) |
| Lens mounts (available) | C-Mount, CS-Mount |
| Max. frame rate at full resolution | 34 fps at ≥200 MByte/s, Mono8 |
| ADC | 12 Bit |
| Image buffer (RAM) | 256 KByte |
| Non-volatile memory (Flash) | 1024 KByte |

### B. LAB EQUIPMENT AND CAMERA

For the experiment, the image was obtained using the following device. The camera acquired the LED-Display test image with the resolution of 2464 × 2056 as in Table 2. This camera provides an image of approximately 5 million pixels. The resolution is sufficient for taking clear and sharp photos. It utilizes supplier technology to minimize electronic noise and thus improve image quality. All pixels are read simultaneously using the 'global shutter' function. This effectively captures fast-moving objects and provides images without distortion. It offers superior performance compared to other image sensors. It is based on supplier technology, which collects more light for each pixel for better image quality. The sensor has a low- power design, which is ideal for camera systems that require long-term operation.

Figure 4 is a screen blocking the computer, monitor, and light source connected to the jig. The screen was installed to block external light sources because excessive light sources cause reflection or overexposure, which affects image quality.

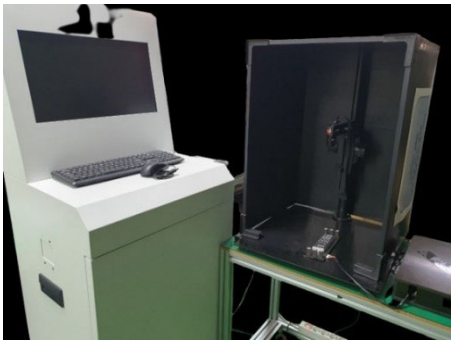

**FIGURE 4.** Camera and Test Computer Equipment.

### C. TRANSFORMER

Transformers [7] from the images are models that process input sequences using the attention mechanism. Figure 5 shows the transformer architecture. Due to these properties, the transformer model is processed in parallel regardless of the length of the input sequence, and has the advantage of fast learning. The Transformer model mainly consists of two parts, an encoder and a decoder. The encoder accepts an input sequence and outputs a fixated-size vector representing the sequence, and the decoder predicts the next word by accepting as input the output vector from the encoder and the words generated by the decoder in the previous time. This codec structure is used in various natural language processing tasks such as machine translation. In the Transformer model, a self-attention mechanism is used to process the input sequence. The self-attention mechanism calculates a weight that indicates how each word is related to other words in the input sequence. We use these weights to adjust each vector of words in the input sequence and compute a new representation. The calculated expression is used as input to the next layer. The Transformer model also introduces the concept of

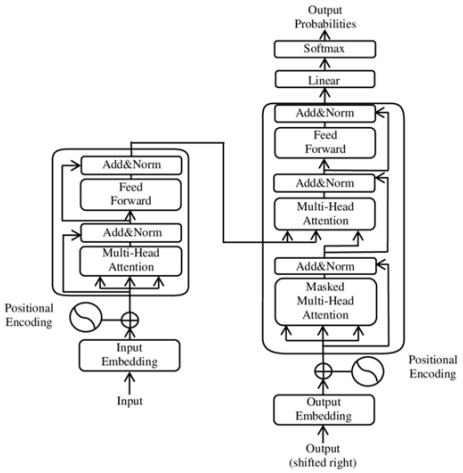


**FIGURE 5.** Transformer architecture.

position embedding, which preserves position information. This allows the Transformer model to store information about the order of the words in the input sequence. It is used for numerous tasks in the field of natural language processing. Representatively, there are machine translation, language modeling, question answering, etc. Recently, the Transformer architecture is used in large-scale language models such as GPT-3. Compared to other deep learning-based natural language processing models, it achieves superb results, especially when the input sequence is long. The reason is that the Transformer model consider the relationship between all words in the input sequence. In addition, the Transformer model is beneficial for large-scale datasets because it has higher learning speed and parallel processing compared to previous models. Model implementation is possible in several deep learning frameworks, such as TensorFlow and PyTorch. Moreover, techniques such as multi-head attention, layer normalization, residual connection, and position-wise feed-forward network is utilized in order to improve the performance of the Transformer model. However, described model is still one of the best operating models in the field of natural language processing. Recently, various deformation models based on it have been proposed, and are expected to be further developed in the future.

### D. YOLOv5

YOLOv5 is a much lighter model compared to YOLOv4 and provides similar performance while requiring less computing resources. In addition, it offers a user-friendly environment and enhanced learning and reasoning speed. As shown in Figure 6, the YOLOv5 architecture is divided into three main parts: Backbone, Neck, and Head. Backbone (Darknet or CSPDarknet) in YOLOv5 is mainly responsible for feature extraction. Darknet is a Convolutional Neural Network (CNN) from the original creators of YOLO. YOLOv5 uses a feature extractor called CSPDarknet53, which is an upgraded version of Darknet53. CSPDarknet53 combines Cross Stage Partial Networks with Darknet53 for better performance and efficiency. Neck (PANet or FPN) is used to combine feature maps of different sizes extracted from Backbone. The importance of it comes from the fact that objects appear in different locations in the image and have different sizes. Path Aggregation Network (PANet) or Feature Pyramid Network (FPN) technology is mainly used. The Head part performs final object detection, predicting the bounding box, class label, and confidence score for each grid cell. It mimics the Head structure of previous models (YOLOv3 or YOLOv4), and uses several techniques such as anchor boxes. It is a deep learning model designed for object detection. Since the YOLO series is designed for real-time object detection, the processing speed is a vital feature of this algorithm. One of YOLOv5 merits is that it reaches very high inference speed, which enables it to be used in a variety of real-time processing operations, such as real-time video streaming and self-driving cars. Another essential characteristic of the algorithm is accuracy, that is the ability to accurately predict an object's position. YOLOv5 offers improved mAP compared to previous versions and effectively detect objects of various sizes and shapes. Accuracy advancement in version 5 significantly improves the overall performance of the YOLO algorithm. Furthermore, YOLOv5 has an easy-to-use interface and powerful customization features, which helps developers adapting it to their own datasets and requirements.

Table 3 compares the performance of each model of YOLOv5. The parameters represent the number of weights the model needs to learn. Models with multiple parameters tend to be more complex and represent the data more accurately. However, too many parameters lead to overfitting problems. 'M' stands for 'Million', and in deep learning it usually represents the number of parameters in millions. FLOP (G): FLOP stands for 'Floating Point Operations Per Second', meaning the number of Indicates the number of floating-point operations. This is an important metric to measure the computational complexity and performance of the model. 'G' stands for 'Giga' or 'Billion', and FLOPs are usually expressed in units of GigaFLOPs or TeraFLOPs. YOLOv5, the model of YOLOv5s, which is fast and relatively accurate, has a small number of parameters and FLOPs compared to other models. It was taken into consideration and used to improve the core network. YOLOv5 requires a large amount of training data to operate satisfactorily, which is a common limitation of deep learning algorithms. In addition, detecting objects in a complex background may be difficult for the algorithm, hence worsening its performance. This is a common flaw in deep learning-based object detection algorithms.

In terms of speed and accuracy, YOLOv5 acts better than its predecessors [8]. Due to the optimized network architecture and the introduction of new technologies, YOLOv5 has been able to achieve higher speeds and higher mAP. YOLOv4 used the backbone of CSPDarknet53, but YOLOv5 improved performance by using a more efficient backbone. Regarding the model size, YOLOv5 offers a smaller model size compared to previous versions while

**TABLE 3.** The performance comparison of different models of Yolov5.

| Model | pixels | mAP(0.5) | mAP(0.5:0.95) | Params(M) | FLOPs(G) |
|---|---|---|---|---|---|
| YOLOv5s | 604x640 | 55.4 | 36.7 | 7.3 | 17.0 |
| YOLOv5m | 604x640 | 63.3 | 44.5 | 21.4 | 51.3 |
| YOLOv5l | 604x640 | 66.9 | 48.2 | 47.0 | 115.4 |
| YOLOv5x | 7604x640 | 66.8 | 50.4 | 87.7 | 218.8 |

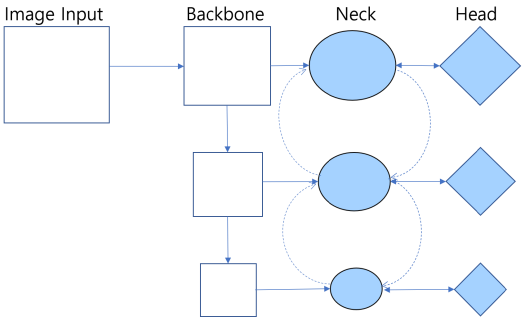


**FIGURE 6.** YOLOv5 Model.

maintaining high performance. This means that you achieve good performance while using fewer computing resources. Thanks to this, YOLOv5 is used even in environments with limited computing resources. Additionally, YOLOv5 is more user-friendly and easier to customize than previous versions. This is a significant advantage that allows developers to easily adapt YOLOv5 to their own datasets and requirements. Also, compared to earlier versions, the code is more concise and easier to understand, and additional features make it more user-friendly.

As shown in Figure 6, YOLOv5 specializes in fast and accurate object detection in images based on neural network architecture. The YOLOv5 network structure has undergone many optimizations and upgrades to maximize efficiency of its performance. YOLOv5's backbone is responsible for extracting features from images. It consists mainly of Convolutional Layers, from which spatial information and features of the image such as texture and color are extracted. YOLOv5 uses a backbone structure called CSPDarknet53 [9]. The neck is used to combine and refine the features extracted from the backbone. YOLOv5 uses Path Aggregation Network (PANet) [10] and Feature Pyramid Network (FPN) [11] to effectively combine features at different scales. This allows the network to better detect objects of different sizes. The head detects the final object. Here, the bounding box's coordinates, the object's class, and confidence of them are predicted. The head of YOLOv5 uses multiple anchor boxes for each grid cell to make these predictions.

YOLOv5 provides different versions (YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x) depending on the size and complexity of the network, which allows users to choose a model that suits their needs and resources.

## III. PROPOSED METHOD

### A. SYSTEM ARCHITECTURE

In the field of computer vision and medical imaging, various artificial intelligence technologies for detecting objects are developed. Typically, CNN's deep learning algorithms detect suspected semiconductor components in an image and draw a bounding box around them. However, conventional deep learning-based CNN algorithms have limitations in real-time detection due to long processing time and low detection accuracy. To improve this point, YOLOv5s is used as the basic object detection model. By feeding images into the object recognition model, the YOLOv5s model detects objects and outputs bounding boxes and class labels. Due to the fact that the improved YOLOv5s model uses transformers, it learn to pay attention to specific parts of an image and consider the context of detected objects, improving the overall accuracy and robustness of the system.

It is possible to train a model combined from a large amount of labeled image datasets end-to-end. In this study, we improved the YOLOv5s model that solve the problem of LED-Display defect detection by inputting CMOS images. We trained a model based on deep learning with Yolov5, which has improved the shortcomings and performance of existing CNN deep learning algorithms.The hyperparameters [12] required for training include the learning rate [13]], optimizer function [14], activation function [15], and number of training epochs [16] that determine the performance of the trained model. Therefore, changing the hyperparameters optimize the performance of the learning model, and appropriate hyperparameters must be applied according to the characteristics of the object or part to be detected. In this study, learning was conducted while modifying hyperparameters using the YOLOv5s model. The model was trained by varying the batch size [17], activation function, optimization function, loss function [18], and number of learning times [19]. Precision and mAP [20] were measured to evaluate the performance of the trained model.

### B. IMPROVED YOLOv5 MODEL

YOLOv5 and Transformer are two major architectures that play an important role in artificial intelligence and deep learning, respectively. When they operate together as one system, their respective strengths work in tandem to further increase the overall performance. YOLO is one of the most widely used deep learning systems for object detection Its key advantages are fast processing and real-time detection capability, which is possible because YOLO processes the entire image at once. This is very useful in situations such as real-time video streaming or CCTV monitoring. In addition, because YOLO considers the entire image, it is good at figuring out the relative positions between objects, which is important for improving object detection accuracy.

On the other hand, Transformer is a model that has revolutionized the field of natural language processing (NLP). The main benefit of the Transformer is its unique structure based on the attention mechanism, which consider all elements of the input simultaneously and capture the relationship between each element. Also, Transformer process the relationships between all inputs in parallel, making the processing very fast, which helps to significantly reduce the training time.

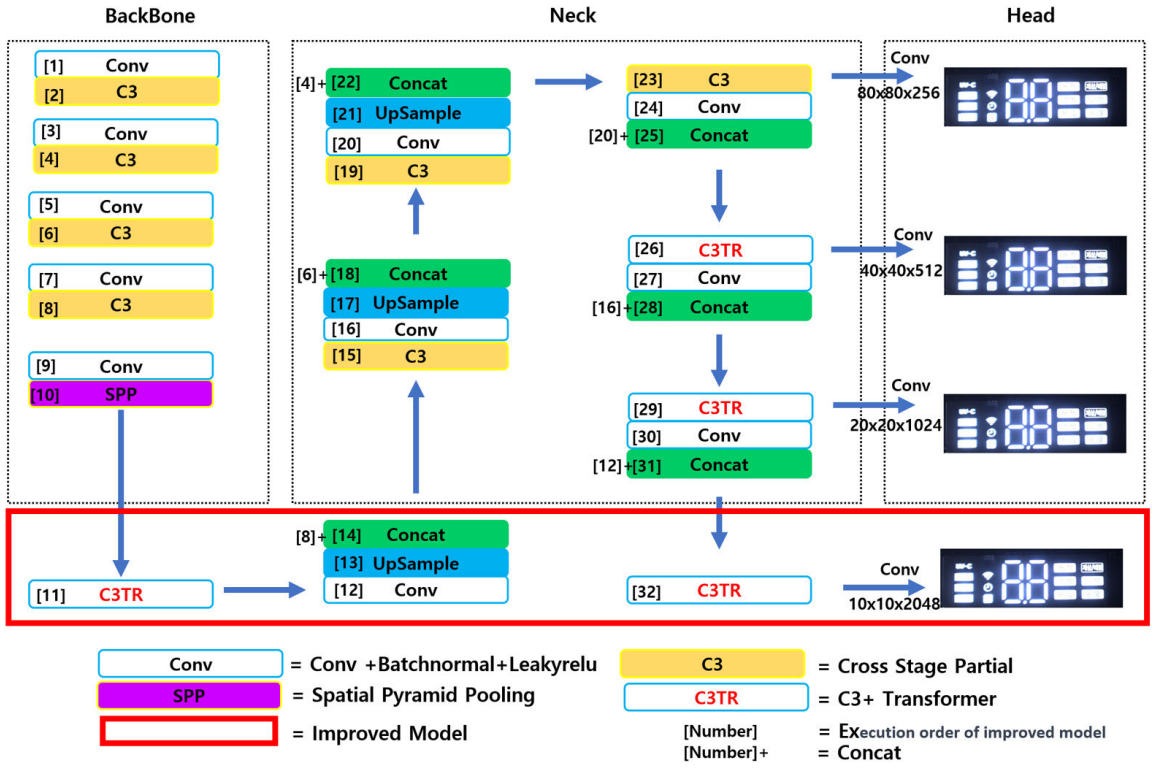


**FIGURE 7.** Improved Model.

Transformers are also good at capturing dependencies in long sequences, which makes them perfect for understanding important contexts in sentences or long sequence data. YOLO quickly and accurately detects and precisely locates objects in images. Translators help understanding the complex relationships between these objects, and more importantly, how they interact with each other. In this way, a system that combines YOLO and Transformer understand images more deeply in terms of detection of objects and their interactions. In this paper, to inspect the LED-Display image, the finished product is connected to the LED jig and the power supply status is applied to the PCB to determine whether it is good or not. Finally, check the LEDs on the PCB, if 15 LEDs are powered on, the final product operates correctly and otherwise it does not. The network configuration is shown in Figure 7 as an improved model from the YOLOv5s model. BACKBONE adds one Transformer Layer to acquire more varied images, changes the existing convolution structure to C3TR [26] and Transformer structures in the HEAD part, as well as adds one existing HEAD, changing from 3 HEAD to 4 HEAD.

The C3TR layer combines a transformer with the C3 layer of CSPDarknet53, and is used in the YOLOv5 object detection model. This layer is responsible for extracting and integrating features from different parts of the image, leveraging the capabilities of the transformer.The C3 layer in CSPDarknet53 is designed for feature extraction, while the transformer is excellent at capturing relationships between different parts of the data. By combining these two functionalities in the C3TR layer, the model extract features from the image and understand the relationships between them, which is crucial for object detection tasks. The C3TR layer takes advantage of the transformer's ability to learn hierarchical representations, from low-level to high-level features of the image. This is combined with the CSPDarknet53's feature extraction capabilities to enable feature learning with a broader context in mind. C3TR supports stable learning by utilizing the transformer's Layer Normalization and Residual Connection, which prevents gradient vanishing/explosion problems in the learning process and enables stable learning in deep networks. C3TR layer is designed to handle multi-scale processing, receiving and integrating feature maps at different scales for processing, resulting in equally well detection of objects of different sizes. The head of the C3TR layer downsamples the feature map, combines it with other feature maps, and passes it through the transformer. Through this process, you learn the characteristics of different images scales. This layer serves as the input feature map for channels 512, 1024 and 2048. C3TR is a layer that combines a transformer with the C3 layer of CSPDarknet53 and is used in the YOLOv5object detection model. This layer is responsible for extracting and integrating features from different parts of the image, leveraging the capabilities of the transformer. There are advantages to using the C3TR layer. In terms of positional independence, transformers are structures that process input data in any order and manage to find positional patterns in images or sequences. This feature is very important for problems such as object detection. For

Transformers, the location of the object is irrelevant, because they see it in the image anyway. Consistent performance obtained with the C3TR layer, regardless of the position of the object. The transformer learns the relationships between all elements of the input, which is favorable when learning features in multiple contexts. This ability, combined with CSPDarknet53's feature extraction capabilities, enables feature learning with a broader context in mind. Additionally, because the features are learned at different scales, scale-insensitive models.

In order to ensure efficient operation, the transformer is designed with a parallel processing structure. Therefore, the C3TR layer correctly calculate using these characteristics of the transformer. This is especially useful when dealing with large images or large amounts of data. The transformer is flexible enough to handle different types of data. It be used effectively in various areas such as image, text, and voice. The C3TR layer leverages the strengths of these converters to easily adapt to all kinds of problems. Translators use the attention mechanism internally, allowing the model to pay more attention to important features and ignore less important ones. It also helps to detect objects in images with complex and varied patterns. The C3TR layer takes advantage of the transformer's ability to learn hierarchical representations, which means from low-level to high-level features of the image, allowing understanding and integrating features at different levels of the image. C3TR supports stable learning by utilizing Transformer's Layer Normalization and Residual Connection, which prevents gradient vanishing/explosion problems in the learning process and enables stable learning in deep networks. Multi-Scale processing is possible, and feature maps at different scales are received and integrated for processing, resulting in equally well detection of objects of different sizes.

In Table 4 Focus layer: The input size depends on the dimensions of the image, but is typically 640 × 640x3. The output size is 320 × 320x64. This layer combines adjacent pixel information to extract better features from high-resolution images. Conv layer: input size is 10 × 10x2048 and output size is 10 × 10x1024. This layer uses 1 × 1 convolution to reduce the number of channels in the feature map. Upsample layer: input size 10 × 10x1024, output size 20 × 20x1024. This layer uses the nearest-neighbor method to increase the size of feature maps. Concatenation layer, input size is 20 × 20x1024 (upsampled features) and 20 × 20x1024 (8th layer output), output size is 20 × 20x2048. This layer connects two functional maps at the channel level. Layer C3: The input size is 20 × 20x2048 and the output size is 20 × 20x1024. This layer uses CSPDarknet53's 3-way convolution block to expand the dimensionality of features and learn cross-dimensional correlations. Conv layer: input size is 20 × 20x1024 and output size is 20 × 20x512. This layer uses 1 × 1 convolution to reduce the number of channels in the feature map. Upsample layer: input size 20 × 20x512, output size 40 × 40x512. This layer uses the nearest-neighbor method to increase the size of feature maps. C3TR with described block structure uses transformer technology to successfully transform the input feature maps and capture long-term dependencies, allowing the YOLOv5 model to perform more precise object detection. Efficient transformer-based learning allows C3TR to capture long-term dependencies by leveraging transformer technology. Transformers work well in sequence modeling and are more effective than traditional methods when applied to nonsequential tasks such as object detection. Long-term dependency is the concept of correlation between earlier and later elements in a sequence or spatio temporal data. In other words, the long-term dependencies are meaningful relationships between the current element and the before/after elements. For example, the contextual dependence between words in a sentence or the relationship between objects positions or movement between successive frames of an image over time is a long-term dependence.

Capturing long-term dependencies [29] means that the model identifies meaningful relationships between former and latter elements of the input data and exploits them appropriately. Models capture long-term dependencies to understand the temporal flow or spatial relationships of data, and then to predict or analyze the next step. In natural language processing, recurrent neural networks (RNNs) [30] make next-step predictions by reflecting past inputs in the current state to capture long-term dependencies which enables understanding contextual relationships between words in a sentence and performing tasks such as language modeling or machine translation. A type of artificial neural network that processes sequence data in a recursive way. RNNs process each element of an input sequence consecutively, capturing long-term dependencies by transferring the state from the previous step to the current one. Due to that, understanding the temporal meaning of input data and predicting or analyzing is possible. It basically has a repeating structure. Each step in the sequence takes as input data the state of the previous step and computes the output of the current step. This output is transferred to the state of the next step and used in the calculation of the next step. This iterative process allows RNNs to capture long-term dependencies. The most basic form is Simple RNN. Simple RNN computes the output of the current step by linearly transforming the inputs and states of the previous step along with the weights, and then apply a non-linear activation function. However, Simple RNNs do not capture long-term dependencies well. To overcome this, variants of RNNs such as Long Short-Term Memory (LSTM) [31] and Gated Recurrent Unit (GRU) [32] have been developed. LSTMs and GRUs have memory cells that control and store input and state information by introducing a gate mechanism. This allows for better capture of long-term dependencies. The LSTM uses gates such as input gate, delete gate, and output gate, whereas GRU uses reset gate and update gate to control the flow of information. It consists of an iterative process of storing and calculating the output of the current step. This enables RNNs to capture long- term dependencies

**TABLE 4.** Improved model architecture.

| Layer | Input | Output |
|---|---|---|
| Focus | 640x640x3 | 320x320x64 |
| Conv | 320x320x64 | 160x160x128 |
| C3 | 160x160x128 | 160x160x128 |
| Conv | 160x160x128 | 80x80x256 |
| C3 | 80x80x256 | 80x80x256 |
| Conv | 80x80x256 | 40x40x512 |
| C3 | 40x40x512 | 40x40x512 |
| Conv | 40x40x512 | 20x20x1024 |
| C3 | 20x20x1024 | 20x20x1024 |
| Conv | 20x20x1024 | 10x10x2048 |
| SPP | 10x10x2048 | 10x10x2048 |
| C3TR | 10x10x2048 | 10x10x2048 |
| Conv | 10x10x2048 | 10x10x1024 |
| Upsample | 10x10x1024 | 20x20x1024 |
| Concat | 20x20x1024, 20x20x1024 | 20x20x2048 |
| C3 | 20x20x2048 | 20x20x1024 |
| Conv | 20x20x1024 | 20x20x512 |
| Upsample | 20x20x512 | 40x40x512 |
| Concat | 40x40x512, 40x40x512 | 40x40x1024 |
| C3 | 40x40x1024 | 40x40x512 |
| Conv | 40x40x512 | 40x40x256 |
| Upsample | 40x40x256 | 80x80x256 |
| Concat | 80x80x256, 80x80x256 | 80x80x512 |
| C3 | 80x80x512 | 80x80x256 |
| Conv | 80x80x256 | 40x40x256 |
| Concat | 40x40x256, 40x40x256 | 40x40x512 |
| C3TR | 40x40x512 | 40x40x512 |
| Conv | 40x40x512 | 20x20x512 |
| Concat | 20x20x512, 20x20x512 | 20x20x1024 |
| C3TR | 20x20x1024 | 20x20x1024 |
| Conv | 20x20x1024 | 10x10x1024 |
| Concat | 10x10x1024, 10x10x1024 | 10x10x2048 |
| C3TR | 10x10x2048 | 10x10x2048 |
| Detect | 80x80x256, 40x40x512, 20x20x1024, 10x10x2048 | |

of input sequences used by various applications, including natural language processing, speech recognition, and time series prediction.

C3TR captures the long-term dependencies of input feature maps through a self-attention mechanism. This allows the model to take into account the wider context and understand global object information for accurate object detection. The right combination of lines and transformers ensures high efficiency with small number of parameters. Therefore, the models is small and lightweight, reducing inference speed and memory requirements. The combination of C3TR and YOLOv5 is convenient, because transformer-based long-term dependency modeling and convolution-based visual feature extraction improves accurate object localization and better classification performance. Capturing long-term dependencies means that the model understands the overall structure and meaning of the input data and uses them appropriately to make predictions or analyzes. It is an important factor in various application fields such as natural language processing, computer vision, and voice processing.

**TABLE 5.** Training DataSets.

| Item | Description |
|---|---|
| Image Total | *986* |
| Training Set | *860* Images (*83%*) |
| Validation Set | *88* Images (*11%*) |
| Testing Set | *38* Images (*8%*) |
| Preprocessing | Auto-Orient: Applied<br>Resize: Stretch to *640x640*<br>Tile: *2* rows x *2* columns |
| Augmentations | Outputs per training example: *3*<br>90° Rotate: Clockwise, Counter-Clockwise |

## IV. EXPERIMENTAL RESULTS

### A. EXPERIMENTAL ENVIRONMENT

The experimental conditions were configured as follows.Improved model used CPU i9 11900k, GPU K80, T4 and P 100, and RAM 52GB as hardware in the experimental environments for applying the YOLO-based object feature search method. All of LED- Display images were used as the image dataset for training, which have been preprocessed beforehand. The pre-learning iteration was carried out up to 30,000 times, among which the weight used was when the loss was the lowest.

### B. DATASETS

Table 5 shows the contents of the data preprocessing step. The contents of the preprocessing are as follows. Preparing LED-Display Images for Training involved obtaining photographs directly from the final production stage of LED products on site. All images were resized to fixed dimensions before being fed into the model. Specifically, for YOLOv5, the typical input image size was set to 640 × 640 pixels, meaning that all input images were resized to 640 × 640 pixels. Additionally, the pixel values of the images were normalized to a range between 0 and 1. The characteristics of the images include a high contrast ratio in the LED display, which is essential for distinguishing between good and defective products due to the significant contrast between the bright LED lights and the dark background. The LED display express various colors, typically generating a range of colors using red, green, and blue (RGB) LEDs. However, in this context, only white is processed to make the light emission familiar to the user's eyes for air conditioners. The resolution of the LED display varies with the number of pixels, as each LED is considered one pixel; hence, the more LEDs, the higher the resolution. There is one LED chip located on the PCB. The LED display images may include noise, which could be due to the camera sensor's noise, the reflection of external light sources, or the irregular brightness of the LED display. Nevertheless, product inspection is conducted in a darkroom-type structure to block out surrounding noise.

- Labelling: Image Selection: First, select the images to be labeled. These images include various states of the LED Display. Only the 'on' state is used for training. Image Loading, Load the selected images into

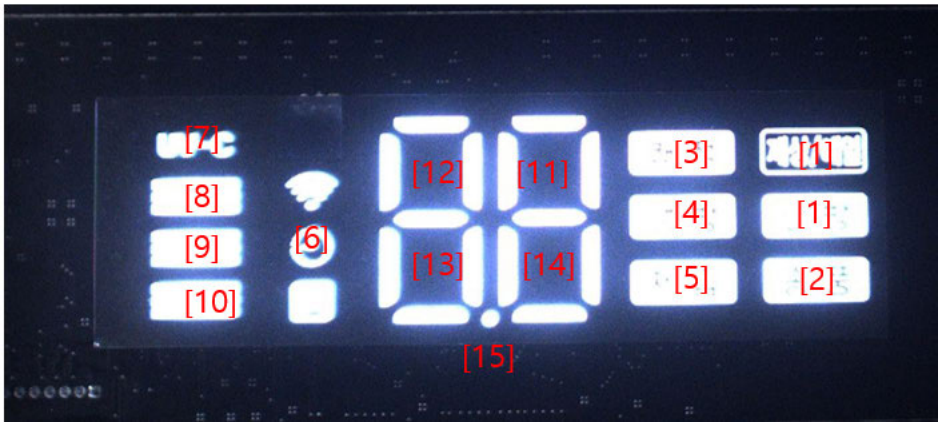


**FIGURE 8.** LED-Display Class.

a labeling tool. The open-source tool LabelImg was used for this purpose.Object Selection, Select the objects to be labeled in the image, distinguishing between LED Chips 1 to 15. Drawing Bounding Boxes, Draw bounding boxes around the selected objects. These boxes indicate the location and size of the objects. Class Label Assignment, Assign class labels (1∼15) to the drawn bounding boxesFigure. LED Chip was assigned label called ''LED_Chip''. Figure 8 is an illustration that classifies the classes of LED-Displays.Label Saving, Save the assigned labels and bounding box coordinates. Typically, this information is saved in a text file. In the case of the YOLO model, one text file is created for each image. This file includes the bounding box coordinates and class labels for all objects in the image. Repeat these steps for all images. Annotation File, For each image, create a text file containing the coordinates of the bounding boxes and the class labels of the corresponding objects. YOLO includes the center coordinates (x, y) of each bounding box, the width and height of the box, and the class of the object. If the LED Chip is included in the image and the center coordinates of the LED Chip are (x, y), and the width and height are w and h, respectively, the annotation file will be labeled in the form of a *.txt file. The LED-Display consists of 15 sections, from 1 to 15. The brightness of each LED light is identical. The LED Chip is located on the PCB. The position of the LED Chip that forms the characters is determined to Display numbers. These chips illuminate the LED lights according to the corresponding signals, serving as a Display that conveys the air conditioner's information. The LED Chip is produced to form the shape of the characters and is inspected at the final assembly stage on the production line. The LED Chip on the PCB board is configured with a plastic structure called a reflector attached on top of it to allow the LED light to emit outward. The LED light is Displayed in the shape of numbers due to the rectangular configuration of the reflector, at which point the quality and defects of the LED are determined. If power is supplied, it is deemed PASS, and if power is not supplied, it is considered defective. The positions of each LED Chip are labeled with numbers from 1∼15, dividing them into classes. They are grouped into three intermediate groups: 1∼5, 6∼10, and 11∼15, designed to identify the part of the entire PCB board where the LED Chip defects occur. The first group, numbers 1∼5, represents the LEDs located on the right, the second group, numbers 6∼10, represents the LEDs in the left, and the third group, numbers 11∼15, represents the LEDs on the center. The difference between each Class is the location of the Chip on the PCB, while the rest of the product's characteristics and functions are identical.

As shown in Figure 9, image labeling involves setting the correct category for each image. This allows computer vision algorithms and machine learning models to understand images and automatically recognize what they contain. Adjacent pixel values are grouped and numbered to identify each object in the binarized image. Black areas of an image is individually selected and labeled. When areas of a region were carefully labeled according to the segments, labeling gave good results in all segments. Each image has been verified and calibrated for accurate labeling for LED Display applications. When an image is entered, the labeling must be accurate because the image area must be recognized and the image processed to obtain a meaningful value. The illuminated areas of the image were individually selected and labeled. When areas of a region were carefully labeled according to segments, labeling gave good results across segments. To apply the LED-Display, each image was checked and corrected to ensure accurate labeling. When an image is input, the labeling must be accurate in that the image area must be recognized and the image processed to obtain a meaningful value.

- Data Augmentation: Data augmentation is used to prevent overfitting and improve the generalization

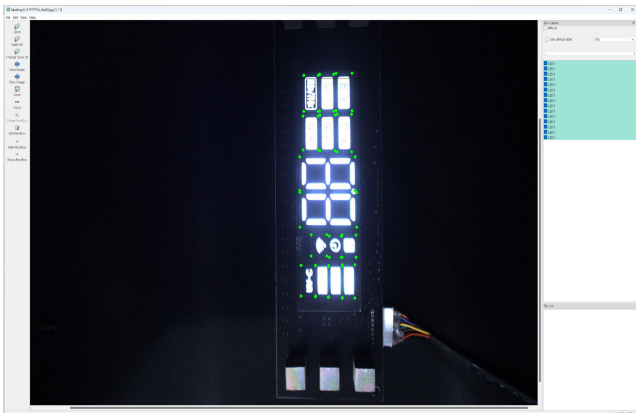

**FIGURE 9.** LED-Display labelling.

performance of the model. YOLOv5 uses various data augmentation techniques such as rotation, scaling, brightness adjustment, and color adjustment. By providing a variety of images through 90-degree rotation, clockwise and counterclockwise direction changes, and direction changes of 15 degrees before and after, we aimed to increase the accuracy of product detection.

- Training, Validation, Test Data: Divide the preprocessed images into training, validation, and test sets. Generally, about 70∼80% (860) of the total data is used as the training set, 10∼15% (88) as the validation set, and 5∼10% (38) as the test set. The training set is used to train the model, the validation set is used to evaluate the performance of the model and adjust the hyperparameters, and the test set is used to evaluate the final performance of the trained model.

Figure 10 shows the appearance of the Train, Validation, and Test Datasets.

### C. PERFORMANCE METRICS

Precision [33] is used in conjunction with recall. This is an indicator of how accurate the prediction results are. That is, since the ratio of correct answers among the detected items is known, it is known how accurate the detection result is. The precision is divided by TP/(TP+FP). Recall is the number of correct answers among GT (ground truth), which is the percentage of correctly detected objects among all the objects to be detected. Recall is divided into TP/(TP+FN). Average precision is calculated by increasing the recall from 0 to 1 by 0.1 units (total of 11 values), and the precision inevitably decreases. Precision values are calculated and averaged for each unit. That is, the average of the precision values over 11 repeatable values is called AP. AP values is calculated for each class, and mAP is obtained by calculating and averaging AP over the total number of classes.

For the empirical assessment of the proposed model, results were validated employing the confusion matrix, a quintessential evaluation metric in binary classification.

The confusion matrix is characterized by the following four fundamental metrics in Table 6.

**TABLE 6.** Confusion Matrix.

| | Predicted Positive | Predicted Negative |
|---|---|---|
| Actual Positive | TP (True Positive) | FN (False Negative) |
| Actual Negative | FP (False Positive) | TN (True Negative) |

**TABLE 7.** Hyperpimeter.

| Hyperpimeter | Value |
|---|---|
| Batch Size | 2,4,8,16, 32, 64 |
| Activation Function | Leaky ReLU |
| Optimizer | SGD, Adam |
| Loss Function | Cross Entropy Loss [24] |
| Number of Epochs | Early Stopping |

- **True Positive (TP):** Represents instances where the actual class was positive, and the model correctly predicted it as positive.
- **True Negative (TN):** Denotes instances where the actual class was negative, and the model accurately predicted it as negative.
- **False Positive (FP):** Refers to instances where the actual class was negative, but the model mistakenly predicted it as positive. Also known as "Type I Error".
- **False Negative (FN):** Represents cases where the actual class was positive, but the model erroneously predicted it as negative. Also known as "Type II Error".

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detectionss}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}} \tag{2}$$

Eq. 1 defines precision, while Eq. 2 defines recall.

$$AP = \int_0^1 p(r)\, dr \tag{3}$$

$$mAP = \frac{1}{C} \sum_{c=1}^{C} AP_c \tag{4}$$

Eq. 3 defines average precision (AP), and Eq. 4 defines mean average precision (mAP).

mAP stands for mean average precision and is widely used as evaluation metric for object detection and instance segmentation tasks in computer vision. In an object detection task, the goal is to identify an object of interest in an image and draw a bounding box around it. The mAP metric measures the effectiveness of the detection algorithm in identifying these objects, taking into consideration both precision and recall. Precision measures the percentage of correctly detected objects, while recall measures the percentage of objects actually detected. mAP is the average precision across all recall levels, and the precision is interpolated between recall levels. The resulting mAP is often used to compare the performance of different object detection models or to evaluate the performance of a single model in a validation set. The higher the mAP, the better the performance on the task. mAP = (AP1 + AP2 + ...+ APn) / n where n is the total number of classes and AP1, AP2, ..., APn are the average precision values for each class.

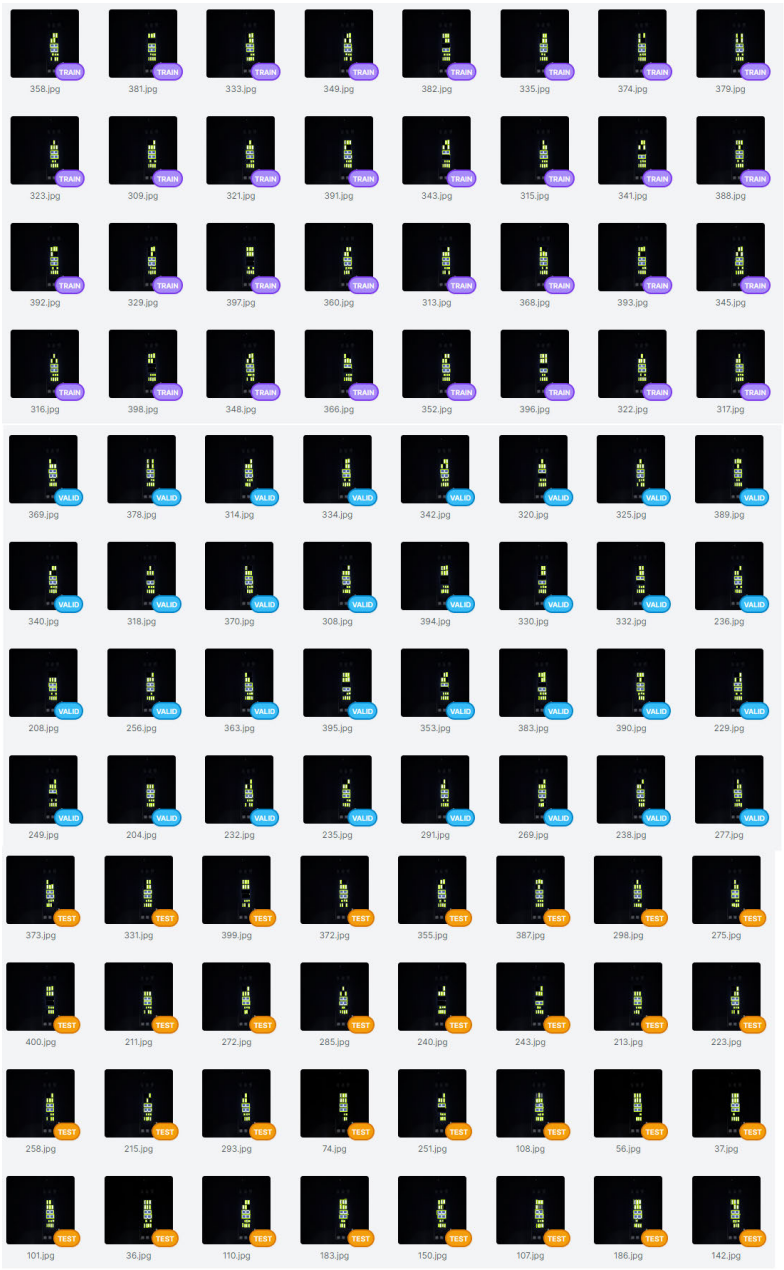

**FIGURE 10.** Train, Validation, Test Dataset.

When setting the hyperparameters, it is essential to consider the model complexity, training time, generalization performance, overfitting and underfitting, optimization algorithm, learning rate, batch size, and the number of epochs as these factors significantly influence the model's performance and training efficiency in Table 7.The GPU memory is limited, for instance, considering the 12GB memory of the NVIDIA Tesla K80 GPU, setting the batch size to 2 restricts the memory usage to a maximum of 12GB. Such a small batch size aids in preventing the model from overfitting to the training data. Specifically, when the batch size is 2, the model becomes less sensitive to the noise in the training data, which enhance the model's generalization performance. A smaller batch size enables more efficient use of the GPU, which accelerate the training speed by at least one fold.

The Adam optimizer is characterized by its ability to adjust the learning rate individually for each parameter. For example, with an initial learning rate of 0.001, the Adam optimizer automatically adjusts the learning rate for each parameter, optimizing the convergence speed.

The loss function measures the difference between the model's predicted values and the actual values. In the YOLO model, three types of losses are typically combined: classification loss, localization loss, and confidence loss. The classification loss is computed using cross-entropy loss, the

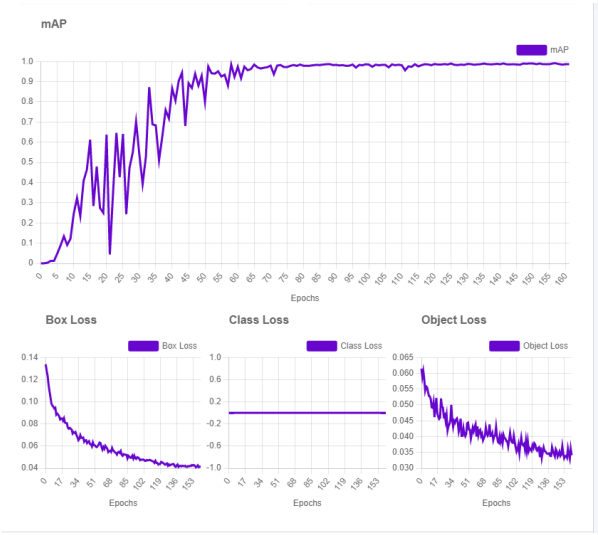


**FIGURE 11.** Train mAP Value.

localization loss is computed using mean squared error, and the confidence loss is used to predict the probability that each bounding box contains an object.

Leaky ReLU is a variant of the basic ReLU function. The basic ReLU function returns the input directly if it is positive and returns 0 if it is negative. However, Leaky ReLU returns a line with a small slope (0.01) for negative inputs, addressing this issue. This allows the model to retain some information from negative inputs, thereby improving the model's performance.

### D. RESULTS

A crucial aspect of the proposed method is the inclusion of the transformer layer in the YOLO architecture. Transformers are recognized for their ability to capture long-term dependencies in the data, a significant advantage over other models that struggle with this aspect. This is particularly vital for object detection tasks, where the context in which an object appears is crucial for its accurate identification.In the YOLO architecture, the transformer layer is added to enhance the model's capability to capture relationships between different parts of an image, thereby improving detection accuracy. Specifically, the self-attention mechanism of the transformer enables the model to weigh the importance of different parts of the image when predicting the presence of an object. This allows the model to capture more global, long-range dependencies across the entire image, rather than merely focusing on local features. Consequently, the proposed model more accurately identify objects in complex scenes with multiple objects and varying backgrounds. The computational cost of the proposed model, an aspect not sufficiently highlighted in the discussion, does not significantly increase the computational burden of the model despite its ability to capture long-term dependencies. This is because the self-attention mechanism parallelized across all positions in the input sequence, making it more efficient than models that process the input sequentially. The improved model demonstrated very satisfactory performance as a result of the transformer's long-term dependencies.

**TABLE 8.** Comparison of Object Detection Models.

| Parameter | Precision | Recall | mAP 0.5 |
|---|---|---|---|
| YOLOv5S | 0.889 | 0.609 | 0.881 |
| YOLOv5m | 0.880 | 0.594 | 0.877 |
| YOLOv5l | 0.870 | 0.596 | 0.860 |
| YOLOv5x | 0.865 | 0.571 | 0.858 |
| Improved model | 0.985 | 0.990 | 0.994 |

**TABLE 9.** Comparison of Object Detection Models by Group.

| LED Chip | Precision | Recall | mAP 0.5 |
|---|---|---|---|
| NO.1~5 | 0.985 | 0.990 | 0.994 |
| NO.6~10 | 0.982 | 0.991 | 0.995 |
| NO.11~15 | 0.980 | 0.988 | 0.992 |
| YOLOv5s | 0.889 | 0.609 | 0.881 |
| Improved model | 0.985 | 0.990 | 0.994 |

Data preprocessing is a very important process in training machine learning models. Preprocessing cleans and converts the raw data into a form that the model understand and learn. The processed data significantly improve the performance of the model. Incomplete or invalid data degrade the model's performance. Preprocessing remove or replace missing values, outliers, noise, etc., which helps to increase the accuracy and stability of the model. Not all features provide useful information. Unnecessary characteristics degrade model performance and extend training time. Preprocessing removes irrelevant or redundant features and selects or extracts important ones. Different features have different units and ranges, which destabilize the training process or make the model overly dependent on some characteristics. Data Scaling solves this problem by transforming all features into the same range. Data preprocessing helps to understand the structure and nature of the data. Therefore, making important decisions in other steps such as feature selection, model selection, and hyperparameter tuning done. Improving the quality of data through preprocessing increases the chances that the trained model will give more consistent and reliable results. Outliers or noise may prevent a model from generalizing. Data preprocessing minimize the influence of these factors. It is a key component of any machine learning pipeline. Raw data is incomplete or irregular, and take many shapes and formats. Converting this data into a form that the model is able to understand and learn improve the performance of the model, streamline the training process, and increase the reliability of the results. Precision represents the proportion of things that are actually true out of the things that the model classifies as true. In other words, it is a metric that evaluates how accurately the model detected the object. A value of 0.988 means that the model detects objects almost flawlessly. Recall represents the proportion of things that the model classifies as true out of the things that are actually true. That is, it is a metric that evaluate many objects the model was able to detect out of all real objects. A value of 0.989 means that the model successfully detected most real objects.

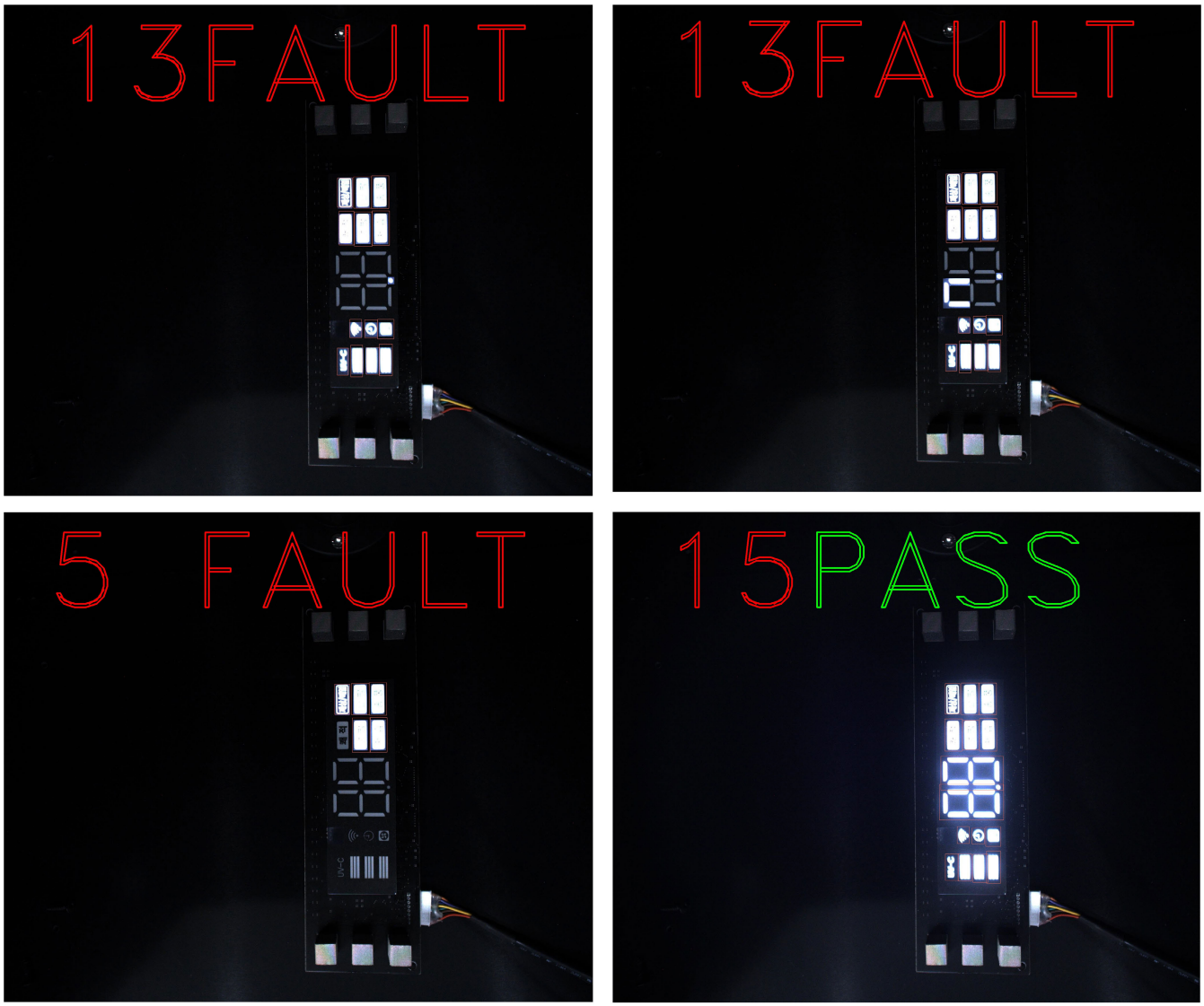


**FIGURE 12.** Detection result using the improved model (selection of good and bad products).

mAP50 (mean Average Precision at 50% IOU): mAP50 is obtained by calculating the Average Precision (AP) for each class and averaging over classes. AP represents the area under the Precision-Recall curve, and is a metric that summarizes the model's performance in one shot. Intersection Over Union (IOU) is a method for calculating the area of overlap between the predicted bounding box and the actual bounding box. 50% in mAP50 means that only when the IOU is greater than 50% it is considered correctly detected. A value of 0.991 means that the model performed very well in every class. These results indicate that the YOLO model has very high accuracy and recall, as well as excellent detection performance for each class, resulting in high confidence in detecting objects. Table 8 presents the result of measuring the accuracy according to the difficulty and label order after adjusting the parameters by batch file size adjustment. The mAP value is expressed as 0.994, the final value representing the final mAP value. Thus, the mAP value improved from 0.889 to 0.994.As shown in the Table 9, the data was preprocessed and trained. 860 training data, 88 validations, and 38 tests were conducted. In the data augmentation, 90-degree rotation and 15-degree left-right rotation were performed. Even if the image of the LED-Display that was a sample was not settled on the jig, it could be inspected. The picture below is a picture rotated left and right, and the three purple, sky blue, and yellow datasets are pictures of data that have gone through training, validation, and testing. As illustrates Table 9, LED- Display images were divided into 15 classes and classified by labeling. It was considered to split into three groups 1∼5, 6∼10, 11∼15 and identify which parts were poorly recognized using a classification system for each group. Divided into divisions and grouped for detection. As shown in the resulting figure, the group with a large image area in the center shows the highest map, and a large image between image and detection is effective. Figure 11 shows the training status as mAP value, and that when the data

training Epoch increases, the rate of not being able to detect Box and Object decreases. As shown in Figure 12,the LED-Display image was finally tested, and if 15 points in the image were found, it passed, otherwise it was treated as defective, and the learned result was inferred. The overall result was satisfactory.

## V. CONCLUSION

The cornerstone of the proposed methodology is the integration of a transformer layer into the YOLO architecture. Transformers are renowned for their superior ability to capture long-term dependencies in the data, a notable advantage over other models that struggle in this aspect. This is particularly critical in object detection tasks, where the context in which an object is situated plays a pivotal role in its accurate identification. Within the YOLO architecture, the transformer layer is incorporated to bolster the model's capacity to discern relationships between disparate segments of an image, thereby enhancing detection accuracy. Specifically, the self-attention mechanism of the transformer bestows the model with the ability to assess the significance of different portions of the image when prognosticating the presence of an object. This enables the model to capture more global, long-range dependencies across the entire image, rather than merely focusing on local features. Consequently, the proposed model identify objects with higher precision in intricate scenes encompassing multiple objects and varying backgrounds. In this study, we refined the YOLOv5 model, which boasts relatively high accuracy among the latest image detection models, to accurately count the number of reel parts. As a model in which the Transformer is incorporated, Backbone 1, Neck 1, Head 1 layers are added, and the transformer module is integrated into the HEAD from 3 CSP modules to C3TR. The detection processing value for Small Device Image exceeds 10.2%, substantiating that performance has been enhanced. The content proposed in this study is highly applicable as a network processing small image for inspecting the quality of LED-Display.

The experiment was conducted with 860 learning data, 88 verification data, and 38 test data. Experiments ascertained that after modifying the anchor box, the accuracy for large parts was elevated and the ROI was identified more precisely. Additionally, relatively numerous instances of false detections were observed compared to the small size when the anchor box was modified, but the performance was experimentally validated by modifying and supplementing the network. The final developed YOLO program is a modified anchor box and network model, and performance evaluation exhibited excellent performance with a 99.4% mAP as a result of training among a total of 986 data. Since different parts of the Display have various types of images, there were limitations related to the lack of time in the research process, labeling training data, more classification, training time, and the finiteness of the system to be invested. Modeling suitable for object detection is more desirable in the industry, so research on modeling is still needed. Models like YOLOvX are constantly updated with the aim to set new research directions and improve modeling. The future trend will be the further development of object detection systems, real-time processing, learning with less data, and object detection models in various configurations. Since object detection involves different types of images, there were limitations considering the lack of time in the research process, the limitations of labeling training data, more classification, training time, and the finiteness of the system to be invested. Modeling suitable for object detection creates more demand to be applied in production, so research on modeling is continuously needed. Models such as YOLOvX are continuously being researched and developed, so we aim to set research directions and improve modeling.

## REFERENCES

[1] L. Auria and R. A. Moro, "Support vector machines (SVM) as a technique for solvency analysis," Tech. Rep., 2008.
[2] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proc. Int. Conf. Eng. Technol. (ICET)*, Aug. 2017, pp. 1–6.
[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.
[4] R. Lv, X. Wang, and T. Yang, "Small object detection with scale adaptive balance mechanism," in *Proc. 15th IEEE Int. Conf. Signal Process. (ICSP)*, vol. 1, Dec. 2020, pp. 361–365.
[5] Y. Chen, D. Dai, J. Pont-Tuset, and L. Van Gool, "Scale-aware alignment of hierarchical image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 364–372.
[6] H. Zhang, H. Chang, B. Ma, N. Wang, and X. Chen, "Dynamic R-CNN: Towards high quality object detection via dynamic training," *Computer Vision—ECCV 2020*. Glasgow, U.K.: Springer, Aug. 2020.
[7] J. Vig, "A multiscale visualization of attention in the transformer model," 2019, *arXiv:1906.05714*.
[8] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of YOLO algorithm developments," *Proc. Comput. Sci.*, vol. 199, pp. 1066–1073, Jan. 2022.
[9] D. Misra, "Mish: A self regularized non-monotonic activation function," 2019, *arXiv:1908.08681*.
[10] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.
[11] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7029–7038.
[12] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning: Methods, Systems, Challenges*. 2019, pp. 3–33.
[13] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1—Learning rate, batch size, momentum, and weight decay," 2018, *arXiv:1803.09820*.
[14] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6555–6564.
[15] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," 2018, *arXiv:1811.03378*.
[16] J. Brownlee, "What is the difference between a batch and an epoch in a neural network," *Mach. Learn. Mastery*, vol. 20, 2018.
[17] F. He, T. Liu, and D. Tao, "Control batch size and learning rate to generalize well: Theoretical and empirical evidence," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–10.
[18] J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. Broeck, "A semantic loss function for deep learning with symbolic knowledge," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5502–5511.
[19] A. Gotmare, N. S. Keskar, C. Xiong, and R. Socher, "A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation," 2018, *arXiv:1810.13243*.

[20] W. Li, T. Liu, and D. Tao, "Scan: Cross domain object detection with semantic conditioned adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 2, pp. 1421–1428.
[21] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova, "Nonlinear approximation and (deep) ReLU networks," *Constructive Approximation*, vol. 55, no. 1, pp. 127–172, 2022.
[22] D. Gao, P. Li, M. Wang, Y. Liang, S. Liu, J. Zhou, L. Wang, and Y. Zhang, "CSF-GTNet: A novel multi-dimensional feature fusion network based on Convnext-GeLU-BiLSTM for EEG-signals-enabled fatigue driving detection," *IEEE J. Biomed. Health Informat.*, early access, Jan. 31, 2023, doi: 10.1109/JBHI.2023.3240891.
[23] S. Zhang, A. E. Choromanska, and Y. LeCun, "Deep learning with elastic averaging SGD," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.
[24] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.
[25] S. Du, B. Zhang, and P. Zhang, "Scale-sensitive IOU loss: An improved regression loss function in remote sensing object detection," *IEEE Access*, vol. 9, pp. 141258–141272, 2021.
[26] M. Dai, M. M. H. Dorjoy, H. Miao, and S. Zhang, "A new pest detection method based on improved YOLOv5m," *Insects*, vol. 14, no. 1, p. 54, Jan. 2023.
[27] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "PP-YOLO: An effective and efficient implementation of object detector," 2020, *arXiv:2007.12099*.
[28] J. Xu, Z. Li, B. Du, M. Zhang, and J. Liu, "Reluplex made more practical: Leaky ReLU," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2020, pp. 1–7.
[29] G. Li, X. Du, X. Li, L. Zou, G. Zhang, and Z. Wu, "Prediction of DNA binding proteins using local features and long-term dependencies with primary sequences based on deep learning," *PeerJ*, vol. 9, May 2021, Art. no. e11262.
[30] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5308–5317.
[31] A. Shewalkar, D. Nyavanandi, and S. A. Ludwig, "Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU," *J. Artif. Intell. Soft Comput. Res.*, vol. 9, no. 4, pp. 235–245, Oct. 2019.
[32] Q. Tao, F. Liu, Y. Li, and D. Sidorov, "Air pollution forecasting using a deep learning model based on 1D ConvNets and bidirectional GRU," *IEEE Access*, vol. 7, pp. 76690–76698, 2019.
[33] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1737–1746.

**JINWOO PARK** received the master's degree in sw convergence from Korea Engineering University. He is currently pursuing the Ph.D. degree with the Department of Smart Factory Convergence, Sungkyunkwan University. He is also with the UCT AI Research Center. In 2022, he has written a dissertation on the detection field of semiconductor components, and has a lot of interest in research in this field. He is especially interested in YOLO and wants to develop research. A paper related to the Department of Smart Factory was published in a journal entitled "YOLOv5 Based Object Detection in Reel Package X-ray Images of Semiconductor Component," in 2022. He has papers and holds more than five patents.

**JIHUN BAE** was born in Daegu, South Korea. He is currently a Junior Student with the Department of Mechanical Engineering, Sungkyunkwan University. During his academic career, he has been involved in the Interactive Treadmill Project with the Rehabilitation Biomechatronics Laboratory, where he gained valuable experience and knowledge. He also applied for a patent related to drone delivery systems, demonstrating his innovative thinking and passion for creating solutions to real-world problems. He is committed to conducting intriguing and challenging research in the field of mechanical engineering.

**JONGEON LIM** was born in Cheongju, South Korea, in 1996. He is a Senior Student in advanced material science and engineering with Sungkyunkwan University, in 2023. In 2020, he participated in an optimization for perovskite green photo detector. He also participated in the contest about AI modeling using processing data. Currently he studies about the field of machine learning and semiconductor process. His research interests include process integration of semiconductor.

**BYEONGCHAN KIM** was born in Siheung, South Korea. He is currently pursuing the bachelor's degree with the Department of Mechanical Engineering, Sungkyunkwan University. In 2022, he participated in an eco-friendly tumbler development project called "Oasis" and produced an MVP, that actually works. Through this project, he improved his understanding of embedded systems. In addition, he participated in a 3-D modeling contest for future technologies to model automatic logistics up and down robots, and through this, he won the Encouragement Award. Currently, he studies about the field of AI and mechanical engineering.

**JONGPIL JEONG** received the bachelor's degree in engineering from Sungkyunkwan University, Suwon, South Korea, and the master's and Ph.D. degrees in computer engineering from Sungkyunkwan University, in 2003 and 2008, respectively. He has been with Sungkyunkwan University, since 2008, and has been an Associate Professor with the Department of Smart Factory Convergence, Sungkyunkwan University, since 2016. He is the Principal Investigator of MAKE UNIC, a key support area for smart manufacturing with Sungkyunkwan University. He has coauthored more than 300 technical articles and holds more than 20 patents. His research interests include smart factory, industrial AI, anomaly detection, manufacturing data analysis, AI-based fault diagnosis and prediction, 5G-based smart manufacturing, industrial IoT applications, AI platforms, cloud platforms, and industrial security. In 2020, he received the "Outstanding Researcher Award," which is given to the Best Professor with Sungkyunkwan University.