Data Science and Engineering (DSE) Record, Volume 6, issue 1.

Food Consumption Measurement Using Computer Vison: A Case Study on Thai Cuisine

Pattadon Thepkan [1] and Jakarin Chawachat [2]

[1] Master's Degree Program in Data Science, Chiang Mai University, Chiang Mai, Thailand
[2] Department of Computer Science, Faculty of Science, Chiang Mai University, Chiang Mai, Thailand
pattadon_th@cmu.ac.th

Abstract. This study aims to develop a system for estimating the portion size and energy of Thai food from images using deep learning techniques. The proposed system supports dietitians and health-conscious individuals by enabling automated and accurate food intake assessment. The system consists of two main components: (1) object detection using YOLOv11 to simultaneously identify food items and reference coins in an image, and (2) food weight estimation using ResNet101, with reference coin serving as physical references for real-world size scaling. The estimated food weight is then used to calculate nutritional values based on a Thai food database. Experimental results demonstrate that annotating object boundaries with Smart Polygon significantly improves model accuracy and stability compared to the traditional Bounding Box method, yielding higher Precision, Recall, F1-score, and mAP. Among the tested models, ResNet101 with coin references achieved the best weight estimation performance, with a Mean Absolute Error (MAE) of 71.12 grams and Root Mean Squared Error (RMSE) of 91.56 grams. This system is suitable for real-world applications in hospitals, restaurants, and personal nutrition tracking.

Keywords: Food Estimation, Object Detection, Nutrient Tracking, Thai Cuisine

1 Introduction

Malnutrition, particularly among elderly patients, is a persistent concern in hospital care. To assess food intake, nutritionists traditionally rely on a precise method of weighing food portions before and after a meal. While this approach is highly accurate, it is time-consuming and impractical for continuous use, especially in high volume healthcare settings where staff resources are limited. Computer vision-based systems offer a promising solution by enabling automated food intake monitoring through image analysis. A previous study by Ruenin et al. (2020) [1] introduced machine learning systems designed to estimate the amount and calories of food consumed by hospital patients. These systems used object detection and regression techniques to process images of meals served in segmented trays. While effective in controlled settings, their use is limited when applied to real-world dining scenarios where dish presentation varies widely. This study proposes a solution that addresses these challenges by combining object detection with real-world scaling. By using YOLOv11 to detect food and coins, and ResNet101 to estimate food weight, the system calculates

nutritional intake from images in an automated and scalable way. This approach aims to support dietary monitoring not only in hospitals but also in real-world environments, where food is served in informal and varied formats.

2 Literature Review

This research aims to develop a food image analysis system capable of automatically classifying food types and estimating their weights for calculating energy and nutrients. Therefore, this chapter reviews the related literature in two main areas: (1) food detection and (2) food weight estimation, presented chronologically. It also explains the techniques, advantages, limitations, and research gaps that inform the approach proposed in this study.

2.1 Food Detection

Zhang, H. et al. (2015) [2] developed Snap-n-Eat, one of the first applications to use Convolutional Neural Networks (CNNs) for classifying food types from mobile phone images. A key feature of the system was its end-to-end capability, handling both food classification and energy estimation from a single image.

Myers, A. et al. (2015) [6] developed Im2Calories, which combined CNNs with the USDA food database to present results in terms of energy and nutritional values. The system was functional on mobile devices and served as one of the foundations for the development of automatic food diaries.

Ege, T., & Yanai, K. (2019) [3] proposed a Multi-Task Learning approach that integrates detection with energy estimation within a single network. This method reduces processing time and system complexity

Tan, M. et al. (2020) [4] introduced EfficientDet, which, although not specifically designed for food-related tasks, has been adapted for food image analysis to reduce model size while maintaining high accuracy—making it suitable for resource-constrained devices.

Nguyen, T. et al. (2024) [5] presented FoodMask, an instance segmentation-based approach to separate individual food items on a single plate. This method addresses the challenge of overlapping and mixed dishes such as Thai food, where traditional bounding boxes struggle to segment components accurately.

2.2 Food Weight Estimation

He, Y. et al. (2013) [7] proposed a method for estimating food weight based on the segmented area of the food and predefined density from a top-down (vertical) camera angle.

Wang, Z. et al. (2018) [8] developed a deep learning-based system to estimate food volume from a single image without requiring a depth camera. This enabled usage on

standard consumer devices, although it still required consistent camera angles and plain backgrounds for accuracy.

Ye, H. et al. (2019) [9] introduced a concept of cross-domain learning by training models on both real and synthetic images. This allowed models to generalize to new image domains without needing retraining, although performance degraded in cases with overlapping foods or cluttered scenes.

Garcia et al. (2023) [10] proposed a system that uses physical reference objects, such as coins or spoons with known dimensions, to calculate the image scale. This scale was then used to convert food bounding boxes into real-world dimensions and estimate weight. The study highlighted the system's simplicity and practical accuracy, making it particularly suitable for use in restaurants and home environments.

Fang, C. et al. (2024) [11] developed a method for estimating food volume from 2D images using 3D point cloud reconstruction. This approach converts flat images into depth-aware structures, allowing for more accurate weight estimation without the need for specialized cameras. However, it requires complex training data and highly accurate segmentation to separate food from background.

3 Data and Methodology

The overall workflow of the proposed system is illustrated in Figure 1, which consists of four sequential components: data preparation, object detection, cropping and weight estimation, and nutrient calculation.

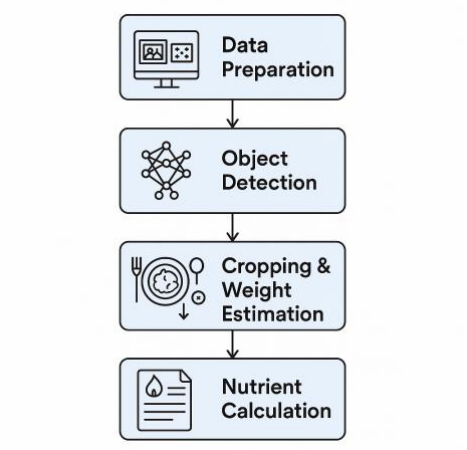

Figure 1. Overall Workflow

3.1 **Data Preparation**

This section describes the steps involved in preparing the dataset used in this study. It includes data collection, data labeling with different annotation strategies, and data preprocessing and augmentation to improve the model's performance and robustness.

- Data Collection: The dataset used in this study consists of top-down view photographs, each containing only two types of objects.
  - Food: chicken rice, crispy pork rice, fried rice, pad thai, and pad see ew
  - Coins: 1 baht, 2 baht, 5 baht, and 10 baht coins

  See Figure 2 for an example of data collection.

Figure 2. Example of Data Collection

- Data Labeling in Roboflow (https://roboflow.com): To support comparison between different levels annotation precision, the dataset was prepared in two versions.
  - Food items were annotated using Bounding Box, while coins were annotated using Smart Polygon.
  - Both food items and coins were annotated using Smart Polygon

  Example of these annotation strategies are shown in Figures 3a and 3b.

Figure 3a. Bounding Box Annotation

Figure 3b. Smart Polygon Annotation

- Data Preprocessing and Augmentation: All images were preprocessed through resizing and normalization to ensure consistency across the dataset. Data augmentation was applied using Roboflow (https://roboflow.com) to enhance model robustness under real-world conditions. Augmentations included random noise, brightness and contrast adjustments, and slight geometric transformations (e.g., shear, blur). Coins and food items were augmented differently to reflect their distinct visual properties and roles in the detection process.

3.2 **Object Detection Model**

This section outlines the procedures used for training and evaluating the YOLOv11 model in this study. It details the model selection, environment setup, training configurations, and performance monitoring to ensure robust object detection results.

- Model Selection: YOLOv11 was chosen for object detection due to its high accuracy and strong performance in multi-class object identification.

- Environment Setup: Training was performed on a Tesla T4 GPU (15GB VRAM) with CUDA 12.4 and driver version 550.54.15. Libraries used include ultralytics, supervision, and Google Drive mount for data management.

- Training Configuration: The model was initialized with yolo11s.pt, using a batch size of 32 for 50 epochs. Input images were resized to 640×640 pixels. Mixed precision (AMP) and caching were enabled to optimize training. Model checkpoints were saved every 10 epochs.

- Evaluation and Fine-Tuning: Loss, accuracy, and mean Average Precision (mAP) were monitored throughout training. Checkpoints were used to track performance and adjust hyperparameters as needed.

3.3 **Cropping and Weight Estimation**

This section describes the data preprocessing, experimental setup, and methods used to estimate food weight with ResNet models. It includes the steps of cropping detected objects, computing scaling factors for coin-referenced predictions, and feeding the processed data into ResNet variants to generate weight predictions.

- Image Preprocessing and Cropping: To prepare inputs for the ResNet model, object coordinates were exported from Roboflow in csv format, containing file names, object classes, and bounding box coordinates. The process involved reading the csv with Pandas, separating food and coin entries, and cropping the objects from the original images using OpenCV. Cropped images were resized

to 224×224 pixels while maintaining aspect ratio, applying black padding where necessary. Resulting images were stored in separate folders for food and coin.

- Experiment Setup: Two experimental conditions were tested: one with a coin reference, using a coin-based scaling factor, and another without the coin reference as the baseline. Each condition was evaluated with three ResNet architectures: ResNet50, ResNet101, and ResNet152.

- Coin-Based Scaling: The real-world width of the coin was used as a reference. The scaling factor was computed by dividing the known coin diameter (e.g., 25.60 mm for a 10-baht coin) by its bounding box pixel width.

- Model Input: Cropped food images, along with the scaling factor (if applicable), were fed into the ResNet model to predict food weight.

3.4 **Nutrient Calculation**

This section details how the estimated food weight is used to calculate nutrient content. Using the Thai Food Composition Database from the Institute of Nutrition, Mahidol University, the system retrieves the nutrient values for each food type and calculates the final nutrient amounts based on the estimated weight.

1) Load the nutrient database file (.json format).
2) Match the predicted food label to its corresponding entry in the database.
3) Apply the formula:

$$Nutrient = \left(\frac{Nutrient\ per\ 100g}{100}\right) \times\ Estimated\ Weight\ (g)$$

According to the Thai Food Composition Database (Institute of Nutrition, Mahidol University, 2015), nutritional values for each food item are standardized per 100 grams of edible portion.

4 **Results and Evaluation**

This section presents the experimental results and evaluation of the models developed in accordance with the methodology described in Section 3. The evaluation is divided into two parts: first, the training and validation evaluation, which compares model performance during training and validation, including both object detection using YOLOv11 and food weight estimation using ResNet50, ResNet101, and ResNet152, ultimately selecting the best-performing model for testing on unseen data; and second, the unseen test evaluation, where the selected models are tested on

previously unseen data to assess their generalization performance, including a comparison between models that use images with and without physical reference objects (coins) to validate the effectiveness of the proposed approach.

4.1 Training and Validation Evaluation

4.1.1 Evaluation of YOLOv11 for Object Detection

To compare the effectiveness of annotation strategies, two YOLOv11 models were trained under identical configurations using either Bounding Box or Smart Polygon labels.

Both models were evaluated using F1-Confidence curves, Precision-Recall curves, and mAP@0.5, focusing on both food items and coins.

- **F1-Confidence Curve Analysis:**
  To compare the effectiveness of annotation strategies, two YOLOv11 models were trained under identical configurations using either Bounding Box or Smart Polygon labels. As shown in Figures 4a and 4b, the Smart Polygon model achieved a peak F1 score of 0.92 at a confidence threshold of 0.255, whereas the Bounding Box model only reached a peak F1 score of 0.64 at confidence threshold of 0.141.

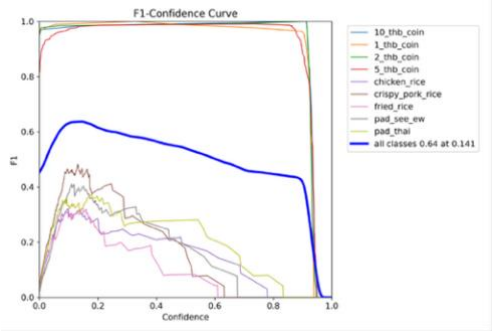

Figure 4a. F1-Confidence curve (Bounding Box)

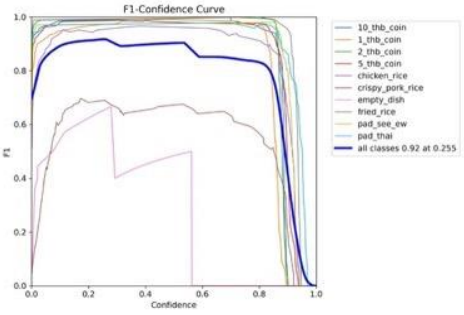

Figure 4b. F1-Confidence curve (Smart Polygon)

- **Precision-Recall (PR) Curve Analysis:**
  The PR curves in Figures 5a and 5b further confirmed the superior performance of the Smart Polygon model, which reached a mAP@0.5 of 0.915, compared to only 0.616 for the Bounding Box model. This difference was particularly notable in detecting complex food categories such as pad_see_ew and fried_rice.

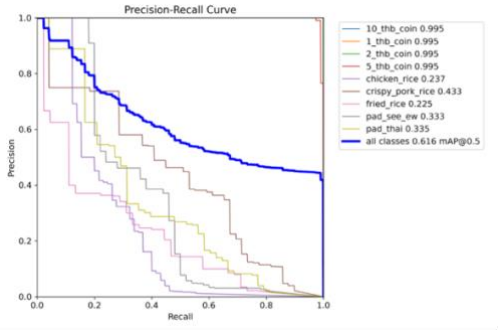

Fig. 5a. Precision-Recall curve (Bounding Box)

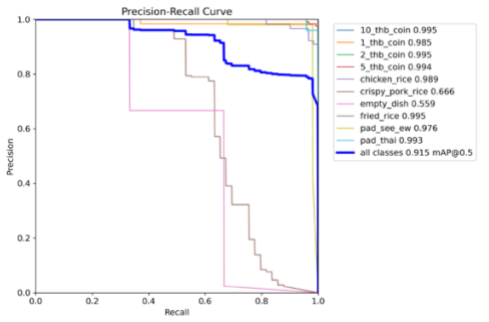

Fig. 5b. Precision-Recall curve (Smart Polygon)

Across both evaluation metrics, Smart Polygon annotation consistently improved model performance, especially for foods with irregular shapes, without compromising coin detection. These results highlight the importance of fine-grained annotation when applying object detection in real-world food analysis scenarios.

4.1.2 Evaluation of ResNet for Food Weight Estimation

Once food items were detected and isolated using YOLOv11, food weight estimation was conducted using ResNet-based regression models. ResNet (Residual Network) architectures are well established in computer vision tasks due to their ability to learn hierarchical features in complex image data. This study compared three variants: ResNet50, ResNet101, and ResNet152. Each was tested under two conditions: with coin reference, using a coin's known diameter to compute a scaling factor (mm/pixel), and without coin reference, used as a baseline with no physical size input. The goal was to evaluate both prediction accuracy and model stability across different network depths. Models were assessed using training and validation loss curves.

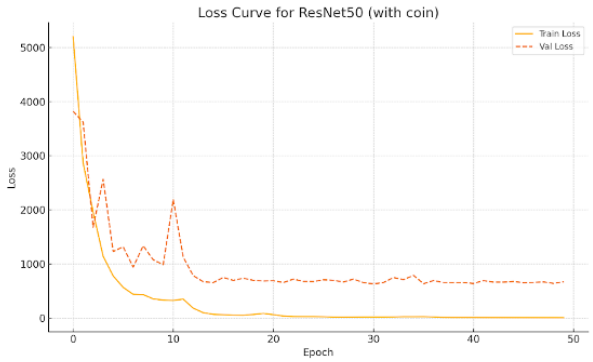

Fig. 6a. Training and validation loss – ResNet50 (with coin)

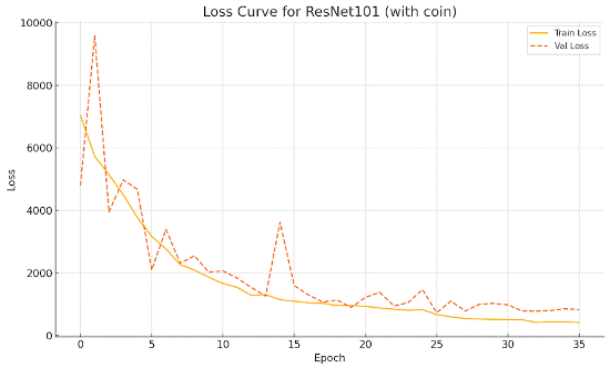

Fig. 6b. Training and validation loss – ResNet101 (with coin)

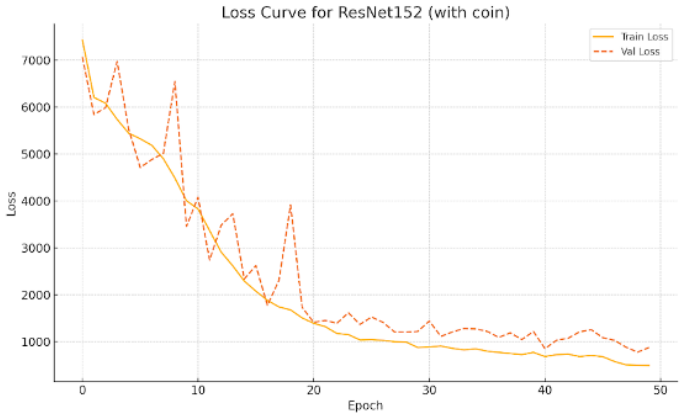

Fig. 6c. Training and validation loss – ResNet152 (with coin)

The training and validation loss curves in Figures 6a, 6b, and 6c show the performance of the three ResNet models with coin reference. All models converged well, but ResNet101 in Figure 6b exhibited the most stable and balanced loss trajectory, indicating consistent learning and minimal overfitting. In contrast, ResNet152 in Figure 6c, despite its deeper architecture, showed higher variance in the validation loss, suggesting sensitivity to noise or potential overfitting. Overall, these curves demonstrate that ResNet101 offered the best trade-off between accuracy and stability during training.

4.2 Testing on Unseen Data

This section evaluates the performance of the proposed system on an unseen dataset to assess its generalization ability. The dataset consists of 100 images that were not used during training or validation, providing a direct and independent test of the system's performance. It includes evaluations of the selected YOLOv11 model for food and coin detection, as well as the ResNet model for weight estimation, examining how effectively they perform in realistic, previously unseen scenarios.

4.2.1 Testing the Selected YOLO Model on Unseen Data

To evaluate the generalization capability of the YOLOv11 model trained with Smart Polygon annotations, the model was tested on an unseen test set that was not used during training or validation.

The evaluation focused on both food classification and coin detection, using standard performance metrics: Accuracy, Precision, Recall, and F1-score. These metrics assess the model's ability to correctly classify objects and avoid false predictions.

The definitions of the metrics are based on the following:

- TP (True Positive): Correctly predicted as the target class
- TN (True Negative): Correctly predicted as not the target class
- FP (False Positive): Incorrectly predicted as the target class
- FN (False Negative): Failed to predict the target class

The metrics are calculated as:

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
- $Precision = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F1-score = 2 \times \frac{Precision \times Recall}{Precision+Recall}$

The overall classification performance of the YOLOv11 model, as shown in Table 1, indicates a high level of accuracy in detecting food items but comparatively lower performance for coin detection. Specifically, food classification achieved an accuracy of 86.00%, an F1-score of 0.896, a precision of 0.867, and a recall of 0.862. In contrast, coin classification only reached an accuracy of 66.00%, with an F1-score of 0.680, precision of 0.688, and recall of 0.678, reflecting the greater challenge in detecting coins due to their small size and reflective surfaces.

Table 1. Overall Classification Performance

Metric	Food Classification	Coin Classification
Accuracy	86.00 %	66.00 %
F1-score	0.896	0.680
Precision	0.867	0.688
Recall	0.862	0.678

As shown in Table 2, the precision, recall, F1-score, and support for each food category in the test dataset highlight the model's performance. Crispy pork rice achieved the highest classification performance, with both precision and recall at 1.00, demonstrating excellent model accuracy for this dish. In contrast, fried rice had the lowest recall at 0.60, likely due to visual similarities with dishes like chicken rice and crispy pork rice. Despite these occasional confusions, most dishes had F1-scores above 0.85, indicating strong overall robustness of the model in distinguishing between the five food categories.

Table 2. Per-Class Performance (Food Categories)

Food Type	Precision	Recall	F1-score	Support
chicken_rice	0.769	1.000	0.870	20
crispy_pork_rice	0.909	1.000	0.952	20
fried_rice	1.000	0.600	0.750	20
pad_see_ew	1.000	0.737	0.848	20
pad_thai	0.800	1.000	0.889	20

Coins serve as critical reference objects for estimating food weight via image scaling, and their classification performance is detailed in Table 3. The 10-baht coin was classified with perfect precision, recall, and F1-score of 1.00, indicating the model could consistently recognize it with high confidence. In contrast, the 1-baht and 5-baht coins had lower performance, particularly the 5-baht coin, which showed a recall of only 0.40, likely due to its small size and reflective surfaces. The 2-baht coin performed moderately well, achieving an F1-score of approximately 0.75.

Table 3. Per-Class Performance (Coin Categories)

Food Type	Precision	Recall	F1-score	Support
1_thb_coin	0.490	0.610	0.530	28
2_thb_coin	0.790	0.750	0.750	27
5_thb_coin	0.480	0.400	0.440	25
10_thb_coin	1.000	1.000	1.000	20

4.2.2 Testing of the Selected Resnet Model on Unseen Data

To assess the effectiveness of the selected ResNet model, this study evaluated ResNet101 (with coin reference) on an unseen test set. The model was tested on 100 food images (20 images per food type) using standard performance metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and a custom-defined Accuracy metric. The formulas are as follows:

- Mean Absolute Error (MAE)
  Measures the average magnitude of errors between predicted and actual values:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|yi - \hat{y}i|$$

- Root Mean Squared Error (RMSE)
  Gives higher weight to larger errors by squaring the differences:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(yi - \hat{y}i)^2}$$

- Accuracy (%), as defined in this study
  The percentage of samples whose prediction error falls within ±MAE of the true value:

$$Accuracy = \left(\frac{No.of\ samples\ where\ |yi - \hat{y}i| \leq MAE}{n}\right) \times 100$$

To determine the benefit of using coin references in the images, the study compared the performance of ResNet101 with coin and ResNet101 without coin using the same test dataset. The dataset included 5 food categories with a total of 100 images. The average food weight in the test set was 209.87 grams, which closely matched the training set average of 208.14 grams, ensuring data consistency across sets.

The comparison in Table 4 shows that including coin references improved estimation accuracy. The use of physical scale (coin diameter) enhanced the model's ability to map pixel area to real-world food weight, particularly for visually ambiguous or irregular portions. Specifically, incorporating the coin reference reduced the Mean Absolute Error (MAE) by approximately 5 grams and the Root Mean Squared Error (RMSE) by around 6 grams. The model with the coin reference also achieved a notably higher accuracy of 56.00%, compared to only 48.00% without the coin, highlighting its improved ability to infer physical scale from pixel dimensions.

Table 4. Comparison of Resnet101 with and without Coin Reference

Model	MAE (g)	RMSE (g)	Accuracy (%)
ResNet101 (With Coin)	71.12	91.56	56.00
ResNet101 (Without Coin)	76.38	97.35	48.00

To further analyze model error patterns, the test set was divided into three weight ranges: Less than 150 grams, Between 150-250 grams, and Greater than 250 grams. This stratification enabled evaluation across different distributions.

As shown in Table 5, the ResNet101 model with coin reference demonstrated varying performance across different food weight ranges. In the <150 g category, the model overestimated the actual average weight of 102.97 g with a predicted average of 181.11 g, resulting in a prediction bias of +78.15 g, an MAE of 78.15 g, and an accuracy of 51.72%. For the 150–250 g range, the model achieved its best performance, showing a small negative bias of -2.02 g, an MAE of 28.38 g, and the highest accuracy of 60.53%. In the >250 g range, the model underestimated the true average weight of 316.30 g by 113.60 g, with an MAE of 114.17 g and an accuracy of 54.55%. These findings highlight the model's overall robustness in the mid-range weights while pointing to areas for improvement in handling smaller and larger portion sizes.

Table 5. Performance of ResNet101(with coin) across Different Weight Ranges

Weight Range	Sample Size	True Avg. (g)	Predicted Avg. (g)	MAE (g)	Prediction Bias (g)	Accuracy (%)
< 150 g	29	102.97	181.11	78.15	+78.15	51.72
150 – 250 g	38	199.03	197.01	28.38	-2.02	60.53
> 250 g	33	316.30	202.60	114.17	-113.60	54.55

As detailed in Table 6, the accuracy and Mean Absolute Error (MAE) of the ResNet101 model with coin reference varied across different food types. For chicken rice, the model predicted an average weight of 181.11 g, slightly above the true average of 175.40 g, with a prediction bias of +0.35 g and an accuracy of 60.00%. Crispy pork rice showed a slightly larger underestimation of 19.69 g, resulting in an MAE of 60.06 g and an accuracy of 55.00%. Fried rice performed best, achieving the highest accuracy of 65.00% and a low bias of +4.09 g. In contrast, pad thai and pad see ew had larger underestimations of around 31 g each, with accuracies of 55.00% and MAEs of over 60 g. These results suggest that the model's predictive accuracy varies across different dishes, reflecting the challenges of estimating weights for visually similar or complex meals.

Table 6. Accuracy and MAE of ResNet101(with coin) by Food Type

Food Type	Sample Size	True Avg. (g)	Predicted Avg. (g)	MAE (g)	Prediction Bias (g)	Accuracy (%)
Chicken_rice	20	175.40	181.11	28.94	+0.35	60.00
Crispy_pork_rice	20	199.85	197.01	60.06	-19.69	55.00
Fried_rice	20	203.15	202.60	35.15	+4.09	65.00
Pad_thai	20	205.90	174.07	64.46	-31.83	55.00
Pad_see_ew	20	265.05	234.02	65.24	-31.03	55.00

5 Conclusion and Discussion

This section briefly summarizes the main conclusions and provides a discussion of the experimental findings. It also highlights potential limitations and suggests future work to enhance the model's performance and expand the application scope.

5.1 **Conclusion**

This research aimed to develop a food image analysis system based on deep learning for object detection, food classification, and food weight estimation from photographs. The system consists of two core modules: the YOLOv11 model for object detection and the ResNet101 model for estimating food weight.

The study compared two annotation techniques-Normal Box and Smart Polygon. Results indicated that Smart Polygon annotations significantly improved YOLOv11's performance across key metrics such as Precision, Recall, F1-score, and mAP. Furthermore, the loss curve of the Smart Polygon-trained model was smoother and more stable, reflecting efficient learning and reduced risk of overfitting.

For weight estimation, ResNet101 (with coin) provided the best trade-off between accuracy and stability, achieving a Mean Absolute Error (MAE) of 71.12 grams and a Root Mean Squared Error (RMSE) of 91.56 grams on the unseen test set.

Key findings from evaluations on unseen data include: YOLOv11 using Smart Polygon annotation achieved average Precision, Recall, and F1-score exceeding 0.96. Food classification reached an Accuracy of 86% and a Macro F1-score of 0.86. Dishes such as Crispy Pork Rice and Pad Thai were classified with high accuracy, while Fried Rice and Pad See Ew were more error prone. Coin classification achieved 66% Accuracy and a Macro F1-score of 0.68, with the 10-baht coin attaining perfect Precision and Recall.

ResNet101 with coin outperformed its version without coin on unseen data. The model with coin reduced MAE by approximately 5 grams and RMSE by 6 grams, and improved accuracy from 48.00% to 56.00%, confirming the benefit of using a coin as a reference object to aid in interpreting physical scale.

An error distribution analysis by weight range showed optimal model performance for mid-range meals (150-250 grams), consistent with the training data average. The model tended to overestimate lighter meals (<150g) and underestimate heavier meals (>250g), demonstrating a regression toward the mean behavior. When grouped by food type, dishes with visually distinct and compact features such as Chicken Rice and Fried Rice yielded lower MAE and higher accuracy. Conversely, dispersed dishes like Pad Thai and Pad See Ew tended to be underestimation.

5.2 **Discussion**

The use of Smart Polygon annotations significantly improved YOLOv11 performance, especially for irregularly shaped objects like food. By eliminating excess background, the model could better learn specific object features, thus enhancing Precision, Recall, and mAP compared to Bounding Box. Among reference objects, the 10-baht coin was identified as the most effective due to its consistent classification accuracy, while other coins, particularly the 1-baht and 5-baht, exhibited higher error rates. Therefore, the 10-baht coin is recommended as the standard reference object for accurate scaling in weight estimation. Regarding food classification, Chicken Rice, Crispy Pork Rice, and Pad Thai demonstrated high classification accuracy, whereas Fried Rice and Pad See Ew encountered greater classification challenges due to similar visual characteristics. The model showed directional estimation bias, for after-meal images, especially those under 130 grams, being significantly overestimated, likely due to their underrepresentation in the training set. Conversely, full-plate meals exceeding 250 grams were consistently underestimated, reflecting a regression toward the mean. Training environment constraints also played a role, as images were taken with limited dish and background diversity, such as plain white dishes or repetitive designs, causing background bias when the model faced unfamiliar environments. Furthermore, the limited size and diversity of the test set relative to real-world scenarios may have introduced high variance in metrics like Accuracy, Precision, and F1-score, ultimately affecting the model's generalizability.

5.3 **Future Work**

Future work should increase the number of after-meal images featuring 10-50% food remnants on the plate to better capture real-world scenarios and reduce overestimation biases. Additionally, implementing prediction correction logic using pixel area thresholds could further improve weight estimation accuracy. For example, using a correction formula as shown:

$$\boldsymbol{weight\ corrected} = \boldsymbol{predicted\ weight} \times \boldsymbol{\alpha} + \boldsymbol{\beta}$$

If $\alpha = 0.8$ and $\beta = -10$g, the model adjusts predicted weight by reducing it 20% and subtracting an additional 10 grams.

This approach is particularly beneficial when the detected food area drops significantly, helping mitigate the tendency of the model to overestimate weight in these scenarios.

References

1. Ruenin, P., Bootkrajang, J., & Chawachat, J. (2020). A System to Estimate the Amount and Calories of Food that Elderly People in the Hospital Consume. In *Proceedings of the 11th International Conference on Advances in Information Technology (IAIT '20)*, Article No. 8, 1–7.
2. Zhang, H., Li, J., & Wang, Y. (2015). Snap-n-Eat: A Mobile Application for Food Image Classification and Calorie Estimation. In Proceedings of the ACM International Conference on Multimedia, 1002-1008.
3. Ege, T., & Yanai, K. (2019). Simultaneous Estimation of Dish Locations and Calories with Multi-Task Learning.IEICE Transactions on Information and Systems, E102-D(7), 1240–1246.
4. Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 10781–10790.
5. Nguyen, T. T., Tran, H. N., & Do, T. M. (2024). FoodMask: Instance Segmentation of Food in Images. Sensors, 24(3), 841.
6. Myers, A., Johnston, N., Rathod, V., et al. (2015). Im2Calories: Towards an Automated Mobile Vision Food Diary. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 1233–1241.
7. He, Y., Xu, C., Khanna, N., Boushey, C. J., & Delp, E. J. (2013). Food Image Analysis: Segmentation, Identification, and Weight Estimation. In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), 1–6.
8. Wang, Z., Liu, X., & Li, Y. (2018). Food Volume Estimation Using Deep Learning from Single Images. IEEE Access, 6, 12345–12352.
9. Ye, H., Xu, C., & Delp, E. (2019). Towards Learning Food Portion From Monocular Images With Cross-Domain Feature Adaptation. arXiv preprint arXiv:2103.07562.
10. Garcia, J., Jones, A., & Smith, B. (2023). Automated Food Weight and Content Estimation Using Computer Vision and Artificial Intelligence. *Sensors, 23*(15), 7660.
11. Fang, C., Lin, S., & Hsu, T. (2024). MFP3D: Monocular Food Portion Estimation Leveraging 3D Point Clouds. arXiv preprint arXiv:2411.1049