

A YOLO-Based Framework for Automated Food Recognition and Nutrient Adequacy Estimation in the Free Nutritious Meal Program

Amelia Prachintya Putri Arifin¹, Wiyli Yustanti², and Mega Mirasaputri Cahyanti³

¹ Information Systems Study Program, Faculty of Engineering, Universitas Negeri Surabaya, Indonesia

² Master's Study Program in Informatics, Faculty of Engineering, Universitas Negeri Surabaya, Indonesia

³ Graduate School of Innovation and Technology Management, Yamaguchi University, Japan

Corresponding author: Wiyli Yustanti (e-mail: wilyyustanti@unesa.ac.id), **Author(s) Email:** Amelia Prachintya Putri Arifin (e-mail: amelia.23054@mhs.unesa.ac.id), Mega Mirasaputri Cahyanti (e-mail: mega_mc@yamaguchi-u.ac.jp)

Abstract The Free Nutritious Meal (MBG) program in Indonesia may not consistently meet nutritional standards, providing an average of 377 of the targeted 531 calories per serving, with insufficient protein and fat. Limited nutritional information from food images makes independent meal assessment difficult. This study develops an image-based food detection system using You Only Look Once (YOLO) to estimate the nutritional adequacy of MBG meals based on Recommended Dietary Allowances (AKG). The study used Knowledge Discovery in Databases (KDD)- guided data management and Rapid Application Development (RAD)- guided system development. A dataset of 1.200 images across 51 food classes was evaluated using YOLO 8-YOLO 12, utilizing 5-fold cross-validation, followed by retraining and a 1.000 iteration bootstrap analysis on 240 test images. Average mAP50 scores were 0.8981, 0.898, 0.896, 0.894, and 0.888 for YOLO 8, YOLO 9, YOLO 11, YOLO 12, and YOLO 10. The Kruskal-Wallis test indicated significant differences among models ($H = 489.946$; $p < 0.001$), and YOLO 8 was selected because it had the highest bootstrap mean rank. Detected food classes were assigned reference serving weights rather than measured mass or volume, converted to nutrient values using the Indonesian Food Composition Table (TKPI), and compared with AKG by school level, age, and gender. Validation against nutritionist assessments across five nutrients produced a mean absolute percentage error of 40.31%, with energy and carbohydrate showing the closest agreement. A sensitivity analysis showed a 20 percent portion-weight error most strongly affected iron and calcium adequacy. Because nutrient values rely on reference assumptions rather than measured mass or volume, the outputs represent an estimate, not a definitive assessment of nutritional adequacy. YOLO 8 was the most suitable model and the highest mean rank assumption-based estimation approach for MBG nutritional screening, with direct mass or volume measurement recommended for future work.

Keywords: YOLO; Food Detection; Nutritional Analysis; Digital Health; Free Nutritious Meal Program

1. Introduction

Nutritional issues among school-age children require attention because malnutrition can hinder cognitive development, education, productivity, adult income, and human resource quality [1]. To address this issue, the government launched the Free Nutritious Meals Program (MBG) for schoolchildren to improve their nutritional intake and haemoglobin levels. Previous research shows that the MBG program reduced anaemia prevalence from 42.6% to 21.7% [2]. However, implementing the MBG still faces challenges related to distribution, quality control, and budget sustainability [1]. Although the MBG program is designed to meet students' nutritional needs, research in Makassar shows that the MBG menu still does not fully meet adolescents' nutritional needs, particularly for carbohydrates, which remain below 25% of the standard [3]. These conditions are difficult to monitor because food information generally consists only of photos and portion-size-based nutritional analyses, without providing age- and gender-specific nutritional information. As a result,

teachers, students, and parents struggle to assess the nutritional adequacy of the meal students receive.

Image-based dietary estimation systems based on the You Only Look Once (YOLO) concept have been applied in various contexts to meet these monitoring needs. For example, researchers have developed an automated system to estimate food weight and nutritional content using computer vision and artificial intelligence (AI) algorithms [4], as well as image-based food nutrient detection and analysis in a cafeteria setting, achieving an mAP50 of 0.873 [5]. A study in Sweden also used YOLO 8 to identify food types and estimate portions using a standard plate model [6]. In Indonesia, research on YOLO-based food detection is also gaining traction, including detecting food calories to prevent obesity [7] and evaluating YOLO 8 on traditional Indonesian foods [8]. These systems are generally limited to estimating total calories using general or foreign nutritional databases such as FatSecret [8], [7], rather than the Indonesian Food Composition Tables (TKPI) and Indonesian Recommended Dietary Allowances (AKG), which account

for age- and gender-based differences in nutritional needs. Nutritional adequacy is also calculated using only a single general standard, rather than standards tailored to each individual as defined by the AKG. Research on YOLO-based food detection in Indonesia [7], [9] is also limited to food classes and stops at recognition, without proceeding to nutritional adequacy analysis.

This gap is further underscored by the Central Asian Food Scenes (CAFSD) dataset, which contains 239 classes and indicates a shift toward assessing macro- and micronutrient values. However, no comprehensive nutritional adequacy estimation system has implemented this [10], a direction that subsequent research has not pursued. However, estimating food portion weight solely from visual images remains a major limitation and challenge in current nutritional analysis systems. As shown in calorie detection studies [11], precise food volume measurement still relies heavily on additional depth data from 3D reconstruction processes, an approach outside the scope of this study. This research is important because the MBG program reaches millions of students every day. However, the public currently lacks easy access to information on food's nutritional adequacy by education level, age, and gender. Therefore, an AI-based system is needed to identify food types and analyze their nutritional content in accordance with the AKG to help nutritionists, schools, and parents monitor students' nutritional quality.

The novelty of this study does not lie in the use of YOLO itself, but rather in how this system simultaneously integrates four components: food object detection results, the TKPI, demographically adjusted AKG requirements per student in accordance with Minister of Health Regulation No. 28/2019, and a manual correction mechanism as a control layer before the data are used for nutritional decision-making. Unlike previous systems that limited calorie calculations by not adjusting for demographics and relied on foreign databases, this study maps detection results directly to the TKPI and AKG, calculated individually by school level, age, and gender. This system also includes a manual correction feature, ensuring that classification errors do not directly affect the final results, as in the "detection only" approach used by the previous system. To support system development, this study compared five versions of YOLO on the MBG dataset to determine the best detection model for implementation. Although this evaluation is not the primary contribution, this comparison goes beyond previous research, which covered models only up to YOLO 11 [12], [13].

Based on this description, this study aims to develop a YOLO-based food detection system that directly calculates nutritional adequacy for students receiving MBG. The proposed framework, YOLO-TKPI-AKG, detects food types from photos, matches them with TKPI data, and compares them with AKG standards based on students' grade level, age, and gender. To select a detection model for this framework, this study tested and compared five YOLO versions. YOLO was selected as the detection backbone because it excels in processing speed and ease of training compared to CNNs and SSDs [14] and has been widely applied in real-time in similar

cases [15], [16]. Comparing the five YOLO versions ensures the system uses a high-accuracy detection model, so the food identification results are reliable. However, this comparison only justifies the backbone selection and is not the main contribution of this study. The main contribution of this study is the YOLO-TKPI-AKG framework, which integrates food detection, national TKPI nutrition data, and demographically adjusted AKG requirements, thereby distinguishing it from previous YOLO-based dietary estimation systems. In this system, a manual correction feature prevents detection errors from affecting nutritional calculations, and it displays the percentage deficiency for each nutrient, along with food examples, as recommendations for nutritional improvement.

II. Method

This study implements and compares YOLO 8 through YOLO 12 to detect MBG food objects and identify the best-performing model for a nutritional adequacy analysis system. This study uses three methods: a literature review, Knowledge Discovery in Databases (KDD), and Rapid Application Development (RAD). The KDD stage consisted of five phases: selection, preprocessing, transformation, data mining, and evaluation [17]. Meanwhile, the RAD phase built the system in three stages: requirements planning, the RAD design workshop, and implementation [18].

A. You Only Look Once Architecture

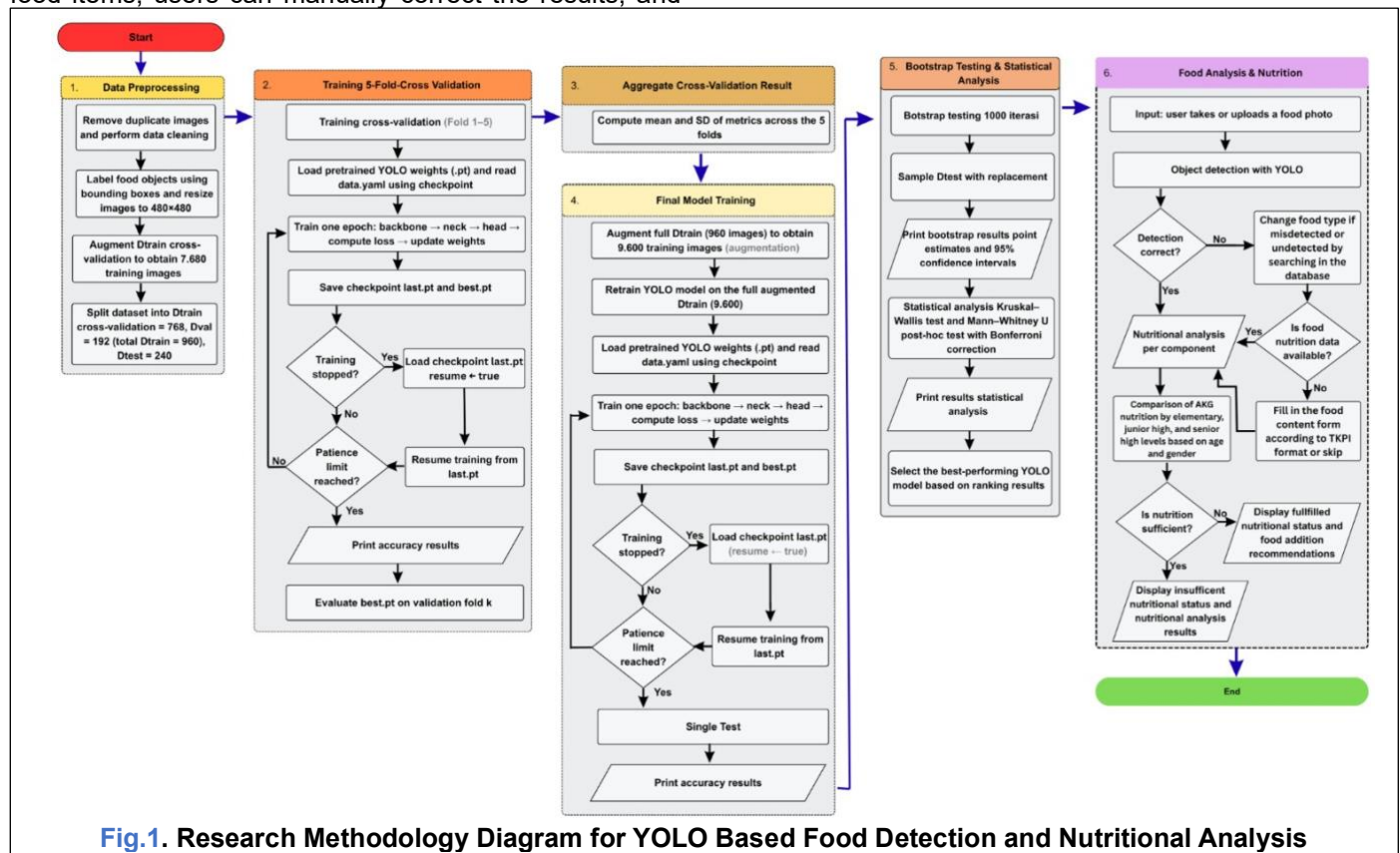
Redmon introduced YOLO in "You Only Look Once: Unified, Real-Time Object Detection." YOLO was developed to replace two-stage detection methods with a faster process. Its architecture has three main parts: a backbone that extracts image features, a neck that combines features from multiple scales, and a head that generates bounding box predictions [19]. The model's performance is evaluated using box loss, confidence loss, and classification loss, with the following key innovations:

- YOLO 8: no longer anchor-based; the backbone section uses C2f blocks.
- YOLO 9: an additional path to preserve gradient accuracy, RepNCSPeLan4 (GELAN) blocks, ADown downsampling, and CBLinear/CBFuse.
- YOLO 10: A ranking-guided CIB design replaces the C2f bottleneck, SCDown downsampling, and the PSA block after the fourth backbone stage.
- YOLO 11: C3k2 and C2PSA blocks to capture deeper inter-feature relationships.
- YOLO 12: Area Attention (A2) and Residual Efficient Layer Aggregation Network (R-ELAN).

Fig. 1 combines two frameworks: the KDD phase (Blocks 1–5), which includes data preprocessing, model development, evaluation, and statistical testing, and the RAD phase (Block 6), which includes system implementation. The KDD phase uses the MBG image dataset as input and produces the best-performing YOLO detection model. Block 1 covers preprocessing: data cleaning, labeling, resizing, segmentation, and augmentation. In Block 2, the YOLO model is trained using 5-fold cross-validation with a checkpoint-based procedure, and Block 3 uses these results for aggregated cross-validation. Block 4 retrains the final model on the full training dataset and a single test set, evaluates it through

bootstrap testing and statistical analysis (Kruskal–Wallis and Mann–Whitney U tests with Bonferroni correction) to select the best-performing model in Block 5. The RAD phase (Block 6) takes user-uploaded photos of MBG trays as input. It generates nutritional status and recommendations: the selected YOLO model detects food items, users can manually correct the results, and

the detected foods are converted into nutritional values using the TKPI and compared to the AKG (based on school grade, age, and gender), with a target of 30 percent per meal. The following subsections describe each stage in detail.



B. Selection

The selection phase focused on data collection. Data for these 51 food categories were collected between June 3 and June 30 using photos of MBG food trays obtained from Google Images and Facebook via search terms such as “MBG,” “photos of MBG food trays,” and “MBG food menu,” as well as from the official Instagram and TikTok accounts of the Nutrition Service Unit (SPPG). The SPPG’s data collection process led the authors to prioritize these official accounts as the primary sources. Only photos showing the entire MBG food tray were used, while blurry, cropped, or irrelevant images were discarded. A duplication check, combined with a review of filenames for 10 pairs of similar photos, ensured that

similarities between images were limited to background or editing style, rather than indicating actual duplication. All photos were used for non-commercial academic research purposes; photos from the SPPG served as the primary source with explicit permission. In contrast, photos from public social media were used under fair use. The data were manually cleaned until 1.200 photos remained, then divided into 960 photos for cross-validation and 240 photos (2.203 instances) for testing. In each fold, 768 photos were used for training and 192 photos for validation. Data augmentation was applied only to the training set, resulting in 9.600 samples, while the validation and test sets continued to use the original photos to prevent data leakage and ensure the validity of the model evaluation.[20], [21]. As shown in Table 1.

Table 1. The Number of Images and Instances in Training and Validation Data for Each Cross-Validation Fold

Fold	Train			Validation	
	Image Original	Image Augmentation	Instance Augmentation	Image	Instance
1	768	7.680	72.387	192	1.648
2	768	7.680	70.701	192	1.817
3	768	7.680	71.229	192	1.766
4	768	7.680	70.550	192	1.833
5	768	7.680	70.607	192	1.827

The standard portion sizes for each food category are based on the Food Exchange List (DBMP) and the Food

Photo Book (Porsimetri), using the Household Measure (URT) as the unit of measurement, and were then

Manuscript received 7 August 2026; Revised 15 August 2026; Accepted 30 September 2026; Available online 20 October 2026

DOI: <https://doi.org/10.35882/ijeemi.v8i4.382>

Copyright © 2026 by the authors. Published by Jurusan Teknik Elektromedik, Politeknik Kesehatan Kemenkes Surabaya Indonesia. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

adjusted to the average serving weights typically served in the MBG kitchen (SPPG) as confirmed by a nutritionist. These values are average estimates and not absolute standards; therefore, serving weights may vary

depending on how they are served at each SPPG. Table 2 presents the number of training data samples, test data samples, serving weights (Serv. Wt.), and serving sizes (URT) for each food category.

Table 2. The Number of instances for each food class in the final model

No	Class Name	Train Instance	Test Instance	Serv. Wt. (g)	URT
1	Beef Floss	1.370	36	30	3 Tablespoon
2	Grape	3.290	28	6	1 Medium Fruit
3	Apple	1.510	25	120	1 Medium Fruit
4	Crispy Chicken	1.520	34	60	1 Upper Breast Piece
5	Fried Chicken	1.460	35	100	1 Upper Breast Piece
6	Teriyaki Beef	1.400	40	20	1 Tablespoon
7	Fried Vermicelli	1.580	34	45	3 Tablespoon
8	Broccoli	1.340	57	25	1 Floret
9	Dragon Fruit	2.435	87	80	1 Medium Piece
10	Steamed Green Beans	1.300	44	20	2 Tablespoon
11	Capcai	1.640	31	50	1 Ladle
12	Fried Squid	1.610	43	15	1 Small Piece
13	Edamame	1.480	29	8	5 Pods
14	Steamed Corn	1.310	62	35	1 Corn Cob
15	Sweet Orange	2.010	72	100	1 Medium Fruit
16	Lime	1.462	24	10	1 Slice
17	Green Peas	1.375	73	25	2 Tablespoon
18	Longan	1.590	27	10	1 Piece
19	Potatoes	1.500	26	100	1 Serving
20	Tempeh Chips	2.000	49	30	1 Handful
21	Crackers	1.420	33	10	1 Sachet
22	Fried Catfish	2.239	35	40	1 Medium Fish
23	Melon	1.760	63	100	1 Large Slice
24	Boiled Noodles	1.629	27	100	1 Serving
25	Fried Tilapia	1.430	29	40	1 Medium Fish
26	Fried Rice	1.470	38	100	2 Rice Paddles
27	Yellow Rice	1.547	40	100	2 Rice Paddles
28	White Rice	2.330	69	100	2 Rice Paddles
29	Papaya	2.317	52	110	1 Piece
30	Bananas	1.410	46	50	1 Piece
31	Rambutan	2.160	75	35	1 Piece
32	Rendang	1.430	38	40	1 Piece
33	Bread	2.420	20	35	1 Slice
34	Snake Fruit	2.090	52	40	1 Medium Fruit
35	Sauce	2.270	42	15	1 Sachet
36	Mustard Greens	2.105	27	15	1 Tablespoon
37	Mix Vegetables	2.454	35	30	2 Tablespoon
38	Lettuce	1.740	58	10	1 Leaf
39	Watermelon	2.715	59	180	1 Slice
40	Soup	1.380	38	250	1 Ladle
41	Spaghetti	1.430	41	250	1 Serving
42	Milk	1.440	43	200	1 Glass
43	Tofu	1.570	43	40	1 Medium Piece
44	Fried Egg	1.999	45	60	1 Whole Egg
45	Omelette	1.860	49	50	1 Whole Egg
46	Boiled Egg	1.410	38	60	1 Whole Egg
47	Tempeh	1.440	35	25	1 Medium Piece
48	Cucumber	1.450	34	15	1 Piece
49	Tomatoes	2.040	71	15	1 Slice
50	Crispy Shrimp	1.380	35	10	1 Medium Piece
51	Steamed Carrot	1.350	37	20	2 Tablespoon
Total		88.867	2.203		

C. Data Preprocessing

The preprocessing stage consists of several steps, namely:

1) Data Labelling and Resizing

All training images were labelled using bounding boxes in Roboflow and resized to 480x480 pixels. Because a single tray may contain multiple types of food, annotations in one image may include multiple bounding boxes. As a result, the 1,200 images produced 11,094 annotated bounding box instances across 51 food classes, as detailed in Table 2. The 480x480 size was chosen for two reasons. First, according to Ultralytics, this size corresponds to a value of 32 (ranging from 32 to 4.096). Since 480 is a multiple of 32, the model can process the images directly without additional adjustments. Second, the data consists of compressed social media screenshots of food photos, resulting in varying quality and resolution. Enlarging images beyond their original resolution does not add new details to the food objects; Ultralytics also recommends selecting image sizes based on dataset characteristics because larger sizes do not always improve performance [22]. Based on these considerations, a resolution of 480 x 480 is deemed sufficient to preserve important information about the food objects while reducing memory requirements and training time compared to larger images.

2) Test, Training and Validation Data Splitting

The dataset was split before augmentation to prevent data leakage, such as identical photos appearing in the training data. These conditions can make model evaluation results inaccurate [23]. Data splitting is carried out on the Google Colab platform. The dataset totals 1.200 images, with .txt-format annotations. The 240 images will be set aside as

validation process. These data are randomly selected using a specific seed value to ensure the selection is reproducible. Test data were used only once, at the end of the process, to measure the model's performance on unseen images. The test data also did not undergo augmentation, so the images remained in their original condition. The remaining dataset of 960 images was then divided into 5 folds using 5-fold cross-validation technique with KFold function from scikit-learn library. This function divides the data into 5 folds, each containing an equal number of images. No image overlap was found between the test set and any fold, or between validation sets across folds.

3) Augmentation

Data augmentation is performed after splitting and applied only to each fold's training set. The goal of augmentation is to increase the number of images used, thereby preventing the model from overfitting. The augmentation process uses the Roboflow platform by uploading the "train" folder, which contains images, labels, and a data yaml file. The augmentation techniques used are horizontal flipping, rotation $\pm 13^\circ$, hue $\pm 25^\circ$, exposure $\pm 15\%$, blur up to 1.2 pixels, noise up to 1.52%, and motion blur, resulting in 9.600 training images and 91.070 instances.

D. Transformation

This stage was performed on the Roboflow platform. After format conversion, the dataset was downloaded as a ZIP file containing the training data, labels, and a data.yaml.

E. Data Mining

The YOLO 8 through YOLO 12 models were trained on Google Colab (T4 GPU) with the same hyperparameters as shown in Table 3:

Table 3. Hyperparameter Values Used for the Training Process in Each YOLO Model

Hyperparameter	Value
Variance Model and Parameter	YOLO 8m: 25.9, YOLO 9c: 25.5, YOLO 10m: 15.4, YOLO 11m: 20.1 and YOLO 12m: 20.2
GFLOPs	YOLO 8m: 44.6, YOLO 9c: 58.5, YOLO 10m: 36.3, YOLO 11m: 38.6, YOLO 12m: 39.3
Pretrained Weights	COCO pretrained, fine-tuned via transfer learning
Optimizer	MuSGD (lr = 0.01, momentum = 0.9)
Learning Rate (LR)	lr0 = 0.01, lrf = 0.01
LR Scheduler	Linear decay (cos_lr = False)
Momentum/weight decay	0.937 (base) /0.0005
Warm up	3 epochs (momentum 0.8, bias LR 0.1)
Epoch	100
Batch Size	8
Image size or imgsiz	480 x 480 pixels
Patience	20
Confidence / IoU threshold	Default (0.001)/0.7
Software	Ultralytics 8.4.117, Python 3.12.13 PyTorch 2.11.0+cu128, CUDA 12.8
GPU and GPU memory	GPU Tesla T4, 15.6GB
System RAM	13.6 GB
Training duration	3 hours
Inference speed	16.2 ms/image

fixed test data and will not be used in the training or

So any differences in performance stem solely from their architectures. Each version was evaluated using 5-fold cross-validation to prevent data leakage: the data was divided into five parts four for training and one for validation on a rotating basis, and then the five results were averaged and their standard deviations calculated. Afterwards, the final model was retrained on the entire training dataset and evaluated using a bootstrap of 1,000 iterations due to the small initial sample size ($n=5$). The results were tested using the Kruskal-Wallis test to determine whether there were differences between models, followed by the Mann-Whitney U test with Bonferroni correction to identify which model pairs differed significantly. During training on Google Colab, checkpoints were saved periodically at the end of each epoch to prevent loss of progress due to technical issues such as Colab session disconnections [24], as shown in Fig.1, blocks 2 and 4. At this stage, the training process relies on several mathematical formulas, described in Eq.(1) through (4), that define how each YOLO model learns from the training data.

During training, each YOLO model minimises a weighted sum of three loss functions: L_{box} which improves bounding-box localization accuracy, L_{cls} which improves food-class prediction accuracy and L_{dfl} which refines object boundaries. The weights λ_{box} , λ_{cls} , and λ_{dfl} determine how much each loss contributes to the total loss, as defined in Eq.(1) [25]:

$$L_{total} = \lambda_{box}L_{box} + \lambda_{cls}L_{cls} + \lambda_{dfl}L_{dfl} \quad (1)$$

The bounding box regression component uses the Complete-IoU ($CIoU$) loss, which evaluates prediction boxes based on overlap, center-to-center distance, and width-to-height ratio, where $1 - IoU$ represents the overlap penalty, $\frac{\rho^2(b, b^{gt})}{c^2}$ is used as a penalty for the squared distance between the center of the prediction box b and the ground truth b^{gt} , normalized by the diagonal of the smallest enclosing box c^2 , while v measures the consistency of the width-to-height ratio and a serves as the balancing weight, as defined in Eq.(2) [26]:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + av \quad (2)$$

Food class prediction uses the Binary Cross-Entropy (BCE) loss, which decreases as the predicted probability approaches the true label and increases when the prediction is incorrect, where N is the number of samples, y_i is the true label (0 or 1), $\hat{y}_i$ is the predicted probability, and $\log$ is the natural logarithm, as defined in Eq.(3) [27]:

$$BCE = -\frac{1}{N} \sum_{i=1}^N y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \quad (3)$$

Distribution Focal Loss (DFL) improves bounding box prediction accuracy. The true object boundary position y may lie between two prediction intervals, y_i and y_{i+1} . DFL encourages the model to assign a higher probability to the interval closest to y , thereby allowing the object boundary position to be determined more accurately, where S_i and S_{i+1} are the softmax probabilities for those two intervals, as defined in Eq.(4) [28]:

$$DFL = -((y_{i+1} - y) \log(S_i) + (y - y_i) \log(S_{i+1})) \quad (4)$$

F. Evaluation

The mAP50 values of the five YOLO versions were

analyzed using the Kruskal-Wallis test and the Mann-Whitney post-hoc test with Bonferroni correction to determine whether the differences in performance among the models were statistically significant and to identify the best model. Detection performance is evaluated using standard metrics, starting with Intersection over Union (IoU), which measures the overlap between the predicted bounding box B and the ground truth B^{gt} . Predictions are classified as true positives (TP) if the IoU value exceeds a certain threshold and as false positives (FP) if it does not, while undetected objects are classified as false negatives (FN), IoU is defined in Eq.(5) [26]:

$$IoU = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|} \quad (5)$$

Based on the TP , TP , and FN values, three metrics are derived. Precision is TP divided by the total number of model detections ($TP + FP$), so the fewer the FP , the higher the value. Recall is TP divided by the total number of actual objects ($TP + FN$), so the fewer objects missed, the higher the value. The $F1 Score$ is twice the product of Precision and Recall divided by their sum, so it is high only if both are high. These three metrics are defined in Eq.(6, 7, 8) [27]:

$$Precision = \frac{TP}{(TP + FP)} \quad (6)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (7)$$

$$F1 Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (8)$$

Average Precision (AP) is calculated as the area under the Precision-Recall curve for each class, where $p(r)$ is the precision at recall r and dr is the differential recall. The mAP50 value is the average AP across N classes at an IoU threshold of 0.5, where i is the class index, as defined in Eq.(9, 10) [29]:

$$AP = \int_0^1 p(r) dr \quad (9)$$

$$mAP_{50} = \frac{1}{N} \sum_{i=1}^N AP_N |IoU = 0.5 \quad (10)$$

To obtain a more reliable estimate of the model's performance, 5-fold cross-validation was performed. The mean ($\bar{x}_m$) and standard deviation (s_m) of each metric m were calculated from the five folds, where $x_{m,k}$ is the value of metric m in the k th fold, and the divisor 4 is derived from the number of folds minus one ($5 - 1$), as defined in Eqs.(11, 12) [30]:

$$\bar{x}_m = \frac{1}{5} \sum_{k=1}^5 x_{m,k} \quad (11)$$

$$s_m = \sqrt{\frac{1}{4} \sum_{k=1}^5 (x_{m,k} - \bar{x}_m)^2} \quad (12)$$

To obtain a more stable estimate of model performance, this study applied bootstrap resampling to 240 test images. From the set of images x , 1,000 bootstrap samples X_b^* were generated with replacement, and the metric θ^* was calculated for each sample. The 95% confidence interval is estimated from the 2.5th ($P_{2.5}$) and 97.5th ($P_{97.5}$) percentiles of the bootstrap results, as

defined in Eq.(13, 14) [31]:

$$X_b^* = \{x_1^*, x_2^*, \dots, x_n^*\}, b = 1, \dots, 1000 \quad (13)$$

$$CI_{95\%} = [P_{2.5}(\theta^*), P_{97.5}(\theta^*)] \quad (14)$$

The five mAP50 values obtained for each version of YOLO were analysed using the Kruskal-Wallis nonparametric test with the H test statistic, where N is the total number of observations, n_i is the number of observations in group j , $R_{i,j}^2$ is the number of ranks in group j , and k is the number of groups, as defined in Eq.(15) [32]:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^k \frac{R_i^2}{n_i} - 3(N+1) \quad (15)$$

If the Kruskal-Wallis test indicates a significant difference, use the Mann-Whitney U test as a post-hoc test between groups. The reported U value is the smaller of U_1 and U_2 , where R_1 is the sum of ranks for the first group, and n_1 and n_2 are the sample sizes of each group, defined in Eq.(16, 17) [33]:

$$U_1 = R_1 - \frac{n_1(n_1+1)}{2}, U_2 = n_1n_2 - U_1 \quad (16)$$

$$U = \min(U_1, U_2) \quad (17)$$

Since there are 10 pairwise comparisons among the five versions of YOLO, the Bonferroni correction was applied to control the family-wise error rate by adjusting the significance level. The null hypothesis is rejected, and the difference is declared statistically significant if the p-value is less than the adjusted significance level, where α is the initial significance level (0.05), m is the number of pairwise comparisons ($m = 10$), α_{adj} is the adjusted significance level ($0.05 / 10 = 0.005$), and p is the p-value from the Mann-Whitney U test, defined in Eqs.(18, 19) [32]:

$$\alpha_{adj} = \frac{\alpha}{m} \quad (18)$$

$$\text{Reject } H_0 \text{ when } p < \alpha_{adj} \quad (19)$$

To evaluate the accuracy of the nutritional estimates, the system's calculations were compared with the assessments of an MBG nutritionist using 30 sample photos per nutrient. In the formula, i denotes the sample order (1 to n) that is summed, y_i denotes the nutrient value from the nutritionist for the i -th sample used as a reference, $\hat{y}_i$ denotes the system's estimated nutrient value for the same sample; and n is used as the divisor for the average ($n = 30$). Mean Absolute Error (MAE) is the average absolute difference between the two values. Root Mean Square Error (RMSE) squares the differences so that larger errors have a greater impact. Mean Absolute Percentage Error (MAPE) expresses the difference as a percentage relative to y_i . Agreement is the percentage of samples with the same nutritional status ("Fulfilled" or "Insufficient"). These four metrics are defined in Eq.(20, 21, 22) [34]:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (20)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (21)$$

$$MAPE = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (22)$$

G. Nutritional Adequacy Calculation

Nutritional values in the TKPI are reported per 100 grams. Since food weight cannot be measured from images, each detected food class is assigned a serving weight (Serv. Wt.) per instance from the DBMP and Porsimetri datasets Table 3. In the formula, i denotes the food class, c_i the number of detected instances, W_i the serving weight per instance, and B_i the total weight of class i

$$B_i = c_i \times W_i \quad (23)$$

In Eq.(23) [35], B_i is the total weight of class i food (g), and c_i is the number of bounding boxes for class i detected by YOLO in a single image. The TKPI value is then converted to a total nutritional value using:

$$N_{i,c} = \frac{B_i}{100} \times TKPI_{i,c} \quad (24)$$

In Eq.(24) [35], $N_{i,c}$ is nutrient c (energy, protein, fat, carbohydrate, fiber, calcium, iron) from the total detected instances of food class i . The total intake for the n detected food classes is:

$$Total_c = \sum_{i=1}^n N_{i,c} \quad (25)$$

In Eq.(25) [36], this value is compared to the daily requirement AKG_c , which is determined based on school level, age, and gender in accordance with the regulation of the Minister of Health of the Republic of Indonesia No. 28 of 2019 (replacing Regulation No. 75 of 2013). Since a single MBG meal provides only a portion of the daily intake, the target per meal is set at 30% of AKG_c , where c denotes the type of nutrient, $Total_c$ is the total content of nutrient c in the detected menu, and $Percentage_c$ is the percentage of the target fulfillment for that nutrient. In Eq.(26, 27) [2]:

$$AKG_c: Target_c = AKG_c \times 0.30 \quad (26)$$

$$Percentage_c = \frac{Total_c}{Target_c} \times 100\% \quad (27)$$

Status is "Fulfilled" if $Total_c \geq Target_c$, and "Insufficient" if $Total_c \leq Target_c$.

III. Results

A. YOLO Model Performance Comparison

In Table 4, the 5-fold cross-validation evaluation divided the data into five subsets to prevent a single data split from biasing the results, which could cause overfitting or underfitting if the model is evaluated using only one training/validation split. This approach ensures that the estimates of the model's performance and stability are more representative. The results show that all five YOLO models perform similarly across all metrics, with differences ranging from about 0.006 to 0.013.

For the F1-Score, YOLO 12 and YOLO 9 tied for the highest score (0.875), followed by YOLO 8 (0.873), YOLO 11 (0.872), and YOLO 10 (0.869). For Precision, YOLO 12 leads (0.902), followed by YOLO 11 (0.892), YOLO 8 (0.891), YOLO 10 (0.890), and YOLO 9 (0.889). For Recall, YOLO 9 had the highest value (0.860), followed by YOLO 8 (0.856), YOLO 11 (0.854), YOLO 12 (0.850), and YOLO 10 (0.848). For mAP50, YOLO 8 and YOLO 9 tied for the highest score (0.898), followed by YOLO 11 (0.896), YOLO 12 (0.894), and YOLO 10 (0.888).

Overall, YOLO 12 ranked highest among the others in Precision and F1-Score, but ranked only fourth in Recall. YOLO 9 records the highest Recall and ties with YOLO 8 in mAP50. YOLO 8 has the highest mAP50, although it

never leads in any other metric. YOLO 11 is neither the highest nor the lowest in any metric. YOLO 10 ranked lowest in F1-Score, Recall, and mAP50, while ranking fourth in Precision, above YOLO 9.

Table 4. Descriptive Statistics: Min, Max, Average and Stdev of Performance Metrics Obtained for Each YOLO Model

Model	Metric	Min	Max	Average	Stdev
YOLO 8	F1-Score	0.862	0.890	0.873	0.011
	Precision	0.881	0.905	0.891	0.009
	Recall	0.841	0.876	0.856	0.013
	mAP50	0.885	0.925	0.898	0.017
YOLO 9	F1-Score	0.863	0.899	0.875	0.014
	Precision	0.879	0.912	0.889	0.012
	Recall	0.840	0.882	0.860	0.017
	mAP50	0.882	0.924	0.898	0.017
YOLO 10	F1-Score	0.856	0.886	0.869	0.012
	Precision	0.864	0.918	0.890	0.021
	Recall	0.835	0.856	0.848	0.008
	mAP50	0.873	0.914	0.888	0.016
YOLO 11	F1-Score	0.860	0.900	0.872	0.016
	Precision	0.880	0.913	0.892	0.012
	Recall	0.835	0.888	0.854	0.020
	mAP50	0.880	0.929	0.896	0.019
YOLO 12	F1-Score	0.860	0.901	0.875	0.015
	Precision	0.889	0.919	0.902	0.013
	Recall	0.831	0.884	0.850	0.020
	mAP50	0.878	0.928	0.894	0.020

After cross-validation, the model is retrained on the entire training dataset and then tested once on 240 test data points it has never seen before. This evaluation is called a single test because it is performed only once on a single combination of those 240 test data points, so the result is a single value per model. Because it is a single test, this value alone cannot indicate how stable the model's performance would be on other data combinations; therefore, a bootstrap test resamples the single test results 1.000 times.

Table 5. Comparison of Point Estimates and 95% Confidence Intervals (CI) YOLO 8- YOLO 12

Model	Metric	Single Test	Bootstrap			
			Point Estimate	CI 95% Lower	CI 95% Upper	SD
YOLO 8	Precision	0.904	0.911	0.883	0.934	0.014
	Recall	0.868	0.868	0.841	0.896	0.014
	F1-Score	0.886	0.889	0.871	0.907	0.008
	mAP50	0.899	0.904	0.883	0.926	0.011
YOLO 9	Precision	0.914	0.914	0.895	0.929	0.008
	Recall	0.861	0.867	0.843	0.892	0.012
	F1-Score	0.887	0.890	0.872	0.907	0.009
	mAP50	0.895	0.900	0.878	0.924	0.011
YOLO 10	Precision	0.906	0.902	0.880	0.921	0.010
	Recall	0.863	0.852	0.827	0.879	0.013
	F1-Score	0.884	0.876	0.857	0.894	0.009
	mAP50	0.880	0.893	0.870	0.916	0.012
YOLO 11	Precision	0.898	0.899	0.876	0.925	0.011
	Recall	0.864	0.868	0.843	0.892	0.012
	F1-Score	0.881	0.883	0.863	0.902	0.009
	mAP50	0.893	0.897	0.877	0.919	0.011
YOLO 12	Precision	0.886	0.890	0.868	0.913	0.011
	Recall	0.865	0.866	0.837	0.895	0.014
	F1-Score	0.875	0.878	0.858	0.895	0.009
	mAP50	0.889	0.894	0.870	0.917	0.012

The average of the 1.000 resulting values serves as the point estimate, while the range from the 2.5th to the 97.5th percentile constitutes the 95% confidence interval (CI). The single test results and the bootstrap point estimate differ only slightly across models, ranging from 0.000 to 0.013. Such a small difference indicates that the performance obtained from a single test on 240 test data points falls within the same range as the results of 1.000 resampling runs; therefore, the bootstrap point estimate can be used as a more reliable basis for comparing model

Manuscript received 7 August 2026; Revised 15 August 2026; Accepted 30 September 2026; Available online 20 October 2026

DOI: <https://doi.org/10.35882/ijeemi.v8i4.382>

Copyright © 2026 by the authors. Published by Jurusan Teknik Elektromedik, Politeknik Kesehatan Kemenkes Surabaya Indonesia. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

performance than relying solely on the results of a single test.

The bootstrap test results in Table 5 show that for Precision, YOLO 9 has the highest value (0.914), followed by YOLO 8 (0.911), YOLO 10 (0.902), YOLO 11 (0.899), and YOLO 12 (0.890). For Recall, YOLO 8 and YOLO 11 both have the highest values (0.868), followed by YOLO 9 (0.867), YOLO 12 (0.866), and YOLO 10 (0.852). For F1-Score, YOLO 9 has the highest value (0.890), followed by YOLO 8 (0.889), YOLO 11 (0.883), YOLO 12 (0.878), and YOLO 10 (0.876). For mAP50, YOLO 8 has the highest score (0.904), followed by YOLO 9 (0.900), YOLO 11 (0.897), YOLO 12 (0.894), and YOLO 10 (0.893).

Table 6. Kruskal-Wallis Test Results for Overall Performance Comparison Among YOLO Models

Metric	H Statistic	df	p-value	Decision ($\alpha=0.05$)
F1-Score	1299.340	4	0.000	Significant (H_0 rejected)
Precision	1800.138	4	0.000	Significant (H_0 rejected)
Recall	797.182	4	0.000	Significant (H_0 rejected)
mAP50	489.946	4	0.000	Significant (H_0 rejected)

The Kruskal–Wallis test was used instead of ANOVA because the bootstrap metric values were not normally distributed and were concentrated at high values. The hypotheses tested were H_0 : no significant difference in performance among the five YOLO models, and H_1 : a significant difference in performance among the five YOLO models. Based on Table 6, all metrics had p-values < 0.001. The test statistics obtained were 1299.340 for

The differences between models are relatively small, ranging from 0.011 to 0.024 across metrics. No single model ranked highest across all metrics: YOLO 9 ranked highest in Precision and F1-Score, YOLO 8 ranked highest in Recall and mAP50, YOLO 11 never ranked highest or lowest on any metric, YOLO 12 ranked lowest in Precision, and YOLO 10 ranked lowest in Recall, F1-Score, and mAP50. Since no clear winner emerged, Kruskal-Wallis and Mann-Whitney U statistical tests were needed to determine whether these performance differences were statistically significant [37].

F1-Score, 1800.138 for Precision, 797.182 for Recall, and 489.946 for mAP50. As all p-values are less than 0.05, H_0 is rejected, indicating a significant difference in performance amongst the YOLO models. The very small p-values result from the large number of bootstrap samples ($n = 1.000$), which makes even small performance differences statistically detectable.

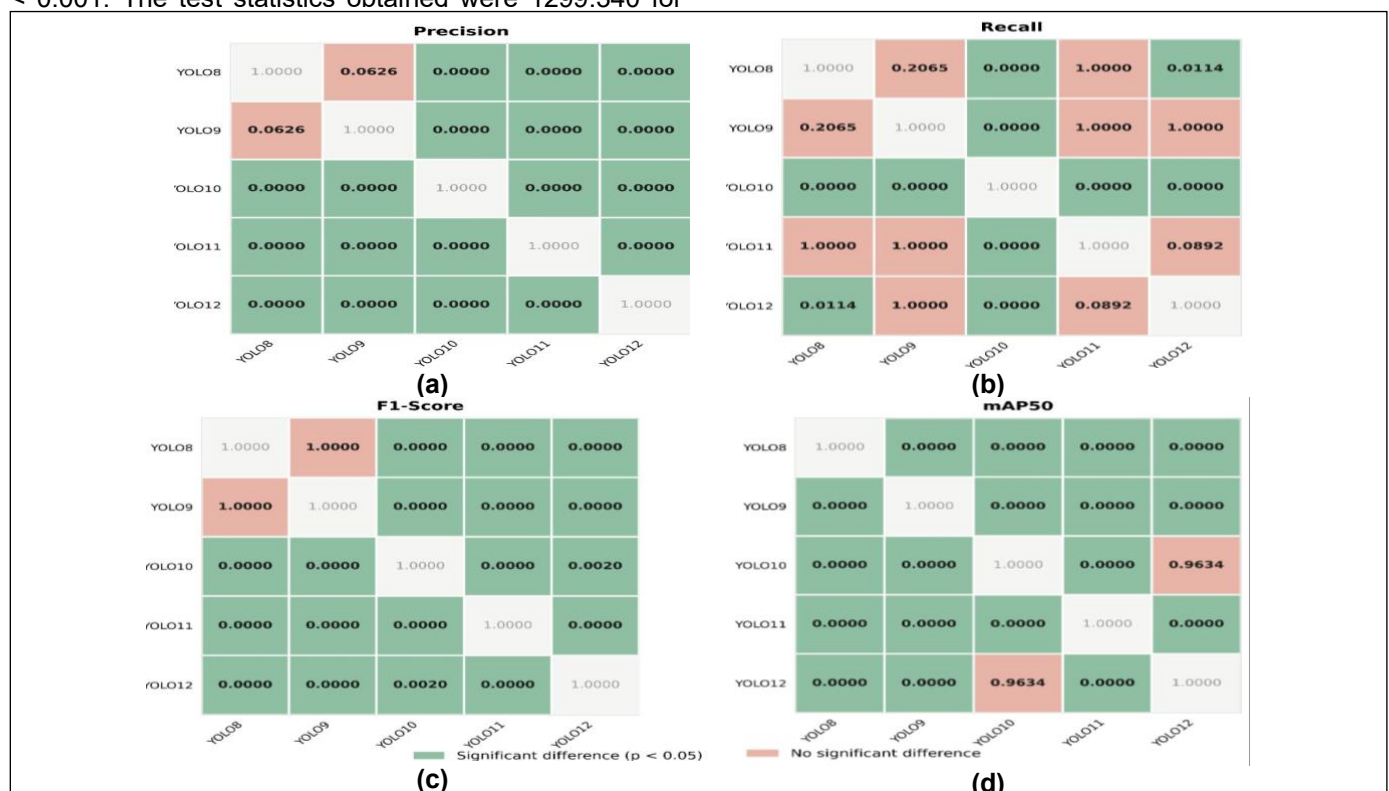


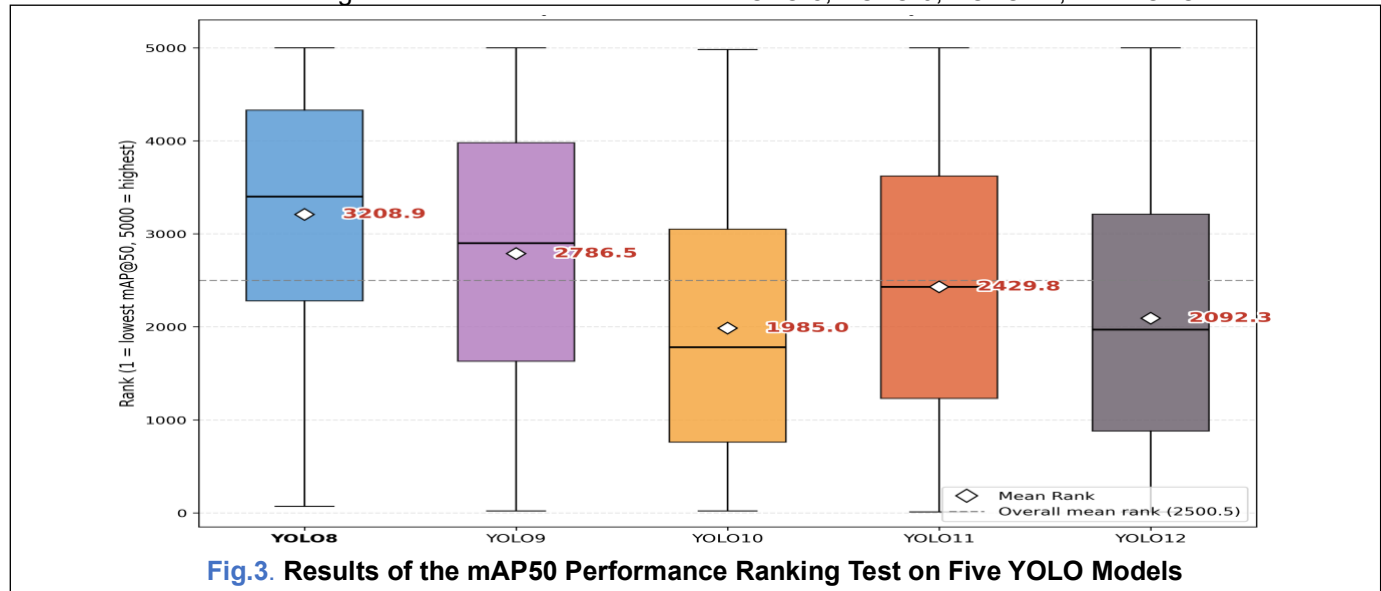
Fig.2. Post-Hoc Mann–Whitney U Test with Bonferroni Correction Across YOLO Models: a. Precision, b. Recall, c. F1-Score, and d. mAP50.

The Mann–Whitney U post-hoc test was used to identify specific differences between model pairs following the Kruskal–Wallis test, since the bootstrap data were not normally distributed. The Bonferroni correction was applied to the 10 comparison pairs to control the family-wise error rate. The post-hoc results show that YOLO 8

and YOLO 9 are the most similar model pair, with no significant differences in Precision in Fig.2a ($p = 0.062$), Recall in Fig.2b ($p = 0.206$), or F1-Score in Fig.2c ($p = 1.0000$). This indicates that the architectural improvement from YOLO 8 to YOLO 9 does not significantly alter these three metrics. In contrast, YOLO 10 shows the most

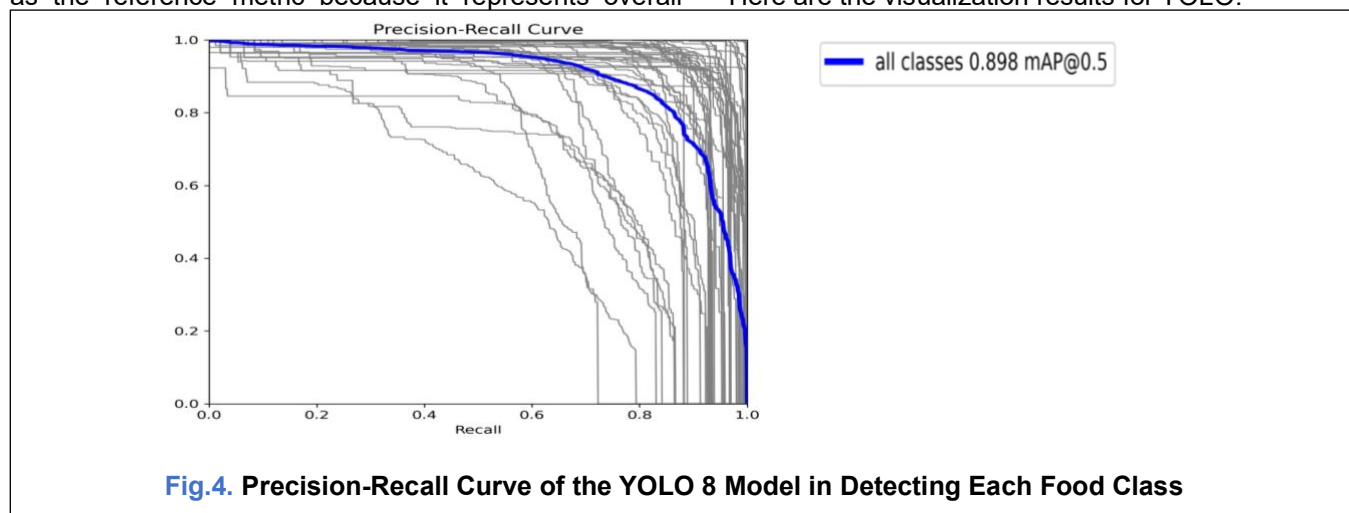
distinctive Recall, as it differs significantly from all other models, indicating that its performance in detecting all objects is statistically different from that of other YOLO versions. For the mAP50 metric in Fig.2d, only YOLO 10 and YOLO 12 showed no significant difference ($p = 0.963$), meaning both have comparable object detection accuracy despite differing on other metrics. Furthermore, YOLO 8, YOLO 9, YOLO 11, and YOLO 12 tend to have similar Recall values, as most pairs among these models show no significant differences.

Overall, Precision, F1-Score, and mAP50 better distinguish performance across YOLO versions, as nearly all model pairs differ significantly. Conversely, Recall is more similar across models and therefore less sensitive for distinguishing performance. These findings indicate that the main differences between YOLO versions are more evident in object detection accuracy and Precision. In contrast, the ability to identify objects in images is relatively similar across several models, particularly YOLO 8, YOLO 9, YOLO 11, and YOLO 12.



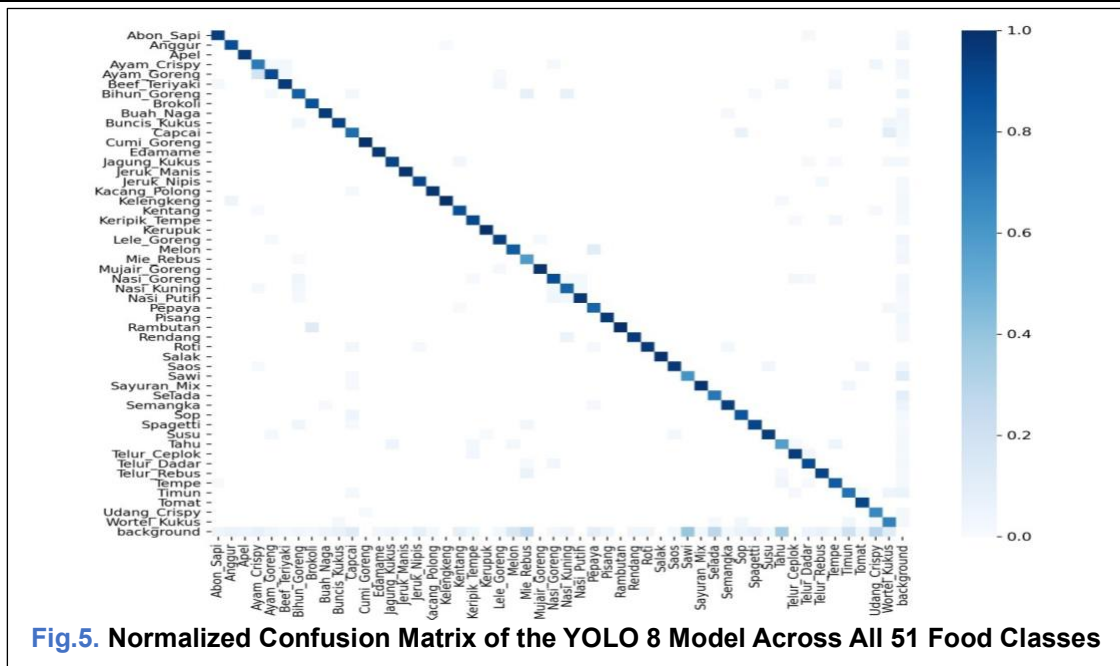
Since no single model consistently outperformed the others across all metrics, the bootstrap mAP50 results (5 models $\times$ 1.000 iterations) are ranked from 1 to 5.000 in Fig.3, with an overall average rank of 2,500.5. Mean rank is used as a benchmark because it is more robust to outliers than the simple average, while mAP50 is chosen as the reference metric because it represents overall

object detection accuracy. YOLO 8 recorded the highest mean rank of 3,208.9, well above the overall average, followed by YOLO 9 at 2,786.5, YOLO 11 at 2,429.8, YOLO 12 at 2,092.3, and YOLO 10 as the lowest at 1,985.0. This highest mean rank value served as the basis for selecting YOLO 8 as the model used by the system. Here are the visualization results for YOLO:



The Precision-Recall curve in Fig.4 shows the YOLO 8 model's detection performance at each confidence threshold, based on a single test evaluation on 240 test data points. This evaluation produced predictions per image, which served as the basis for the Precision-Recall curve, the confusion matrix, and the performance table by class. The results of this single test evaluation show a precision of 0.904, a recall of 0.868, an F1 score of 0.886, and an mAP50 of 0.899. The average curve (blue line)

stays in the high-precision region up to a fairly high recall before dropping sharply as it approaches a recall of 1.0, indicating that the model can generally detect food objects accurately before its performance weakens on more difficult or less certain predictions. Meanwhile, the gray lines (per-class curves), which decline earlier in the low-recall region, indicate performance variation across classes, likely due to classes with fewer samples or objects that are harder to recognize.



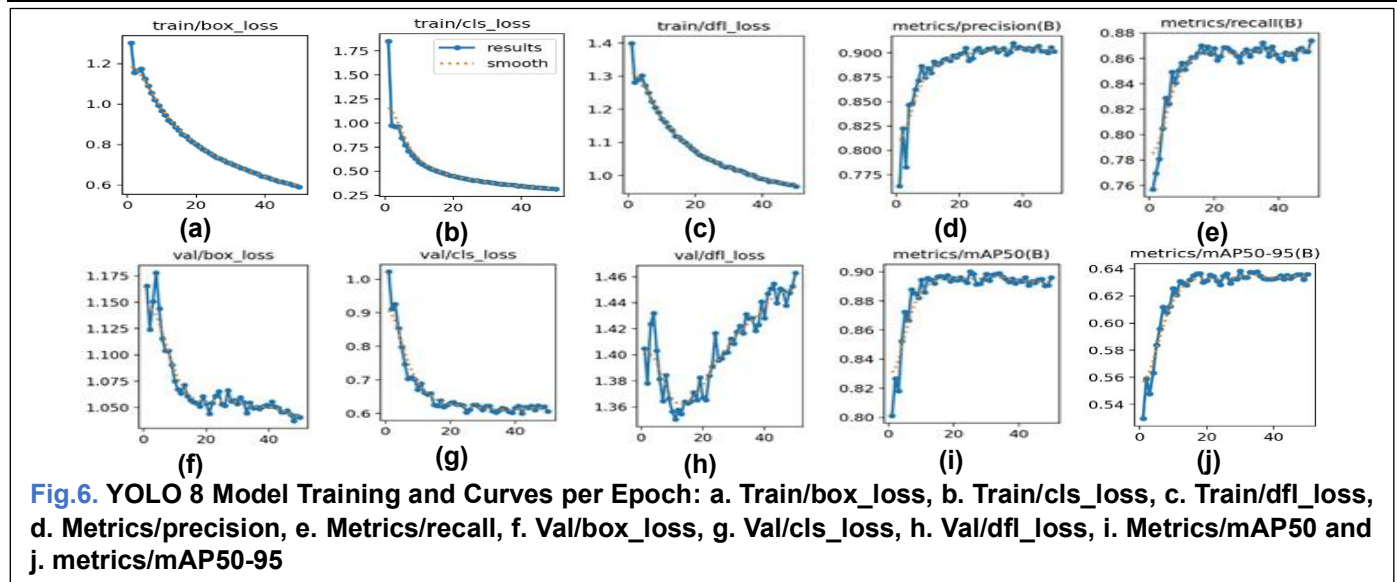
In Fig.5 overall, YOLO 8 performed excellently on most food classes, with six classes achieving a perfect mAP50 score of 0.995, namely Milk, Apple, Grape, Broccoli, Lime, and Dragon Fruit, and most other classes scoring above 0.850. However, some classes exhibited two distinct error patterns. First, detection failure in Green_Peas showed high Precision (0.938) but low Recall (0.329). This was caused by the small object size and their tendency to cluster, making it difficult for the model to distinguish their boundaries. Second, the model misclassified

Crispy_Chicken and Fried_Chicken. Both recorded an AP50 below 0.830 due to similarities in shape and color. Fried_Rice_Vermicelli also had the lowest Precision at 0.616 because other classes were often misclassified as this class. These two patterns generally appear in classes with high visual similarity or small object sizes; they do not reflect a general weakness of the model, as evidenced by the rice and egg groups, which remain consistently strong with an mAP50 above 0.890. Table 7 shows the data distribution.

Table 7. Precision, Recall, and mAP Results per Class to Complement the Confusion Matrix Analysis

No	Class Name	Precision	Recall	mAP50
1	Beef Floss	0.989	0.944	0.967
2	Grape	0.987	1.000	0.995
3	Apple	0.989	1.000	0.995
4	Crispy Chicken	0.777	0.819	0.828
5	Fried Chicken	0.728	0.743	0.822
6	Teriyaki Beef	0.866	0.975	0.949
7	Fried Vermicelli	0.616	0.853	0.841
8	Broccoli	0.991	1.000	0.995
9	Dragon Fruit	0.953	0.920	0.936
10	Steamed Green Beans	0.861	0.841	0.908
11	Capcai	0.855	0.903	0.920
12	Fried Squid	0.976	0.964	0.993
13	Edamame	0.935	0.998	0.991
14	Steamed Corn	0.967	0.951	0.976
15	Sweet Orange	0.977	0.889	0.963
16	Lime	0.975	1.000	0.995
17	Green Peas	0.938	0.329	0.360
18	Longan	0.744	0.852	0.823
19	Potatoes	0.914	1.000	0.995
20	Tempeh Chips	0.861	0.857	0.871
21	Crackers	1.000	0.926	0.965
22	Fried Catfish	0.830	0.977	0.954
23	Melon	0.881	0.841	0.882
24	Boiled Noodles	0.806	0.667	0.717
25	Fried Tilapia	0.966	0.989	0.993

No	Class Name	Precision	Recall	mAP50
26	Fried Rice	0.883	0.997	0.974
27	Yellow Rice	0.973	0.925	0.992
28	White Rice	0.967	0.957	0.986
29	Papaya	0.974	0.865	0.937
30	Bananas	0.916	0.957	0.953
31	Rambutan	0.982	0.987	0.978
32	Rendang	0.937	0.816	0.847
33	Bread	0.978	0.900	0.902
34	Snake Fruit	0.976	1.000	0.995
35	Sauce	0.826	0.881	0.923
36	Mustard Greens	0.853	0.647	0.764
37	Mix Vegetables	0.976	0.971	0.994
38	Lettuce	0.741	0.593	0.693
39	Watermelon	0.933	0.966	0.987
40	Soup	0.811	0.868	0.905
41	Spaghetti	0.871	0.824	0.898
42	Milk	1.000	0.990	0.995
43	Tofu	0.828	0.767	0.827
44	Fried Egg	0.937	0.956	0.962
45	Omelette	0.901	0.878	0.896
46	Boiled Egg	0.926	0.895	0.963
47	Tempeh	0.900	0.800	0.842
48	Cucumber	1.000	0.577	0.779
49	Tomatoes	0.962	0.901	0.934
50	Crispy Shrimp	0.908	0.563	0.656
51	Steamed Carrot	0.758	0.594	0.639

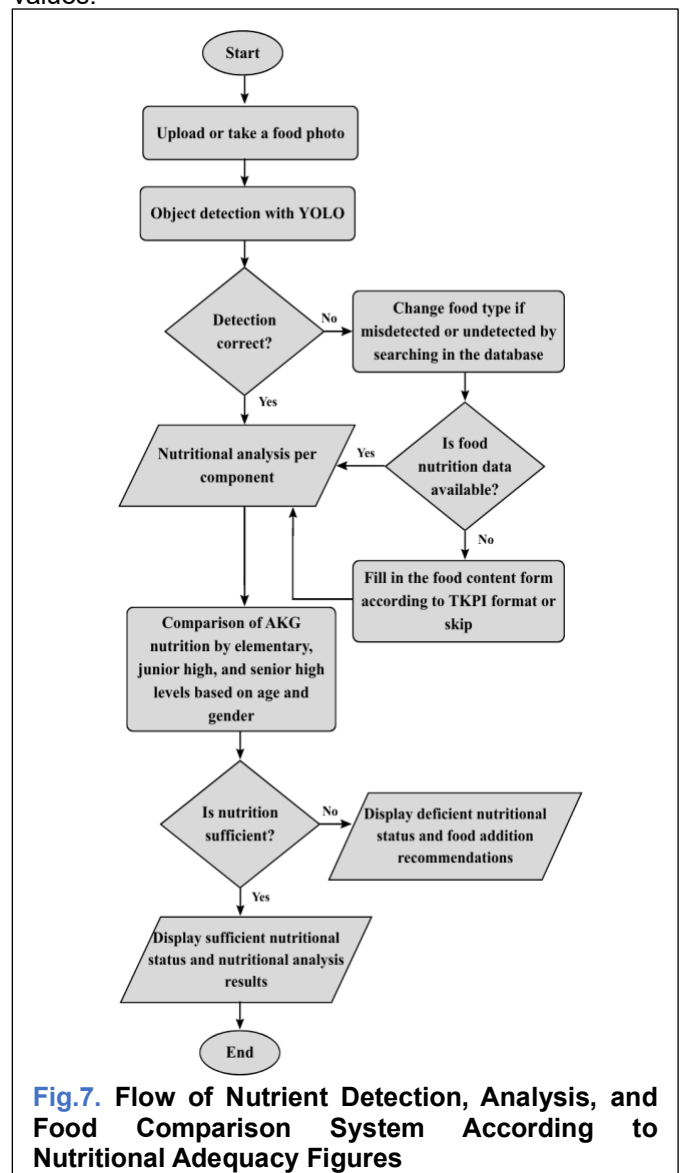


Loss curves and performance metrics during YOLO 8 training and validation. Fig.6a box_loss shows the error in predicting bounding box position and size. Fig.6b cls_loss shows the error in classifying food types. Fig.6c, df_l_loss, shows the error in refining the bounding box. All three values decrease consistently until they stabilize, indicating that the learning process is proceeding smoothly. Fig.6d, Precision, shows the proportion of correct detections among all predictions, while Fig.6e, Recall, shows the proportion of actual food objects that were successfully detected. Both metrics increase rapidly and then remain stable. On the validation data, Fig.6f, Fig.6g and Fig.6h val/df_l_loss represent the same three loss metrics train/box_loss, train/cls_loss, train/df_l_loss but measured on previously unseen data. val/box and val/cls decrease and stabilize, while val/df_l_loss decreases around the tenth epoch and then gradually increases, indicating mild overfitting. However, this does not significantly affect overall detection performance, as shown in mAP50 Fig.6i and mAP50-95 Fig.6j which represent detection accuracy at an IoU threshold of 0.5 remained high and stable until the end of the training process.

B. System Development

At this stage, the entire design from the RAD phase is implemented as a nutritional analysis system, based on the system development results. The system workflow shown in Fig.7 begins when the user uploads or takes a photo of food, which the YOLO model then processes to detect the food type automatically. If the detection results are incorrect or if some food items are not detected, the user can correct them using the database search feature. If nutritional data for that food item are not yet available, the system provides a form to fill out in the TKPI format, or the user can skip this step. Once the food data are confirmed, the system displays a nutritional analysis by food component. Then it compares total intake to the AKG for each educational level (elementary, junior high, senior high) based on age and gender. If the comparison results indicate that nutritional needs are not met, the system will display an “Insufficient” status along with recommendations for additional foods. Conversely, if nutritional needs are met, the system displays an

“Fulfilled” status along with the complete nutritional analysis results. This workflow includes four main system features: the home page, revision of detection results, results of each component, and comparison of nutritional values.



1) Home Page

The home page appears when the user opens the system. This page displays two buttons: Take Photo, which opens the device's camera, and Upload, which opens the gallery. The user's submitted photos serve as input data for the YOLO detection process. Fig.8a shows the home page.

2) Revision of Detection Results

This feature lets users correct incorrectly detected foods or add food types the system did not detect using the "Revise" button. If a detection error occurs, users can click on the food's name, and the system will display a search menu within the database. If the food type in question is not yet available in the database, users can enter the information manually via the "Unknown" form by filling in the food's name and nutritional content, or skip this step. Meanwhile, if a food item is not detected, users can click the "Add Component" menu, then draw a bounding box around the food item to search the database for the appropriate food type. If that food item is also not available in the database, users can fill out the "unknown" form in the same way by entering the food's name and its nutritional content. This feature helps reduce YOLO detection errors, thereby improving the accuracy of nutritional analysis. However, portion size, recipe variations, and food preparation methods can still affect the results, as shown in Fig.8b.

3) Results of Each Component

This feature displays detailed nutritional information for each food type detected by TKPI, including energy, carbohydrates, protein, fat, zinc, fiber, and calcium. This allows users to view the nutritional content of each food type, as shown in Fig.8b.

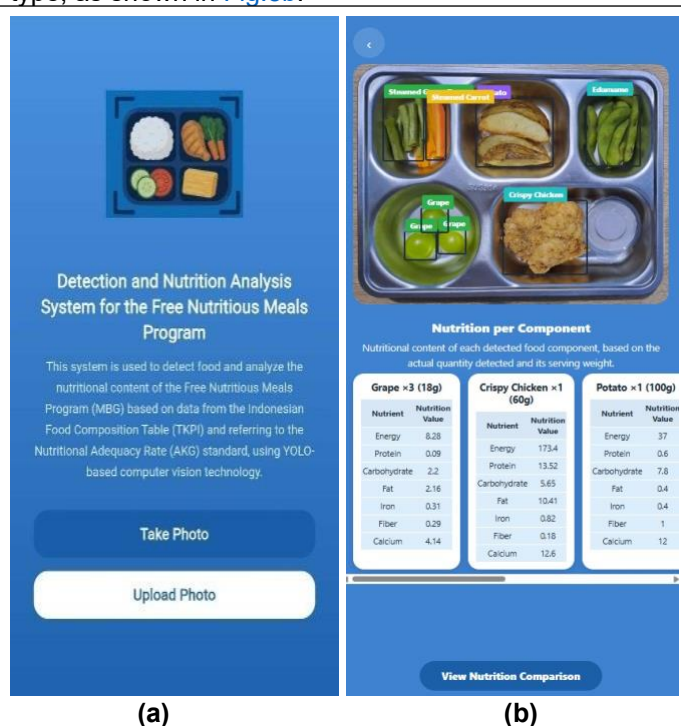


Fig.8. Display of the System Interface: a. Home Page, and b. Nutrient Detection and Analysis Results

4) Comparison of Nutritional Values

This feature displays the total nutritional content of all detected foods and compares it to the AKG-based one by educational level, age, and gender, as shown in Fig.9a.

The results are displayed as a percentage of the 30% daily nutritional requirement target. The nutritional content calculated in this study is based on standard portion assumptions for each food class, as referenced in the TKPI, rather than on the actual measured mass or volume of the food captured in the image. This approach was adopted because the current system lacks depth sensing or volumetric estimation capability and therefore provides an approximation of nutritional adequacy rather than a precise measurement. If there is a lack of nutritional content, the system will provide recommendations for nutritional fulfilment, as shown in Fig.9b.

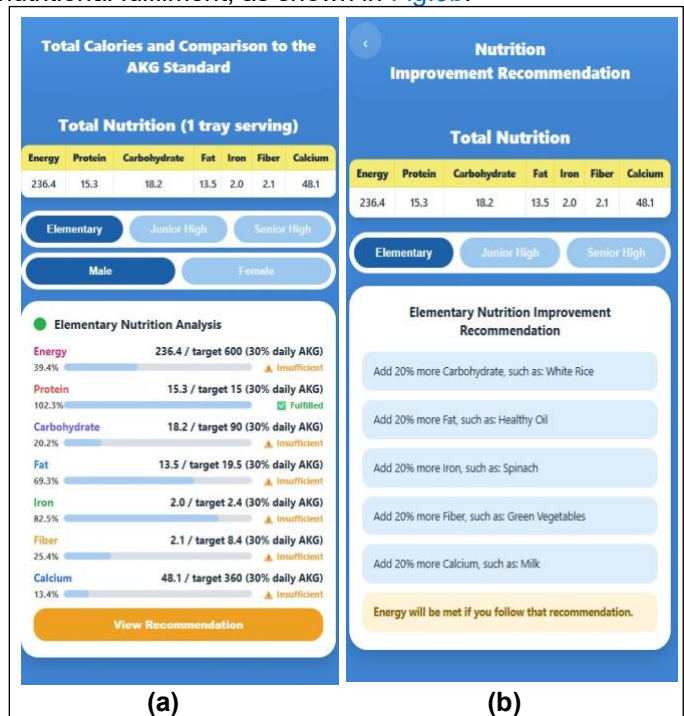


Fig.9. Display of the System Interface: a. AKG Comparison Results at Each Level of Elementary, Junior High, and High School Education According to Gender and b. Provision of Nutrition Fulfilment Recommendation

C. Testing System

System testing was conducted in two stages: black-box testing and a comparison of the system's results with those of a nutritionist using the MAE, RMSE, and MAPE metrics. Black-box testing was used to ensure that each feature functioned according to the RAD functional design in Table 8 without examining the code structure, while MAE, RMSE, and MAPE were used to measure the system's nutritional calculation error rates relative to the nutritionist's reference values. Of the seven black-box scenarios, all produced results as expected. The "Take Photo" and "Upload" features successfully opened the camera and gallery, and the system successfully processed photos using YOLO and generated bounding boxes. When revising the detection results, users can correct misidentified foods, add undetected menu items, and fill out the "unknown" form, which is successfully saved. The system also successfully displays nutritional analysis by component, total nutritional values, and a comparison with the AKG, along with recommendations if the nutritional status is classified as deficient.

Table 8. Black Box Testing Results on Each Feature

No	Testing Scenarios	Expected results	Results
1.	Take photos of food	System opens the camera to take photos of MBG trays	✓
2.	Upload food photos	The system opens the gallery to select a photo.	✓
3.	Object detection using YOLO	The system processes user-submitted photos with YOLO and generates <i>bounding boxes</i> .	✓
4.	Revision of detection results	a. The user presses the wrong food <i>bounding box</i> , and the system <i>searches the database</i> to replace the incorrectly detected food type; the results appear in the nutrition analysis. b. The user presses the button to add the missed menu item, and then the system <i>searches the database</i> to determine the type of food that was missed. c. The system displays an unknown form if the food type searched in the database does not exist. The user fills in the form, and the system successfully saves it and displays it.	✓ ✓ ✓
5.	Component nutrition analysis	The system displays the nutritional results of the components.	✓
6.	Calculation total nutritional component	The system correctly displays the total nutritional content of the components.	✓
7.	AKG Comparison	The system displays the status of nutritional comparison analysis and additional recommendations if the nutritional status is deficient.	✓

In addition to functional testing, quantitative analysis was performed using MAE, RMSE, and MAPE on 30 sample photos of MBG trays with menus tailored as shown in Table 2, and the system's estimates were then compared with the assessments of MBG nutrition experts as listed in Table 9. The overall MAE was 30.55, RMSE was 63.64, and MAPE was 40.31%. Energy (20.86%) and carbohydrates (21.08%) had the lowest MAPE values, indicating closer agreement with the nutritionist. In comparison, fat (50.94%) and fiber (68.91%) showed the highest errors because RGB images have limited ability to estimate the actual mass or volume of food. Compared with studies [38] and [39], this system achieved comparable estimates for energy and better estimates for

carbohydrates, although protein and fat still showed higher errors. These differences may be due to the difficulty of using RGB images to identify food preparation methods and the food's actual weight. Of the seven nutrients calculated, only five were validated because nutritionist reference data for calcium and iron were not available. The MAE, RMSE, and MAPE results show that fat and fiber had the highest errors, at 50.94% and 68.91%, respectively. This is likely because the reference serving weights were used without accounting for the food's actual mass or volume. Therefore, a sensitivity analysis was conducted to determine the extent to which changes in serving weight affect AKG.

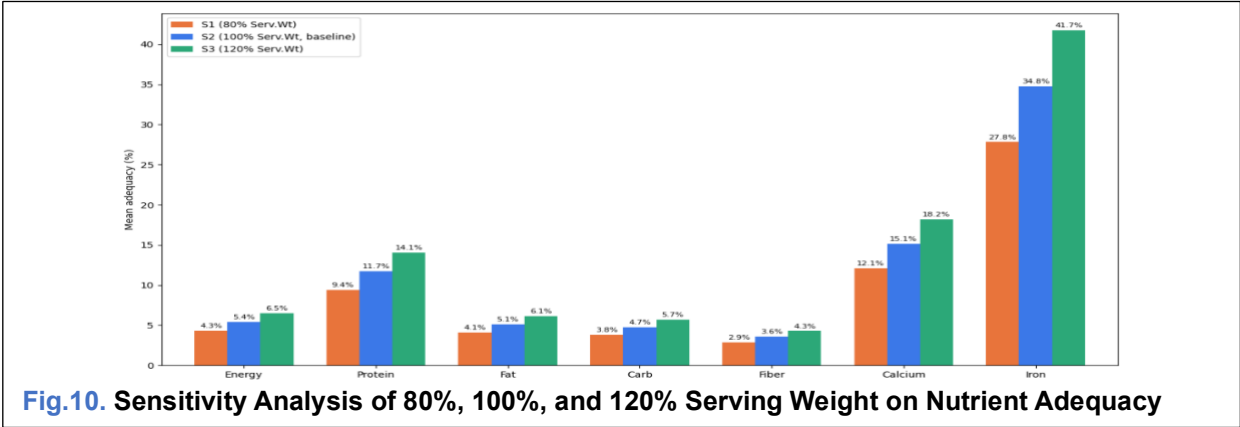
Table 9. MAE, RMSE, and MAPE Results from System-Based Nutritional Estimates Compared to Nutritionist Assessments

Nutrient	MAE	RMSE	MAPE (%)
Energy	113.06	138.78	20.86
Protein	9.51	11.62	39.75
Fat	8.62	10.55	50.94
Carbohydrate	16.08	21.01	21.08
Fiber	5.46	17.45	68.91
Overall	30.55	63.64	40.31

D. Sensitivity Analysis

Unlike MAE, RMSE, and MAPE, which require data from nutrition experts, the sensitivity analysis was performed via simulation, allowing analysis of all seven nutrient components, including calcium and iron. This study used 240 photos of MBG trays from the test dataset, which were grouped into six AKG categories (elementary, middle, and high school levels, each further grouped by gender), and evaluated based on three portion size scenarios: 80%, 100% (baseline), and 120%, as shown in Table 2 and Fig. 10, which shows that nutritional adequacy increased with serving weight. Iron increased from 27.8% at the 80% serving weight to 34.8% at the baseline and

41.7% at the 120% serving weight. Calcium increased from 12.1% to 15.1% and 18.2%, while protein increased from 9.4% to 11.7% and 14.1%. Meanwhile, energy, fat, carbohydrates, and fiber changed by only about 1–2 percentage points. This differs from the MAPE results, which measure overall system error, including errors in food detection and nutrient composition, whereas sensitivity measures only the impact of changes in portion weight. Thus, the high errors in fat and fiber are more influenced by limitations in nutrient composition detection. At the same time, portion-weight accuracy more strongly influences the adequacy of iron and calcium.



IV. Discussion

A. Interpretation of Results

The 5-fold cross-validation results show comparable performance across the five YOLO models, with an average mAP50 of 0.898 for YOLO 8, 0.898 for YOLO 9, 0.896 for YOLO 11, 0.894 for YOLO 12, and 0.888 for YOLO 10. YOLO 12 recorded a Precision score of 0.902, while YOLO 9 achieved a Recall score of 0.860. This indicates that newer architectures do not always produce higher scores on every evaluation metric. The Kruskal-Wallis test confirmed statistically significant differences across all metrics, with an H value of 489.946 for mAP50 and $p < 0.001$. A follow-up Mann-Whitney U test with Bonferroni correction showed that YOLO 8 and YOLO 9 did not differ significantly in precision, Recall, or F1-score. At the same time, only YOLO 10 and YOLO 12 did not differ significantly in mAP50. Based on the bootstrap mean rank, YOLO 8 achieved the highest score of 3.208.9, followed by YOLO 9, YOLO 11, YOLO 12, and YOLO 10; therefore, it was selected as the most stable model for implementation. Class-by-class analysis showed that most objects were detected well; however,

Green_Peas had a low Recall of 0.329 due to its small size and clustering, while Crispy_Chicken and Fried_Chicken recorded an AP50 below 0.830 due to visual similarity. Detection errors generally took the form of false positives in the background. Black box testing showed that the system functioned as designed; however, nutritional estimates still contained errors with an MAE of 30.55, an RMSE of 63.64, and a MAPE of 40.31%. The lowest errors were observed in energy (20.86%) and carbohydrates (21.08%), while fat (50.94%) and fiber (68.91%) had the highest errors because the study used standard serving weights rather than actual measurements. A sensitivity analysis of the 240 photos of MBG trays showed that increasing serving weights from 80% to 120% raised the adequacy of iron from 27.8% to 41.7%, calcium from 12.1% to 18.2%, and protein from 9.4% to 14.1%, while energy, fat, carbohydrates, and fiber changed by only about 1–2%. This suggests that portion-weight accuracy has a greater impact on iron and calcium adequacy. In contrast, the large errors in fat and fiber are more closely tied to limitations in nutrient composition estimates.

B. Comparison with other researchers

Table 10. A Comparison of Previous Research on Food Detection Using the YOLO Model

Author(s), Year	Method	Limitations	Output
G. Chrintz-Gath, M. Daivadanam, L. Matta, and S. McKeever, 2025 [6]	YOLO 7, 8, and 9	The test uses only 6 images, so the results are not very accurate. Similar foods are often difficult to distinguish. Nutritional adequacy has not been calculated.	mAP50: YOLO 8(0.704), YOLO 7 (0.694) and YOLO 9 (0.689).
A. N. bin A. Faris, N. H. A. Razak, and M. M. Din, 2024 [40]	YOLO 8, 9, and 10	There are only 12 food groups. These figures are merely calorie estimates, they do not include other nutrients or comparisons of nutritional needs.	mAP50: YOLO 8(0.880), YOLO 9(0.890) and YOLO 10(0.910)
F. Romadhon et al. 2023 [8]	YOLO 8	Limited to 13 food categories and a small dataset, it is less accurate for layered foods and estimates only calories, not other nutrients.	Precision(0.940), Recall(0.900), and F1-score(0.910)
A. Karabay, H. A. Varol, and M. Y. Chan, 2025 [10]	YOLO 8 (n, s, m, l, and xl)	Evaluates only one generation of YOLO, without comparing different versions of YOLO. Performs only detection, without nutrient estimation or comparison to nutritional requirements.	mAP50: YOLO8n (0.570), YOLO8s (0.612), YOLO 8m (0.652), YOLO 8l (0.659), and YOLO 8xl (0.677)
A. Sharma, V. Kumar, and L. Longchamps, 2024 [13]	YOLO 8, 9, 10, 11 and CNN	Limited training data. Complex backgrounds make recognition difficult. No nutritional estimates.	mAP50: YOLO8 (0.921), YOLO9 (0.935), YOLO10 (0.872), YOLO11 (0.921), and CNN (0.821)

As shown in Table 10, compared to previous studies, this study used 1.200 images from 51 MBG food classes, which were photographed using smartphones under various lighting conditions. A nutritional study based on the Swedish Plate Model [6] used 3.707 images from 42 classes with standardized serving conditions and compared YOLO 7–YOLO 9, with a performance difference of about 2% and without statistical testing. A study on food detection for calorie estimation [40] used 1.337 images from 12 classes and compared YOLO 8–YOLO 10. In contrast, a study on Indonesian food detection [8] used YOLO 8 on 13 classes with 1,680 data points after augmentation, with an 80:20 split. On 100 separate test images, the model achieved an average precision of 0.940, a recall of 0.900, and an F1 score of 0.910, with errors primarily occurring in overlapping objects. The CAFSD study [10] used 21.306 images from 239 classes sourced from the web, photos, and videos; however, the mAP50 was only 67.7–69.9% due to similarities between classes. A study on weed detection [13] used 2.348 images and compared YOLO 8–YOLO 11 with a single split, where YOLO 9 achieved an mAP50 of 93.5%.

This study compared YOLO 8–YOLO 12 using 5-fold cross-validation, 1,000 bootstrap iterations, and the Kruskal–Wallis and Mann–Whitney tests with Bonferroni correction. YOLO 8 achieved an mAP50 of 0.899, a precision of 0.900, and a recall of 0.870. The number and similarity of classes, image sources and conditions, and validation rigor primarily influence performance differences across studies. Although its performance is lower than that reported in [13], this study's results are higher than those in [6] and [10] and were obtained through a much more rigorous validation process. The errors found are also consistent with [6] and [8], particularly misclassifications of visually similar foods, such as Crispy_Chicken and Fried_Chicken.

C. Limitations

This study has several limitations. The dataset is still limited to the MBG menu. It therefore does not cover all types of Indonesian food, and nutritional calculations use a standard portion in Table 3 because the model cannot yet directly measure food mass or volume, so results may differ from actual portion sizes. Some images are sourced from social media, with variations in quality, lighting, compression, and shooting angles that can affect detection accuracy. At the same time, data imbalance in certain classes such as Green_Peas, Crispy_Shrimp, and Steamed_Carrot results in lower Recall than classes with distinctive visual features like Grape, Apple, and Broccoli. Additionally, menu and condition variations across regions can affect the model's out-of-sample performance. At the same time, the manual correction feature still relies on user knowledge and may therefore affect the accuracy of reported nutritional estimates. Further testing with users and a more diverse range of MBG menus in real-world schools is still needed to assess the system's external validity.

D. Implications

YOLO 8 was selected as the final system model because its performance proved to be the most consistent and

stable empirically, underscoring that architectural choices should be based on concrete evidence, not merely on following the latest version. In practice, YOLO-TKPI-AKG can convert photos of food trays into information about food types and estimated nutritional values, helping schools and MBG administrators monitor students' nutritional adequacy. Nutritionists can use these data to evaluate menus and program outcomes, while parents can get a simple overview of their children's nutritional adequacy by age and gender. When used routinely, these data can also identify patterns of malnutrition at the school or regional level. For MBG and SPPG administrators, the system's results can support menu evaluation, program planning, and compliance with nutritional standards under Ministry of Health Regulation No. 28 of 2019. However, this system is a decision-support prototype, not a clinical diagnostic tool or a definitive nutritional assessment; the results are estimates, and determining a student's nutritional status still requires evaluation by a nutritionist or doctor.

V. Conclusion

This study achieved both objectives: comparing the performance of five YOLO versions and developing a nutrient analysis system based on AKG for elementary, middle, and high school levels. Based on the single test evaluation, YOLO 8 was selected as the final model, with an mAP50 of 0.899, Precision of 0.904, Recall of 0.868, and an F1-Score of 0.886; bootstrap ranking analysis confirmed it as the best model. This study's main contribution is to show that selecting a YOLO version should be based on empirical performance comparisons, rather than the assumption that the latest version is always better. This study also produced the MBG nutritional analysis system, which integrates food detection with nutritional adequacy analysis based on the AKG, with all features passing black-box testing. The system's manual correction feature only addresses classification errors. It does not eliminate other sources of uncertainty, such as assumptions about serving weights, recipe variations, preparation methods, ingredient composition, or differences in actual food weights. Thus, the system's nutritional estimates are approximations, not fully accurate assessments. The study's limitations include a dataset that is still limited to the MBG menu, as well as nutritional estimates that still rely on assumptions about standard serving weights. Future research should develop image-based portion volume estimates, expand the dataset, and validate the system in real schools to measure accuracy and acceptance under actual conditions of use by students, teachers, and SPPG staff.

Acknowledgment

The author would like to thank Surabaya State University for its institutional support, as well as all those who contributed their assistance.

Funding

The authors did not receive any financial support for the research, writing, and publication of this article.

Author Contributions

Amelia designed, conducted, analyzed, and drafted the research and manuscript, while Wiyli and Mega provided methodological supervision and feedback. All authors approved the final manuscript.

Data Availability

The MBG 51 class food dataset was processed using Roboflow and is available upon reasonable request.

Code Availability

The source code for data processing, models, and systems is available upon reasonable request.

Ethical Approval

This study does not involve human subjects and therefore does not require formal ethical approval.

Consent To Participate

Not applicable, as this study did not involve the direct participation of human subjects.

Consent for Publication

Not applicable.

Competing Interests

The authors declare that there are no conflicts of interest affecting this study.

References

- [1] G. D. Dirawan and A. Sukriadi, "Beyond School Feeding : Makan Bergizi Gratis (MBG) as a Life-Course Nutrition and Human Capital Policy in Indonesia," *Pinisi J. Educ.*, vol. 6, no. 2, pp. 99–114, 2026, doi: 10.26858/c3jnx65.
- [2] R. Rimbawan, R. Nurdiani, P. H. Rachman, Y. Kawamata, and Y. Nozawa, "School Lunch Programs and Nutritional Education Improve Knowledge, Attitudes, and Practices and Reduce the Prevalence of Anemia: A Pre-Post Intervention Study in an Indonesian Islamic Boarding School," *Nutrients*, vol. 15, no. 4, 2023, doi: 10.3390/nu15041055.
- [3] Faramitha, Hendrayati, and Nadimin, "Analysis of Energy and Nutrient Adequacy in Two Menu Variants of the Free Nutritious Meal (MBG) Program for Adolescents in Makassar City," *Aceh Nutr. J.*, vol. 10, no. 4, pp. 1166–1174, 2025, doi: 10.30867/action.v10i4.3008.
- [4] S. Kokare and P. Kandekar, "Food Detection and Nutrition Analysis by Image," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, pp. 306–312, 2024, doi: 10.22214/ijraset.2024.64899.
- [5] B. Gonzalez, G. Garcia, S. A. Velastin, H. GholamHosseini, L. Tejeda, and G. Farias, "Automated Food Weight and Content Estimation Using Computer Vision and AI Algorithms," *Sensors*, vol. 24, no. 23, pp. 1–19, 2024, doi: 10.3390/s24237660.
- [6] G. Chrintz-Gath, M. Daivadanam, L. Matta, and S. McKeever, "Image-Based Dietary Assessment Using The Swedish Plate Model: Evaluation of Deep Learning–Based You Only Look Once (YOLO) Models," *JMIR Form. Res.*, vol. 9, pp. 1–15, 2025, doi: 10.2196/70124.
- [7] A. Sani *et al.*, "Identification Food Nutrition and Weight Prediction Using Image Processing," *J. Appl. Electr. Eng.*, vol. 9, no. 1, pp. 28–36, 2025, doi: 10.30871/jaee.v9i1.9492.
- [8] 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, pp. 1–7, 2023, doi: 10.1051/e3sconf/202346502057.
- [9] G. C. Utami, C. Raras, A. Widiawati, and P. Subarkah, "Detection of Indonesian Food to Estimate Nutritional Information Using YOLOv5," *J. Inf. Commun. Technol.*, vol. 12, no. 2, pp. 158–165, 2023, doi: 10.34148/teknika.v12i2.636.
- [10] A. Karabay, H. A. Varol, and M. Y. Chan, "Improved Food Image Recognition by Leveraging Deep Learning and Data-Driven Methods with an Application to Central Asian Food Scene," *Sci. Rep.*, vol. 15, no. 1, pp. 1–12, 2025, doi: 10.1038/s41598-025-95770-9.
- [11] Y. Shi *et al.*, "Calorie Detection in Dishes Based on Deep Learning and 3D Reconstruction," *Comput. Electron. Agric.*, vol. 229, pp. 1–12, 2025, doi: 10.1016/j.compag.2024.109832.
- [12] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, "YOLOv8 to YOLOv11: A Comprehensive Architecture In-Depth Comparative Review," *Rekayasa Sist. dan Teknol. Inf.*, vol. 10, no. 2, pp. 341–354, 2026, doi: 10.29207/resti.v10i2.6598.
- [13] A. Sharma, V. Kumar, and L. Longchamps, "Comparative performance of YOLOv8, YOLOv9, YOLOv10, YOLOv11 and Faster R-CNN models for detection of multiple weed species," *Smart Agric. Technol.*, vol. 9, p. 100648, 2024, doi: 10.1016/j.atech.2024.100648.
- [14] G. More, O. Patil, O. More, M. More, S. Suryavanshi, and M. Mali, "Comparison of Object Detection Algorithms CNN, YOLO and SSD," *Int. J. Sci. Res. Technol.*, vol. 1, no. 11, pp. 137–144, 2024, doi: 10.5281/zenodo.14186397.
- [15] S. Patil, S. Patil, V. Kale, and M. Bonde, "Food Item Calorie Estimation Using YOLOv4 and Image Processing," *Int. J. Comput. Trends Technol.*, vol. 69, no. 5, pp. 69–76, 2021, doi: 10.14445/22312803/IJCTT-V69I5P110.
- [16] 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, 2024, doi: 10.1134/S1547477124010059.
- [17] R. Shankar and A. Yip, "Transforming Patient Feedback Into Actionable Insights Through Natural Language Processing: Knowledge

- Discovery and Action Research Study," *JMIR Form. Res.*, vol. 9, pp. 1–14, 2025, doi: 10.2196/69699.
- [18] M. A. Fauzi, H. Tribiakto, A. Moniva, F. Amir, I. K. Ilyas, and E. Utami, "Systematic Literature Reviews on Rapid Application Development Information System," *Bull. Comput. Sci. Electr. Eng.*, vol. 4, no. 1, pp. 57–64, 2023, doi: 10.25008/bcsee.v4i1.1181.
- [19] R. Sapkota and M. Karkee, "Ultralytics YOLO Evolution: An Overview of YOLO26, YOLO11, YOLOv8 and YOLOv5 Object Detectors for Computer Vision and Pattern Recognition," pp. 1–16, 2026, doi: 10.48550/arXiv.2510.09653.
- [20] K. Alomar, H. I. Aysel, and X. Cai, "Data Augmentation in Classification and Segmentation: A Survey and New Strategies," *J. Imaging*, vol. 9, no. 2, p. 46, Feb. 2023, doi: 10.3390/jimaging9020046.
- [21] T. M. Omoniyi *et al.*, "The Effect of Data Augmentation on Performance of Custom and Pre-Trained CNN Models for Crack Detection," *Appl. Sci.*, vol. 15, no. 22, p. 12321, Nov. 2025, doi: 10.3390/app152212321.
- [22] X. Du, Y. Sun, Y. Song, W. Chi, L. Dong, and X. Zhao, "Impact of Input Image Resolution on Deep Learning Performance for Side-Scan Sonar Classification: An Accuracy–Efficiency Analysis," *Remote Sens.*, vol. 17, no. 14, pp. 1–24, 2025, doi: 10.3390/rs17142431.
- [23] A. Ichwani, R. I. Kesuma, A. Setiawan, I. E. Wicaksono, and R. Hanifah, "Preventing Data Leakage in Classification via Integrated Machine Learning Pipelines: Preprocessing, Feature Transformation, and Hyperparameter Tuning," *J. Tek. Inform.*, vol. 7, no. 1, pp. 391–410, 2026, doi: 10.52436/1.jutif.2026.7.1.5490.
- [24] J. Terven, D. M. C. Esparza, and J. A. R. Gonzales, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach. Learn. Knowl. Extr.*, vol. 5, no. 4, pp. 1680–1716, Nov. 2023, doi: 10.3390/make5040083.
- [25] Z. Hu, W. Chen, D. Yang, Z. Wang, and P. Wang, "Enhancing Object Detection in Remote Sensing Images with Improved YOLOv8 Model," *Sci. Rep.*, vol. 15, pp. 1–14, 2025, doi: 10.1038/s41598-025-26536-6.
- [26] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and Better Learning For Bounding Box Regression," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, pp. 12993–13000, 2020, doi: 10.1609/aaai.v34i07.6999.
- [27] S. M. A. H. Shah, A. Rizwan, G. Atteia, and M. Alabduhafith, "CADFU for Dermatologists: A Novel Chronic Wounds & Ulcers Diagnosis System with DHuNeT (Dual-Phase Hyperactive UNet) and YOLOv8 Algorithm," *Healthc.*, vol. 11, no. 21, pp. 1–22, 2023, doi: 10.3390/healthcare11212840.
- [28] X. Li *et al.*, "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 1–14, 2020, doi: 10.48550/arXiv.2006.04388.
- [29] J. Wei, A. A. Arry, K. A. M. Rezali, M. Z. M. Yusoff, H. Ma, and K. Zhang, "A Review of YOLO Algorithm and Its Applications in Autonomous Driving Object Detection," *IEEE Access*, 2025, doi: 10.1109/ACCESS.2025.3573376.
- [30] J. Allgaier and R. Pryss, "Cross-Validation Visualized: A Narrative Guide to Advanced Methods," *Mach. Learn. Knowl. Extr.*, vol. 6, no. 2, pp. 1378–1388, 2024, doi: 10.3390/make6020065.
- [31] S. F. Mokhtar, Z. Md Yusof, and H. Sapiri, "Confidence Intervals by Bootstrapping Approach: A Significance Review," *Malaysian J. Fundam. Appl. Sci.*, vol. 19, no. 1, pp. 30–42, Feb. 2023, doi: 10.11113/mjfas.v19n1.2660.
- [32] N. K. Rajpoot, P. D. Singh, B. Pant, and M. Sharma, "An Intelligent Framework for Real-Time Heart Stroke Monitoring and Early Detection Using Machine Learning," *Cureus J. Comput. Sci.*, vol. 2, 2025, doi: 10.7759/s44389-025-04004-6.
- [33] U. Ahmad, M. A. Rosyid, and Mardison, "Abnormal Shapes Identification of Gedong Mango Using Dimensionless Shape Factors in Image Processing," *Earth Environ. Sci.*, vol. 1038, 2022, doi: 10.1088/1755-1315/1038/1/012045.
- [34] J. Terven, D. M. Cordova Esparza, J. A. R. González, A. Ramírez Pedraza, and E. A. Chávez Urbola, "A Comprehensive Survey of Loss Functions and Metrics in Deep Learning," *Artif. Intell. Rev.*, vol. 58, no. 7, p. 195, Apr. 2025, doi: 10.1007/s10462-025-11198-7.
- [35] M. Nasari, G. A. Tarigan, N. A. P. Masaling, S. Joddy, and K. A. Minor, "Indonesian Food Detection with YOLO and Calorie Estimation Using Lightweight LLMs," *IEEE Xplore*, vol. 25, 2025, doi: 10.1109/ISRITI68345.2025.11393164.
- [36] I. Hantoro, J. Wang, A. J. Löhr, F. G. A. J. Van Belleghem, B. Widianarko, and A. M. J. Ragas, "Assessing Age- Specific Variability in Microplastic Intake Through Seafood Consumption: A Case Study in Central Java, Indonesia," *J. Food Sci.*, vol. 91, no. 3, 2026, doi: 10.1111/1750-3841.70909.
- [37] F. N. Nwobi and F. C. Akanno, "Power comparison of ANOVA and Kruskal–Wallis Tests When Error Assumptions Are Violated," *Adv. Methodol. Stat.*, vol. 18, no. 2, pp. 53–71, 2021, doi: 10.51936/lgt2135.
- [38] C. Sergio *et al.*, "Beyond the Nutrition5k Project: Data Curation and Deep Learning Algorithms to Predict the Nutritional Composition of Dishes from Food Images," *Epidemiol. Biostat. Public Heal.*, vol. 20, no. 1, pp. 150–152, 2025, doi: 10.54103/2282-0930/29429.
- [39] P. Ma, C. P. Lau, N. Yu, A. Li, and J. Sheng,

"Application of deep learning for image-based Chinese market food nutrients estimation," *Food Chem.*, vol. 373, p. 130994, Mar. 2022, doi: 10.1016/j.foodchem.2021.130994.

[40] A. N. bin A. Faris, N. H. A. Razak, and M. M. Din, "Deep Learning for Meal Recognition and Calorie Estimation," *5th Int. Conf. Inf. Technol. Secur.*, vol. 5, no. 1, pp. 357–362, 2024, doi: 10.32664/ic-itechs.v5i1.1679.

AUTHOR BIOGRAPHY



Amelia Prachintya Putri Arifin is an undergraduate student in the Information Systems program at the Faculty of Engineering, Surabaya State University. He has experience in the field of digital forensics and in designing the City of Surabaya's government service system, which was showcased at

the exhibition. His research interests include computer vision, deep learning, and digital image analysis, particularly AI-based object detection integrated with national nutrition standards. she is also interested in developing data-driven information systems to support targeted decision-making. His research focuses on the application of artificial intelligence to build accurate, measurable, and practical systems to address real-world problems and support technological innovation.



Wiyli Yustanti is an Associate Professor in the Master's Program in Informatics at Universitas Negeri Surabaya (Unesa), Indonesia. She earned her Bachelor's degree in Statistics, Master's degree in Informatics, and Doctorate in Statistics from the Sepuluh

Nopember Institute of Technology (ITS), Indonesia. Her research interests include machine learning, data mining, text mining, classification, clustering, big data, and computational modeling of multimodal data, with applications in healthcare, education, social media, and enterprise systems. She has authored more than 70 scientific publications and contributed to books on Python Programming, Deep Learning, and Machine Learning. Her work emphasizes data-driven computational methods for complex real-world challenges.



Mega Mirasaputri Cahyanti Is an Assistant Professor at the Graduate School of Technology Management, Yamaguchi University, Japan. She holds a Ph.D. from Yamaguchi University, an MSc in the Global Master Program of Tourism and Management from National Chiayi University, Taiwan, and a

Master of Management and a Bachelor's degree in Economics from Brawijaya University, Indonesia. Her research interests include innovation management,

technology application, disaster management, resilience, and interdisciplinary technology-driven solutions. Her interdisciplinary academic background enables her to integrate technology, management, and community perspectives in addressing real-world challenges. She has conducted research in Indonesia and Japan and contributed to internationally funded projects involving innovative approaches to disaster preparedness, community resilience, and technology-driven solutions.