Efficient packaging defect detection: leveraging pre-trained vision models through transfer learning

Wiwi Prastiwinarti[1], Mera Kartika Delimayanti[2], Hendra Kurniawan[3,4], Yoga Putra Pratama[1], Hanin Wendho[1], Rizky Adi[2]
[1]Department of Printing Engineering and Publishing, Politeknik Negeri Jakarta, Jakarta, Indonesia
[2]Computer and Informatics Engineering, Politeknik Negeri Jakarta, Jakarta, Indonesia
[3]Informatics Engineering, Universitas Maritim Raja Ali Haji, Tanjungpinang, Indonesia
[4]Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa, Japan

Article Info

Article history:

Received Oct 31, 2023
Revised Feb 18, 2024
Accepted Mar 16, 2024

Keywords:

Defect detection
EfficientNet
MobileNet
ResNet
Transfer learning

ABSTRACT

The inspection of packaging defects is a crucial aspect of maintaining the quality of industrial production, especially in the case of boxed products. This study introduces a novel approach for detecting physical defects in product packaging boxes by integrating image processing with deep learning, specifically transfer learning with two images as an input. The proposed method utilizes both top view and side view images of the packaging to determine its condition, a significant departure from the conventional single image input. Our approach incorporates 16 pre-trained model variants from EfficientNetV2, MobileNetV3, and ResNetV2 for transfer learning as feature extractors. The experimental findings demonstrate that the best model that leverages EfficientNetV2 variant achieves 100% accuracy and F1 score in terms of classification performance. However, the most optimal model in terms of classification performance and inference speed was the one that leveraged ResNetV2 variant. This model scored 95% accuracy and 95.24% F1 score, with an inference speed of 91 ms per prediction.

Corresponding Author:

Wiwi Prastiwinarti
Department of Printing Engineering and Publishing, Politeknik Negeri Jakarta
Jakarta, Indonesia
Email: wiwi.prastiwinarti@grafika.pnj.ac.id

1. INTRODUCTION

The inspection of package defects is a crucial aspect of maintaining the quality of industrial production, especially in the case of boxed products. Traditionally, this process has been carried out manually, a method that is not only inefficient but also prone to errors. With the advancement of technology, particularly in the field of computer vision, it has become possible to develop a system where machines can automatically detect defects on product packages, thereby replacing manual labor [1]. Manual inspection, on the other hand, presents several drawbacks such as slow speed, low detection efficiency, extensive man-hours, and substantial consumption of material and site resources [2]. The repetitive nature of the task can lead to visual fatigue, impairing perception over time. This is due to the fact that different individuals, and even the same individual under varying conditions, tend to inspect the package differently, which makes it challenging to establish a unified standard of judgment [1]. Computer vision relies heavily on computational systems to emulate the human visual function, enabling extraction, processing, and analyzing information from tangible objects. A higher level of objectivity characterizes the outcomes, as they can promptly and precisely identify defects in the box packaging. Additionally, these outcomes provide a complete analysis of the defect parameters, enabling the determination of whether the packaging material meets the required standards or is defective [3].

In recent years, both the software and hardware technologies of computer vision systems have been significantly improved with the growth of electronic technology. The subfield of machine learning, known as "deep learning", focuses on creating and applying methods that let neural networks learn from and make predictions about big, complex datasets. Multiple layers of nodes, also called neurons, are interconnected in deep learning [4]. In computer vision, deep learning has shown exceptional effectiveness in recent years. In contemporary times, deep learning has demonstrated notable efficacy in computer vision. The insensitivity of pattern identification in images has been proven concerning elements such as backdrop, lighting, colour, form, size, and intensity [5]–[8]. The design and operation of these networks are influenced by the physical and functional properties of neural networks in the human brain. The input data undergoes processing within these neural networks to autonomously extract and acquire knowledge of features-these traits' abstraction level increases as they traverse the network's tiers. Deep learning can be attributed to their capacity to effectively grasp intricate patterns and representations from unprocessed data [3]. This is especially desirable when it comes to detecting intricate surface faults in commercial or industrial environments. In addition, flaws must not only be identified, but also their precise dimensions and types ascertained before moving on to the next step of the process [9].

Defect detection techniques powered by deep learning give the network the flexibility to find specific flaws based on the data set. In addition, the parameters of the network that are learned while working with one network can be applied to other networks that are comparable in order to produce high success rates for surface defect detection. Consequently, a number of studies have utilized a computer vision strategy to support in classification, defect detection, and quality inspection for fabric defect [3], [10], cylinder and glass bottle [4], [11], [12] and box packaging [1], [8]. For instance, Jeyaraj and Nadar [3] had proposed developing a quick and efficient classification system for fabric defects. The authors have employed a learning feature for defect classification and have achieved a satisfactory classification accuracy by employing a deep learning algorithm [3]. Moreover, Liu *et al.* [10] presented an approach for fabric defect detection based on generative adversarial networks (GAN). It has been demonstrated that this method assisted in fine-tuning the semantic segmentation network to detect defects in varying conditions more accurately.

In another study, surface defect detection was conducted on a cylinder liner dataset. Due to the irregular shape, variety, and small size of the surface defects, cylinder liner defect detection based on machine vision is a challenging task. Gao *et al.* [4] used experiments to determine which model was the most effective. The dataset was then augmented using a modified augmentation technique incorporating the region of interest's automatic extraction technique with conventional augmentation techniques. The results indicate that the detection accuracies were superior to those of conventional methods [4], [11]. In addition, a Wavelet Transform Multiscale Filtering algorithm has been offered for the experiment of defect detection on glass bottles to reduce the influence of texture and increase the robustness to localization error [13]. The experimental findings demonstrate that the framework achieves the highest performance compared to various conventional approaches [12].

Furthermore, research has been conducted on the identification of defects in packaging boxes. Yang *et al.* [8] introduced support vector machine (SVM) with a radial basis function kernel as a model used for detecting defects in logistics packaging boxes and has successfully detected three types of defects. It satisfies manufacturers requirements regarding defect classification and recognition in machine vision detection systems. Therefore, this method is an effective solution for detecting defects in logistics packaging boxes [1], [8], [14]. Despite its effectiveness in its own scenario, the approach from the previous study has significant limitations. In response to these limitations, our study proposes a solution that utilizes both top view and side view images of the packaging as inputs to determine if the package is defective or not. However, the dataset we have is a small dataset [15]. To address this issue, we employed deep learning with transfer learning. This technique leverages a pre-trained model to transfer knowledge from one domain to another. It is faster and less time-consuming than training a model from random initialization [16], [17]. Moreover, it is particularly suitable for training with a limited dataset because pre-trained models have already been trained on large datasets, thus retaining knowledge from these extensive datasets [16]–[18].

2. METHOD

2.1. Dataset

In this study, we used an Industrial quality control of packages kaggle dataset that was open to the public [15]. This dataset consists of computer-generated images designed to resemble packages produced on an industrial production line. This dataset is composed of 400 RGB images from a ‘virtual’ production line. Each package consists of two images, one top-view image and one side-view image. This means there are only 200 package examples in this dataset. These packages are divided into two classes, ‘intact’ and 'damaged', indicating the condition of the package box. Specifically, 100 packages are intact and the other 100 are damaged. The images contained in the dataset are presented in PNG format with a dimension of 960×540 pixels. The dataset samples are illustrated in Figures 1 and 2.

Figure 1(a) and Figure 1(b) display a pair of images depicting a package in normal condition, without any defects. The package box appears to be in perfect shape, with no visible dents. Conversely, Figure 2 presents a pair of images showing a defective package. Figure 2(a) and Figure 2(b) showcase a package with a noticeable defect, which is a dent on the box. In the side-view image, the dent is located on the top of the box and is highlighted with a red line. Similarly, in the top-view image, the dent is on the side of the box and is also marked with a red line.

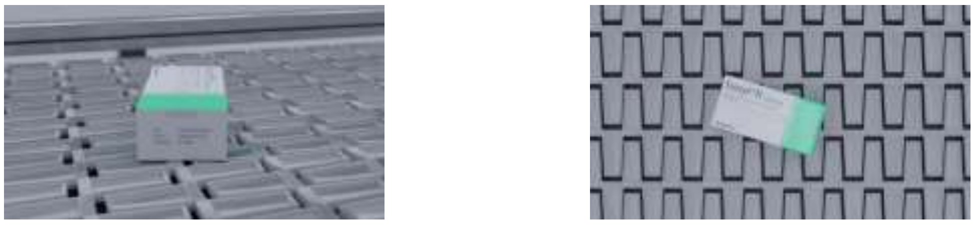
(a) (b)

Figure 1. Normal package sample (a) side-view and (b) top-view

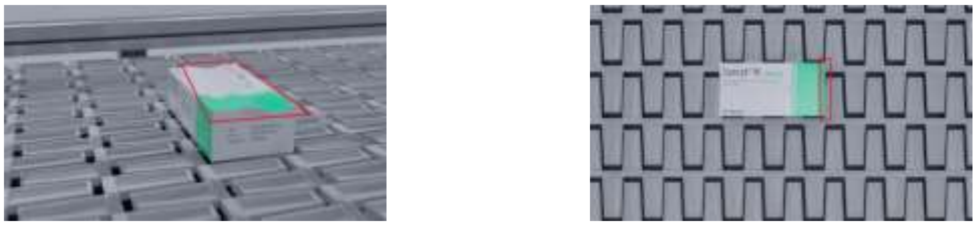
(a) (b)

Figure 2. Defect package sample (a) side-view and (b) top-view

To accommodate the training and evaluation purposes, we split the dataset into three parts, which are training, validation, and test sets. The training and validation sets are utilized during the training phase. Specifically, the training data is used to train the model, while the validation set is used for hyperparameters tuning. The test set, on the other hand, is used to evaluate the performance of the trained model. We split the dataset with a ratio of 80:10:10 randomly. We partitioned the dataset using an 80:10:10 ratio due to the small size of our dataset. By using an 80:10:10 split, we can maximize the data available for training. The detailed distribution of the dataset is depicted in Table 1, which presents the total amount of data for each split set along with its corresponding package class.

Table 1. Dataset distribution

	Intact package images		Damaged package images		
Split Set	Side-view	Top-view	Side-view	Top-view	Total images
Training	80	80	80	80	320
Validation	10	10	10	10	40
Test	10	10	10	10	40

2.2. Preprocessing

The preprocessing stage is the stage where the raw images from dataset are transformed into a format that can be used for training models effectively [19]. In our study, we opted for a simple preprocessing approach, which involves reading the raw images as pixel numbers, resizing them, and then normalizing. An example of this can be seen in Figure 3, where the images may appear squished after resizing. Despite this, we chose to retain the images in this state, opting for resizing over cropping. This decision was influenced by the variable positioning of the package box within each image. It is not always centrally located, and cropping could potentially result in the package box being omitted.

In our study, we also explored two distinct image color spaces to understand their effect on the detection of defects in package box shapes. The first dataset maintained the original RGB color scheme of the images, shown in Figure 3, while the second was transformed into grayscale, illustrated in Figure 4. This approach was designed to assess the influence of color information on defect detection. By converting to grayscale, we could concentrate exclusively on the shape characteristics of the package boxes, thereby eliminating any potential interference from color variations. This experiment aimed to determine whether grayscale images would be sufficient for accurate defect detection, potentially simplifying the model input and reducing computational requirements. However, with resizing, we can ensure the package box's consistent presence in the image. As for the final size of the resizing, it follows the default input size of each pre-trained model stated in Table 2.

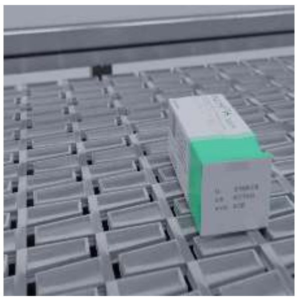
Figure 3. Resized RGB image

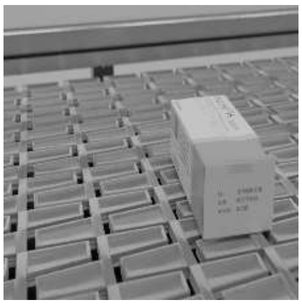
Figure 4. Resized grayscale image

2.3. Transfer learning

Technique knows as transfer learning was utilized in our research. This method leverages a pre-trained model to transfer knowledge from one domain to another, providing a more efficient and less time-consuming alternative to training a model from random initialization [16], [17]. By leveraging the knowledge from the pre-trained model, transfer learning proved to be beneficial when working with small dataset like our dataset [16]–[18]. In our implementation, we employed the pre-trained model as the feature extraction layer of our models and only train the rest of the network [20], [21], given its prior training on large datasets, making it an suitable approach for our small datasets. Consequently, transfer learning enables us to harness the capabilities of pre-existing models, thereby reducing computational time and resources, while still ensuring robust performance. The pre-trained model utilized used for transfer learning in our study are EfficientNetV2 [22], MobileNetV3 [23], and ResNetV2 [24] variants which trained using ImageNet dataset [25]. These models were selected due to their classification as lightweight to medium models [26], which are ideal for a possible real-world scenarios such as fast-paced production line. The advantage of using a lightweight model is the minimal inference time, which is crucial for accommodating the possibility of rapid pace of a production line. Detailed information about each pre-trained model used in this research is provided in Table 2.

Table 2. Pre-trained models used in this study

Pre-trained model	Variant	Default input image size	Parameters without top layer (million)
EfficientNetV2 [22]	EfficientNetV2-21k-S	384×384	20.33
	EfficientNetV2-21k-M	480×480	53.15
	EfficientNetV2-21k-L	480×480	117.75
	EfficientNetV2-1k-S	384×384	20.33
	EfficientNetV2-1k-M	480×480	53.15
	EfficientNetV2-1k-L	480×480	117.75
	EfficientNetV2-21k-ft1k-S	384×384	20.33
	EfficientNetV2-21k-ft1k-M	480×480	53.15
	EfficientNetV2-21k-ft1k-L	480×480	117.75
MobileNetV3 [23]	MobileNetV3-dm1.00-S	224×224	1.53
	MobileNetV3-dm0.75-S	224×224	1.03
	MobileNetV3-dm1.00-L	224×224	4.23
	MobileNetV3-dm0.75-L	224×224	2.73
ResNetV2 [24]	ResNetV2-50	224×224	23.56
	ResNetV2-101	224×224	42.63
	ResNetV2-152	224×224	58.33

Table 2 presents detailed information about each pre-trained model, its variant, default input image size, and parameters without the top layer. Each model listed in the table has multiple variants, but we only utilized the ones specified. In total we trained and evaluated 16 model variants in our experiments. Consisting of three distinct mode architectures, EfficientNetV2 [22], MobileNetV3 [23], and ResNetV2 [24].

Our study incorporated nine variants of EfficientNetV2. The suffixes S, M, and L in a variant name represent the architecture type: ‘Small’, ‘Medium’, or ‘Large’, respectively. We also employed four variants of MobileNetV3, with the suffixes S and L indicating ‘Small’ and ‘Large’ architectures, respectively. The labels dm1.00 and dm0.75 signify depth multipliers of 1.00 and 0.75 [23]. Furthermore, we utilized three variants of ResNetV2, with the numbers 50, 101, and 152 at the end of the variant names representing the number of layers in the model [24], [27]. The default input image size for both MobileNetV3 and ResNetV2 is 224×224. However, EfficientNetV2 differs, with a size of 384×384 for the ‘Small’ architecture and 480×480 for the ‘Medium’ and ‘Large’ architectures [22]–[24], [27]. The parameters listed in Table 2 exclude the top layer, as we used the pre-trained models solely for feature extraction. By looking on Table 2 it reveals that MobileNetV2 has the fewest average parameters, followed by ResNetV2, while EfficientNetV2 boasts the highest average parameters.

2.4. Model architecture

In our experiments, we employed a custom-made model architecture, diverging from the conventional approach of using purely pre-trained models. This decision was chosen because of the unique structure of our dataset, which includes two images as input, while pre-trained models typically support only a single image input. Even though we used a custom-made model architecture; we still utilize pre-trained models as the image feature extractor. The illustration of our custom-made model can be found in Figure 5.

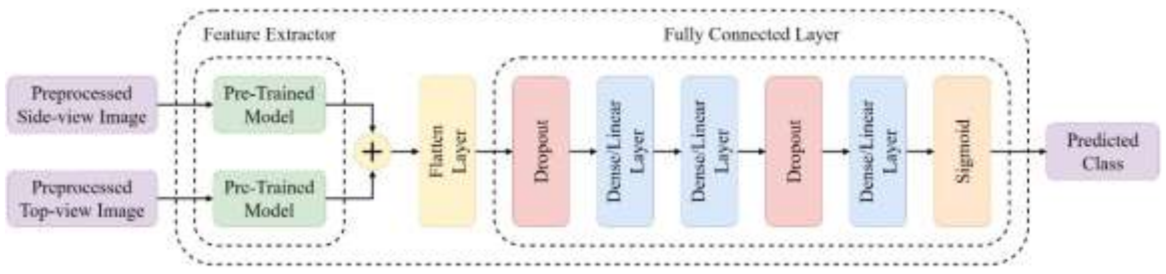

Figure 5. Custom made model architecture

The custom model architecture depicted in Figure 5 is designed to accept two images as input and yield a predicted class output, indicating whether the package is defective or normal. The process begins with the model receiving two image inputs, each of which is fed into separate instances of pre-trained models. These pre-trained models serve as image input feature extractors. The features extracted from each pre-trained model are then concatenated and flattened. These flattened features are subsequently fed into a network of fully connected layers, comprising multiple dropouts and dense/linear layers, finished by sigmoid activation function. The sigmoid function was chosen due to its ability to map the input to a value between 0 and 1 [28], making it suitable for our classification task. The output of the sigmoid function represents the final output of our model, which is the predicted image class. The hyperparameters setting used during the training phase of this model architecture are detailed in Table 3.

Table 3. Hyperparameters setting

Hyperparameter	Value
Optimizer	Adam
Loss function	BinaryCrossEntropy
Learning rate	1e-3, 1e-4, 1e-5
Batch size	8, 16, 32
Dropout	0.2
Maximum epoch	200
Early stopping monitor	Validation Accuracy
Early stopping mode	Max
Early stopping patience	10
Early stopping min delta	0.00

As detailed in Table 3, we experimented with three distinct values for both the learning rate and batch size. For the learning rate, we conducted three separate experiments for each model variant, utilizing all the presented learning rates to identify the optimal model. In contrast, for the batch size, we selected a single value for each model variant. The chosen batch size was the largest that could be accommodated within the GPU VRAM capacity. Table 3 also outlines the early stopping settings implemented in our experiments. This feature was utilized to conserve computing power in the absence of any improvement. With early stopping enabled, the training process is automatically stopped if no improvement is observed based on the provided settings.

2.5. Evaluation

The performance of our image-based classification model was evaluated by using recall, precision, F1 score, accuracy, and inference speed metrics. Recall measures the proportion of actual positives that are correctly identified. Precision quantifies the number of true positive instances among the predicted positives. The F1 score is the harmonic mean of precision and recall, providing a balance between these two metrics [29]. Accuracy, on the other hand, gives us a holistic view of the overall correctness of the model.

The terms TP, TN, FP, and FN used in (1), (2), and (4) stands for True Positive, True Negative, False Positive, and False Negative. These terms describe the condition of between predicted and actual label. While the term t_i used in (5) is the time the model takes for each inference process. The definition of TP, TN, FR, and FN are shown in Table 4.

Table 4. TP, TN, FP, FN definition

Predicted	Actual	Definition
Defect	Defect	True positive (TP)
Defect	Non-defect	False positive (FP)
Non-defect	Defect	False negative (FN)
Non-defect	Non-defect	True negative (TN)

3. RESULTS AND DISCUSSION

The classification performance of the proposed approach has been evaluated using the test dataset. The findings were presented as tables. The utilization of transfer learning for two image input models shows a promising result presented in Tables 5-7. We conducted the experiments multiple times with different learning rates and image color space. The results of best model variants we got using the RGB image dataset detailed in Table 5.

Table 5. Test results (RGB image dataset)

Pre-trained model variant	Optimal learning rate	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)	Inference Speed (ms)
EfficientNetV2-21k-S	1e-3	**100**	**100**	**100**	**100**	131
EfficientNetV2-21k-M	1e-3	80	77.78	87.5	70	177
EfficientNetV2-21k-L	1e-5	85	84.21	88.89	80	247
EfficientNetV2-1k-S	1e-3	80	81.82	75	90	131
EfficientNetV2-1k-M	1e-3	90	88.89	**100**	80	182
EfficientNetV2-1k-L	1e-5	80	80	80	80	250
EfficientNetV2-21k-ft1k-S	1e-5	85	85.71	81.82	90	130
EfficientNetV2-21k-ft1k-M	1e-3	85	82.35	**100**	70	182
EfficientNetV2-21k-ft1k-L	1e-4	80	81.82	75	90	260
MobileNetV3-dm1.00-S	1e-3	65	69.57	61.54	80	81
MobileNetV3-dm0.75-S	1e-3	75	73.68	77.78	70	**80**
MobileNetV3-dm1.00-L	1e-4	80	77.78	87.5	70	86
MobileNetV3-dm0.75-L	1e-3	80	77.78	87.5	70	84
ResNetV2-50	1e-3	95	95.24	90.91	**100**	91
ResNetV2-101	1e-3	70	70	70	70	138
ResNetV2-152	1e-3	80	77.78	87.5	70	131

As presented in Table 5, the best model for RGB image input was the EfficientNetV2 on the 21k-S variant, which achieved a perfect accuracy and F1 score of 100%. However, when considering inference speed, the MobileNetV3-dm0.75-S model outperformed others, averaging 80 ms per prediction, or 12.5 predictions per second. Despite its speed, the performance of MobileNetV3-dm0.75-S was inferior compared to other models. Therefore, the optimal balance between performance and inference speed was found in the ResNetV2-50 model, which demonstrated an accuracy of 95% and an F1 score of 95.24%, while maintaining a reasonable average prediction speed of 91 ms. The information presented in Table 5 is the performance of the model from the RGB image, as for the grayscale image input is presented in Table 6.

Table 6. Test results (grayscale image dataset)

Pre-trained model variant	Optimal learning rate	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)	Inference speed (ms)
EfficientNetV2-21k-S	1e-4	85	82.35	**100**	70	128
EfficientNetV2-21k-M	1e-3	90	90	90	**90**	183
EfficientNetV2-21k-L	1e-5	80	81.82	75	**90**	252
EfficientNetV2-1k-S	1e-3	90	88.89	**100**	80	130
EfficientNetV2-1k-M	1e-4	65	58.82	71.43	50	181
EfficientNetV2-1k-L	1e-4	75	73.68	77.78	70	246
EfficientNetV2-21k-ft1k-S	1e-3	**95**	**94.74**	**100**	**90**	132
EfficientNetV2-21k-ft1k-M	1e-3	90	90	90	**90**	184
EfficientNetV2-21k-ft1k-L	1e-4	90	88.89	**100**	80	250
MobileNetV3-dm1.00-S	1e-3	80	75	**100**	60	**81**
MobileNetV3-dm0.75-S	1e-3	60	69.23	56.25	**90**	**81**
MobileNetV3-dm1.00-L	1e-3	85	82.35	**100**	70	84
MobileNetV3-dm0.75-L	1e-3	75	76.19	72.73	80	84
ResNetV2-50	1e-4	80	81.82	75	**90**	92
ResNetV2-101	1e-5	75	78.26	69.23	**90**	144
ResNetV2-152	1e-3	80	80	80	80	128

Table 6 presents the best model variants resulting from grayscale image input, and contrary to the previous table, none of these models achieved a perfect 100% performance score. The top performing model is now EfficientNetV2-21k-ft1k-S, which scored 95% in accuracy and 94.74% in F1 score. In terms of inference speed, EfficientNetV2 still lags other models, with MobileNetV3 being the fastest on dm1.00-S and dm0.75-S variants, clocking in at 81 ms per prediction. However, speed does not necessarily equate to superiority. The model with the best performance in terms of inference speed is EfficientNetV2-21k-ft1k-S, the best model for grayscale image input, with an inference speed of 132 ms per prediction. To better understand how the color space of image input affects model performance, Table 7 is presented with an information of performance difference between models using RGB image input and grayscale image input in terms of accuracy and F1 score.

Table 7. RGB and grayscale image input performance differences

Pre-trained model variant	RGB image input		Grayscale image input		Performance difference	
	Accuracy (%)	F1 Score (%)	Accuracy (%)	F1 Score (%)	Accuracy (%)	F1 Score (%)
EfficientNetV2-21k-S	100	100	85	82.35	-15	-17.65
EfficientNetV2-21k-M	80	77.78	90	90	12.5	15.71
EfficientNetV2-21k-L	85	84.21	80	81.82	-5.88	-2.84
EfficientNetV2-1k-S	80	81.82	90	88.89	12.5	8.64
EfficientNetV2-1k-M	90	88.89	65	58.82	**-27.78**	**-33.83**
EfficientNetV2-1k-L	80	80	75	73.68	-6.25	-7.9
EfficientNetV2-21k-ft1k-S	85	85.71	95	94.74	11.76	10.54
EfficientNetV2-21k-ft1k-M	85	82.35	90	90	5.88	9.29
EfficientNetV2-21k-ft1k-L	80	81.82	90	88.89	12.5	8.64
MobileNetV3-dm1.00-S	65	69.57	80	75	**23.08**	**7.81**
MobileNetV3-dm0.75-S	75	73.68	60	69.23	-20	-6.04
MobileNetV3-dm1.00-L	80	77.78	85	82.35	6.25	5.88
MobileNetV3-dm0.75-L	80	77.78	75	76.19	-6.25	-2.04
ResNetV2-50	95	95.24	80	81.82	-15.79	-14.09
ResNetV2-101	70	70	75	78.26	7.14	11.8
ResNetV2-152	80	77.78	80	80	0	2.85

As depicted in Table 7, not all model variants benefit from using grayscale images instead of RGB. Out of 16 model variants, only 9 show a positive impact when using grayscale images as model input. The most significant improvement is seen in MobileNetV3-dm1.00-S, with an increase of 23.08% in accuracy and 7.81% in F1 score. Conversely, the worst impact is observed in EfficientNetV2-1k-M, with a performance deterioration of -27.78% in accuracy and -33.83% in F1 score. This suggests that using grayscale input for our custom-made model architecture does not guarantee an improvement in model performance. It is evident that each model variant has its best version, whether using RGB image input or grayscale image input. The combination of the best model test results for each model variant is detailed in Table 8.

Table 8. Best test results

Pre-trained model variant	Image input color space	Accuracy (%)	F1 Score (%)	Precision (%)	Recall (%)	Inference speed (ms)
EfficientNetV2-21k-S	RGB	**100**	**100**	**100**	**100**	131
EfficientNetV2-21k-M	Grayscale	90	90	90	90	183
EfficientNetV2-21k-L	RGB	85	84.21	88.89	80	247
EfficientNetV2-1k-S	Grayscale	90	88.89	**100**	80	130
EfficientNetV2-1k-M	RGB	90	88.89	**100**	80	182
EfficientNetV2-1k-L	RGB	80	80	80	80	250
EfficientNetV2-21k-ft1k-S	Grayscale	95	94.74	**100**	90	132
EfficientNetV2-21k-ft1k-M	Grayscale	90	90	90	90	184
EfficientNetV2-21k-ft1k-L	Grayscale	90	88.89	**100**	80	250
MobileNetV3-dm1.00-S	Grayscale	80	75	**100**	60	81
MobileNetV3-dm0.75-S	RGB	75	73.68	77.78	70	**80**
MobileNetV3-dm1.00-L	Grayscale	85	82.35	**100**	70	84
MobileNetV3-dm0.75-L	RGB	80	77.78	87.5	70	84
ResNetV2-50	RGB	95	95.24	90.91	**100**	91
ResNetV2-101	Grayscale	75	78.26	69.23	90	144
ResNetV2-152	Grayscale	80	80	80	80	111

Presented in Table 8 that models produced by EfficientNetV2 architecture are the top amongst two other model architecture in terms of average performance, but it suffers from the slow inference speed compared to two other models' architecture. On the other hand, MobileNetV3 architecture produces the fastest models, but has subpar performance on average. The most optimal pre-trained model architecture in our experiments was ResNetV2, it has a promising performance average, while maintaining a notably fast inference speed, albeit not as fast as MobileNetV3. Meaning that each pre-trained architecture is suitable for different use cases. If classification performance is the most important, then EfficientNetV2 is the correct choice. If inference speed is the only one that matters, then MobileNetV3 is the perfect choice. Lastly, if classification performance and inference speed are equally important then ResNetV2 is the optimal choice.

Table 8 shows the best model we got was a model with the pre-trained EfficientNetV2 as the feature extractor on the 21k-S variants with a perfect score of 100% accuracy and F1 score, which use RGB image as the input. Despite the perfect performance score, it ranked second if we include the inference speed in the calculation. The model that demonstrated the best balance between classification performance and inference speed is model with a pre-trained ResNetV2-50 variant as the feature extractor with a score of 95% accuracy and 95.24% F1 score, and 91 ms inference speed, which also use RGB image as the input. Although the results are promising, please keep in mind that this performance was gained using a small computer-generated dataset, which may differ from a big real-world dataset. However, if the real-world image datasets are similar with the dataset we used on our experiments, it may have comparable results with our experiments.

4. CONCLUSION

This study introduces a novel approach for detecting physical defects in product packaging boxes by integrating image processing with deep learning, specifically transfer learning with two images as an input. The proposed method utilizes both top and side view images of the packaging to determine its condition, a significant departure from the conventional single-image input. Our approach incorporates 16 pre-trained model variants from EfficientNetV2, MobileNetV3, and ResNetV2 for transfer learning as feature extractors. We also experimented with multiple image input color spaces, namely RGB and grayscale. Our experiments yielded promising results. The best model, which leverages EfficientNetV2-21k-S as a feature extractor, achieved a perfect 100% accuracy and F1 score in terms of classification performance. However, the most optimal model in terms of classification performance and inference speed was the one that leveraged ResNetV2-50 as a feature extractor. This model scored 95% accuracy and 95.24% F1 score, with an inference speed of 91 ms. Both models used RGB images as input. Interestingly, we found that using grayscale images as input does not necessarily improve model performance. For future research, it may be worthwhile to conduct experiments with real-world datasets instead of computer-generated ones. Exploring real-time model approaches such as YOLO could also be beneficial. While our findings show promising results, they were obtained using a small computer-generated dataset. Therefore, the performance may differ when using real-world datasets. However, if the real-world dataset is similar to the one used in our experiments, our approach and models could implement effectively to identify defects in package boxes with good results.

ACKNOWLEDGEMENTS

We would like to express our highest gratitude to Politeknik Negeri Jakarta for supporting this research through Associate Professor Accelerated Research Grant Schema 2023.

REFERENCES

[1] Y. Wu and Y. Lu, "An intelligent machine vision system for detecting surface defects on packing boxes based on support vector machine," *Measurement and Control*, vol. 52, no. 7–8, pp. 1102–1110, Sep. 2019, doi: 10.1177/0020294019858175.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[3] P. R. Jeyaraj and E. R. Samuel Nadar, "Computer vision for automatic detection and classification of fabric defect employing deep learning algorithm," *International Journal of Clothing Science and Technology*, vol. 31, no. 4, pp. 510–521, Aug. 2019, doi: 10.1108/IJCST-11-2018-0135.

[4] C. Gao, F. Hao, J. Song, R. Chen, F. Wang, and B. Liu, "Cylinder Liner Defect Detection and Classification based on Deep Learning," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 8, 2022, doi: 10.14569/IJACSA.2022.0130818.

[5] M. Ifeanyi Akazue, R. Elizabeth Yoro, B. Ogheneovo Malasowe, O. Nwankwo, and A. Arnold Ojugo, "Improved services traceability and management of a food value chain using block-chain network: a case of Nigeria," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 3, pp. 1623–1633, Mar. 2023, doi: 10.11591/ijeecs.v29.i3.pp1623-1633.

[6] T.-T.-H. Vu, D.-L. Pham, and T.-W. Chang, "A YOLO-based Real-time Packaging Defect Detection System," *Procedia Computer Science*, vol. 217, pp. 886–894, 2023, doi: 10.1016/j.procs.2022.12.285.

[7] A. Abdullahi, N. Azah Samsudin, M. Rasidi Ibrahim, M. Syariff Aripin, S. K. Ahmad Khalid, and Z. Ali Othman, "Towards IR4.0 implementation in e-manufacturing: artificial intelligence application in steel plate fault detection," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 20, no. 1, pp. 430–436, Oct. 2020, doi: 10.11591/ijeecs.v20.i1.pp430-436.

[8] X. Yang, M. Han, H. Tang, Q. Li, and X. Luo, "Detecting Defects With Support Vector Machine in Logistics Packaging Boxes for Edge Computing," *IEEE Access*, vol. 8, pp. 64002–64010, 2020, doi: 10.1109/ACCESS.2020.2984539.

[9] Y. Chen, Y. Ding, F. Zhao, E. Zhang, Z. Wu, and L. Shao, "Surface Defect Detection Methods for Industrial Products: A Review," *Applied Sciences*, vol. 11, no. 16, p. 7657, Aug. 2021, doi: 10.3390/app11167657.

[10] J. Liu, C. Wang, H. Su, B. Du, and D. Tao, "Multistage GAN for Fabric Defect Detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 3388–3400, 2020, doi: 10.1109/TIP.2019.2959741.

[11] Y. Hammoudi, I. Idrissi, M. Boukabous, Y. Zerguit, and H. Bouali, "Review on maintenance of photovoltaic systems based on deep learning and internet of things," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 26, no. 2, pp. 1060–1072, May 2022, doi: 10.11591/ijeecs.v26.i2.pp1060-1072.

[12] X. Zhou *et al.*, "A Surface Defect Detection Framework for Glass Bottle Bottom Using Visual Attention Model and Wavelet Transform," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2189–2201, Apr. 2020, doi: 10.1109/TII.2019.2935153.

[13] J. Rostami, P. W. Tse, and M. Yuan, "Detection of broken wires in elevator wire ropes with ultrasonic guided waves and tone-burst wavelet," *Structural Health Monitoring*, vol. 19, no. 2, pp. 481–494, Mar. 2020, doi: 10.1177/1475921719855915.

[14] M. K. Delimayanti, U. P. Lestari, W. Prastiwinarti, R. A. Ryanari, R. A. Prasetyo, and M. K. Ismail, "Development of Integrated Web Application for Fishery Trading in Indonesia," in *2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Dec. 2022, pp. 493–497. doi: 10.1109/ICITISEE57756.2022.10057770.

[15] C. Vorhemus, "Industrial Quality Control of Packages." 2022. [Online]. Available: https://www.kaggle.com/datasets/christianvorhemus/industrial-quality-control-of-packages

[16] A. K. Bitto and I. Mahmud, "Multi categorical of common eye disease detect using convolutional neural network: a transfer learning approach," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 4, pp. 2378–2387, Aug. 2022, doi: 10.11591/eei.v11i4.3834.

[17] S. Adebayo, H. Oluwatobi Aworinde, A. O. Akinwunmi, A. Ayandiji, and A. Olalekan Monsir, "Convolutional neural network-based crop disease detection model using transfer learning approach," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 1, pp. 365–374, Jan. 2022, doi: 10.11591/ijeecs.v29.i1.pp365-374.

[18] H. Toba, H. Bunyamin, J. E. Widyaya, C. Wibisono, and L. S. Haryadi, "Masking preprocessing in transfer learning for damage building detection," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 12, no. 2, pp. 552–559, Jun. 2023, doi: 10.11591/ijai.v12.i2.pp552-559.

[19] T. N. Minh, M. Sinn, H. T. Lam, and M. Wistuba, "Automated Image Data Preprocessing with Deep Reinforcement Learning," *arXiv*, 2018, [Online]. Available: http://arxiv.org/abs/1806.05886

[20] Z. N. K. Swati *et al.*, "Brain tumor classification for MR images using transfer learning and fine-tuning," *Computerized Medical Imaging and Graphics*, vol. 75, pp. 34–46, Jul. 2019, doi: 10.1016/j.compmedimag.2019.05.001.

[21] U. Dudekula and P. N., "Linear fusion approach to convolutional neural networks for facial emotion recognition," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 25, no. 3, pp. 1489–1500, Mar. 2022, doi: 10.11591/ijeecs.v25.i3.pp1489-1500.

[22] M. Tan and Q. V. Le, "EfficientNetV2: Smaller Models and Faster Training," in *Proceedings of Machine Learning Research*, 2021, vol. 139, pp. 10096–10106.

[23] A. Howard *et al.*, "Searching for MobileNetV3," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 1314–1324. doi: 10.1109/ICCV.2019.00140.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Networks," in *European Conference on Computer Vision*, Cham: Springer, 2016, pp. 630–645. doi: 10.1007/978-3-319-46493-0_38.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.
[26] O. KATAR and İ. F. KILINÇER, "Automatic Classification of White Blood Cells Using Pre-Trained Deep Models," *Sakarya University Journal of Computer and Information Sciences*, vol. 5, no. 3, pp. 462–476, Dec. 2022, doi: 10.35377/saucis...1196934.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.
[28] D.-H. Lee, Y.-T. Kim, and S.-R. Lee, "Shallow Landslide Susceptibility Models Based on Artificial Neural Networks Considering the Factor Selection Method and Various Non-Linear Activation Functions," *Remote Sensing*, vol. 12, no. 7, p. 1194, Apr. 2020, doi: 10.3390/rs12071194.
[29] R. F. Naryanto and M. K. Delimayanti, "Machine learning approach for prediction model on biomass characteristic analysis," 2023. doi: 10.1063/5.0115617.

BIOGRAPHIES OF AUTHORS

Wiwi Prastiwinarti specialization in Printing and Packaging Technology, Her research area are Color Reproduction, Printing and Packaging Technology. She is a lecturer at Department of Printing Technology and Publishing at Politeknik Negeri Jakarta. She is a Head of Printing Technology and Publishing Departement at Politeknik Negeri Jakarta. She has supervised and co-supervised more than 50 undergraduate students. She has an intelectual property rights from ministry of Law and Human Rights, Indonesia. In last 3 years, She has authored or coauthored in packaging technology and Internal Conference at IEEE. She can be contacted at email: wiwi.prastiwinarti@grafika.pnj.ac.id.

Mera Kartika Delimayanti received the Doctoral degree in Electrical Engineering and Computer Science from Kanazawa University, Japan with specialization in Bioinformatics. Her research areas are Machine Learning, Deep Learning, Bioinformatics, Medical Informatics, and Biomedical Engineering. Now, she is a lecturer at Department of Computer and Informatics Engineering, Politeknik Negeri Jakarta. She is a Head of Information and Communication Technology Unit at Politeknik Negeri Jakarta. She has supervised and co-supervised more than 80 undergraduate students and 2 masters students. She has authored or coauthored more than 30 publications: 15 proceedings and 15 journals, with 5 H-index Scopus. She was awarded as Exemplary lecturer at 2023 in Politeknik Negeri Jakarta. She can be contacted at email: mera.kartika@tik.pnj.ac.id.

Hendra Kurniawan received the B.S. degree in informatics engineering from Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, in 2007, the M.Sc.Eng. degree in computer science from the National Taiwan University of Science and Technology, Taiwan, in 2010, and Ph.D. in computer science from Kanazawa University, Japan, in 2023. His research interests include privacy-preserving data mining, machine learning, and edge AI security and privacy issues. He is a lecturer at the Department of Informatics Engineering, Faculty of Engineering and Maritime Technology, Universitas Maritim Raja Ali Haji. He is currently a research collaborator at Kanazawa University, Japan. He can be contacted at email: hendra@umrah.ac.id; hendra-kurniawan@se.kanazawa-u.ac.jp.

Yoga Putra Pratama Specialization in Electrical Engineering , Artificial Intelegence and Electronics. He is a lecturer at Printing Technology and Publishing Departement at Politeknik Negeri Jakarta. His research area are artificial intelegence, computer vision, and machine learning. He can be contacted at email: yoga.putra.pratama@grafika.pnj.ac.id.

Hanin Wendho is currently a fourth-year student at Politeknik Negeri Jakarta majoring in Printing Technology and Publishing Departement. His research area are color reproduction, printing and packaging. He can be contacted at email: haninwendho.tgp20@mhsw.pnj.ac.id.

Rizky Adi is currently a fourth-year student at Politeknik Negeri Jakarta majoring in Informatics Engineering. His research areas of interest include Machine Learning, Deep Learning, and Natural Language Processing. He grew up in Bogor and was inspired to pursue a career in Informatics Engineering after joining the student science club in high school. He has completed external studies in AI Research at Bisa.AI and in machine learning at Bangkit Academy. He can be contacted at email: rizky.adi.tik20@mhsw.pnj.ac.id or rizkyzadadi@gmail.com.