Journal of Applied Informatics and Computing (JAIC)
Vol.10, No.3, June 2026, pp. 2287~2294
e-ISSN: 2548-6861

Benchmarking YOLOv12 Variants for Indonesian Traditional Cuisine Detection

Fauzan Firdaus [1]*, Lidya Ningsih [2], Aminah Indahsari Marsuki [3]***, Angel Metanosa Afinda [4]*****
* Software Engineering Study Program, School of Computing, Telkom University, Bandung
** Informatics Study Program, School of Computing, Telkom University, Bandung
*** Telecommunication Engineering Study Program, School of Electrical Engineering, Telkom University, Bandung
**** Computer Engineering Study Program, School of Electrical Engineering, Telkom University, Bandung
fauzanfirdausff@telkomuniversity.ac.id [1], telulidyaningsih@telkomuniversity.ac.id [2], aminahindahsarim@telkomuniversity.ac.id [3], angelmetanosa@telkomuniversity.ac.id [4]

Article Info

Article history:

Received 2026-03-04
Revised 2026-04-27
Accepted 2026-05-05

Keyword:

YOLOv12,
Object Detection,
Indonesian Cuisine,
Food Detection,
Benchmark Analysis.

ABSTRACT

YOLOv12 is one of the latest YOLO versions currently. Several studies have proven that YOLOv12 has better performance compared to previous versions. YOLOv12 itself has five model variants based on its architectural complexity, namely nano, small, medium, large and extra larges. This study tests the performance of YOLOv12 model variants (n, s, m, l, x) for traditional Indonesian culinary detection using a domain-specific object detection dataset. The dataset contains 718 images with 720 bounding-box instances annotated across 20 culinary classes, divided into 418/150/150 images for training/validation/testing. Data processing was performed in Roboflow with automatic orientation and stretching resizing to 640×640, while the training split was enriched using augmentation (horizontal and vertical flips) to increase sample diversity. All YOLOv12 variants were trained with the same configuration and environment, for 50 epochs using the Ultralytics framework with default hyperparameters on an NVIDIA A100-SXM4 80GB GPU. On the validation set, all variants achieved high detection accuracy (mAP@0.5 = 0.985–0.991), while differences emerged under a more stringent localization criterion (mAP@0.5:0.95). The best overall localization performance was achieved by YOLOv12-L (mAP@0.5:0.95 = 0.874), while YOLOv12-N provided the fastest inference (0.8 ms/image) with competitive accuracy (mAP@0.5:0.95 = 0.822). These findings provide preliminary guidance for selecting YOLOv12 variants based on the trade-off between accuracy and speed.

I. INTRODUCTION

Real-time object detection plays a crucial role in modern food-related applications, offering benefits ranging from personal health tracking to automated meal service [1], [2], [3]. For example, mobile diet apps have surged to hundreds of millions of users, addressing the previously tedious manual record-keeping challenges, creating demand for systems that can automatically identify foods and assess nutrition in real time [4]. By detecting food items through a smartphone camera, users can easily obtain calorie and nutrient estimates, which supporting healthier eating habits [5], [6], [7]. Furthermore, object detection has been implemented in various industries [8], [9]. For example, a cafeteria cashier system uses computer vision technology to recognize food and instantly calculate the bill [10]. This can help reduce queues. On the other hand, there are many recent researches on deep learning, including food detection as a case study. A YOLOv5 model for detecting Indonesian food achieved over 98% accuracy. With high detection accuracy, a food detection system can provide good nutritional estimates as well [5]. In addition, this can also impact consumer well-being and the operational efficiency of a business [11].

Detecting food in images is one of the most challenging tasks in computer vision due to the complex visual characteristics of cuisine [12], [13]. Food types in the world

often encounter the challenge of high intra-class variability [14], [15]. Even the same type or class of food can appear different, depending on the ingredients, presentation, and lighting conditions [16], [17]. Another challenge is that even different types or classes of food can look similar [18]. In practice, multiple foods may be served together, creating challenges of occlusion and overlap [19]. Furthermore, many traditional dishes lack large labelled datasets. There is no single universal model for all foods, and local cuisines (such as Indonesian cuisine) are underrepresented in global food recognition research [20]. Multiple studies have faced various challenges. According to reports, classic feature-based methods only get about 47% accuracy [16]. Even with advanced segmentation and machine learning, accuracy levels off in the mid-80% range for dozens of dish classes [16]. These limitations necessitate the implementation of deep learning detectors. Researchers have discovered that incorporating two-stage detectors into single-stage models can enhance performance. Combining Faster R-CNN with YOLO, for instance, makes it easier to recognize food in a dataset with 100 categories [20].

The original YOLO (You Only Look Once) family model [21], [22], [23] has had a big effect on object detection research by allowing for high speed without losing accuracy. The first version of YOLO algorithm came up with a single-stage architecture that makes detection a single-network regression problem. This mechanism enables real-time operation at around 45 frames per second [21]. The YOLO architecture has been improved with new features over the years (v2 to v12) to make it more accurate and fix problems that were found in earlier versions [24]. Enhancements such as multi-scale prediction, advanced feature fusion, and attention modules have consistently improved performance. For instance, the fifth-generation model (YOLOv5) was shown to be faster and more accurate than the fourth-generation model (YOLOv4) [5]. The seventh-generation YOLOv7 [25] raised the bar even higher by getting 56.8% AP on the COCO benchmark while still keeping frame rates in real time. Recently, a comprehensive benchmark of YOLOv3 to YOLOv12 was published [26], and it was reported that the more complex YOLOv12 architecture offers little advantage in general settings, which increases the computational burden without significantly improving accuracy. However, despite YOLO's rapid evolution, no research has so far been conducted on the application of YOLOv12 to food object detection specifically for Indonesian cuisine. This raises an open question about how the latest YOLO models perform on more detailed culinary data.

In this paper, we benchmark YOLOv12 [27] variants for traditional Indonesian cuisine detection by conducting an empirical study of the latest YOLO models in this domain. We use a representative open-source dataset of traditional Indonesian dishes [28] and train several YOLOv12-based variant models with different scales and model configurations. Computation time and detection evaluation metrics (mAP) were used to measure model quality. Unlike accuracy-focused studies, this study also examines the accuracy-efficiency trade-off across YOLOv12 variants, supported by per-class confusion and preliminary out-of-distribution error analysis. Therefore, this study contributes a controlled empirical benchmark and practical model-selection insight for Indonesian traditional cuisine detection, rather than a new object detection architecture.

II. METHOD

A. Research Flow

The main goal of this study was to make sure that YOLOv12 model variants were compared fairly. The flow of the study is shown in Figure 1. The first step is to get the dataset ready. The goal of this step is to make a dataset of traditional Indonesian foods. This preparation includes splitting the data and augmenting the data until it is ready for training. The prepared data is then used to train all of the YOLOv12 model variants in the same way, with the same input resolution and training scheme. At this stage, the training process is made sure that all YOLOv12 model variants are on the same settings.

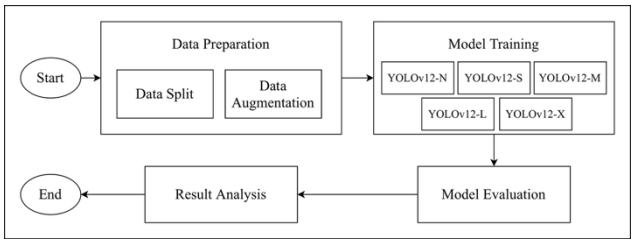

Figure 1. Research Pipeline

After the training phase, we used standard object detection metrics to test how well each YOLOv12 model did overall and for each class. The evaluation results were then looked at to see how accurate the detection was, how fast the computer was, and how the model acted in different situations. The qualitative quality of the results and the quantitative performance of the predictions were both examined in this section of the result analysis.

B. Dataset and Data Preparation

This study utilizes an object detection dataset of traditional Indonesian cuisine [28]. This dataset was annotated with bounding boxes across 20 classes. There are 718 unique images and 720 object instances in the dataset. This means that most images have only one dish, but some have more than one. The YOLO bounding box format (class ID, normalized x/y center, width, height) is used to label each image. The dataset includes several examples of traditional Indonesian dishes *Mie-Aceh*, *Pempek-Palembang*, and *Rendang*.

The dataset provider labelled each image and checked it by hand to make sure the object class was correct. Table I shows the number of object instances per class in the dataset. The counts of instances show that some classes have more than 60

instances and others have fewer than 30, which means that the classes are not evenly distributed.

TABLE I
NUMBER OF OBJECT INSTANCES PER CLASS

Class	Instances
Ayam-Goreng-Lengkuas	61
Mie-Aceh	60
Ayam-Betutu	60
Bubur-Manado	58
Rujak-Cingur	31
Pempek-Palembang	31
Tahu-Telur	30
Laksa-Bogor	30
Rawon-Surabaya	30
Rendang	30
Lumpia-Semarang	30
Sate-Ayam	30
Ayam-Bumbu-Rujak	30
Kerak-Telor	30
Gulai-Ikan-Mas	30
Soto-Lamongan	30
Papeda	30
Gado-Gado	30
Sate-Lilit	30
Gudeg-Jogja	29

The dataset was split into 418 training images, 150 validation images, and 150 test images to make sure that each split had the same number of classes. After that, augmentation was conducted only on the training set by flipping images both horizontally and vertically. The total number of training samples increased to 963 images, while the validation and test sets were kept unchanged to reduce the risk of data leakage during evaluation.

Figure 2. Sample Images from the Dataset

The input was made the same by resizing all the images to 640×640 pixels and automatically aligning them. The majority of YOLO-based training pipelines use 640×640 as the default image size, which is why this choice was made. Some pictures of the food classes in the dataset are shown in Figure 2. These samples indicate that the dataset contains variations in food presentation, lighting conditions, camera viewpoints, and backgrounds.

C. YOLOv12 Variants

When it first appeared, YOLO (You Only Look Once) became a very popular object detection method because of how accurate and fast it was [21]. Currently, YOLOv12 is the latest version of the YOLO family. Essentially, this version features an attention-based architecture that has been proven to facilitate the search for small and overlapping objects [27]. This version also maintains the inference speeds seen in previous YOLO versions. The YOLOv12 architecture adds spatial attention mechanisms to the backbone and neck to help the model combine features from different scales without making the architecture too deep.

Ultralytics provides YOLOv12 in five different sizes: nano (YOLOv12-N), small (YOLOv12-S), medium (YOLOv12-M), large (YOLOv12-L), and extra-large (YOLOv12-X). The difference between these model variants lies in the size of the network used. The nano version is considered the most effective version of YOLOv12 under limited resource conditions, while the extra-large version is the most effective under conditions that prioritize high-accuracy predictions and powerful GPUs.

TABLE II
NUMBER OF OBJECT INSTANCES PER CLASS

Model Variant	Parameters (M)	FLOPs (G)	Description
YOLOv 12-N	2.6	6.5	Ultra-lightweight model for fast inference on edge devices
YOLOv 12-S	9.3	21.4	Small model optimized for mobile and embedded systems
YOLOv 12-M	20.2	67.5	Balanced model for general-purpose real-time detection
YOLOv 12-L	26.4	88.9	High-capacity model for higher accuracy
YOLOv 12-X	59.1	199.0	Most accurate model, suitable for server-side deployment

The training flow and architectural parts are the same for all variants. For example, they all have a feature pyramid and a detection head without anchors. They have different amounts of parameters, FLOPs (floating point operations per second), and run times. Table II shows the most important parts of the five YOLOv12 versions that were used in this study [27]. To ensure the comparisons were fair, all of the models were trained on the same dataset, with the same input resolution (640×640), augmentation method, and training plan.

D. Training Configuration

The training configuration was carried out on the same setup. The official Ultralytics framework was used to train all

of the YOLOv12 model variants in this study. To conduct this study, we used a Google Colab Pro+. The GPU used is NVIDIA A100-SXM4 80GB with CUDA version 12.4. Additionally, the PyTorch version used is 2.9.0, and the Python version is 3.12.12.

Table 3 lists all the hyperparameters used to train all the different versions of YOLOv12. These are the optimization settings, input resolution, and the learning rate policy that the Ultralytics framework sets by default. We began training with weights that had already been trained from the official YOLOv12 release. This speeds up convergence and usually makes generalization better than starting from scratch. To keep the benchmark setting under control, all variants were trained in the same experimental conditions. The only difference was the scale of the YOLOv12 model that was chosen. All experiments had the same data splits, augmentations, image resolutions, and training times.

TABLE III
TRAINING CONFIGURATION FOR ALL YOLOV12 VARIANTS

Parameter	Settings
Epochs	50
Optimizer	SGD
Initial Learning Rate	0.01
Momentum	0.937
Weight Decay	0.0005
Batch Size	Default (based on GPU memory)
Input Size	640x640

During training, model checkpoints were saved automatically when the model did the best on the validation set (highest mAP50-95). Then, the model with the best validation score is used to test the test set. Default Ultralytics hyperparameters were used for fair comparison across YOLOv12 variants.

III. RESULT AND DISCUSSION

A. Detection Performance Overview

Each model variant was trained individually. After training, the next step was to analyse the evaluation results. We test the five YOLOv12 variations on a set of 150 images that show 151 object instances from 20 different food classes. The main focus of the evaluation is on core detection metrics, such as Precision, Recall, mean Average Precision at an IoU threshold of 0.5 (mAP@0.5), and mean Average Precision averaged over IoU thresholds from 0.5 to 0.95 (mAP@0.5:0.95) [29], [30].

Table IV shows a summary of these performance metrics for all versions of YOLOv12. The variants of YOLOv12-M, YOLOv12-L, and YOLOv12-X have the highest mAP@0.5:0.95 values (0.859, 0.874, and 0.870, respectively). Although the models achieved high mAP values, these results should be interpreted as in-domain performance on the prepared dataset. The dataset is relatively small, with 718 images and 720 object instances, and the validation and test sets come from the same overall dataset source. Therefore, the possibility of dataset-specific learning cannot be fully ruled out. Further evaluation on larger and more diverse external datasets is required to better assess model generalization and reduce the risk of overfitting.

TABLE IV
YOLOV12 EVALUATION RESULTS

Model	Precision	Recall	mAP @0.5	mAP @0.5:0.95
YOLOv12-N	0.913	0.937	0.985	0.822
YOLOv12-S	0.960	0.961	0.991	0.845
YOLOv12-M	0.975	0.990	0.991	0.859
YOLOv12-L	0.964	0.977	0.991	0.874
YOLOv12-X	0.970	0.975	0.991	0.870

The YOLOv12-N model has only 2.56 million parameters, but it still gets a great mAP@0.5 of 0.985 and a mAP@0.5:0.95 of 0.822. This shows that even small models in the YOLOv12 family can find objects in datasets that are hard to see. But smaller versions don't always do as well on some classes that have small visual differences or fewer training samples.

Overall, the results show that making the model bigger does make it more accurate, especially when it comes to harder detection tasks. But after intermediate scales, the improvement is slow. The importance of the trade-off between scenario accuracy and efficiency is underscored by the rapidity of inference and its rapid implementation.

B. Per-Class Performance

In this section, a per-class analysis is conducted to provide a more detailed analysis of how each YOLOv12 variant can separate 20 traditional Indonesian food classes. All model variants received good evaluations overall, but for some classes, there were still errors that need to be noted. Some dishes were well predicted, while others were difficult to predict. This is because they have similar colours, textures, or presentation styles. For example, the same dish can look very different in different images due to lighting, camera angles, and the presentation style of the dish itself.

Starting from the nano version of YOLOv12, the confusion matrix in Figure 3 shows some mistakes very clearly. As an example, *Laksa-Bogor* is incorrectly labelled as *Soto-Lamongan*, *Ayam-Goreng-Lengkuas* incorrectly labelled as *Sate-Ayam*, and *Rujak-Cingur* is incorrectly labelled as *Pempek-Palembang*. Even though these are wrong, most classes still show high diagonal dominance, which means that most categories are very precise. Some classes such as *Ayam-Betutu*, *Bubur-Manado*, and *Rawon-Surabaya* are examples of classes that have been predicted very well.

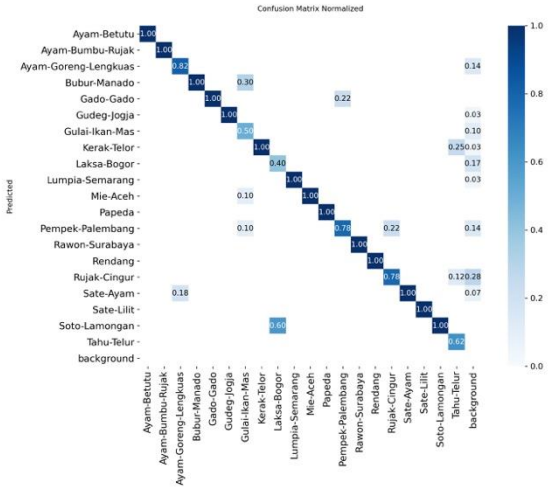

Figure 3. Normalized Confusion Matrix for YOLOv12-N

The nano version cannot accomplish as good a job of predicting classes as YOLOv12-S in Figure 4. There are fewer and smaller mistakes in classification, but there are still some small mistakes with dishes that look alike, like *Gulai-Ikan-Mas* and *Tahu-Telur*. The pattern of predictions as a whole is more stable. For instance, *Ayam-Goreng-Lengkuas* and *Gulai-Ikan-Mas* classes are more reliable and have fewer false positives.

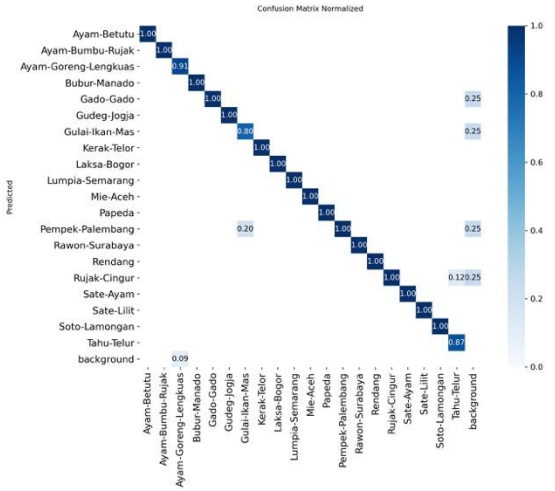

Figure 4. Normalized Confusion Matrix for YOLOv12-S

Figure 5 shows how well YOLOv12-M works. The matrix shows that the predictions are more accurate because most of them are on the diagonal. It's interesting that hard classes like *Gulai-Ikan-Mas* and *Tahu-Telur* have a high precision score. This means that the model is better at seeing differences.

Figure 6 shows that YOLOv12-L has stronger diagonal dominance, which means that its predictions are more confident and consistent. There are still some small confusions, especially with *Tahu-Telur* and *Ayam-Goreng-Lengkuas*. There are also fewer false positives for background classes, which means better localization and fewer detection failures.

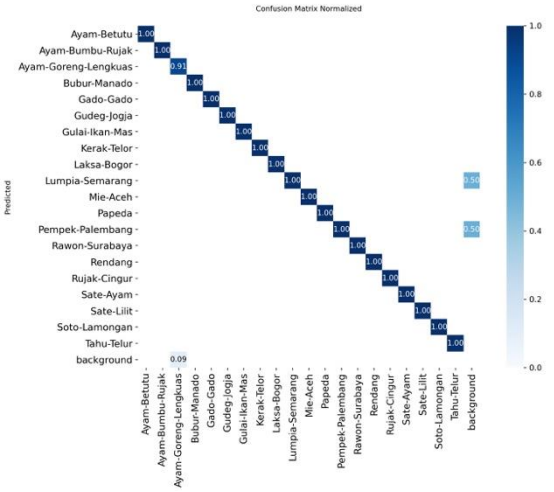

Figure 5. Normalized Confusion Matrix for YOLOv12-M

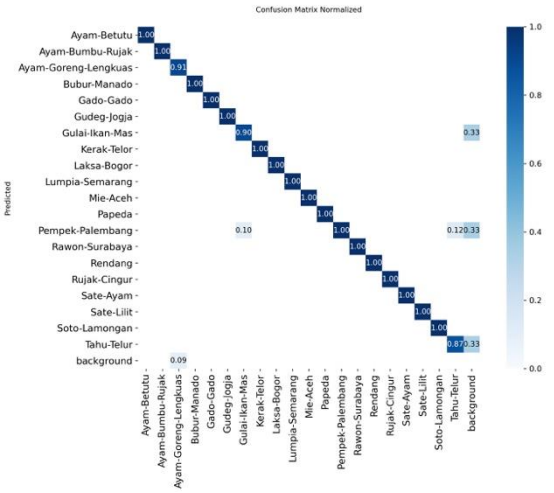

Figure 6. Normalized Confusion Matrix for YOLOv12-L

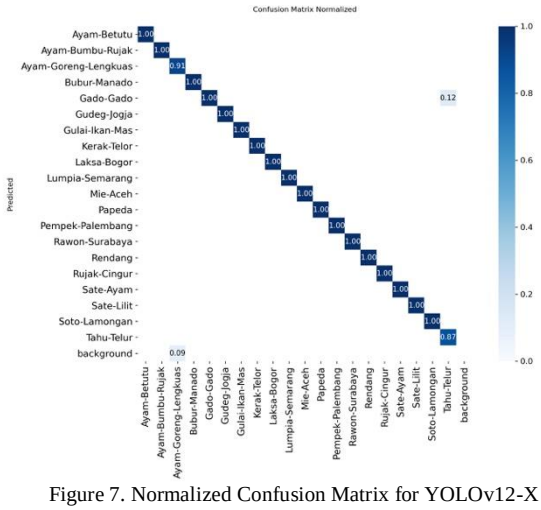

Figure 7. Normalized Confusion Matrix for YOLOv12-X

Figure 7 shows that YOLOv12-X makes an almost perfect confusion matrix. The predictions are very close to the diagonal, and there aren't many mistakes in putting them in the right class. This shows that the model can see small things

in pictures. The extra-large version clearly does better with more space, since it performs the same in all classes.

The confusion matrix shows that as the model gets more complicated, the accuracy for each class goes up. Even the smallest variant gives good results, but the bigger YOLOv12 model is clearly better for high-precision tasks that involve visually different categories, like traditional food.

C. Training, Inference Time, and Efficiency

Detection accuracy is important, but in real-world situations, models need to be not only accurate but also quick to run, cheap to train, and use little memory. This is especially important in real-time or edge-based situations, like mobile apps and embedded systems, where the speed of inference, the size of the model, and the time it takes to train are all very important for an object detection system to work.

Inference time was measured using the same evaluation environment as the training process. All models were tested on an NVIDIA A100-SXM4 80GB GPU using the Ultralytics framework, with an input resolution of 640×640 pixels. The reported inference time is expressed as milliseconds per image (ms/image) and was obtained from the model evaluation process on the test set using the default Ultralytics evaluation setting. Since the measurement was performed on GPU-based inference, the latency may differ when the models are deployed on CPU-based or edge-device environments.

TABLE V
INFERENCE TIME AND MODEL FILE SIZE FOR YOLOV12 VARIANTS

Model	Training Time (h)	Inference Time (ms/image)	Model File Size (MB)
YOLOv12-N	0.13	0.8	5.5
YOLOv12-S	0.14	1.4	18.9
YOLOv12-M	0.18	2.5	40.8
YOLOv12-L	0.27	5.5	53.6
YOLOv12-X	0.36	8.2	119.1

To evaluate this trade-off, we measured the average inference time per image and the training duration over 50 epochs on an NVIDIA A100 GPU. After the training was done, we downloaded the final model file size (.pt). Table V shows that the model became much more complicated when it went from YOLOv12-N to YOLOv12-X. The YOLOv12-N model was the smallest, trained in just 0.13 hours, had an inference latency of 0.8 milliseconds per image, and made a model file that was only 5.5 MB. It took 0.36 hours to train YOLOv12-X, and the final model was 119.1 MB in size. It took 8.2 milliseconds to conduct the inference.

From the results, we can see that both YOLOv12-M and YOLOv12-L are good balance versions because they have high detection performance and don't need too much processing power. In practice, when choosing a model, one thing that should be considered is not just the accuracy measures. Training time, inference latency, and memory limits are important considerations since there is no free and unlimited hardware for a real-time system.

D. Error Analysis on Out-of-Distribution Data

The YOLOv12 model did well on the validation and test sets, but when it was tested on images that weren't in the original dataset, it showed that it couldn't generalize as well as it could have. This shows how hard it is for the model to work with data that isn't in the training set. This is especially true when the styles, lighting, or presentations of the dishes are different from what it learned.

Figure 8. Undetected *Pempek-Palembang* class in Out-of-Distribution Image

Figure 9. Missclassification Sample of *Sate-Ayam* as *Gudeg-Jogja*

Figure 8 shows one case where the model doesn't find the *Pempek-Palembang* class. The model works well with these kinds of test images, but it doesn't make a bounding box in this case. Some possible explanations are that the training set doesn't have many examples that look like the test images, that the test images are very bright, and that the overhead view is very different from the samples in the dataset.

Figure 9 shows another issue with the Chicken-Satay class. The model incorrectly predicts that the dish is Gudeg-Jogja with a score of 0.68. Both dishes are brown, and they come with other side dishes, which makes it more likely that people

will get confused. This mistake shows that the model is still sensitive to a lot of variation within classes and visual overlap between classes. This is common in Indonesian food because of the different regions and how food is served.

These results show that training needs to include more types of images, especially for categories where there is a lot of difference within the same class. In the future, targeted augmentation, domain adaptation, or few-shot learning could help make things more stable in the future.

E. Overall Trade-offs and Recommendations

Quantitatively, YOLOv12-L improved mAP@0.5:0.95 from 0.822 in YOLOv12-N to 0.874, representing a gain of 0.052, but its inference time increased from 0.8 ms/image to 5.5 ms/image, or about 6.9 times slower. YOLOv12-X further increased inference time to 8.2 ms/image and model size to 119.1 MB, but its mAP@0.5:0.95 slightly decreased to 0.870 compared with YOLOv12-L. This indicates that larger models do not always provide proportional accuracy gains, and YOLOv12-L offers the best localization accuracy, while YOLOv12-N provides the fastest inference with competitive accuracy.

The YOLOv12-X model variant is the best variant in terms of overall prediction accuracy. Therefore, this variant may be considered for scenarios that prioritize accuracy, such as server-side inference, provided that further validation is conducted on more diverse real-world data. However, it should be noted that this model variant may not be suitable for cases with limited hardware environments, due to the long prediction time it takes.

However, the smaller model variants, YOLOv12-N and YOLOv12-S, have very fast inference times and small file sizes. Therefore, these two variants may be more promising candidates for resource-limited applications such as mobile devices and embedded systems, although additional deployment testing is still required. However, it should also be noted that these two models have less accuracy than the larger model variants.

On the other hand, YOLOv12-M and YOLOv12-L provide a middle ground for this situation. Both models offer substantial alternatives in terms of detection accuracy, inference time, and model size. These variants performed well in image prediction while maintaining a reasonable computational time. Both models may serve as balanced candidates for systems with moderate GPU resources.

Ultimately, model selection must align with the computational limitations and requirements of the application. This study does not state which model variant is superior, as the choice of variant ultimately depends on the available resources and conditions.

IV. CONCLUSION

This study evaluates the performance of five YOLOv12 variants, namely nano, small, medium, large, and extra-large. The focus is on detecting 20 categories of traditional Indonesian dishes. All models were trained and tested under identical conditions and environment using labelled datasets with diverse visual characteristics. The results show that all YOLOv12 variants achieve high accuracy, with mAP@0.5 scores above 0.98, and performance increases with variants from nano to extra-large. YOLOv12-X provides the highest overall accuracy and per-class consistency. At the same time, this variant requires the most computational resources.

In contrast, YOLOv12-N and YOLOv12-S offer fast inference and lighter file sizes. Both models show potential for lightweight applications, but their practical use should still be validated under real deployment conditions. Meanwhile, the medium and large variants balance detection performance with reasonable efficiency. While benchmark results are promising, results from out-of-distribution testing indicate that model performance is still sensitive to variations in image style, composition, or intra-class diversity. Therefore, the use of higher-quality datasets in terms of resolution and quantity must be considered. These findings indicate that YOLOv12 variant selection should be tailored to application needs, performance targets, and operational constraints, while recognizing that broader real-world validation is still needed. However, since the dataset size is relatively limited, the high detection performance should be interpreted cautiously and should not be considered as conclusive evidence of generalization to all real-world Indonesian cuisine images.

REFERENCES

[1] A. Tuomi and M. P. Ascenção, "Intelligent automation in hospitality: exploring the relative automatability of frontline food service tasks," *J. Hosp. Tour. Insights*, vol. 6, no. 1, pp. 151–173, Nov. 2021, doi: 10.1108/JHTI-07-2021-0175.

[2] M. Gerasimchuk and A. Uzhinskiy, "Food Recognition for Smart Restaurants and Self-Service Cafes," *Phys. Part. Nucl. Lett.*, vol. 21, no. 1, pp. 79–83, Feb. 2024, doi: 10.1134/S1547477124010059.

[3] G. I. Alkady, "A Deep Learning-Powered Web Service for Optimal Restaurant Recommendations Based on Customers Food Preferences," in *2024 16th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, Jun. 2024, pp. 1–4. doi: 10.1109/ECAI61503.2024.10607587.

[4] M. Han, J. Chen, and Z. Zhou, "NutrifyAI: An AI-Powered System for Real-Time Food Detection, Nutritional Analysis, and Personalized Meal Recommendations," Oct. 21, 2024, *arXiv*: arXiv:2408.10532. doi: 10.48550/arXiv.2408.10532.

[5] G. C. Utami, C. R. Widiawati, and P. Subarkah, "Detection of Indonesian Food to Estimate Nutritional Information Using YOLOv5," *Teknika*, vol. 12, no. 2, pp. 158–165, Jun. 2023, doi: 10.34148/teknika.v12i2.636.

[6] F. Romadhon *et al.*, "Food Image Detection System and Calorie Content Estimation Using Yolo to Control Calorie Intake in the Body," *E3S Web Conf.*, vol. 465, p. 02057, 2023, doi: 10.1051/e3sconf/202346502057.

[7] T. Selvaraju, V. Dakshinamurthi, G. Badurudeen, N. Prabhakaran, P. Gopi, and M. A. N. Hussain, "Food detection and estimation of calories and other macro-nutrients and features to support healthy lifestyle," *AIP Conf. Proc.*, vol. 3279, no. 1, p. 020129, Apr. 2025, doi: 10.1063/5.0263063.

[8] A. Dhelia, S. Chordia, and K. B, "YOLO-based Food Damage Detection: An Automated Approach for Quality Control in Food Industry," in *2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Oct. 2024, pp. 1444–1449. doi: 10.1109/I-SMAC61858.2024.10714664.

[9] N. Rane, "YOLO and Faster R-CNN object detection for smart Industry 4.0 and Industry 5.0: applications, challenges, and opportunities," Oct. 25, 2023, *Social Science Research Network, Rochester, NY*: 4624206. doi: 10.2139/ssrn.4624206.

[10] M. Y. Wu, J. H. Lee, and C. Y. Hsueh, "A Framework of Visual Checkout System Using Convolutional Neural Networks for Bento Buffet," *Sensors*, vol. 21, no. 8, p. 2627, Jan. 2021, doi: 10.3390/s21082627.

[11] J. W. Park, Y. H. Cho, M. K. Park, and Y. D. Kim, "Consumer Usability Test of Mobile Food Safety Inquiry Platform Based on Image Recognition," *Sustainability*, vol. 16, no. 21, p. 9538, Jan. 2024, doi: 10.3390/su16219538.

[12] X. Huang *et al*., "Application of Image Computing in Non-Destructive Detection of Chinese Cuisine," *Foods*, vol. 14, no. 14, p. 2488, Jan. 2025, doi: 10.3390/foods14142488.

[13] A. Sanatbyek, A. Karabay, H. A. Varol, and M. Y. Chan, "Deep Object Recognition-Based Analysis of Diverse Culinary Landscapes," in *2025 IEEE International Conference on Image Processing (ICIP)*, Sep. 2025, pp. 1127–1132. doi: 10.1109/ICIP55913.2025.11084477.

[14] N. Zheng, X. Song, W. T. Tang, S.-K. Ng, L. Nie, and R. Zimmermann, "Unsupervised Few-Shot Food Recognition With Intra-Class Variation and Inter-Class Similarity Modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 12, pp. 12138–12151, Dec. 2025, doi: 10.1109/TCSVT.2025.3585925.

[15] D. Pandey *et al*., "Object Detection in Indian Food Platters using Transfer Learning with YOLOv4," in *2022 IEEE 38th International Conference on Data Engineering Workshops (ICDEW)*, May 2022, pp. 101–106. doi: 10.1109/ICDEW55742.2022.00021.

[16] A. H. Rangkuti, J. M. Kerta, R. Y. Mogot, and V. H. Athala, "Identification of Indonesian Traditional Foods Using Machine Learning and Supported by Segmentation Methods," *JOIV Int. J. Inform. Vis.*, vol. 8, no. 4, pp. 2324–2335, Dec. 2024, doi: 10.62527/joiv.8.4.2545.

[17] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – Mining Discriminative Components with Random Forests," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 446–461. doi: 10.1007/978-3-319-10599-4_29.

[18] J. Dai, X. Hu, M. Li, Y. Li, and S. Du, "The multi-learning for food analyses in computer vision: a survey," *Multimed. Tools Appl.*, vol. 82, no. 17, pp. 25615–25650, Jul. 2023, doi: 10.1007/s11042-023-14373-6.

[19] N. Aditama and R. Munir, "Indonesian Street Food Calorie Estimation Using Mask R-CNN and Multiple Linear Regression," in *2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T)*, Mar. 2022, pp. 1–6. doi: 10.1109/ICPC2T53885.2022.9776804.

[20] M. Nadeem, H. Shen, L. Choy, and J. M. H. Barakat, "Smart Diet Diary: Real-Time Mobile Application for Food Recognition," *Appl. Syst. Innov.*, vol. 6, no. 2, p. 53, Apr. 2023, doi: 10.3390/asi6020053.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," May 09, 2016, *arXiv*: arXiv:1506.02640. doi: 10.48550/arXiv.1506.02640.

[22] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7263–7271. Accessed: Jan. 16, 2026. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2017/html/Redmon_YOLO9000_Better_Faster_CVPR_2017_paper.html

[23] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 08, 2018, *arXiv*: arXiv:1804.02767. doi: 10.48550/arXiv.1804.02767.

[24] I. Kurmashev, V. Semenyuk, A. Lupidi, D. Alyoshin, L. Kurmasheva, and A. Cantelli-Forti, "Study of the Optimal YOLO Visual Detector Model for Enhancing UAV Detection and Classification in Optoelectronic Channels of Sensor Fusion Systems," *Drones*, vol. 9, no. 11, p. 732, Nov. 2025, doi: 10.3390/drones9110732.

[25] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," Jul. 06, 2022, *arXiv*: arXiv:2207.02696. doi: 10.48550/arXiv.2207.02696.

[26] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions," Mar. 17, 2025, *arXiv*: arXiv:2411.00201. doi: 10.48550/arXiv.2411.00201.

[27] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-Centric Real-Time Object Detectors," Feb. 18, 2025, *arXiv*: arXiv:2502.12524. doi: 10.48550/arXiv.2502.12524.

[28] President University, "Indonesian-Traditional-Cuisine Computer Vision Model." 2023. [Online]. Available: https://universe.roboflow.com/president-university-y2m5p/indonesian-traditional-cuisine

[29] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics*, vol. 10, no. 3, p. 279, Jan. 2021, doi: 10.3390/electronics10030279.

[30] A. Badithela, T. Wongpiromsarn, and R. M. Murray, "Evaluation Metrics for Object Detection for Autonomous Systems," Oct. 19, 2022, *arXiv*: arXiv:2210.10298. doi: 10.48550/arXiv.2210.10298.