Edu Komputika 13 (1) (2026)

Edu Komputika Journal

https://journal.unnes.ac.id/journals/edukom/

Performance Analysis of Convolutional Neural Network Methods Using VGG16 and YOLOv8n for Bottle Waste Sorting in Computer Vision Applications

Akbar Sujiwa✉, Nailul Hasan, Fajar Timur, and Reffany Choiru Rizkiarna

Department of Physics, Faculty of Engineering and Science, Universitas Pembangunan Nasional "Veteran" Jawa Timur, Surabaya, 60294, Indonesia

Article Info

Article History:
Received: 4 February 2026
Revised: 8 August 2026
Accepted: 14 August 2026

Keywords:
Computer Vision, Convolutional Neural Network, VGG16, Waste Sorting, YOLOv8n

Abstract

This study examines the performance of two artificial intelligence models—VGG16 and Yolov8n—in sorting plastic bottles using computer vision. The objective is to individually assess the classification and detection performance of these models with constrained computational resources and training data. The dataset consists of 300 original images for two classes (bottle and other), and is split into 210 training, 45 validation and 45 test images. The images were taken in different lighting conditions and orientations to simulate the real waste sorting situation . Both models were trained and evaluated on CPU based hardware to simulate a constrained computing environment. The VGG16 was evaluated using classification metrics like accuracy, precision, recall, and F1-score, while the YOLOv8n was evaluated using object detection metrics like precision, recall, F1-score, mAP@0.5, and frame processing speed (FPS). The accuracy of the VGG16 model was 91% on the test set. The mAP@0.5 of YOLOv8n was 0.56 with an average processing speed of 47.12 FPS, while the average processing speed of VGG16 was 5.17 FPS. These results indicate that VGG16 had a good performance on image-level classification, while YOLOv8n had a higher processing efficiency and better object-localization performance in the studied conditions. Further evaluation on embedded hardware is required to establish the suitability of YOLOv8n for practical real-time waste-sorting applications.

✉ Correspondence Address:
Fakultas Teknik dan Sains, Universitas Pembangunan Nasional "Veteran" Jawa Timur, Jl. Raya Rungkut Madya No. 1, Gunung Anyar, Surabaya, Jawa Timur 60294.
E-mail: akbarsujiwa.ft@upnjatim.ac.id

ISSN 2252-6811
E-ISSN 2599-297X

INTRODUCTION

Waste management is a pressing environmental issue, particularly for plastic and glass bottles, due to their high volume and the environmental impact if not handled properly. Manual waste sorting is still common, but it is prone to error, requires significant time and effort, and offers inconsistent accuracy (Aberger et al., 2025). Even in some areas of Indonesia, plastic waste is not properly sorted, resulting in waste accumulation in landfills and serious environmental pollution, both on land and in water (Dewi, 2022). This situation disrupts ecosystems, such as clogged waterways and increased flooding risks, and poses a threat to marine life from microplastics carried into the ocean. In 2024, plastic waste in Indonesia is projected to account for around 13.98% of total national waste generation, with an estimated volume of 9.9 million tons (Nizar et al., 2025). With such a high volume of plastic waste, Indonesia needs to utilize an integrated waste management system from upstream to reduce the potential for environmental pollution. Upstream management encompasses effective waste reduction, sorting, and recycling before it reaches landfills. This approach focuses not only on waste processing but also on preventing waste generation through increased public awareness, the implementation of circular economy-based policies, and the use of modern technology.

One strategic step that can be implemented is automating waste sorting systems using computer vision and artificial intelligence (AI) technology. This approach allows for the rapid and accurate identification and classification of waste types, particularly plastic bottles, without relying on human labor (Kumar et al., 2024; Lin et al., 2024; Yang et al., 2024). Computer vision technology and Convolutional Neural Network (CNN) methods can be used to automatically identify, classify, and sort waste types based on images or videos (Altikat et al., 2022; Sujiwa et al., 2025). CNNs work by extracting visual features such as shape, color, and texture from images, enabling them to recognize various materials, including plastic, paper, glass, and metal, with high accuracy. In the context of waste management, CNN can be applied to smart waste sorting systems to replace the time-consuming and error-prone manual sorting process (Shi et al., 2021).

However, using CNNs presents its own challenges, particularly reduced accuracy when training data is insufficient. CNN relies heavily on the diversity and quality of the dataset to optimally recognize visual patterns. When training data is limited or imbalanced between classes, the model tends to overfit, which means it adapts too closely to the training data and fails to generalize to new data (Ian Goodfellow et al., 2016). Furthermore, CNNs are also sensitive to variations in lighting, shooting angles, and backgrounds. In the context of waste sorting, non-uniform image conditions, such as reflections on plastic bottles or overlapping piles of trash, can reduce classification accuracy (Peng, 2025). Another challenge is the need for significant computational resources, especially when training models with deep and complex architectures. For instance, conventional CNN models such as VGG16, while effective in image classification tasks, require large amounts of labeled data and high computational power, making them less practical for real-time waste sorting applications or for deployment on low-power devices (Suganda Girsang et al., 2023).

While extensive studies have focused on high-end GPU-based performance, benchmarking of low-cost, CPU-based edge deployment for small-scale recycling applications remains limited. This study addresses this gap by providing empirical evidence on the trade-offs between conventional classification models and modern single-stage detectors under constrained computing environments. Although the dataset size is limited, transfer learning and data augmentation were employed to improve training feasibility; however, these techniques cannot completely eliminate the risk of overfitting. Therefore, this study serves as a preliminary evaluation of deep learning approaches for plastic waste sorting. VGG16 and YOLOv8n were compared based on the same application objective, namely plastic bottle recognition, while considering their different capabilities in image classification, object detection, and computational efficiency.

To address these limitations, various approaches have been developed, such as data augmentation, transfer learning, and the implementation of more efficient object detection models like YOLO (You Only Look Once). These approaches allow models to recognize objects directly from images without requiring prior segmentation and can improve accuracy even with limited datasets (Redmon et al., 2015). YOLO is a rapidly developing object detection algorithm, consisting of several versions ranging from YOLOv1 to YOLOv8, where each version brings improvements in terms of speed, efficiency, and detection accuracy. In this study, YOLOv8n (nano) was used because it has a small model size and low computational requirements, but is still able to provide good detection results. The model was selected to facilitate adaptation

for low-power computing platforms such as the Raspberry Pi, thereby supporting the efficient deployment of computer vision-based waste sorting systems in real-world environments. This study also aims to evaluate the extent to which improvements in accuracy, detection speed, and computational efficiency can be achieved compared to conventional CNN methods in computer vision-based waste sorting applications.

RESEARCH METHODS

This study employs a comparative experimental design to analyze which model—conventional CNN (VGG16) or YOLOv8n—performs better when trained on a small-scale dataset of plastic bottle images. The primary objective is to evaluate the effectiveness, accuracy, and efficiency of both models under limited data conditions and constrained computational resources, particularly in recognizing visually similar bottles with varying brands and sizes.

A. Research Design

The research process began with collecting a dataset consisting of images of several plastic bottles of different brands and sizes. Data was collected manually by photographing objects from various angles and lighting conditions to increase dataset diversity. Data preprocessing, such as image resizing, color normalization, and data augmentation (rotation, flipping, and contrast and brightness adjustments), was performed to make the model more robust to image variations. The overall research workflow is illustrated in the flow diagram shown in Figure 1.

Then the two models are trained on the same dataset. The first model uses a conventional CNN as a baseline model. The second model uses YOLOv8n, which has a more efficient and faster architecture. We chose YOLOv8n because it can detect objects in real-time with low computational complexity, making it a potentially suitable model for further development on low-powered devices such as Raspberry Pi.

After training both models, we evaluated the performance using several key metrics: accuracy, precision, recall, F1-score, and detection speed (FPS). We also compared the utilization of the computation resource (CPU and memory) to evaluate the system efficiency. The results of this evaluation will show how much YOLOv8n can improve the performance of the garbage detection system compared to a classic CNN.

The final stage of the research involved evaluating the performance of both models by measuring metrics such as accuracy, precision, recall, F1-score, and detection speed (FPS). These test results were used to determine the extent to which YOLOv8n improved performance compared to a classic CNN in detecting and classifying plastic bottles of different brands and sizes.

B. Data Acquisition

The data acquisition process in this study was conducted manually by collecting images of several types of plastic bottles of various brands and sizes. Each bottle was photographed using a high-resolution camera from various angles and lighting conditions to increase data diversity. This approach aims to ensure the model can recognize objects well despite variations in orientation, light intensity, and background.

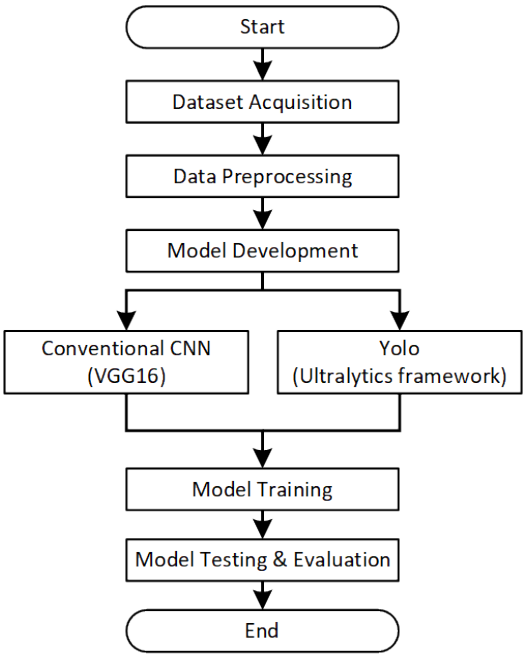

Figure 1. Research workflow

Before the data is used for training, a preprocessing stage is performed to standardize the image size and quality. Each image is resized to a fixed dimension, namely 224 × 224 pixels for conventional CNN models such as VGG16 and 640 × 640 pixels for the YOLOv8n model to match the input structure of each architecture. In addition, color normalization is performed by changing the pixel value range to a scale of [0,1] to accelerate the convergence process and reduce sensitivity to lighting differences.

The dataset structures used in this study differ according to the type of model applied,

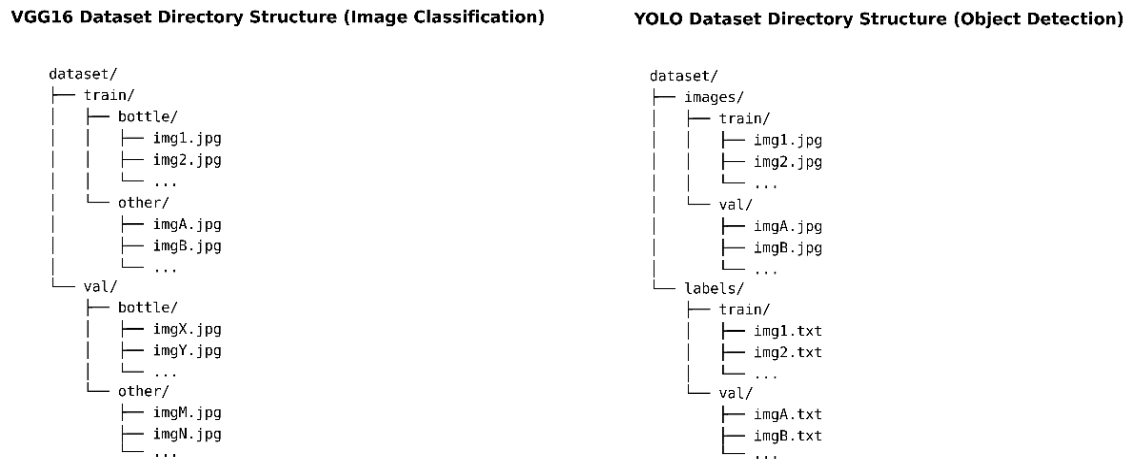

Figure 2. Dataset structure of VGG16 and YOLOv8 models

namely VGG16 and YOLOv8. For the VGG16 model, the dataset is organized into a relatively simple folder hierarchy, with images grouped into class-specific directories within the train and validation folders. This structure does not require any additional label files, as the folder names themselves represent the class categories.

In contrast, the YOLOv8 model uses a more complex dataset structure. It has image folders and a separate labels directory that contains text (.txt) files that define the bounding box coordinates and object class for each image. So the YOLO dataset is divided into two parts: images and labels, whereas the VGG16 dataset uses only the class-based folder organization. The two dataset structures are shown in Figure 2. The VGG16 dataset is simpler, while the YOLO dataset is more complex because of the label files needed for object detection.

To increase the variety and expand the training data set, data augmentation was done by rotation, horizontal flipping, and adjustment of brightness and contrast. These augmentation techniques were used to help the model generalize to different image conditions and to prevent overfitting, especially due to the small data set. Moreover, all images were labeled according to the object category, i.e., different types of plastic bottles, using tools such as LabelImg for YOLOv8n training and a classification directory system for conventional CNN models.

The dataset was pre-processed and split into three sets, 70% training data, 15% validation data, and 15% testing data. This division was done so that the model could not only recognize data it had seen during training but also generalize to new data it had not seen before. Through this systematic acquisition and pre-processing process, the hope is that the CNN and YOLOv8n models can be optimally trained to recognize and classify plastic bottles based on their shape and visual characteristics.

C. Data Augmentation and Cross-Validation Strategy

An ablation study was performed to evaluate the effect of preprocessing data in a limited training data scenario by training both models with and without the data augmentation pipeline (random rotations, flips, and brightness adjustments). Moreover, in an effort to ensure the statistical reliability and avoid the bias from a single random train-test split, a 5-fold cross-validation strategy was applied to the pre-processed dataset, and the final metrics are an average over all the folds. (Vabalas et al., 2019).

D. Model Implementation

The obtained and pre-processed data were then used to train two different models, a traditional CNN (VGG16) and YOLOv8n, to compare their performance in detecting and classifying plastic bottle waste in this step.

We chose the VGG16 model as a representative of the classic CNN architecture, since it has a simple but effective structure, consisting of 13 convolutional and 3 fully connected layers. This model served as a baseline to assess the ability of conventional CNNs to recognize visual patterns in plastic bottle images. Training was performed using transfer learning using pre-trained weights from the ImageNet dataset, allowing the model to adapt to new datasets with relatively limited data. Training parameters, such as learning rate and batch size, were adjusted to achieve model convergence without overfitting.

Meanwhile, YOLOv8n (You Only Look Once version 8 - nano) is used as a modern, more efficient, and faster object detection approach. This model has a lightweight architecture while maintaining a high level of accuracy in detecting objects directly in a single stage (single-stage detection). YOLOv8n was trained using the same dataset as VGG16, with labels determined through the bounding box annotation process. Training was performed using a standard configuration with the Adam optimizer and loss functions that include objectness loss, classification loss, and bounding box regression loss.

Furthermore, YOLOv8n was chosen for its suitability for further development on low-computing devices, such as the Raspberry Pi, enabling the implementation of a real-time automated waste-sorting system in the field. The model was implemented in Python using PyTorch and the Ultralytics YOLO framework.

E. Testing and Evaluation

The testing and evaluation phase was conducted to assess the model's performance in recognizing and classifying plastic bottles based on the training results. The main objective of this phase was to compare the accuracy, detection speed, and computational efficiency between a conventional CNN model (VGG16) and a modern object detection model (YOLOv8n).

A separate test set, distinct from the training and validation data, was used for the testing phase, to ensure that the model was tested on completely unseen data. Metrics of accuracy, precision, recall, and F1-score were calculated to evaluate the VGG16 model. These metrics describe the performance of the model on the classification of plastic bottle types. We also employed a confusion matrix to evaluate the ability of the model to differentiate each class based on its visual features.

YOLOv8n, in contrast, was evaluated on the mean Average Precision (mAP) metric at an IoU of 0.5. Inference time for each image was also measured to evaluate real-time detection speed. All tests were run on a computer with an Intel Core i5-7300HQ processor, 16 GB RAM, without an external GPU, so all the training and inference processes were done using the CPU resources.

Thus, we try to simulate computing conditions closer to those of low-power devices and, therefore, to get a realistic picture of the model performance before its further development for its implementation on embedded systems such as the Raspberry Pi.

The test results will compare conventional CNNs and YOLOv8n in terms of accuracy, inference speed, and resource efficiency. This analysis will enable the study to assess the extent to which YOLOv8n can improve the performance of a computer vision-based waste-sorting system compared with the classic CNN approach.

One limitation of this study is the relatively small dataset, which may introduce uncertainty and limit model generalizability. Sample dependence can also arise from images of the same physical bottles. Environmental variables, such as lighting and object orientation, were not quantitatively recorded. Besides, VGG16 and YOLOv8n perform different tasks; hence, a direct metric comparison is not appropriate. The results should therefore be considered as a preliminary assessment under the conditions investigated. Future work should use larger, more diverse datasets and perform condition-specific analyses, uncertainty estimation, and augmentation ablation to provide more robust evidence of model stability and generalization.

RESULT AND DISCUSSION

A. Model Performance Evaluation

As shown in Figure 3, there is a significant difference in the computational process efficiency of the two models. The VGG16 model took about 1879.9 seconds (31.33 minutes) to finish 20 training epochs on a CPU, while the YOLOv8n model completed the same in 905.5 seconds (15.09 minutes). This indicates that YOLOv8n is twice as fast to train as VGG16, which requires twice as much training time as YOLOv8n. The difference in training speed can be attributed primarily to architectural design: VGG16 is a deep convolutional neural network with a large number of parameters and dense layers, which increases computational load and memory usage. YOLOv8n (You Only Look Once v8 nano) is built with a lightweight and optimized structure, with efficient convolutional blocks and advanced backbone optimization that drastically cut down processing time with minimal accuracy loss.

From a practical perspective, the reduced training time of YOLOv8n offers remarkable advantages, especially when training resources are scarce or when the model needs to be retrained often, as in adaptive or real-time systems. VGG16, although it provides strong feature extraction and slightly higher classification accuracy, may not be suitable for applications requiring quick training due to its prolonged training time. Therefore, as demonstrated in Figure X, YOLOv8n provides a

superior balance between efficiency and performance, making it a more viable option for rapid deployment, experimentation, and real-world object detection scenarios where processing speed is a critical factor.

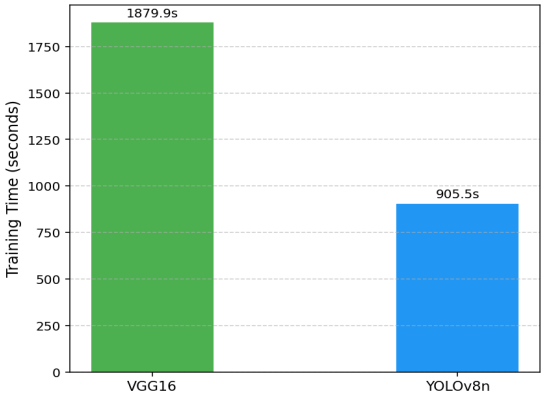

Figure 3. Training time comparison between VGG16 and YOLOv8n

As depicted in Figure 3, VGG16 required 1879.9 seconds to complete 20 epochs, whereas YOLOv8n finished in 905.5 seconds. This behavior is directly linked to the structural differences between the two architectures. VGG16 relies on sequential convolutional layers followed by dense fully connected layers containing millions of parameters, resulting in heavy matrix multiplications during backpropagation on a CPU. In contrast, YOLOv8n utilizes an optimized backbone and an efficient gradient flow via its CSP-structure, which substantially reduces computational overhead and accelerates convergence time without compromising learning stability.

Based on the performance results shown in Figure 4, VGG16 and YOLOv8n demonstrated distinct strengths for their respective tasks. VGG16 achieved an accuracy of 0.91, precision of 0.93, recall of 0.91, and F1-score of 0.91 on image-level classification, indicating consistent performance in plastic bottle recognition. In contrast, YOLOv8n, evaluated as an object detection model, achieved a precision of 0.82, a recall of 0.94, an F1-score of 0.88, and an mAP@0.5 of 0.56. The higher recall indicates that YOLOv8n detected a larger proportion of relevant objects, while its mAP@0.5 reflects its object localization and detection performance. These results are therefore interpreted separately according to the different tasks performed by each model rather than treating classification accuracy and detection mAP as equivalent metrics.

Overall, VGG16 outperformed YOLOv8n in accuracy, precision, and F1 score, making it more suitable for classification tasks under limited training data and computational constraints. Meanwhile, YOLOv8n's higher recall reflects its strength in object detection scenarios where identifying all possible instances is prioritized over classification precision. These results highlight the trade-off between accuracy-oriented and detection-oriented architectures, showing that while YOLOv8n is optimized for real-time, broad object detection, VGG16 remains more reliable for small-scale image classification, delivering stable, consistent performance under limited computational resources. The performance comparison in terms of processing speed demonstrates a significant difference between YOLOv8 and VGG16. As shown in the FPS summary, YOLOv8 achieved an average frame rate of 47.12 FPS, with a maximum of 85.11 FPS and a minimum of 1.38 FPS. This indicates that YOLOv8 is capable of performing near real-time detection under standard CPU-based conditions. In contrast, VGG16 recorded an average frame rate of only 5.17 FPS, with a relatively stable deviation (±0.30) and a maximum speed of 5.57 FPS. The substantial performance gap highlights YOLOv8's architectural optimization for real-time object detection, which employs a lighter backbone and parallel processing mechanisms.

B. Data Augmentation and Cross-Validation

An ablation study was conducted to evaluate the impact of data augmentation on the VGG16 model. Without augmentation, the model achieved an accuracy of 93.75%, precision of 100.00%, recall of 87.50%, and F1-score of 93.33%. After applying data augmentation, the accuracy increased to 95.00%, while the recall improved to 92.50%, and the F1-score increased to 94.87%. Although the precision slightly decreased from 100.00% to 97.37%, it remained high, indicating that the augmented model achieved a better balance between precision and recall. These results demonstrate that data augmentation enhances the generalization capability of VGG16 by exposing the model to a more diverse set of training samples.

The robustness of the VGG16 classifier was further evaluated using a 5-fold cross-validation strategy on the pre-processed dataset. The model achieved consistently high

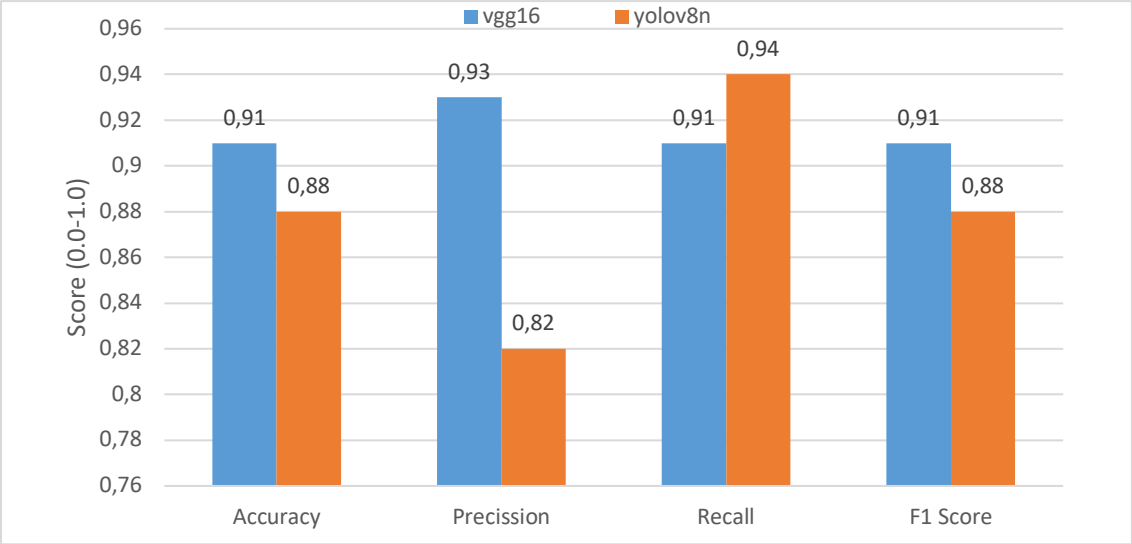

Figure 4. Performance comparison of accuracy, precision, recall, and F1 score between VGG16 and YOLOv8n models

performance across all folds, with accuracy values ranging from 98.33% to 100%. The average accuracy, precision, recall, and F1-score obtained were 99.33%, 99.35%, 99.33%, and 99.33%, respectively. These results indicate that the VGG16 model provides stable classification performance and maintains strong generalization capability across different data partitions.

C. Performance Evaluation under Real-World Conditions

Real-world testing reveals a highly significant difference in the obtained data, particularly regarding the interpretation of screen images when comparing VGG16 and YOLOv8n. In the VGG16 model, background misclassification occurs when the background is not included in the training dataset. False positive cases were also observed for VGG16 in some test scenarios. The model misclassified background regions as objects (e.g., bottles) even when there was no actual object in the frame (Figure 5). This could be due to VGG16 being a traditional CNN primarily built for classification. VGG16 does not possess contextual spatial information, and it treats the entire image as a whole rather than explicitly analyzing the boundary of the object. Consequently, variations in lighting, texture, or environmental background can lead to false detections or overfitting toward familiar background features from the training data.

In contrast, the YOLOv8n model was more robust and accurate in distinguishing between objects and backgrounds, as shown in Figure 6. This region-based detection mechanism enables it to analyze local image features and spatial relationships, making it more efficient in distinguishing background noise from actual objects. Thus, in scenes with complex or unfamiliar backgrounds, YOLOv8n generated fewer false positives. This strength highlights YOLOv8n's advantage in practical applications, particularly in real-time environments where background variability and dynamic lighting conditions are common.

Figure 6. Feature extraction result of yolov8n on image background

Overall, the real-world testing results suggest that VGG16 performs reliably under controlled, uniform conditions, but its accuracy declines when encountering unseen or cluttered backgrounds. Meanwhile, YOLOv8n exhibits stronger generalization capability, maintaining

consistent object detection performance even in diverse environmental contexts.

A significant gap in computational efficiency is observed when comparing the inference speeds of VGG16 (Figure 7) and YOLOv8 (Figure 8). As summarized in the FPS results, YOLOv8 achieved an average frame rate of 47.12 FPS, with a maximum of 85.11 FPS and a minimum of 1.38 FPS. The standard deviation is relatively higher (±10.90). Real-time processing capability was consistently maintained during the entire process. This confirms the model is optimized for rapid object detection tasks. In contrast, VGG16 demonstrated a significantly lower average frame rate of 5.17 FPS, with a narrow deviation range (±0.30), indicating stable but much slower inference performance. The maximum speed of VGG16 reached only 5.57 FPS, confirming its limitation for real-time applications.

This clear contrast indicates the architectural difference between the two models: YOLOv8n employs a lighter, parallelized detection system, enabling rapid inference—even though object localization is more computationally intensive. By contrast, VGG16 is a deep classification network that processes the entire image sequentially through its convolutional layers without any regional optimization, making it slower to execute. Therefore, the better object localization performance of YOLOv8n makes it more suitable for real-time object detection, in particular in embedded systems with a live video stream, whereas VGG16 performs better for static image classification under the experimental conditions.

Instant FPS values comparison between VGG16 and YOLOv8n is shown in Figure 8. The data reveals a significant difference in real-time performance between the two models, with YOLOv8n demonstrating a higher frame rate at each processing interval compared to VGG16.

Based on the summary table, the average FPS of YOLOv8n is approximately 47.12, whereas VGG16 achieves only around 5.17 FPS. This indicates that YOLOv8n performs roughly ten times faster than VGG16 when processing image frames. The substantial improvement in YOLOv8n's frame rate highlights its architectural efficiency, particularly its ability to perform detection and classification simultaneously within a single convolutional network pass.

In contrast, VGG16—originally designed for image classification rather than object detection—requires additional computational steps, leading to slower real-time performance. Consequently, YOLOv8n is more suitable for real-time object detection applications, especially those requiring high frame rates, such as video surveillance, robotics, and embedded vision systems.

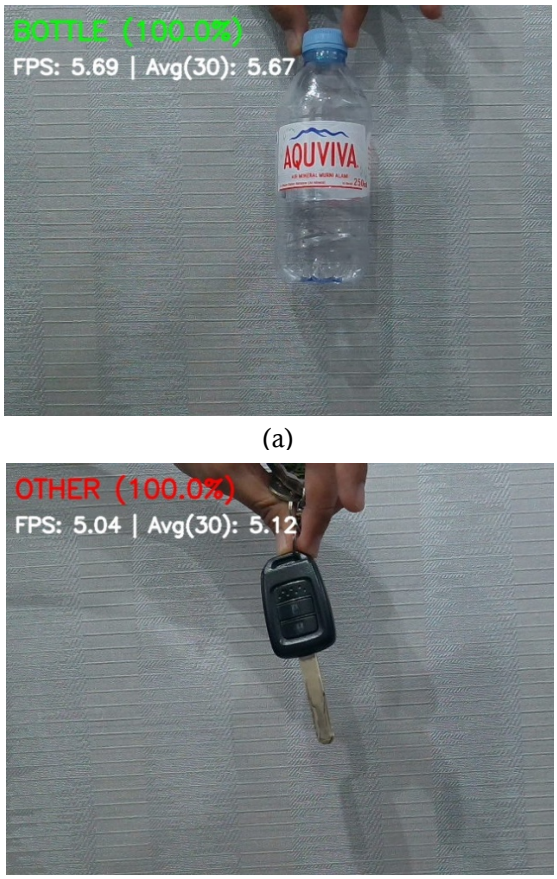

Figure 7. Results of object classification using VGG16: (a) image recognized as a bottle, and (b) image recognized as other (non-bottle) class.

CONCLUSION

Based on the experiments conducted using a small-scale plastic bottle dataset, VGG16 achieved an image-level classification accuracy of 91% on the evaluated test set, with a precision, recall, and F1-score of 0.93, 0.91, and 0.91, respectively. These results indicate strong image-level classification performance under the experimental conditions. Meanwhile, YOLOv8n, evaluated as an object detection model, achieved a precision of 0.82, a recall of 0.94, an F1-score of 0.88, and an mAP@0.5 of 0.56. YOLOv8n also achieved an average processing speed of 47.12 FPS, compared with 5.17 FPS for VGG16 on the specified CPU, demonstrating higher processing efficiency.

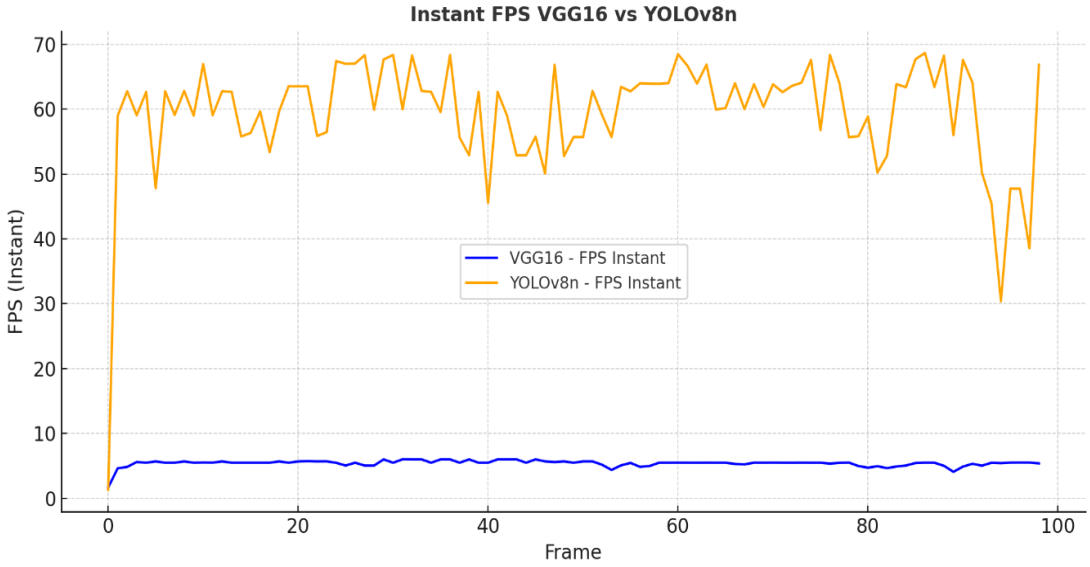

Figure 8. Comparison of instant FPS between VGG16 and YOLOv8n

The results should be interpreted within the scope of the dataset, test conditions, and hardware used in this study. The assessment of model robustness and generalization is limited by the relatively small dataset, lack of quantitatively controlled lighting and orientation conditions, and the possible dependence of images from the same physical bottles. Moreover, the classification metrics and object detection metrics are the results of different evaluations and thus should not be taken as directly equivalent. Hence, the results show the performance of each model for its own task rather than the overall superiority of one architecture over the other.

Future work will be to evaluate the models with larger and more diverse datasets with controlled lighting and orientation conditions and repeated experiments to better assess model stability and generalization. For future work, the implementation of YOLOv8n on Raspberry Pi or other embedded hardware will also be studied, with a direct evaluation of the processing speed and detection performance on the target hardware before making any claims about embedded-system suitability.

REFERENCES

Aberger, J., Shami, S., Häcker, B., Pestana, J., Khodier, K., & Sarc, R. (2025). Prototype of AI-powered assistance system for digitalisation of manual waste sorting. *Waste Management*, *194*, 366–378. https://doi.org/10.1016/J.WASMAN.2025.01.027

Altikat, A., Gulbe, A., & Altikat, S. (2022). Intelligent solid waste classification using deep convolutional neural networks. *International Journal of Environmental Science and Technology*, *19*(3), 1285–1292. https://doi.org/10.1007/S13762-021-03179-4/METRICS

Dewi, N. M. N. B. S. (2022). Studi literatur dampak mikroplastik terhadap lingkungan. *Sosial Sains Dan Teknologi*, *2*(2), 239–250.

Ian Goodfellow, Aaron Courville, & Yoshua Bengio. (2016). *Deep Learning*. 1–23. http://www.deeplearningbook.org

Kumar, M., Kumar, S., Luhach, A. K., & Tiwari, A. (2024). Advancements in Waste Segregation Through Machine Learning and Integrating AI for Sustainable Waste Management. *Proceedings of the 2024 13th International Conference on System Modeling and Advancement in Research Trends, SMART 2024*, 328–334. https://doi.org/10.1109/SMART63812.2024.10882258

Lin, Y. H., Mao, W. L., & Fathurrahman, H. I. K. (2024). Development of intelligent Municipal Solid waste Sorter for recyclables. *Waste Management*, *174*, 597–604. https://doi.org/10.1016/J.WASMAN.2023.12.040

Nizar, M., Putra, A., Zahrani, N. A., Zahra, T. A., Bella, B. C., Hariyadi, A. G., Fadhila, D. S., Akrom, S., Abiyyu, A., Rini, R., Firdausi, K., Justicio, M. N., Kamalul

Albar, A., & Firmansyah, P. (2025). Sampah Plastik sebagai Ancaman terhadap Lingkungan. *Aktivisme: Jurnal Ilmu Pendidikan, Politik Dan Sosial Indonesia*, *2*(1), 154–165. https://doi.org/10.62383/AKTIVISME.V2I1.725

Peng, S. (2025). Multi-object extraction technology for complex background based on faster regions-CNN algorithm in the context of artificial intelligence. *Service Oriented Computing and Applications*, *19*(1), 15–27. https://doi.org/10.1007/S11761-024-00401-2/METRICS

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Shi, C., Tan, C., Wang, T., Wang, L., Alwaeli, M., Pizó, J., & Gołaszewska, M. (2021). A Waste Classification Method Based on a Multilayer Hybrid Convolution Neural Network. *Applied Sciences 2021, Vol. 11, Page 8572*, *11*(18), 8572. https://doi.org/10.3390/APP11188572

Suganda Girsang, A., Dharma Saputra, A., & Yanrie, V. (2023). Performance Comparison between VGG16 and Inception V3 for Organic Waste and Recyclable Waste Classification. *International Journal of Intelligent Systems and Applications in Engineering*, *11*(2), 557–563. https://ijisae.org/index.php/IJISAE/article/view/2711

Sujiwa, A., Hasan, N., & Aruan, N. M. (2025). TensorFlow Based AI Training for Translating Indonesian Sign Language (BISINDO). *Best : Journal of Applied Electrical, Science and Technology*, *7*(1), 29–34. https://doi.org/10.36456/BEST.VOL7.NO1.10312

Vabalas, A., Gowen, E., Poliakoff, E., & Casson, A. J. (2019). Machine learning algorithm validation with a limited sample size. *PLOS ONE*, *14*(11), e0224365. https://doi.org/10.1371/JOURNAL.PONE.0224365

Yang, T., Yang, J., Fang, H., Ji, T., & Chen, W. (2024). Development of intelligent waste sorting system of low-value recyclable waste in Xiamen. *Proceedings of the Institution of Civil Engineers - Waste and Resource Management*, *177*(3), 154–166. https://doi.org/10.1680/JWARM.23.00010