Comparison of Deep Learning Models for Singlsyn Defect Detection and Classification

Manida Bhonsawanwong*, Akarima Pengubon*
Department of Mathematics and Computer Science, Faculty of Science
Chulalongkorn University
Bangkok, Thailand
6434497023@student.chula.ac.th,
6434495823@student.chula.ac.th

Noboru Sonehara
Department of Mathematics and Computer Science, Faculty of Science
Chulalongkorn University
Bangkok, Thailand
Institute for Policy Studies
Tsuda University
Tokyo, Japan
sonehara@math.sc.chula.ac.th,
sonehara@tsuda.ac.jp

Akihisa Kodate
Institute for Policy Studies
Tsuda University
Tokyo, Japan
kodate@tsuda.ac.jp

Masamichi Nakamura
Production Engineering Department
Tamagawa Seiki Co.,Ltd.
Nagano, Japan
masamichi-nakamura@tamagawa-seiki.co.jp

Kazuto Kojima
University Invitation and Cooperation Promotion Office
Iida City Government
Nagano, Japan
idaigaku@city.iida.nagano.jp

Nagul Cooharojananone
Department of Mathematics and Computer Science, Faculty of Science
Chulalongkorn University
Bangkok, Thailand
Nagul.C@chula.ac.th

***Abstract*— The use of computers for inspecting Singlsyn parts before shipment has resulted in occasional errors when distinguishing between usable (OK) and non-usable (NG) parts. Initially, the company's classification system exhibited inaccuracies, sometimes misclassifying OK parts as NG, achieving an accuracy rate of 90%, necessitating manual sorting of the misclassified parts, which was time-consuming. To enhance the accuracy and efficiency in differentiating between OK and NG parts—an inherently challenging task—deep learning techniques were introduced, particularly object detection methods. The target accuracy for this research was set at 97%. A total of eight models were tested, with various parameters adjusted throughout the process. The SSD-Mobilenet-V2-FPNLite-320 model delivered the best results, improving the system's accuracy in detecting OK and NG parts from 90% to 99.58%.**

Keywords—Object detection, Object classification, Product quality inspection, Deep learning, MobileNetV2

I. Introduction

Singlsyn is an ultra-thin, absolute-angle detector known for its exceptional reliability under high temperature, vibration, and shock conditions. Its design enhances reliability and reduces production costs by eliminating the need for a coil on the rotor. By shaping the rotor core uniquely and adjusting the gap between the rotor and stator cores, variations in the output voltage amplitude are produced. The compact, ultra-thin design of the built-in model allows for installation in very limited spaces. Figure 1 (Tamagawa Seiki Co., Ltd. 2022) shows the six lineups of Singlsyn.

Fig. 1. Six lineups of Singlsyn (Tamagawa Seiki Co.,Ltd. 2022)

Before components produced by the company can advance to the next stage of production, they must pass a quality test to determine whether they are usable (OK) or defective and therefore unusable (NG). Currently, the company uses a computer program to conduct these quality tests, achieving an accuracy rate of 90%. However, due to the high visual similarity between OK and NG parts, the program occasionally misclassifies 10% of the components, often identifying OK parts as NG and discarding them. While a 10% error rate may appear relatively low, the large-scale production in the factory results in a substantial number of misclassified parts. These incorrectly classified parts require manual review by employees, causing significant delays due to the need for re-inspection and re-sorting. Upon further inspection by employees, it was discovered that some OK parts had been mistakenly rejected. This misclassification leads to additional time spent reevaluating the discarded parts.

In response to this issue, the company seeks to enhance accuracy to 97% by incorporating AI. The effectiveness of each tested method will be evaluated using the mean average precision (mAP) metric.

This paper aims to identify the most effective method and model for detecting and distinguishing between OK and NG parts using deep learning techniques. The detection process will involve analyzing images of the components. The dataset comprises three types of images: (1) OK: images of OK parts that have been correctly identified as OK by the company's equipment; (2) OK-NG: images of OK parts that have been mistakenly classified as NG; and (3) NG: images of NG parts that have been accurately identified as NG by the company's equipment.

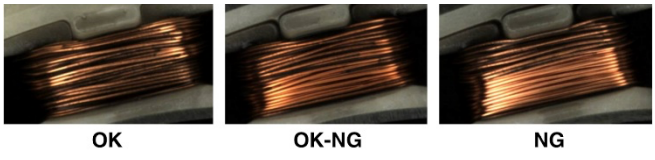

Fig. 2. Examples of OK, OK-NG, and NG

In this experiment, the model that achieved the highest mAP was the SSD-MobileNet-v2-FPNLite-320 detection. This model, which uses Single-Shot Multibox Detection

(*Manida Bhonsawanwong and Akarima Pengubon are co-first authors.*)

(SSD) with MobileNetV2 as the feature extractor and incorporates Feature Pyramid Network Lite (FPN-Lite) to enhance object detection across various scales, significantly improved the accuracy of detecting OK and NG components from 90% to 99.58%.

Section II introduces related studies on object detection and classification in inspection images. Section III presents the methodology, and Section IV verifies the experimental design. In Section V, we analyze the evaluation and experimental results, and Section VI draws deployment, and Section VII delivers our conclusions.

II. Related Works

In recent years, the fields of object detection and object classification have made significant advancements, particularly in the development of Convolutional Neural Networks (CNNs). These networks have become crucial tools in tasks such as quality inspection and defect detection. Notably, popular models like SSD MobileNet, YOLO, and EfficientDet have been widely applied in various scenarios for object detection and classification. Numerous studies have demonstrated the effectiveness of these models in enhancing accuracy and efficiency in inspecting complex objects, especially in industries that require precision, such as manufacturing and product quality control.

Quality control in manufacturing, particularly in casting processes, is crucial for ensuring product performance and competitiveness. Traditional inspection methods, which involve manual checks or basic vision systems, often fall short in terms of accuracy and speed. Recent advances in deep learning, especially Convolutional Neural Networks (CNNs), have significantly improved the efficiency of quality inspection. CNNs have been widely applied to analyze images of cast products, enhancing the precision of defect detection and classification. This paper highlights the use of CNNs for inspecting surface defects in cast products, specifically submersible pump impellers. By employing a CNN architecture with advanced data augmentation techniques, the proposed model effectively addresses data imbalance issues and achieves superior performance compared to previous approaches. The integration of data augmentation not only enhances the CNN model's ability to identify defects but also demonstrates its suitability for detailed surface inspection. Future work will focus on refining this approach to classify and locate various types of defects, such as porosity, flash burr, and cracks, further advancing the quality control process in casting industries [1].

Recent advancements in deep learning have significantly impacted object detection and classification across various fields, including environmental monitoring and industrial quality control. Federico Zocco and colleagues (2024) presented a novel approach to marine debris detection using Autonomous Underwater Vehicles (AUVs) equipped with an optimized version of EfficientDets, a state-of-the-art object detection model. By refining the architecture—specifically reducing BiFPN layers and increasing the depth of the class/box subnets—they achieved up to 2.6% improvements in Average Precision (AP) across different models without increasing GPU latency. Their work also introduced a new public dataset (WPBB) for detecting in-water plastic debris and utilized simulation tools like Unity and Gazebo to enhance the realism of testing conditions, highlighting the potential of deep learning for accurate object detection in complex environments [4].

In recent studies, the classification and detection of cyanosis have gained attention through the use of lightweight deep learning models such as MobileNet. A modified MobileNet architecture was introduced to classify peripheral and central cyanosis using a dataset of 1300 images collected from various cyanosis-related datasets. To enhance performance, data augmentation techniques were applied to the training set. The model achieved impressive results with a validation accuracy of 95% and test accuracy of 97%, outperforming previous methods like Simple Convolutional Neural Networks (SCNNs) and fine-tuned VGG16 models, which yielded lower validation accuracies of 79% and 82%, respectively [3].

Recent advancements in image processing for road damage detection have highlighted the potential of using deep learning models like CNNs and YOLOv5. Among these, CNN models have shown superior accuracy, achieving up to 93.34% compared to YOLOv5's 73.34%. Roboflow has played a crucial role in optimizing these models by providing tools for efficient data annotation, augmentation, and training. By utilizing Roboflow, researchers can enhance model performance and streamline the road damage detection process, facilitating real-time data collection and analysis. This integration of Roboflow enables more accurate and cost-effective road maintenance solutions, ultimately improving road safety and reducing labor costs [2].

A recent study evaluated the Road Damage Dataset (RDD 2018) for its suitability in road inspection systems based on the ASTM D6433-18 standard. The research involved re-annotating the dataset to include 19 types of pavement distress and assessing its compatibility using the YOLOv8 deep learning model. The results indicated that the dataset was not ideal for this purpose due to an imbalanced distribution of instances among classes and varying image quality. The YOLOv8 model achieved a mean Average Precision (mAP) of 37.40%, highlighting the challenges posed by insufficient data in certain classes and inconsistent image quality. The study suggests that to enhance the dataset's utility, it is necessary to augment underrepresented classes, standardize image capture techniques, and use high-resolution cameras [10].

Based on previous papers, it has been shown that there is an increase in capability when using deep learning techniques. Therefore, we intend to apply deep learning in the detection and classification of Singlsyn in this study.

III. Methodology

In this research, we divided the process into seven steps: (1) Dataset construction, where all images were provided by Tamagawa Seiki, (2) Data labeling to identify the positions used to separate usable (OK) and unusable (NG) parts, (3) Data preprocessing, where the labeled data undergoes preprocessing to reduce complexity and enhance model training efficiency, (4) Data augmentation on training set to enable the model to learn from more diverse data, (5) Model selection and training with the prepared dataset to achieve the highest accuracy in object detection and classification, (6) Evaluation on the test set to measure the model's performance and accuracy in detecting the targeted objects, and (7) Deployment of the evaluated model for real-world use by

simulating it through Streamlit to create a user-friendly interface that conveniently displays object detection results. Figure 3 illustrates our methodology, summarizing all the steps mentioned.

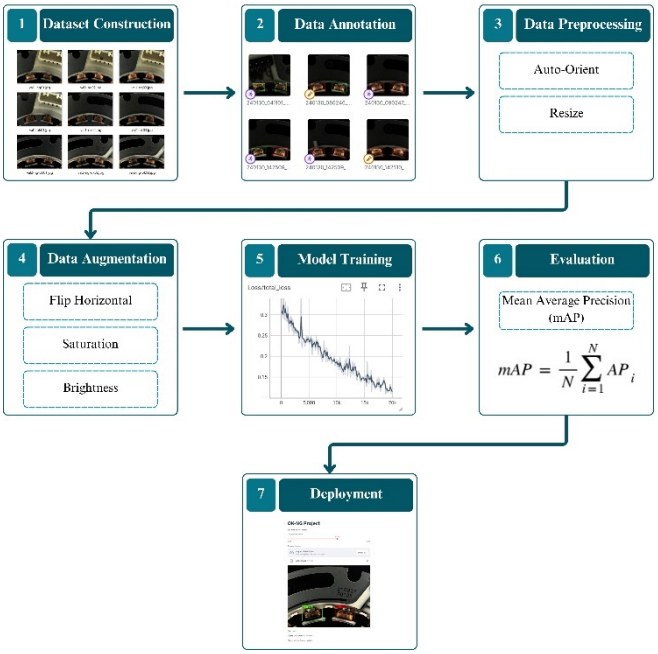

Fig. 3. Methodology

IV. Experiments

A. Dataset Construction

In this research, the company provided a dataset of 3,764 images of Singlsyn in JPG and BMP formats. Each image displays 2 objects of interest, categorized into 2 classes: 3,082 images of usable parts (OK), where both objects are usable, and 684 images of unusable parts (NG), where at least 1 of the 2 objects is unusable. Figure 4 shows an example from our dataset. We then divided the dataset into 3 parts: 80% for the training set (3,011 images), 10% for the validation set (377 images), and 10% (376 images) for the test set.

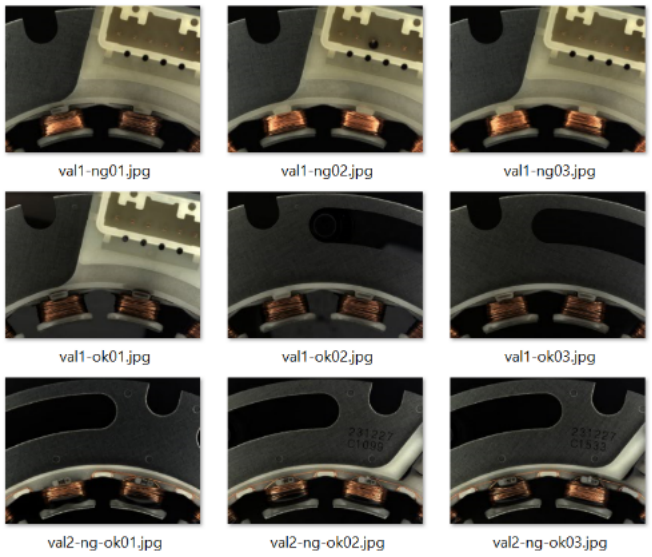

Fig. 4. Examples images of singlsyn dataset

B. Data Annotation

We annotated the data to train a model for object detection and classification, where the objects were categorized into two classes: OK (green) and NG (red). We chose Roboflow for annotating all the images because it has a user-friendly interface, allows for rapid annotation, and supports exporting annotated datasets in multiple formats such as YOLOv8, YOLOv9, Tensorflow Object Detection, Tensorflow TFrecord, and Pascal VOC, among others. All images were annotated by manually creating bounding boxes, which involved precisely defining the boundaries and positions of the objects of interest. After annotating all 3,764 images, the number of annotations totaled 7,527, with 6,534 classified as OK and 993 as NG. Figures 5 and 6 show the dataset analytics and sample annotations, respectively.

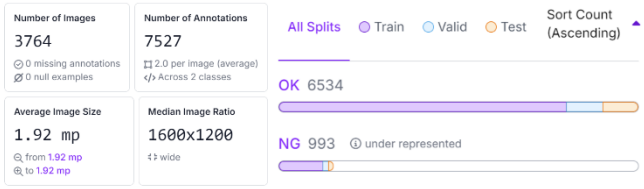

Fig. 5. Dataset analytics

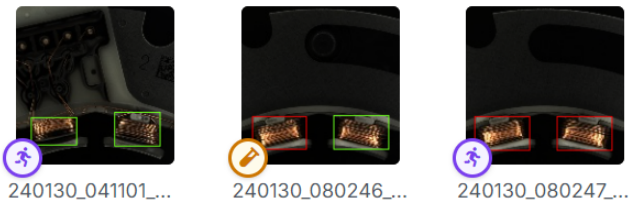

Fig. 6. Examples of annotations

C. Data Preprocessing

Before we began training the model, we performed data preprocessing on all the images in our dataset. Initial data preparation is crucial for improving data quality and making it suitable for the selected model. Preprocessing helps reduce data complexity, minimizes potential noise, and enhances the model's training efficiency. All preprocessing steps were conducted in Roboflow. The process included: (1) Auto-Orienting all images to ensure they have the correct orientation, which is vital to prevent the model from misinterpreting the data, and (2) Resizing the images using the "Stretch to 640x640" function to ensure that all image sizes are consistent with the model's requirements, thereby enabling the model to process the data efficiently and consistently.

D. Data Augmentation

We employed Data Augmentation techniques in the training set to increase data diversity, enhancing the model's efficiency and accuracy in object detection and classification. This technology plays a crucial role in helping the model learn from a wider variety of examples, increasing flexibility, and reducing the issue of overfitting. The augmentation steps we implemented include (1) Flip horizontal, (2) Random adjust saturation between -20% and +20%, and (3) Random adjust brightness between -20% and +20%, as shown in figure 7. After augmentation, the dataset was divided as follows: 9,032 images (92%) for the training set, 377 images (4%) for the validation set, and 376 images (4%) for the test set.

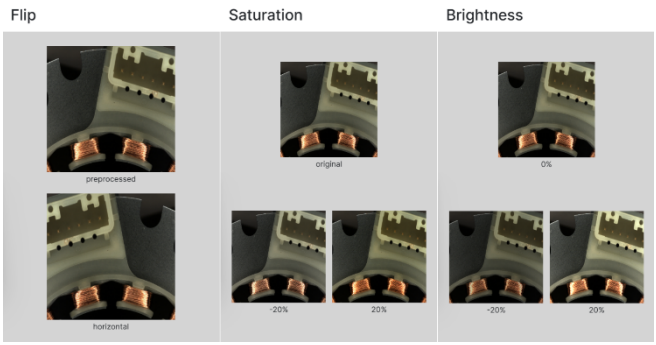

Fig. 7. Images augmentation

E. Model Training

The next crucial step is model training, which will be conducted using Python on the Google Colaboratory (Colab) platform. Google Colab provides an interactive environment for writing and executing code, making it an ideal space for training machine learning models.

1) Uploading the Dataset: Following this, the dataset is uploaded to Google Colab. Exported from Roboflow, this dataset is utilized within the Colab environment for model training.

2) Model Selection: Several models were evaluated for their performance, and among them, the SSD-MobileNet-v2-FPNLite-320 model, a Single-Shot Multibox Detection (SSD) model that uses MobileNetV2 as the feature extractor and incorporates Feature Pyramid Network Lite (FPN-Lite) to enhance the detection of objects across different scales, demonstrated the best results for object detection tasks. After selecting this model, it was downloaded to Google Colab and stored in a directory named "mymodel."

3) Setting Training Parameters: Key training parameters, specifically the number of training steps and batch size, were configured. The "**steps**" parameter represents the number of iterations the model will undergo during training, while the "**batch size**" refers to the number of images processed in each iteration. In this case, the batch size was set to 16, meaning that the model was trained on 16 images per step. The number of steps was set to 20,000, indicating that the model would undergo 20,000 iterations, each with 16 images.

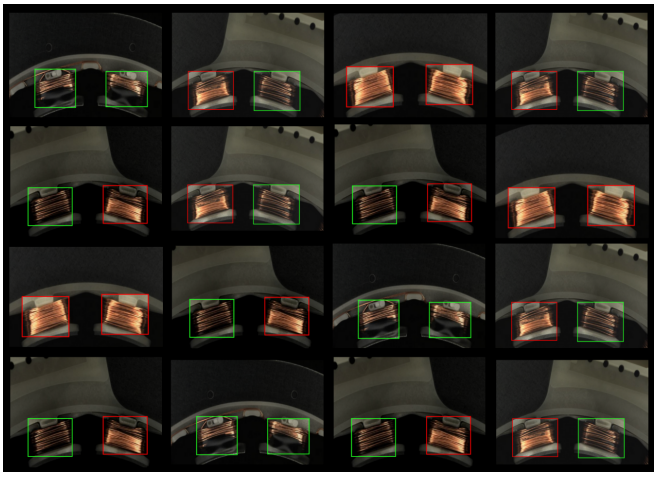

Fig. 8. Example of a batch

4) Training Loss: While training, we can monitor the progress of the model using TensorBoard as shown in figure 9, a tool provided by TensorFlow. Loss represents the ratio of incorrect detections made by the model. During the training session, TensorBoard displays real-time loss. If the loss value continues to decrease, it indicates that the model is improving.

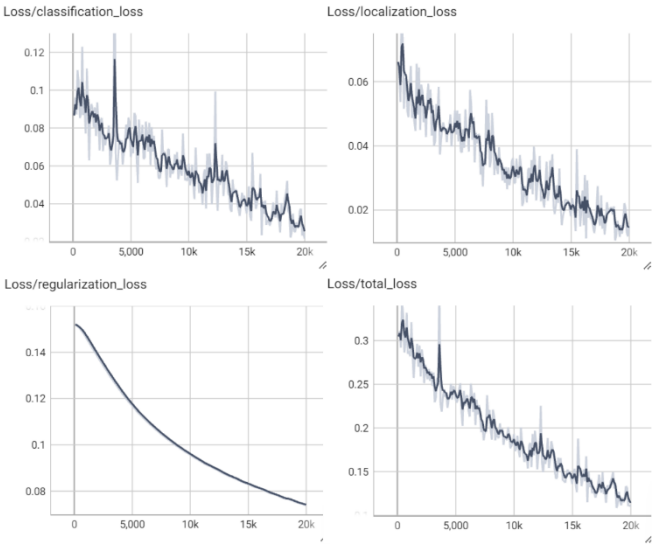

Fig. 9. Training loss

F. Challenges Encountered During Training and Solutions

1) Problem: During the training process, a significant issue arose where the model frequently misclassified NG instances as OK. This misclassification was suspected to result from an imbalance in the dataset, where NG instances were underrepresented compared to OK instances.

2) Solution: To address this issue and optimize training results, adjustments were made to the model's configuration file. Specifically, the "Alpha ($\boldsymbol{\alpha_t}$)" in Focal Loss Equation (1) The Focal Loss (FL) value indicates the model's prediction accuracy. A low FL value shows accurate predictions, especially for easy instances, while a high FL value highlights poor predictions, particularly for difficult instances.

$$Focal\ Loss(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \qquad (1)$$

Where:

- p_t is the model's estimated probability for the NG class.
- $\boldsymbol{\alpha_t}$ is the weighting factor that gives importance to the NG class.
- $\boldsymbol{\gamma}$ is the focusing parameter that reduces the relative loss for well-classified examples, putting more focus on hard, misclassified examples.

The $\boldsymbol{\alpha_t}$ parameter was modified from **0.25** to **0.75**. It controls the emphasis placed on NG instances relative to OK instances. By increasing $\boldsymbol{\alpha_t}$ to 0.75, the model's focus was shifted to prioritize the correct identification of NG instances. This adjustment aimed to mitigate the imbalance issue and enhance the model's ability to accurately differentiate between NG and OK instances. After the parameter was modified, the problem was resolved.

V. Evaluation and Results

A. Evaluation

To assess the performance of our model, we employed mean average precision (mAP) as the primary metric. mAP is widely recognized in object detection tasks for evaluating the accuracy of models by considering both precision and recall across different confidence thresholds. mAP calculation is shown in Equation (2).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{2}$$

Where:

- N is the total number of classes.
- AP_i is the average precision for the i-th class, calculated as the area under the precision-recall curve for that class.

We opted to use mAP with an Intersection over Union (IoU) threshold of 0.5, which means that an object detection is considered correct if the overlap between the predicted bounding box and the ground truth box has an IoU value of 0.5 or higher, as shown in Equation (3).

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{3}$$

B. Results

We measured the performance using mAP and utilized a test set consisting of 652 OK objects and 100 NG objects. From the entire set of experiments, it was found that the best method is the SSD-MobileNet-v2-FPNLite-320 detection model, trained for 20,000 steps with a batch size of 16 and an Alpha value set to 0.75. The accuracy of the test set achieved an mAP of 99.58%.

1) Comparison table: In this experiment, we tested a total of 8 models and found that the SSD-MobileNet-v2-FPNLite-320 detection model provided the best results as shown in TABLE I.

TABLE I. mAP Comparisons of Selected Models

No.	Model name	Mean Average Precision (mAP)
		IoU Threshold = 0.5
1	roboflow 3.0 object detection (fast)	98.6 %
2	yolo v8 detection	99.5 %
3	yolo v9-c detection	99.1 %
4	yolo v9 (gelan) detection	99.4 %
5	ssd-efficientdet-d0 -512 detection	97.72 %
6	ssd-efficientdet-d1 -640 detection	98.51 %
7	ssd-mobilenet-v2-fpnlite-640 detection	99.42 %
8	**ssd-mobilenet-v2-fpnlite-320 detection**	**99.58%**

2) Confusion Matrix:

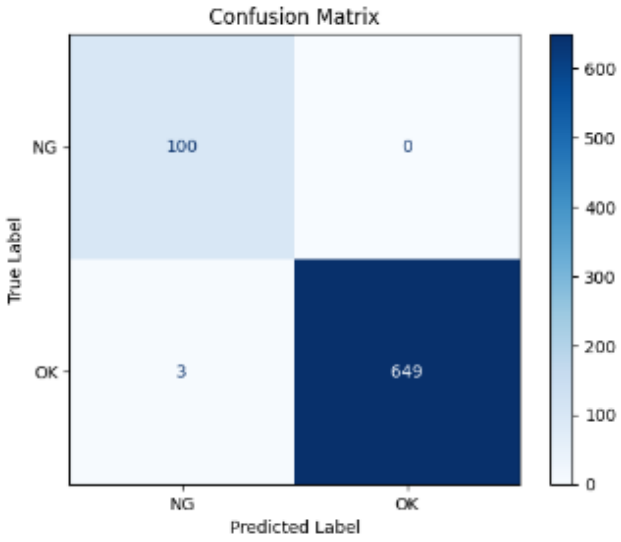

Fig. 10. Confusion matrix

As shown in figure 10:

- Out of 100 NG objects, the model correctly predicted all 100 as NG.
- Out of 652 OK objects, the model correctly identified 649 as OK and incorrectly classified 3 OK objects as NG.

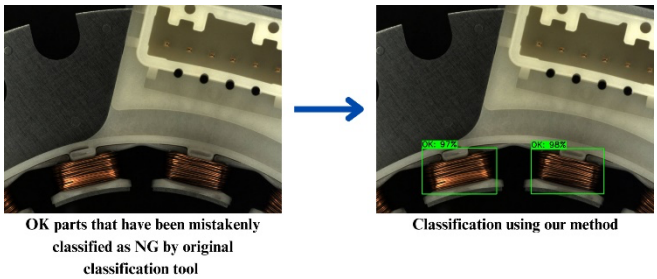

Fig. 11. Correct classification of OK parts using proposed method

As shown in Figure 11, this is an example image of OK parts that were mistakenly classified as NG by the original classification tool. Using our method, the parts were correctly classified as OK.

VI. Deployment

After evaluating the models using the mAP metric in the previous stage, we selected the model with the highest accuracy for deployment. The chosen model is the SSD-MobileNet-v2-FPNLite-320. At this stage, we simulated the deployment using Streamlit as a tool for creating the user interface (UI) for the model's operation and Visual Studio Code (VS Code) as the development environment. In this deployment model, users can upload images and receive object detection results immediately. The Streamlit interface is designed to allow users to adjust the confidence threshold as needed for greater flexibility, as shown in figure 12.

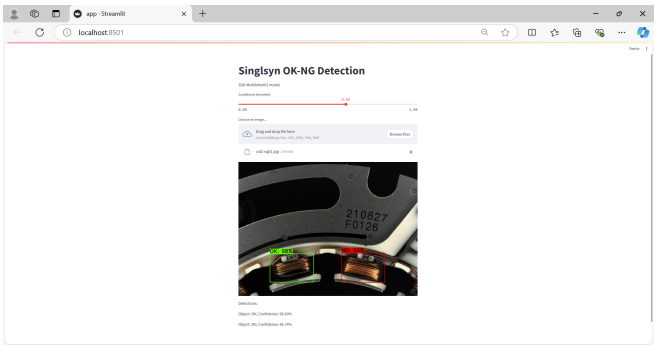

Fig. 12. Interface of Singlsyn OK-NG Detection system

VII. Conclusion

In the process of sorting usable (OK) and unusable (NG) parts, incorrect classification leads to the need for rechecking, which causes delays for employees. This paper presents the implementation of deep learning techniques to assist in detection and classification using image data provided by the company. Several models were tested, and parameters were adjusted, resulting in the SSD-MobileNet-v2-FPNLite-320 detection model. With optimized parameter settings, this model achieved the highest accuracy, improving from a previous accuracy of 90% to 99.58%, as measured by mean Average Precision (mAP), thereby significantly reducing the workload and time required by employees.

However, since there are 684 images of NG and 3,082 images of OK, it is evident that OK data outnumbers NG data by a factor of 4.5. This data imbalance presents a challenge. Future improvements could involve increasing the amount of NG data to balance the dataset, which would enable the model to make more accurate and equitable predictions.

Acknowledgment

A part of this research was supported by Iida City, Nagano Prefecture, Japan. And was supported by Tamagawa Seiki Co.,Ltd., Nagano, Japan.

References

[1] S. Oh, J. Cha, D. Kim, and J. Jeong, “Quality Inspection of Casting Product Using CAE and CNN,” in International Conference on Imaging, Signal Processing and Communications, 2020, pp. 34-48.

[2] D. Deepa, A. Sivasangari, R. Roonwal, and R. Nayan, "Pothole Detection using Roboflow Convolutional Neural Networks," in International Conference on Intelligent Computing and Control Systems, 2023, pp. 560-564.

[3] L. Mpova, T. Shongwe, and A. Hasan, "Classification and Detection of Cyanosis Images on Lightly and Darkly Pigmented Individual Human Skins Using a Fine-tuned MobileNet Architecture," in International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems, 2023, DOI: 10.1109/ICABCD59051.2023.10220464

[4] F. Zocco, T.-C. Lin, C.-I. Huang, H.-C. Wang, M. O. Khyam, and M. Va, "Towards More Efficient EfficientDets and Real-Time Marine Debris Detection," in IEEE Robot. Autom. Lett., vol. 8, no. 4, 2023, pp. 2134-2141.

[5] Tensorflow, “TensorFlow 2 Detection Model Zoo,” 2021. [Online]. Available: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

[6] Ultralytics, “Ultralytics YOLOv8,” 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[7] K.-Y. Wong, "YOLOv9," GitHub, 2022. [Online]. Available: https://github.com/WongKinYiu/yolov9

[8] E. Juras, “TensorFlow Lite Object Detection on Android and Raspberry Pi,” 2024. [Online]. Available: https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi

[9] Roboflow, "Roboflow: Data preparation and training platform," 2024. [Online]. Available: https://roboflow.com

[10] A. Mulyanto, R. Sari, and A. Muis, “Road Damage Dataset Evaluation Using YOLOv8 for Road Inspection System,” in International Conference on Computer and Automation Engineering, 2024, pp. 403-407.