I.J. Image, Graphics and Signal Processing, **2025, 6, 1-18**
Published Online on December 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2025.06.01

Web-Based Waste Detection Using YOLOv8 and Classification Performance Comparison: MobileNet and EfficientNet

Apriandy Angdresey*
Department of Informatics Engineering, Universitas Katolik De La Salle Manado, 95000, Indonesia
E-mail: aangdresey@unikadelasalle.ac.id
ORCID iD: https://orcid.org/0000-0003-1310-1671
*Corresponding Author

Indah Yessi Kairupan
Department of Informatics Engineering, Universitas Katolik De La Salle Manado, 95000, Indonesia
Email: ikairupan@unikadelasalle.ac.id
ORCID iD: https://orcid.org/0009-0003-4281-1960

Andre Gabriel Mongkareng
Department of Informatics Engineering, Universitas Katolik De La Salle Manado, 95000, Indonesia
E-mail: 20013004@unikadelasalle.ac.id
ORCID iD: https://orcid.org/0009-0008-0076-0233

Received: 11 November, 2024; Revised: 20 March, 2025; Accepted: 26 August, 2025; Published: 08 December, 2025

Abstract: Environmental pollution resulting from waste is a critical global challenge that significantly affects both the environment and public health, especially in countries like Indonesia. Effective waste management and recycling depend on accurately detecting and classifying different waste types. This study tackles this challenge by evaluating the YOLOv8s algorithm for object detection and conducting a comparative analysis of two mobile-optimized convolutional neural networks (CNNs), *MobileNetV2* and *EfficientNet*, for waste classification. The YOLOv8s model established a promising baseline for detection, achieving a mean Average Precision (mAP@50) of 0.621 on the hold-out test set. *MobileNetV2* proved to be the superior architecture in the classification task, attaining a higher accuracy of 94.4% compared to *EfficientNet's* 87.8%. Additionally, *MobileNetV2* demonstrated significantly greater computational efficiency, with a processing time of 229 *ms* per step, in contrast to *EfficientNet*'s 606 *ms* per step. These findings confirm that combining YOLOv8s for detection and *MobileNetV2* for classification provides a robust and efficient pathway for developing automated waste management systems.

Index Terms: Waste Detection, Object Classification, YOLOv8s Algorithm, MobileNetV2, EfficientNet

1. Introduction

Environmental pollution caused by waste is one of the primary problems faced by many countries worldwide, including Indonesia [1]. Waste that is not managed properly not only pollutes the environment but also endangers the health of humans and other living things [2]. Therefore, an effective and efficient solution is needed to detect and classify types of waste, thereby improving waste management and recycling. Waste is a material or object that has lost its value or function and can no longer be used. The waste problem arises not only due to increased production but also due to a lack of knowledge and awareness of proper waste management [3]. Additionally, the waste management methods currently used are often less effective and efficient in the process of sorting and separating waste.

The growth in the amount of data or information has continued to increase significantly in recent years [4]. Additionally, the development of data processing technology has enabled the utilization of large amounts of data with innovative approaches [5]. Deep learning, a branch of artificial intelligence, leverages complex patterns of data to generate accurate outputs. Implementations of deep learning include computer vision, natural language processing, and

voice recognition. Computer vision encompasses various tasks, including image classification, object detection, image recovery, segmentation, and human pose estimation [6].

Object detection involves identifying the location of a predetermined target object. This process is divided into two main components: first, assessing the category and probability associated with the object, and second, pinpointing the specific location of the target object through the use of bounding boxes [7]. One of the prominent methods in object detection is YOLO (You Only Look Once), renowned for its efficiency and ability in detecting objects in images or videos, owing to its ability to accurately recognize various items [8]. The latest version of this algorithm, YOLOv8, offers significant performance improvements over its predecessor. Research indicates that YOLOv8 exhibits a high level of precision across various applications, including human detection [9], fruit detection [10], object detection in transportation [11], and medical image analysis [12].

Furthermore, image classification involves categorizing images or objects within images into predefined classes. Convolutional neural networks (CNNs) are a crucial component of deep learning, with a primary focus on image processing [13]. This approach uses convolution, pooling, and mathematical operations on multidimensional data, enabling effective image analysis and classification. Several studies have demonstrated that CNN implementations achieve high accuracy in classifying images across various domains. However, different CNN architectures have unique characteristics and performance levels. *MobileNet* and *EfficientNet* are two architectures known for their high efficiency and accuracy [14]. *MobileNet* is tailored for devices with limited resources [15], whereas *EfficientNet* optimizes network parameters to achieve superior performance with high efficiency [16]. In a study [17], *EfficientNet* was compared with *ResNet101* for classifying lung images from CT scans, achieving an accuracy of 88% for *EfficientNet* and 77% for *ResNet101*. *EfficientNet* has also been utilized to classify images related to retinopathy, achieving an accuracy of 98% [16]. *MobileNetV2* has been utilized to classify various types of tumors, competing with architectures like *ResNet50V2*, *InceptionV3*, and *InceptionResNetV2*. It achieved the highest precision at 85% [15]. Additionally, *MobileNetV2* has demonstrated strong performance in non-medical applications, such as waste classification. When applied to five waste categories using transfer learning, it achieved a training accuracy of 95.28% and a validation accuracy of 89.48% [18].

The primary objective of this study is to evaluate and compare the performance evaluation of different CNN architectures, including the *YOLOv8* object detection algorithm, for waste classification tasks. This technical assessment aims to identify the most effective model suitable for potential integration into a future web-based platform. Although the current work focuses on algorithmic evaluation, the broader vision of this research is to support the development of accessible and scalable solutions for waste classification that can contribute to improved waste management practices and public awareness of environmental conditions. This includes, in future work, the implementation of an online system that could be used for educational and self-learning purposes by the public.

The remainder of this manuscript is organized as follows: Section 2 provides a review of the literature on web-based waste detection using YOLOv8 and compares the classification performance of CNN architectures, *MobileNet*, and *EfficientNet*. Moreover, Section 3 outlines the methodology used in the proposed system, detailing the flow from data collection to model validation. The results and performance analysis of the proposed work are discussed in Section 4. Finally, Section 5 presents conclusions and suggestions for future work.

2. Literature Review

Deep learning, introduced in the early 2000s, emerged alongside machine learning algorithms such as Support Vector Machines, Multi-Layer Perceptron, and Artificial Neural Networks. As a subset of machine learning, deep learning is notable for using complex neural network architectures to discern intricate patterns within the data. Its primary strength lies in its ability to automatically learn from and represent complex features without manual human programming [19]. This capability has led to the development of various specialized network architectures in the field of artificial intelligence. Deep learning's efficiency in processing and analyzing large datasets has proven invaluable across various domains, including computer vision, natural language processing, voice recognition, and financial analysis [20]. Among these applications, computer vision has gained significant attention due to its rapid evolution. It enables machines to comprehend and analyze high-level features in visual content, encompassing still images and video. The field of computer vision encompasses several subdomains, including object and event recognition, object detection, motion tracking in video, object segmentation, pose and movement estimation, situational modeling, and image restoration [21]. These advancements in deep learning and computer vision are pushing the boundaries of artificial intelligence and transforming numerous industries and aspects of daily life.

Environmental pollution from improperly managed waste is a pressing global issue, prompting extensive research into advanced waste detection and classification technologies. YOLO (You Only Look Once) is a widely popular object detection algorithm designed to detect objects in real-time with high speed and accuracy [8]. YOLO's architecture consists of three main components: the backbone (for feature extraction), the neck (for further processing and refinement of features), and the head (for predicting object locations and classes). This structure enables YOLO to perform object detection efficiently and accurately [21]. YOLO has evolved from YOLOv1 to the current YOLOv8, developed by the Ultralytics team. YOLOv8 incorporates significant upgrades, including CSPNet for the backbone, a combination of FPN and PAN for the neck, and a decoupled design for the head, enhancing overall accuracy and speed [22].

Recent studies have highlighted the efficacy of YOLOv8 in improving object detection accuracy across various applications. For instance, a study aimed at improving aerial object detection using the YOLOv8-*s* algorithm compared it with other methods using drone images and achieved a mAP of 91.7% [23]. Another study using apple image objects for growth monitoring proposed a model based on YOLOv8-*n*, named YOLOv8-*shufflenet-ghost-se*, which achieved a mAP of 88.3% [10]. Additionally, a study detecting wrist abnormalities using a dataset of 20,327 labeled images evaluated various YOLO versions and found YOLOv8-*m* to perform best, with precision at 84%, sensitivity at 92%, and mAP at 95% [8]. These studies highlight the potential of YOLOv8 and deep learning techniques in advancing waste detection systems, providing promising solutions to environmental sustainability challenges.

CNNs have shown remarkable advancements in image classification tasks. As a cornerstone algorithm in deep learning, particularly in computer vision, CNNs excel at extracting intricate patterns and features from visual data. This sophisticated subset of deep neural networks finds applications across a wide spectrum, including image classification, object detection, segmentation, and image generation. The hierarchical architecture of CNNs, which mimics the human visual system, enables them to learn and differentiate complex features in data at various levels of abstraction. By leveraging convolutional layers, pooling operations, and non-linear activation functions, CNNs effectively capture and comprehend spatial hierarchies and local patterns within image data, allowing them to generalize well to unseen data [24], [25]. CNNs have become the go-to solution for tasks related to computer vision and speech recognition, particularly adept at handling datasets with inherent spatial relationships, such as image data. The network architecture of CNNs consists of a series of hierarchical stages in the learning process. For instance, in object recognition tasks, initial layers extract general features, while subsequent layers build more complex representations, culminating in features closely resembling the original object [17]. The power of CNNs lies in their ability to automatically learn relevant features from raw data, eliminating the need for manual feature engineering. This capability has revolutionized computer vision, enabling breakthroughs in facial recognition, autonomous vehicles, medical image analysis, and more. For example, one study [12] aimed to create a computerized system capable of automatically evaluating and classifying microstructure images of steel materials using the CNN algorithm. The system achieved an accuracy of 94.5% on test data with 160 images. Another study [26] used garbage image objects as data to develop a system that classifies garbage into various types, improving waste management and reducing manual activities. The CNN model, trained on 2513 garbage images, achieved 96% accuracy and 91% precision.

As research in deep learning continues to advance, CNNs remain at the forefront of innovation, with newer architectures and techniques constantly being developed to improve performance and efficiency. Among these, *MobileNet* and *EfficientNet* architectures have emerged as leading contenders due to their optimized performance in resource-constrained environments and high computational efficiency [27]. *MobileNet*, designed for devices with limited computational power, uses depthwise separable convolutions to significantly reduce parameters and computational load without compromising accuracy, making it suitable for mobile and embedded applications. *EfficientNet* employs a compound scaling method to uniformly scale network width, depth, and resolution, achieving better performance with fewer resources. This architecture has proven effective across various image classification tasks, often outperforming traditional models with larger parameter sizes [13]. Both *MobileNet* and *EfficientNet* have been effectively utilized in diverse domains, showcasing their versatility and robustness. They have been used in medical imaging for disease detection, autonomous vehicles for object recognition, and real-time video analysis for surveillance systems. Their ability to maintain high performance with lower computational requirements makes them ideal for developing efficient and scalable deep learning models.

The integration of web-based platforms for waste detection represents a significant advancement in accessibility and usability. These platforms enable users to easily upload waste images, facilitating automatic detection and classification processes [28]. This approach not only streamlines waste management practices but also promotes community engagement in environmental sustainability efforts. Recent literature highlights the importance of comprehensive datasets for training and evaluating deep learning models in waste detection applications. Public datasets available on platforms like Kaggle have been instrumental in developing accurate and scalable models for waste classification [29]. These datasets encompass a wide range of waste types, ensuring the models' ability to generalize and effectively classify diverse waste materials. The integration of YOLOv8 for object detection with *MobileNet* or *EfficientNet* for image classification creates a powerful framework for web-based waste detection systems. This study aims to expand on existing research by conducting a comparative analysis of these architectures, thereby contributing to the advancement of sustainable waste management practices through innovative technological solutions.

3. Methodology

The system framework for web-based waste detection and classification comprises several key stages. Initially, the process begins with data collection, where a comprehensive dataset of waste images is gathered from various sources. Following this, the collected images undergo labeling, which includes specific annotations for detection (drawing bounding boxes around waste objects) and classification (assigning labels to the images). Subsequently, the images undergo preprocessing, involving operations such as auto-orienting to adjust the image orientation, resizing to ensure uniform image dimensions, and filtering to remove any null or corrupted images. The dataset is then split into three parts: train data for training the model, validation data for tuning the model and preventing overfitting, and testing data for evaluating the model's performance.

In this phase, we implement selected deep learning algorithms, including *YOLOv8* for object detection and *MobileNet* or *EfficientNet* for image classification. We define key parameters for the model, such as batch size, Intersection over Union (IoU), optimization method, and learning rate. Finally, we evaluate the model's performance using various metrics, including Mean Average Precision at IoU 0.5 (mAP@0.5), confusion matrix, precision, recall, loss, and the Receiver Operating Characteristic (ROC) curve. This structured approach ensures that each critical aspect of developing and evaluating the web-based waste detection and classification system is thoroughly addressed, thereby enhancing accuracy and efficiency in waste management practices. The entire process is illustrated in the framework shown in Figure 1, which outlines the sequential flow from data collection to model validation.

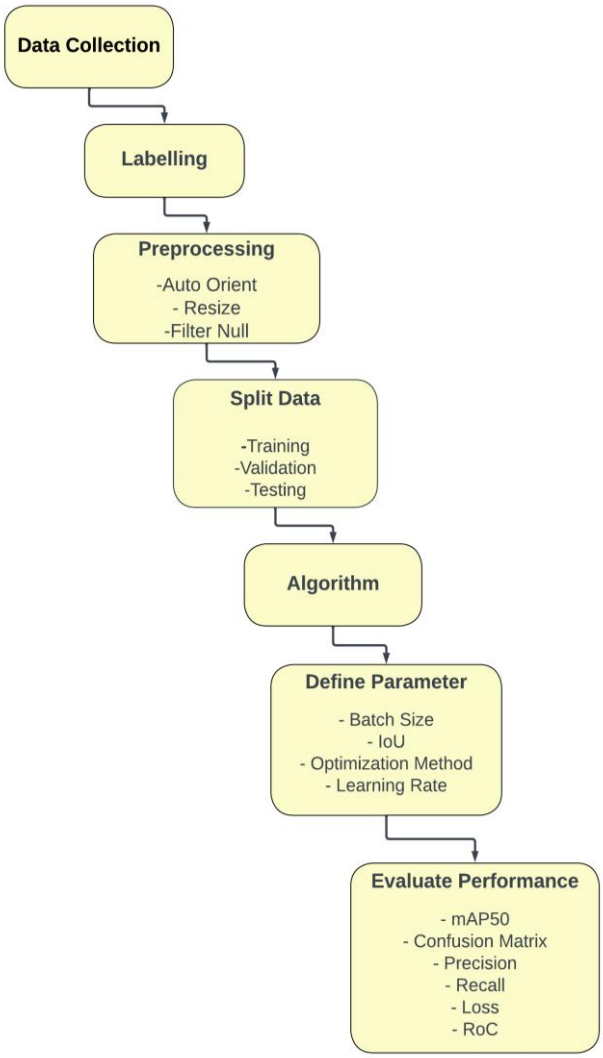

Fig. 1. Research Methods of Web-Based Waste Detection and Classification

3.1 Collect Data

The first phase of this study involved collecting data through waste images sourced from various avenues, including online databases, local waste management facilities, and user-generated content. This data encompasses a range of waste categories, such as *plastic*, *metal*, *organic waste*, *paper*, and *glass.* The goal was to ensure that the data utilized in this study were both diverse and representative of the different types of waste typically encountered. Most of the data was gathered using the Kaggle platform, which provides a broad selection of relevant public datasets. The primary source for the object detection task was the Waste Recycling Plants (WaRP) Dataset, while the classification task was based on the TrashNet dataset; however, not all images from these sources were utilized. Moreover, the various public datasets were integrated to create a more comprehensive and representative dataset. The detection dataset consisted of 4,403 images, while the classification dataset included 13,059 images. This methodology enabled the development of a more accurate model that can identify a diverse range of waste categories. The choice of the Kaggle platform was motivated by the difficulties related to independent data collection and the availability of publicly accessible datasets. The primary objective of this research is to identify waste within images and classify it into two main categories: organic and inorganic. Out of the 12 waste categories identified, only five were selected for their relevance to the research goals.

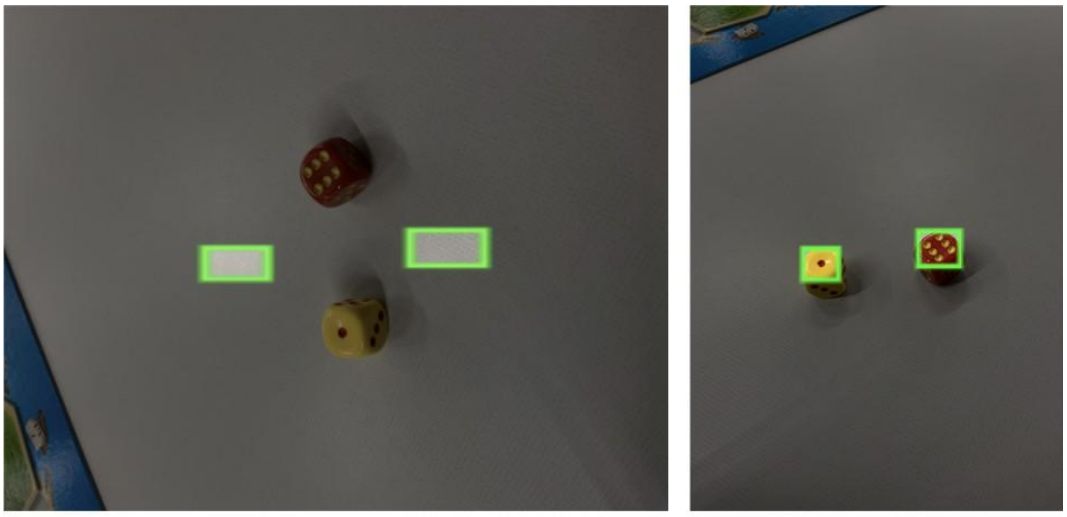

Fig. 2. An Example of Auto-Orient

3.2 Labeling

In this process, the images in the newly compiled dataset will be labeled. For object detection, the labeling process involves drawing bounding boxes around the waste objects to establish ground truth data. Each bounding box is carefully placed to accurately encompass the object, ensuring precise detection during model training. For image classification, the labeling process is automated using the TensorFlow library, which enables efficient categorization of images into predefined classes. This automation streamlines the preparation of the dataset for training the classification model.

3.3 Pre-Processing

In the preprocessing phase, the collected images undergo several enhancements to improve their quality and suitability for analysis. This includes auto-orientation to correct image orientation, resizing images to a standard resolution, and normalizing pixel values to ensure consistency across images. Additionally, the dataset is augmented using techniques such as rotation, flipping, and cropping to increase the model's robustness against variations in the data. Lastly, any images with null values are filtered out to ensure a clean and reliable dataset.

1. Auto Orient

Auto-orientation is a crucial feature in image processing and computer vision that ensures all images in a dataset are correctly aligned, regardless of their original orientation at capture. It addresses the discrepancy between how digital cameras store images (often in a fixed orientation with EXIF metadata indicating display orientation) and how they should be visually represented. By automatically adjusting images based on their EXIF data, auto-orientation creates a uniform dataset where pixel coordinates consistently align with visual content. This uniformity is crucial for training accurate machine learning models, particularly for tasks such as object detection and classification. It prevents common errors in computer vision projects caused by misaligned bounding boxes or annotations, ultimately enhancing the model's ability to interpret and analyze visual data accurately. While generally recommended, auto-orientation can be optionally disabled in specific scenarios where consistent orientation is guaranteed, and processing speed is prioritized. Figure 2 shows an example of auto-orientation.

2. Resize

The next pre-processing step is resizing. Since the new dataset combines images from various sources, the size of each image may differ. For example, in Figure 3, images for detection are resized to 640 × 640 pixels, while images for classification are resized to 256 × 256 pixels. The resizing process adjusts all images in the dataset to these specified dimensions, stretching or compressing them as needed. This ensures that all images have consistent sizes, which is crucial for training machine learning models. Consistent image dimensions help maintain uniformity in the input data, enabling the models to process the images more effectively and achieve better performance.

3. Filter-Null

The final step in the pre-processing phase involves filtering out null images. During the labeling process, some images may lack detectable objects, which are referred to as null images. The filter null feature in RoboFlow is employed to remove these images from the dataset. This step is vital, as it enables the model to learn more effectively by concentrating on images that contain actual objects, ensuring that the training data is both relevant and valuable. The

model can focus on significant features by discarding irrelevant images, ultimately enhancing its performance and accuracy during training.

Fig. 3. An Example of Resize

3.4 Split-Data

The dataset is divided into training, validation, and testing subsets, with 80% of the data allocated for training, 10% for validation, and the remaining 10% for testing. This division allows the model to be trained on a substantial amount of data while also being evaluated on unseen samples, which is crucial for assessing its generalization capabilities. The primary purpose of the training data is to enable the model to recognize patterns and features within images. By analyzing these images, the model gains an understanding of the visual characteristics of various objects and learns to distinguish between them. Through exposure to numerous examples, the model can identify the unique attributes of each object type.

Validation data is utilized to assess the model's performance during the training process. The model learns from the training data and periodically evaluates its accuracy using the validation data. This approach ensures that the model does not simply memorize the training images but rather learns to generalize its understanding to new images. The validation data serves as a form of practice tests, allowing for adjustments and fine-tuning of the model to improve its performance. Furthermore, testing data is used to evaluate the model's performance after training has been completed. Once the model has undergone complete training and refinement with both the training and validation data, it is assessed using the testing data. This dataset is distinct from the training and validation sets, ensuring an unbiased evaluation of the model's capabilities.

3.5 Define Parameter

Once the dataset has been divided into training, validation, and testing subsets, defining the parameters is the next essential step before training the model. By establishing these parameters in advance, a structured framework is created for the model’s learning process. This strategic approach enables fine-tuning the model’s complexity, reduces the risk of overfitting, and ultimately enhances overall performance.

1. Batch size refers to the number of training examples used in each iteration of the training process. A higher batch size speeds up the training process but requires more memory, while a smaller batch size might introduce more noise in gradient estimation but can sometimes lead to better generalization. The optimal batch size depends on the specific problem, available computational resources, and desired trade-offs between speed, memory usage, and model performance. In this study, we used a batch size of 16.
2. The Intersection over Union (IoU) metric measures the accuracy with which an object detection model predicts the location of an object using bounding boxes. In this study, an IoU threshold of 0.5 is used. IoU is calculated by dividing the area of intersection between the predicted bounding box and the ground truth bounding box by the area of their union. A high IoU value indicates that the model’s prediction closely matches the actual annotation
3. Optimization methods are crafted to effectively adjust model parameters to minimize the error or loss function during training. Various methods offer distinct benefits in terms of convergence speed, memory efficiency, and robustness to different data types and models. Selecting the appropriate optimization method typically requires experimentation and consideration of the problem’s specific characteristics and the data. In this study, the optimizer used is *Adamax*, a variant of *Adam* that is based on the infinity norm. *Adamax* is a first-order gradient-based optimization method that adjusts the learning rate according to data characteristics, making it well-suited for learning time-variant processes.

4. The learning rate plays a crucial role in machine learning optimization as it directly influences the speed and effectiveness of a model's learning process from data. Determining the optimal learning rate involves striking a balance between convergence speed and stability to achieve peak performance.

3.6 Algorithm

In this study, we included two primary methodologies to facilitate image-based waste identification and categorization. We employed the YOLOv8 algorithm for object detection, the most recent iteration of the YOLO series, specifically engineered for direct object recognition in a single step. This method operates by concurrently analyzing the entire image, yielding a forecast of the object's location and categorization in one step. YOLOv8 has many technological enhancements, including anchor-box-free detection, a backbone architecture utilizing CSPNet for rapid feature extraction, and a head configuration that distinguishes between classification and box regression tasks. For picture categorization, we employed a CNN methodology, contrasting two architectures: *MobileNet* and *EfficientNet*. *MobileNet* is engineered for efficiency on resource-limited devices, employing depthwise separable convolution. *EfficientNet* employs a balanced scaling methodology to modify the network's depth, breadth, and resolution. This work evaluates the efficacy of both designs in garbage picture categorization, focusing on accuracy and computational efficiency across diverse testing settings. *MobileNet* and *EfficientNet* are two convolutional neural network architectures that demonstrate a satisfactory equilibrium between model complexity and accuracy, as indicated by the literature review. *MobileNet* exhibits relatively consistent performance in low-to-moderate classification tasks, whereas *EfficientNet* typically achieves greater accuracy at comparable model sizes. Consequently, the selection of a model architecture is significantly influenced by the trade-off between the efficient utilization of computational resources and the accuracy of classification.

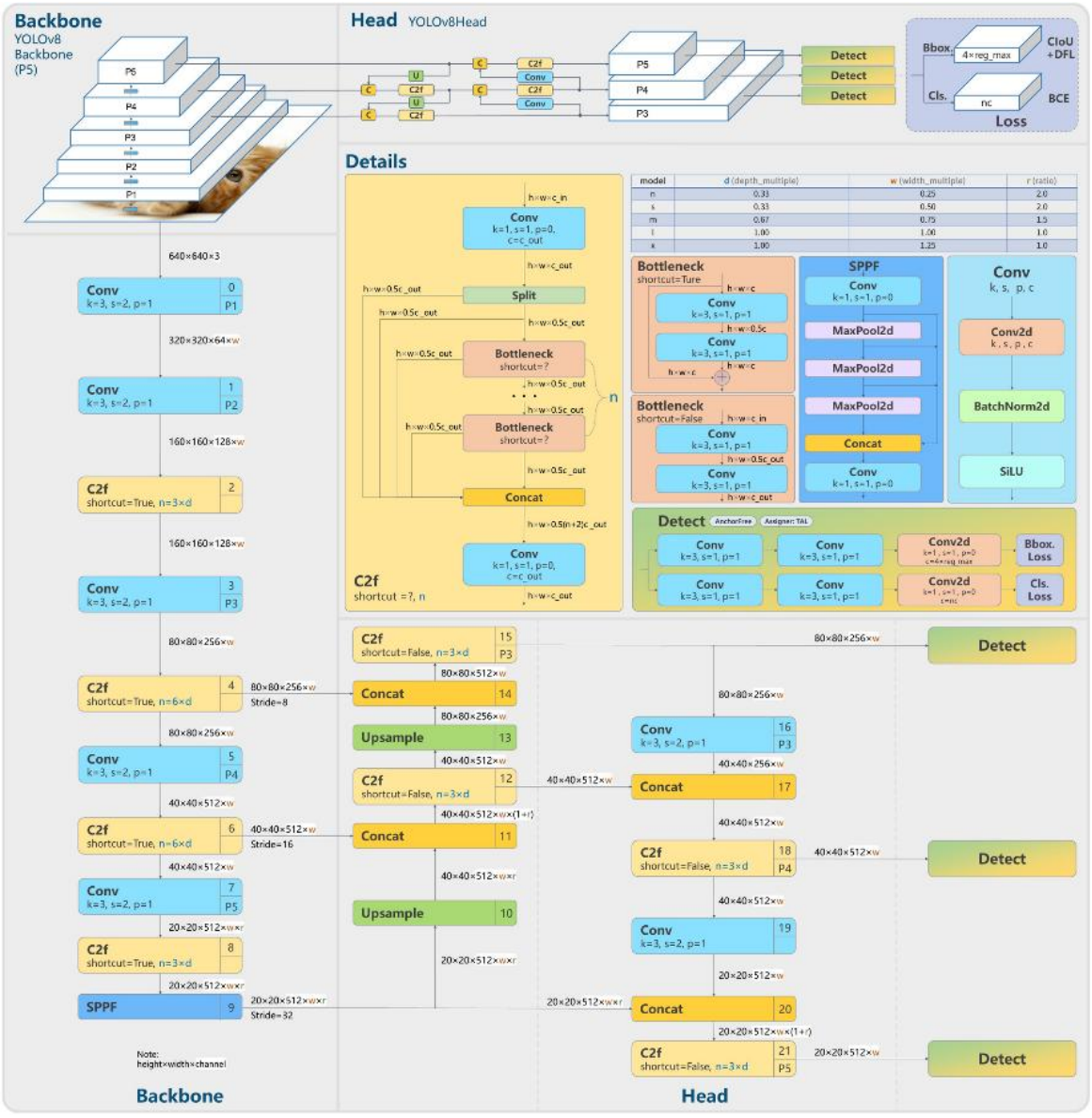

Fig. 4. YOLOv8 architecture overview. Adapted from Jocher [30].

1. YOLOv8 represents the pinnacle of evolution in the YOLO algorithm series, marking a significant advancement in object detection technology. This latest iteration builds upon the strengths of its predecessors while introducing innovative enhancements that push the boundaries of speed and accuracy. As a single-stage detector, YOLOv8 adheres to the core YOLO principle of processing entire images in a single forward pass, predicting bounding boxes and class probabilities simultaneously. This approach remains pivotal in maintaining YOLO's efficiency, facilitating real-time object detection across diverse applications. Key innovations in YOLOv8 include the adoption of an anchor-free detection methodology, which simplifies the detection process and enhances performance on small objects. The architecture incorporates an upgraded backbone utilizing CSPNet (Cross Stage Partial Network) for more efficient feature extraction, complemented by a neck that integrates elements from FPN (Feature Pyramid Network) and PAN (Path Aggregation Network). This fusion enhances the model's ability to detect objects across various scales with greater precision. YOLOv8 also introduces a decoupled head design, separating bounding box prediction from classification tasks to optimize each task individually, thereby improving the overall accuracy. Additionally, the model employs a refined loss function that effectively balances components of the detection task during training, resulting in more robust performance. These architectural enhancements, combined with optimizations in the training process, enable YOLOv8 to achieve state-of-the-art results in both speed and accuracy. This makes it particularly well-suited for real-time computer vision applications in fields such as autonomous driving, surveillance, and robotics. Figure 4 illustrates the architecture of YOLOv8 [30].
2. *MobileNetV2* represents a significant advancement in efficient neural network architecture, specifically designed to optimize performance on mobile and resource-constrained devices. Building upon its predecessor, *MobileNetV2* introduces key innovations that strike a delicate balance between computational efficiency and accuracy. The architecture's cornerstone features include linear bottlenecks and inverted residuals, which work in tandem to reduce the model's parameter count while preserving its ability to capture complex features. At the heart of *MobileNetV2* lies the depthwise Separable Convolution, a technique that dramatically reduces the number of parameters and computational requirements compared to standard convolutions, enabling the model to achieve impressive results in image classification with a significantly smaller footprint. Figure 5 illustrates the architecture of *MobileNetV2*, showcasing its carefully orchestrated structure composed of various layers and blocks, each serving a specific purpose in enhancing the network's inclusive efficiency and effectiveness. The model begins with standard convolutional layers to process input features, followed by a sequence of depthwise separable convolutions that form the backbone of its efficiency. The introduction of inverted residual blocks in *MobileNetV2* expands the number of channels before applying depthwise convolution and then contracts them afterward, facilitating the capture of richer features with fewer parameters. Bottleneck designs further compress information flow, thereby reducing computational overhead. Additionally, the integration of squeeze-and-excitation blocks enhances the model's ability to prioritize essential features by dynamically recalibrating channel-wise feature responses. This sophisticated arrangement of components enables *MobileNetV2* to efficiently extract and process complex visual features, making it an ideal choice for deployment on mobile devices and other platforms with limited computational resources. Despite its resource efficiency, *MobileNetV2* maintains competitive performance against more resource-intensive architectures.

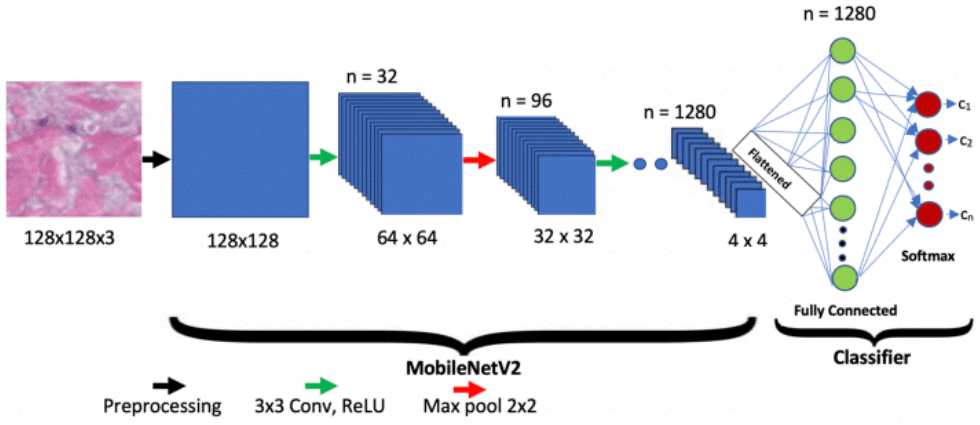

Fig. 5. Architecture of MobileNetV2

3. *EfficientNet* represents a groundbreaking development in Convolutional Neural Networks (ConvNets), introducing a novel approach to network scaling that significantly advances both efficiency and accuracy. At its core, *EfficientNet* is built upon compound scaling, a method that harmoniously scales three critical dimensions of network architecture: depth, width, and input resolution. This innovative approach stems from the understanding that balancing these dimensions is crucial for achieving optimal performance. Unlike previous models that typically scaled only one or two dimensions, *EfficientNet* ensures balanced growth across

all three dimensions as the model size increases. This results in a family of models that not only achieve higher accuracy but also require remarkably fewer parameters and reduced computational resources compared to their predecessors. The architecture of *EfficientNet* is meticulously designed to leverage this compound scaling principle effectively, as demonstrated in Figure 6. It begins with *EfficientNet-B0*, an optimized baseline model derived through neural architecture search. This model serves as the foundation for larger, more powerful models (*B1* to *B7*), which are constructed using the compound scaling method. Each successive model scales up in depth, width, and resolution according to carefully tuned scaling coefficients. *EfficientNet's* scaling strategy allows it to capture increasingly complex features and patterns as model size grows, without the inefficiencies typically associated with scaling up neural networks. This versatility results in a series of models that offer a range of performance-efficiency trade-offs, suitable for diverse applications and computational budgets. Furthermore, *EfficientNet* integrates modern CNN techniques, including depthwise separable convolutions, squeeze-and-excitation blocks, and efficient activation functions, thereby enhancing both performance and efficiency. By combining innovative scaling with optimized architecture, *EfficientNet* sets new benchmarks in the field, achieving superior accuracy with significantly fewer parameters and floating-point operations (FLOPs) compared to traditional convolutional neural networks (ConvNets). This makes it an excellent choice for various computer vision tasks, particularly in scenarios where high performance and computational efficiency are critical.

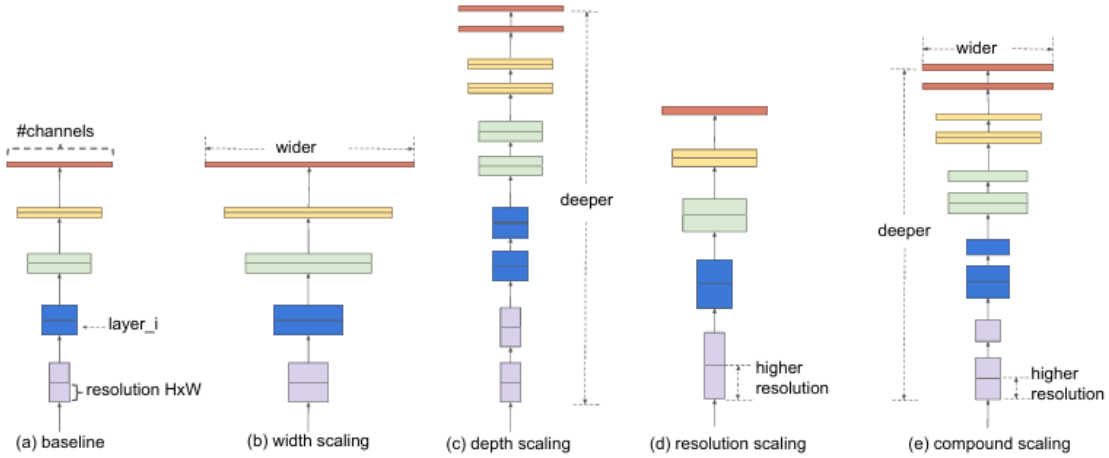

Fig. 6. Architecture of *EfficientNet*

The selection of *MobileNetV2* and *EfficientNet* as algorithms for this study is based on their comparable characteristics and performance profiles. Both architectures are designed to be computationally efficient while maintaining competitive accuracy. Unlike larger models such as *VGG* or *ResNet*, *MobileNetV2* and *EfficientNet* strike a favorable balance between model size and performance. Their similar design philosophies, which prioritize optimizing resource us- age without significant accuracy trade-offs, make them ideal candidates for comparison in scenarios where computational efficiency is crucial. This choice enables a meaningful evaluation of lightweight yet powerful models within the context of the current research objectives.

3.7 Validation

After completing all preparatory processes, the next critical step is model training. During this phase, the model learns from the prepared dataset to optimize its parameters for the given task. Once training concludes, the evaluation phase becomes pivotal. Evaluation involves assessing the model's performance across various metrics such as mean Average Precision (mAP), Confusion Matrix, Receiver Operating Characteristic (ROC) curves, and loss functions. These metrics collectively provide insights into how well the model generalizes to new data and performs the intended tasks. Evaluating these results is essential to determine whether the model meets the desired performance benchmarks.

1. mAP is a commonly used evaluation metric in object detection that measures a model's general performance. It assesses how accurately the model identifies objects across various classes and predicts their locations. The calculation of mAP is based on Equation 1.

$$mAP = \frac{1}{n}\sum_{t=1}^{n} AP_t \quad (1)$$

2. A confusion matrix is a structured table that summarizes the performance of a machine learning classification model. It juxtaposes the predicted labels from the model with the actual labels from the dataset, categorizing outcomes into four distinct groups: true positives (TP, which are correct predictions of the positive class), true

negatives (TN, accurate predictions of the negative class), false positives (FP, instances incorrectly predicted as positive when they are negative), and false negatives (FN, instances incorrectly predicted as negative when they are positive). Accuracy measures the ratio of correct predictions to the total cases evaluated, as shown in Equation 2. Additionally, precision quantifies the proportion of true positive predictions among all positive predictions, as detailed in Equation 3. Meanwhile, recall, also referred to as sensitivity, gauges the proportion of actual positives that are correctly identified, as shown in Equation 4. The F1 score serves as the harmonic mean of precision and recall, providing a unified metric that balances both elements and is calculated using Equation 5.

$$accuracy = \frac{TN+FP}{TN+FP+FN+TP} \tag{2}$$

$$Precision = \frac{TP}{(TP+FP)} \tag{3}$$

$$recall = \frac{TP}{TP+FN} \tag{4}$$

$$F1_{Score} = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall} \tag{5}$$

3. Receiver Operating Characteristic (ROC) demonstrates the diagnostic capability of a binary classifier system as its discrimination threshold varies. The Area Under the ROC Curve (AUC-ROC) is a comprehensive measure of classification performance across all possible thresholds, calculated using Equation 6

$$ROC = \int_0^1 TPR\left(FPR^{-1}(t)\right)dt \tag{6}$$

4. Loss is a metric that quantifies the disparity between a model's predictions and the actual target values. It measures the model's performance and directs the training process, intending to minimize this loss to improve the model's accuracy and generalization capabilities. The calculation of loss is represented by Equation 7.

$$loss = \frac{1}{N}\sum_{t=1}^{N}\sum_{c=1}^{C} y_{t,c} log_{(p_{t,c})} \tag{7}$$

4. Performance Evaluation

This section will examine the evaluation of performance, divided into two main components: the experimental setup and the experimental results, accompanied by a detailed discussion. The experimental setup will detail the methodologies and configurations employed to conduct thorough assessments of the algorithms and models under investigation. This encompasses aspects such as dataset selection, preprocessing steps, parameter settings, and the hardware specifications utilized during experimentation. Following the setup, we will present the results obtained from the evaluation process. These results will be analysed and discussed in detail, with a focus on key metrics such as accuracy, precision, recall, and computational efficiency. This comprehensive analysis aims to provide valuable insights into the effectiveness and robustness of the approaches examined in relation to the research objectives.

4.1 Experimental Setup

This part provides a comprehensive overview of the model architecture, data preparation, training process, and evaluation methodology. It details all relevant parameters, algorithms, and procedures utilized in the development and testing of the model, thereby aiding in comprehension, validation, and potential reproduction of this research by other scholars. The dataset employed in this study comprises two components: an image classification set and an object detection set. The classification dataset comprises 13,059 images categorized into *Glass*, *Cardboard*, *Paper*, *Food*, and *Plastic*. For object detection, 4,403 images containing 14,101 annotated objects are utilized. Both datasets are split into training, validation, and testing sets with an 80:10:10 ratio. A batch size of 16 is employed during training. This value was selected as a pragmatic balance between training stability and computational resources, ensuring consistent gradient updates without exceeding the GPU memory (VRAM) limitations of the hardware used in this study. Object detection images are resized to 640 x 640 pixels, while classification images are scaled to 256 x 256 pixels. Evaluation for object detection utilizes an Intersection over Union (IoU) threshold set at 0.5. This conventional threshold is integral to the mAP50 metric calculation and aligns with standard benchmarks. It is also well-suited for datasets with potential variations in image quality, as it avoids unfairly penalizing valid detections due to minor inaccuracies in bounding boxes. Model optimization uses the *Adamax* algorithm with a learning rate of 10−3. These parameters are carefully chosen to strike a balance between computational efficiency and model performance across both tasks. Furthermore, the simulations for this study were conducted on an Intel Core i7-12700H processor (16 CPUs, ~2.3 GHz) with NVIDIA CUDA V12.5.40 (16 GB) support.

4.2 Experimental Results

In this section, we present the experimental results, which comprise three main components: training outcomes, detection testing, and a comparative analysis of classification results using CNN architectures, specifically *MobileNet* and *EfficientNet*. The training results will detail the performance metrics achieved during the model training phase, including accuracy, loss curves, and convergence trends across epochs. Following this, the detection testing segment will evaluate the model's effectiveness in identifying and localizing objects within test images, utilizing metrics such as precision, recall, and the F1-score, with reference to the previously set IoU threshold. Lastly, we will compare the classification results obtained from *MobileNet* and *EfficientNet* architectures, analyzing metrics such as classification accuracy and inference speed to discern the strengths and trade-offs of each model in handling the dataset's diverse image categories. These analyses aim to provide a comprehensive understanding of the models' performance.

The training process concluded after 80 epochs as the model's learning progress plateaued, with minimal improvements noted. This automatic termination was due to a callback parameter configured to 10, which halted training if no significant changes were observed in the last 10 epochs. The training results are shown in Figure 7. The graphs illustrate the typical training progression of a YOLOv8 model, underscoring its effectiveness in object detection tasks. The decreasing loss curves for box, classification, and DFL losses across training and validation sets indicate that YOLOv8 is successfully learning to localize and classify objects. Improvements in precision, recall, and mean Average Precision (mAP) metrics further validate the model's growing accuracy in object detection. YOLOv8's architecture effectively balances precise bounding box predictions (box loss) and accurate class predictions (*cls loss*). The similar trends observed in training and validation metrics suggest strong generalization without overfitting. The mAP50 and mAP50-95 curves demonstrate the model's robust performance across various Intersection over Union (IoU) thresholds, a notable strength of the YOLO family. These results reaffirm YOLOv8's standing as a high-performance, real-time object detection model, showcasing significant advancements over its predecessors in both accuracy and efficiency.

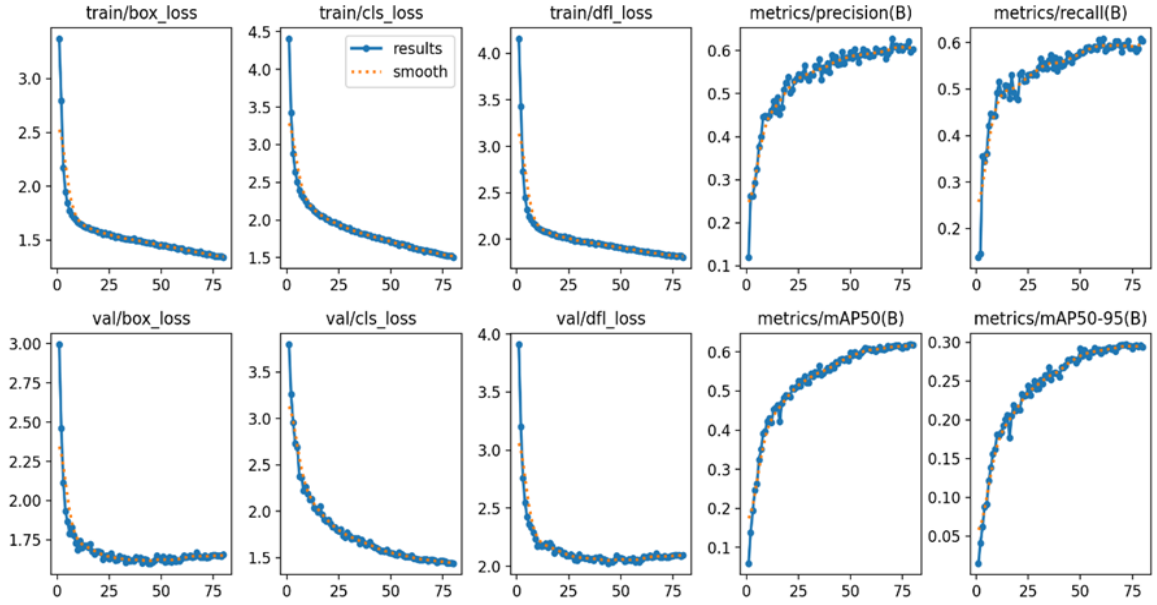

Fig. 7. Training Results Metrics for YOLOv8

Table 1 presents a comparative analysis of optimizer performance for the object detection task utilizing a pretrained YOLOv8s model on a dataset comprising 4,403 images. We compared two prominent optimizers: *Adamax* and Stochastic Gradient Descent (SGD), with both models trained for three epochs to establish a baseline. *Adamax* achieved significantly higher performance, with an mAP50-95 of 0.1943, compared to SGD's 0.0003. Similar trends were observed in mAP50 scores and validation losses. *Adamax* consistently yielded lower losses in bounding box regression (*val/box* loss), classification (*val/cls* loss), and distribution focal loss (*val/dfl* loss), indicating more stable convergence and improved generalization under these training settings. Based on these results, *Adamax* was adopted as the primary optimizer for all subsequent experiments in this study.

Table 1. Cross-Validation: *AdamW* vs *SGD* Optimizer Performance

Optimizer	**Epochs**	**mAP50-95**	**mAP50**	***Val/box_loss***	***val/cls_loss***	***val/dfl_loss***
AdamMax	3	0.1943	0.4051	1.6954	1.9743	1.7863
SGD	3	0.0003	0.0014	4.0226	5.3629	5.9037

Following the selection of the optimizer, a preliminary experiment was conducted to determine the optimal data partitioning strategy. The results of this comparison, presented in Table 2, reveal that the 80:10:10 split yielded superior outcomes on the primary evaluation metric. Specifically, the model trained using this 80:10:10 configuration achieved an mAP50−95 score of 0.3273, surpassing the 0.3266 score attained with the 70:15:15 split. While the mAP50 score was marginally higher for the 70:15:15 configuration, the improved performance on the more stringent mAP50-95 metric, coupled with comparable validation losses, indicates a distinct advantage in allocating a larger portion of the data for training. This empirical evidence supports the conclusion that prioritizing the size of the training set is beneficial for this particular task. As a result, the 80:10:10 split has been established as the standard partitioning scheme for all subsequent experiments conducted in this study.

Table 2. Evaluation Metrics of the Final Model Across Data Splits

Metric	Data Splits	
	70:15:15	**80:10:10**
metrics/mAP50 − 95	0.3266	0.3273
metrics/mAP50	0.6133	0.6096
val/box loss	1.6224	1.6184
val/cls loss	1.3558	1.3563
val/dfl loss	1.7175	1.7080

The significant improvement in the model's object detection capabilities over 80 epochs is presented in Table 3. Initially, the model performs poorly, with an mAP50 of 1.506% in the first epoch. However, it rapidly improves, reaching an mAP50 of 42.209% by epoch 10 and 61.752% by epoch 80. Both box and classification losses decrease substantially, indicating better accuracy in predictions. Precision and recall also improve steadily, reaching approximately 60% by epoch 80, demonstrating a balance between avoiding false positives and detecting true positives. Although the rate of improvement slows in the later epochs and some minor overfitting is noted, the final model shows balanced performance, making it suitable for practical applications.

Table 3. Model Object Detection Capabilities Over 80 Epochs

Epochs	Box Loss	Classification Loss	Dist. Focal Loss	Precision	Recall	mAP 50
1	33.727	44.087	0.0119	0.13762	0.05924	0.01506
2	27.963	34.222	34.282	0.2613	0.14616	0.13854
3	2.17	28.846	2.728	0.26175	0.35509	0.1939
4	19.485	26.388	2.447	0.29287	0.34455	0.24619
5	18.424	2.508	2.319	0.32351	0.36172	0.26318
6	1.769	23.943	2.242	0.37691	0.4214	0.32443
7	1.731	23.287	22.085	0.39949	0.44701	0.35137
8	17.055	22.944	21.705	0.44557	0.44279	0.39198
9	16.908	22.513	21.519	0.44763	0.44244	0.39704
10	16.616	22.011	2.126	0.44414	0.49221	0.42209
...	...	...	...	...	...	...
75	13.587	15.404	18.279	0.60171	0.58806	0.60972
76	13.474	15.322	18.197	0.6088	0.58815	0.61574
77	1.345	15.273	18.143	0.61029	0.58112	0.61287
78	13.462	15.212	18.211	0.62121	0.58714	0.61783
79	13.463	15.213	18.212	0.59639	0.60857	0.61922
80	13.365	15.02	18.037	0.60298	0.60422	0.61752

Moreover, the subsequent critical phase of our study centers on assessing the model's effectiveness in detecting waste objects. The results from a comprehensive two-stage evaluation are detailed in Table 4. This table distinguishes between the model's performance on the validation set at the final training epoch and its performance on the completely unseen hold-out test set. This methodological differentiation provides a clear insight into the model's learned capabilities and its ability to generalize to new data. The results demonstrate excellent generalization, with the model exhibiting slightly superior performance on the unseen test set compared to the final validation set. Specifically, the model achieved a final mAP50 score of 0.3290 and an mAP50 of 0.6210 on the test set, in contrast to scores of 0.3257 and 0.6176 on the validation set, respectively. This outcome confirms that the training process effectively mitigated overfitting, resulting in a robust model. The evaluation, utilizing standard object detection metrics, provides a precise and reliable assessment of the model's capability in identifying and localizing waste objects.

Table 4. Performance Comparison of the Final Model on Validation and Test Sets

Metric	Validation Set	Testing Set
Precision	0.6432	0.6380
Recall	0.5603	0.5750
mAP 50	0.6176	0.6210
mAP 50-95	0.3257	0.3290

Table 5. Performance Comparison of *MobileNetV2* and *EfficientNet*

Model	*MobileNetV2*	*EfficientNet*
Epochs	100	100
Accuracy	0,9816	0,9332
Loss	0,0524	0,2892
Precision	0,9829	0,9237
Recall	0,9807	0,8982
Val_Acc	0,9334	0,9292
Val_Loss	0,2709	0,3133
Val_Precision	0,9356	0,9083
Val_Recall	0,9311	0,8821

The next phase involves training the classification models, specifically *MobileNetV2* and *EfficientNet*, as detailed in Table 5. The training outcomes for both models highlight notable performance discrepancies. Each model was trained for 100 epochs, with *MobileNetV2* demonstrating superior performance across several key metrics. It achieved a higher accuracy of 0.9816, compared to EfficientNet's 0.9332, along with a significantly lower loss of 0.0524, compared to EfficientNet's 0.2892. These metrics suggest that *MobileNetV2* offers improved overall prediction accuracy and a more suitable model fit.

Table 6. *MobileNetV2* Testing Results

Class	Precision	Recall	F1-score
Glass	0.76	0.96	0.85
Cardboard	0.96	0.97	0.97
Paper	0.92	0.99	0.95
Food	0.99	0.89	0.94
Plastic	0.94	0.96	0.95

In terms of precision and recall, *MobileNetV2* outperformed *EfficientNet*, achieving precision and recall values of 0.9829 and 0.9807, respectively. This superior performance indicates fewer false positives and a higher ratio of true positives detected compared to *EfficientNet*, which recorded a precision of 0.9237 and a recall of 0.8982. An evaluation of the validation data also showed that *MobileNetV2* had a slight edge in accuracy, with a score of 0.9334 compared to 0.9292 for *EfficientNet*, along with a lower validation loss of 0.2709 compared to 0.3133. Additionally, *MobileNetV2* achieved a validation precision of 0.9356 and a recall of 0.9311, both of which surpassed the corresponding metrics of *EfficientNet*, which were 0.9083 and 0.8821, respectively. This highlights the *MobileNetV2's* superior capacity to generalize to unseen data. Therefore, *MobileNetV2* consistently demonstrated enhanced accuracy, precision, recall, and loss throughout both the training and validation phases, establishing it as a more effective model than *EfficientNet* for this particular task. The *MobileNetV2* architecture showcased robust and well-balanced performance on the testing dataset, as illustrated in Table 6. The model attained an impressive overall accuracy of 94.41% along with a low loss of 0.2427, further supported by substantial aggregate precision (0.9460) and recall (0.9436). This effectiveness is underscored by its strong class-wise metrics, achieving high F1-scores for *Cardboard* (0.97), *Paper* (0.95), and *Plastic* (0.95).

A more thorough error analysis was conducted to investigate the specific performance variations, particularly concerning the *Glass* class, which demonstrated a lower F1-score of 0.85 and a precision of 0.76. This reduced precision is primarily attributed to a challenging failure mode characterized by visual ambiguity. An analysis of the confusion matrix (as shown in Figure 8) reveals that misclassifications occur when transparent plastic objects are incorrectly identified as *Glass*, due to their inherent similarities in transparency and reflections. In contrast, Figure 9 presents the confusion matrix of the *EfficientNet* architecture, enabling a direct comparison of misclassification patterns across both models. In addition to this class-specific analysis, the overall training dynamics were examined in detail. While minor fluctuations are noted in the validation curves, these can be ascribed to the high variance in the composition of the validation batches. To proactively mitigate the risk of overfitting, an early stopping mechanism was implemented with a patience of 3 epochs, monitored based on the designated metric. The model's ability to complete its

entire training cycle without interruptions confirms that significant overfitting was successfully avoided, allowing the model to converge to a stable and effective state.

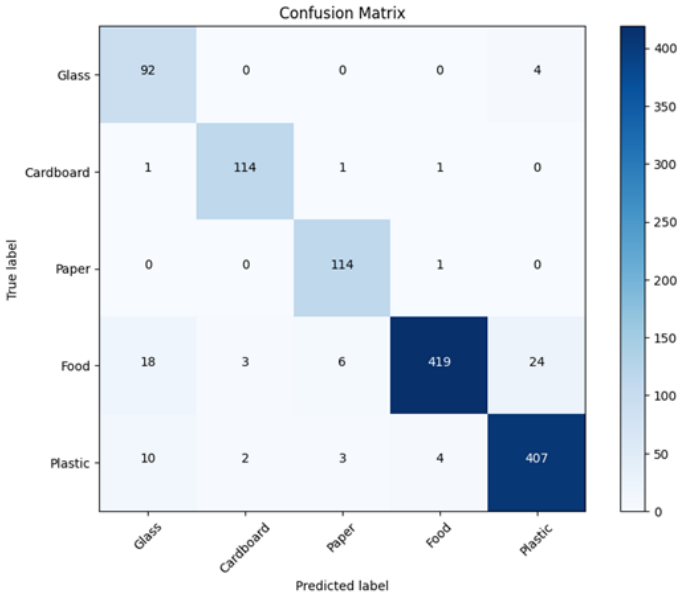

Fig. 8. The result of the Confusion Matrix of MobileNet*V2*

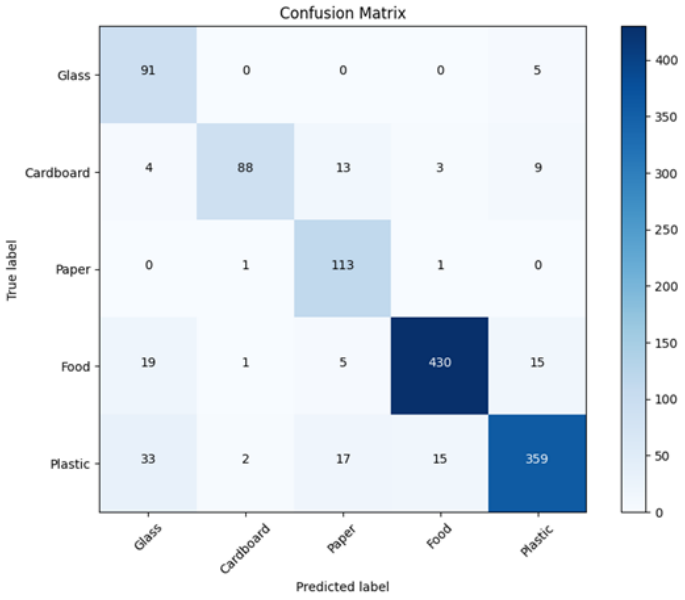

Fig. 9. The result of the Confusion Matrix of *EfficientNet*

In contrast, while *EfficientNet* demonstrates respectable performance, it falls short of *MobileNetV2* in several critical areas, as illustrated in Table 7. A thorough analysis of its class-wise metrics reveals a tendency to prioritize recall over precision. For example, it achieves very high recall rates for the *Glass* (0.95) and *Paper* (0.98) classes; however, this comes with notably lower precision scores of 0.62 and 0.76, respectively. The lower precision associated with the *Glass* class is attributed to the same visual ambiguity with transparent plastic objects discussed earlier, highlighting this as a dataset-specific challenge.

Table 7. *EfficientNet* Testing Results

Class	Precision	Recall	F1-score
Glass	0.62	0.95	0.75
Cardboard	0.96	0.75	0.84
Paper	0.76	0.98	0.86
Food	0.96	0.91	0.94
Plastic	0.93	0.84	0.88

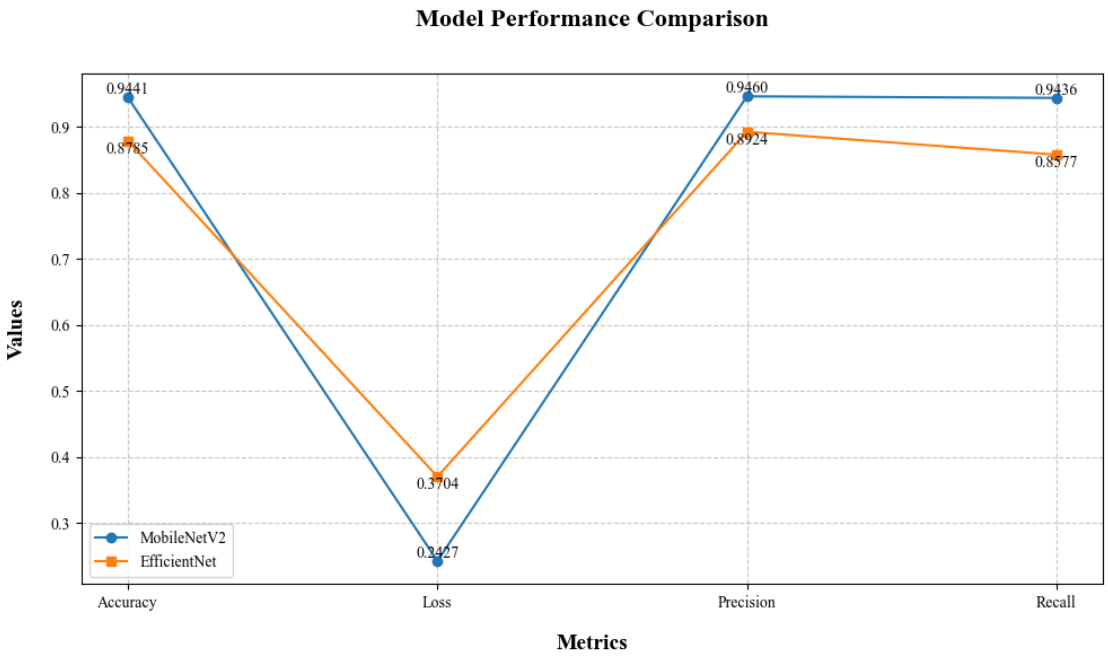

Fig. 10. Performance Comparison Between *MobileNetV2* and *EfficientNet*

This imbalance ultimately affects its overall performance, leading to a lower accuracy of 87.85% and a considerably higher validation loss of 0.3704 compared to *MobileNetV2*. Moreover, *EfficientNet* exposed the reduced computational efficiency, requiring 606 milliseconds per step. Therefore, while *EfficientNet* is a viable option, it is outperformed by *MobileNetV2*, which offers a more balanced, accurate, and efficient performance for this classification task, as summarized in Figure 10. Moreover, the ROC results for the *MobileNetV2* model are shown in Figure 11, while Figure 12 presents the ROC results for the *EfficientNet* model. Both models demonstrate excellent performance, as indicated by their ROC curves and AUC values. However, *EfficientNet* exhibits a slight edge over *MobileNetV2*, achieving perfect AUC scores for three classes (*Glass*, *Cardboard*, and *Paper*, with an area of 1), in contrast to *MobileNetV2*, which secured a perfect AUC score only for *Paper* (*area = 1*) and a slightly lower AUC score of 0.98 for Plastic. This suggests that, although *MobileNetV2* performs exceptionally well, *EfficientNet* may have a marginally improved ability to differentiate between positive and negative instances across certain classes. Therefore, both models are highly effective for this classification task, with *EfficientNet* showing a slight advantage in AUC scores across a broader range of classes. This analysis aligns with the earlier findings from the training and testing phases, offering a comprehensive perspective on the models' performance.

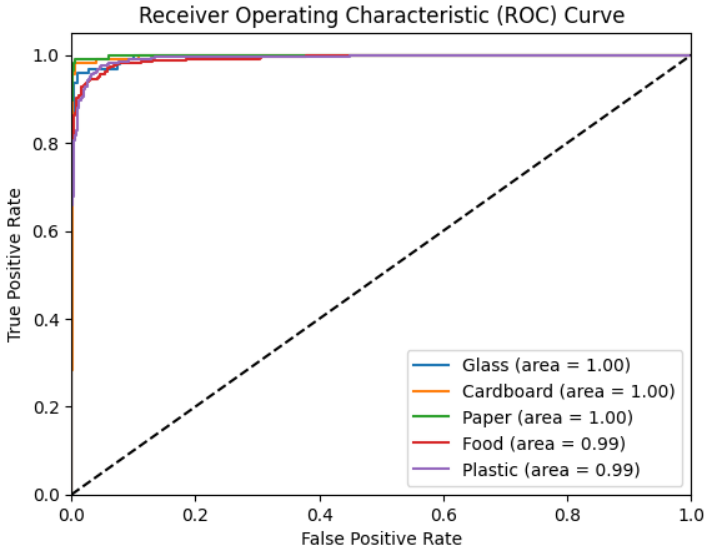

Fig. 11. ROC Curve for *MobileNetV2* Model

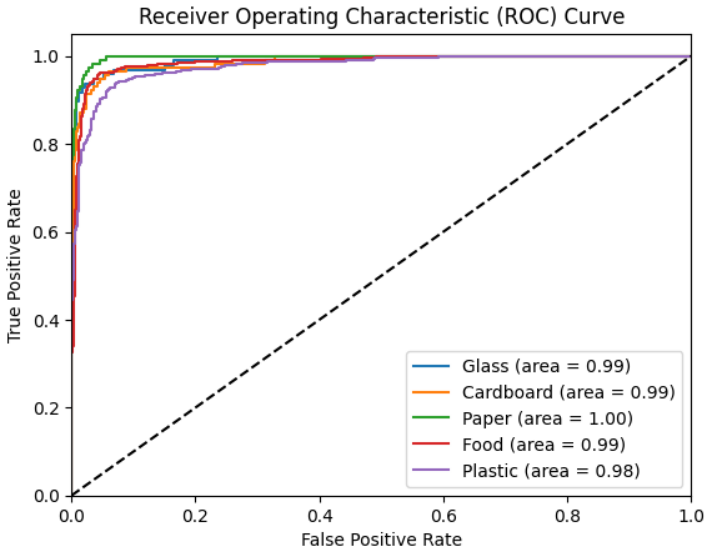

Fig. 12. ROC Curve for EfficientNet Model

4.3 Limitations of the Study

A limitation of this study pertains to the performance of the object detection model. While the YOLOv8-*s* model established a promising baseline, its resulting mean Average Precision (mAP 50-95) score of approximately 0.33 indicates a moderate level of accuracy. While functional for initial identification, this performance level suggests that the model may still be prone to missing particular objects or generating inaccurate bounding boxes in complex, real-world scenarios. Therefore, while effective as a proof of concept, further fine-tuning would be necessary to achieve the high reliability required for a production-grade system. Furthermore, specific performance variations were identified in the *MobileNetV2* classification model. Upon detailed error analysis, the lower precision of 76% for the *Glass* class can be attributed to a challenging failure mode rooted in visual ambiguity. Analysis of the confusion matrix shown in Figure 8 reveals that this is primarily caused by misclassifications where transparent plastic objects are incorrectly identified as *glass* due to inherent similarities such as transparency and reflections.

The overall training dynamics were also scrutinized to assess model stability. While minor fluctuations are present in the validation curves, these are attributed to the high variance in the composition of validation data batches rather than systemic instability. An early stopping mechanism was implemented with a patience of 3 epochs to proactively mitigate potential overfitting, monitored on the *val_loss* metric. The model's ability to complete its full training cycle without being halted confirms that significant overfitting was avoided and that the model successfully converged. Another significant limitation is the potential domain gap between the curated dataset used for training and the conditions of a real-world operational environment. The dataset primarily consists of images featuring relatively clean, well-defined objects, which may not fully represent the complexity of an actual waste stream. Real-world scenarios often involve cluttered, overlapping, and partially occluded items, as well as deformed or soiled objects. Consequently, the performance metrics reported in this study should be considered an idealized benchmark. The model's performance is anticipated to degrade when deployed in a more chaotic, industrial setting without domain-specific fine-tuning on more representative data.

Therefore, while the study highlights the computational efficiency of the models, the applicability for real-time systems requires careful contextualization. The *MobileNetV2* classifier, for instance, achieved an inference speed of 229 *ms/step*, translating to approximately 4.3 frames per second (FPS). Although this processing speed is sufficient for many human-interactive applications, it may not meet the demands of high-throughput industrial scenarios, such as a rapidly moving sorting conveyor, where significantly higher frame rates are often essential. Therefore, the real-time capability of the proposed system is application-dependent, and further optimization or deployment on more powerful hardware may be necessary to satisfy the requirements of high-speed industrial automation.

5. Conclusion

This study thoroughly evaluated the efficacy of the YOLOv8s algorithm for object detection and conducted a comparative analysis of two CNN architectures: *MobileNetV2* and *EfficientNet*, in the context of waste image classification. The findings reveal that YOLOv8s excels in detection tasks, achieving an impressive mean Average Precision (mAP) of 61% at a 0.5 IoU threshold. A notable trade-off between accuracy and computational efficiency was identified in the classification tasks. *MobileNetV2* surpassed *EfficientNet* in accuracy, obtaining 98% compared to *EfficientNet*'s 93%. Moreover, *MobileNetV2* exhibited greater efficiency, requiring only 229 *ms*/processing step, in contrast to *EfficientNet*'s 606 *ms*. Both models showcased robust performance in ROC analysis, with *MobileNetV2's*

strengths particularly notable. As a result, *MobileNetV2* emerges as the more balanced architecture for this classification task, offering an excellent combination of high accuracy and quicker processing speed. The primary aim of this study was to establish a foundational analysis to identify the most effective models. While this work sets a clear performance baseline, a comprehensive investigation into real-world deployment logistics was considered beyond the scope of this research. Future studies should build on these findings to explore critical deployment factors, such as the specific memory requirements of the models, their inference throughput in batch processing scenarios, and the overall scalability necessary for integration into large-scale waste management systems. For immediate development, it is recommended to use higher-quality images and ensure consistent object placements to further enhance classification results. Additionally, to expand the scope of architectural comparisons, future research could incorporate other modern, mobile-optimized CNN architectures to enhance the generalizability of the findings presented in this study.

References

[1] I. M. D. Adnyana, K. Mahendra, and S. Meesam Raza, "The Importance of Green Education in Indonesia: An Analysis of Opportunities and Challenges," *Education Specialist*, vol. 1, pp. 61–68, Nov. 2023, doi: 10.59535/es.v1i2.168.

[2] I. Tyagi, R. R. Karri, N. M. Mubarak, and M. H. Dehghani, "Emerging pollutants in the aqueous solution," in *Sustainable Remediation Technologies for Emerging Pollutants in Aqueous Environment*, 2023. doi: 10.1016/B978-0-443-18618-9.00010-3.

[3] G. Herrera-Franco, B. Merchán-Sanmartín, J. Caicedo-Potosí, J. B. Bitar, E. Berrezueta, and P. Carrión-Mero, "A systematic review of coastal zone integrated waste management for sustainability strategies," 2024. doi: 10.1016/j.envres.2023.117968.

[4] P. Gao, Z. Han, and F. Wan, "Big data processing and application research," in *Proceedings - 2020 2nd International Conference on Artificial Intelligence and Advanced Manufacture, AIAM 2020*, 2020. doi: 10.1109/AIAM50918.2020.00031.

[5] A. Cuzzocrea and E. Damiani, "Privacy-Preserving Big Data Exchange: Models, Issues, Future Research Directions," in *Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021*, 2021. doi: 10.1109/BigData52589.2021.9671686.

[6] V. Wiley and T. Lucas, "Computer Vision and Image Processing: A Paper Review," *International Journal of Artificial Intelligence Research*, vol. 2, no. 1, 2018, doi: 10.29099/ijair.v2i1.42.

[7] J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, "Deep learning in computer vision: A critical review of emerging techniques and application scenarios," *Machine Learning with Applications*, vol. 6, 2021, doi: 10.1016/j.mlwa.2021.100134.

[8] A. Ahmed, A. S. Imran, A. Manaf, Z. Kastrati, and S. M. Daudpota, "Enhancing wrist abnormality detection with YOLO: Analysis of state-of-the-art single-stage detection models," *Biomed Signal Process Control*, vol. 93, 2024, doi: 10.1016/j.bspc.2024.106144.

[9] J. Da Wu and C. H. Chang, "Driver Drowsiness Detection and Alert System Development Using Object Detection," *Traitement du Signal*, vol. 39, no. 2, 2022, doi: 10.18280/ts.390211.

[10] B. Ma *et al.*, "Using an improved lightweight YOLOv8 model for real-time detection of multi-stage apple fruit in complex orchard environments," *Artificial Intelligence in Agriculture*, vol. 11, 2024, doi: 10.1016/j.aiia.2024.02.001.

[11] Y. Wang *et al.*, "VV-YOLO: A Vehicle View Object Detection Model Based on Improved YOLOv4," *Sensors*, vol. 23, p. 3385, Nov. 2023, doi: 10.3390/s23073385.

[12] P. K. Samanta, A. Basuli, N. K. Rout, and G. Panda, "Improved Breast Cancer Detection from Ultrasound Images Using YOLOv8 Model," in *2023 IEEE 3rd International Conference on Applied Electromagnetics, Signal Processing, and Communication, AESPC 2023*, 2023. doi: 10.1109/AESPC59761.2023.10390341.

[13] S. Kato, A. Oshita, T. Kubo, and M. Todai, "Classification of Steel Microstructure Image Using CNN," in *Lecture Notes on Data Engineering and Communications Technologies*, vol. 189, 2024. doi: 10.1007/978-3-031-46970-1_6.

[14] A. Angdresey, L. Sitanayah, and E. Pantas, "Comparison of the Convolutional Neural Network Architectures for Traffic Object Classification," in *Proceedings - 2023 10th International Conference on Computer, Control, Informatics and its Applications: Exploring the Power of Data: Leveraging Information to Drive Digital Innovation, IC3INA 2023*, 2023. doi: 10.1109/IC3INA60834.2023.10285791.

[15] R. Indraswari, R. Rokhana, and W. Herulambang, "Melanoma image classification based on MobileNetV2 network," *Procedia Comput Sci*, vol. 197, pp. 198–207, 2022, doi: https://doi.org/10.1016/j.procs.2021.12.132.

[16] Q. Abbas, Y. Daadaa, U. Rashid, M. Z. Sajid, and M. E. A. Ibrahim, "HDR-EfficientNet: A Classification of Hypertensive and Diabetic Retinopathy Using Optimize EfficientNet Architecture," *Diagnostics*, vol. 13, no. 20, 2023, doi: 10.3390/diagnostics13203236.

[17] M. Shao *et al.*, "Comparison of EfficientNet-B0 and Res2Net Models for Distinguishing Malignant and Benign Pulmonary Nodules," May 29, 2024. doi: 10.21203/rs.3.rs-4397415/v1.

[18] A. Angdresey, I. Y. Kairupan, and A. G. Mongkareng, "Leveraging Convolutional Neural Networks for Multiclass Waste Classification," *Journal of Applied Informatics and Computing*, vol. 9, no. 4, pp. 1137–1145, Aug. 2025, doi: 10.30871/jaic.v9i4.9373.

[19] N. Sharma, R. Sharma, and N. Jindal, "Machine Learning and Deep Learning Applications-A Vision," *Global Transitions Proceedings*, vol. 2, no. 1, 2021, doi: 10.1016/j.gltp.2021.01.004.

[20] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," 2021. doi: 10.1007/s42979-021-00815-1.

[21] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," *Multimed Tools Appl*, vol. 82, no. 6, 2023, doi: 10.1007/s11042-022-13644-y.

[22] M. R. Sholahuddin *et al.*, "Optimizing YOLOv8 for Real-Time CCTV Surveillance: A Trade-off Between Speed and Accuracy," *Jurnal Online Informatika*, vol. 8, no. 2, 2023, doi: 10.15575/join.v8i2.1196.

[23] Y. Li, Q. Fan, H. Huang, Z. Han, and Q. Gu, "A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition," *Drones*, vol. 7, no. 5, 2023, doi: 10.3390/drones7050304.

[24] M. M. Krishna, M. Neelima, M. Harshali, and M. V. G. Rao, "Image classification using Deep learning," *International Journal of Engineering and Technology(UAE)*, vol. 7, 2018, doi: 10.55041/ijsrem27484.
[25] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00444-8.
[26] P. Beleya, "Challenges in Enhancing Solid Waste Management towards Sustainable Environment: Local Council Perspectives," Nov. 2019.
[27] A. Howard *et al.*, "Searching for mobileNetV3," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-October, pp. 1314–1324, Oct. 2019, doi: 10.1109/ICCV.2019.00140.
[28] M. Basheer Ahmed *et al.*, "Deep Learning Approach to Recyclable Products Classification: Towards Sustainable Waste Management," *Sustainability*, vol. 15, p. 11138, Nov. 2023, doi: 10.3390/su151411138.
[29] H. Abdu and M. H. Mohd Noor, "A Survey on Waste Detection and Classification Using Deep Learning," *IEEE Access*, vol. PP, p. 1, Nov. 2022, doi: 10.1109/ACCESS.2022.3226682.
[30] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

Authors' Profiles

Apriandy Angdresey is an Assistant Professor in the Department of Informatics Engineering at Universitas Katolik De La Salle Manado, Indonesia. He earned his Bachelor of Engineering in Informatics Engineering from Universitas Katolik De La Salle Manado in 2013 and a Master of Science in Computer Science and Information Engineering from Chang Gung University, Taiwan, in 2017. He has authored and co-authored several peer-reviewed journal articles and conference papers indexed by Scopus. His research interests include big data analysis, computational intelligence, and applied machine learning.

Indah Yessi Kairupan is a Lecturer in the Department of Informatics Engineering at Universitas Katolik De La Salle Manado, Indonesia. She received her B.Eng. in Informatics Engineering from the same university in 2012 and her master's degree in Computer Science and Information Engineering from Chang Gung University, Taiwan, in 2016. She has published several peer-reviewed journal articles and conference papers indexed in Scopus. Her research interests focus on decision support systems, embedded software systems, and disaster mitigation.

Andre Gabriel Mongkareng earned his Bachelor of Engineering degree in Informatics Engineering from Universitas Katolik De La Salle Manado, Indonesia, in 2024. During his undergraduate studies, he participated in the Bangkit Program - a prestigious career readiness initiative supported by the Indonesian government in collaboration with Google and Coursera. He is currently employed as a Relationship Officer at Maybank Indonesia. His research interests focus on the applications of machine learning in real-world systems and services.

How to cite this paper: Apriandy Angdresey, Indah Yessi Kairupan, Andre Gabriel Mongkareng, "Web-Based Waste Detection Using YOLOv8 and Classification Performance Comparison: MobileNet and EfficientNet", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.17, No.6, pp. 1-18, 2025. DOI:10.5815/ijigsp.2025.06.01