

AI-Driven Food Monitoring for Waste Reduction and Malnutrition Prevention

Axelle Penninger

Student number: 01908321

Supervisor: Prof. dr. Bruno Volckaert

Counsellor: Ir. Thomas Dupont

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering

Academic year 2024-2025

Acknowledgment

I would like to express my sincere gratitude to everyone who supported me throughout my Computer Science Engineering studies and the completion of this thesis.

First and foremost, I extend my deepest appreciation to my supervisor, Prof. Dr. Volckaert, for providing me with the opportunity to work on this fascinating topic. Your insightful feedback, thoughtful guidance during our monthly meetings, and continuous support played a crucial role in the development and completion of this work.

I would also like to thank Ir. Dupont, my counsellor, for the timely and helpful support regarding Obelisk-related questions. Your guidance was valuable in addressing the technical challenges encountered during this investigation.

My sincere thanks go to the ILVO team—Sebastiaan Verbesselt, Liam van Langenhove, and Said Benaissa—for your assistance with dataset-related inquiries and your constructive feedback, which enhanced the quality of this research.

Throughout my academic journey, I have been fortunate to be surrounded by friends who made this experience both enriching and enjoyable. To my study companions, thank you for the moments of shared concentration in the library, the countless hours of collaborative work, and the much-needed moments of relaxation that brought balance to intense academic periods. To my roommates and close friends, your support and companionship turned daily routines into memorable experiences, for which I am truly thankful.

To my significant other, thank you for your constant support, the much-needed moments of relaxation, and for always making me laugh when I needed it most.

Finally, I would like to express my heartfelt gratitude to my parents and family for your unwavering support, encouragement, and belief in me. Your understanding and patience during challenging periods have been the foundation upon which I could build my academic and personal growth.

Thank you all!

Axelle Penninger

22 May 2025

This master's dissertation is part of an exam. Any comments formulated by the assessment committee during the oral presentation of the master's dissertation are not included in this text.

AI-Driven Food Monitoring for Waste Reduction and Malnutrition Prevention

Ing. Axelle Penninger

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering

Supervisor: Prof. Dr. Bruno Volckaert, Counsellor: Ir. Thomas Dupont

Ghent University, Faculty of Engineering and Architecture

Academic year 2024-2025

Abstract

Malnutrition and food waste represent interconnected challenges in residential care centers that are typically addressed in isolation. Despite the provision of nutritionally balanced meals, significant portions remain uneaten, resulting in inadequate nutrient intake and considerable waste. Existing monitoring approaches rely on labor-intensive manual methods or automated systems that focus solely on served meals, overlooking actual consumption patterns and food waste. Moreover, these systems fail to take into account care-specific challenges, such as blended or mixed foods. This work presents LeftoVision, a computer vision-based pipeline designed to address both nutrient intake and food waste estimation through the analysis of paired images of meal plates before and after consumption. The system is trained and evaluated on a custom dataset collected by the Flanders Research Institute for Agriculture, Fisheries and Food (ILVO), containing images of meals served in residential care centers before and after consumption. The standard pipeline uses semantic segmentation to identify and isolate food components, followed by weight estimation to quantify the amounts. Nutrient intake is calculated by querying a food composition database using the identified items and their estimated weights. All information is stored in the Obelisk Core data platform and visualized via an interactive dashboard, providing healthcare professionals insights into residents' nutrient intake patterns, waste trends, and menu optimization opportunities. We evaluated three pipeline configurations. The standard approach, discussed above, demonstrated superior performance (per component: Mean Absolute Error: 22 g, Root Mean Square Error: 40 g, Concordance Correlation Coefficient: 0.53). The region-of-interest variant aimed at enhancing weight estimation through plate isolation and the dual-model approach for weight estimation distinguishes between untouched and partially consumed meals. LeftoVision establishes a foundation for nutrition and waste monitoring in residential care settings, facilitating data-driven interventions that simultaneously improve resident health outcomes and reduce food waste.

Keywords

Nutrient Intake Monitoring, Food Waste Estimation, Food Weight Estimation, Semantic Segmentation

AI-Driven Food Monitoring for Waste Reduction and Malnutrition Prevention

Ing. Axelle Penninger

Supervisor: Prof. Dr. Bruno Volckaert
Counsllor: Ir. Thomas Dupont

Abstract—Malnutrition and food waste represent interconnected challenges in residential care centers that are typically addressed in isolation. Despite the provision of nutritionally balanced meals, significant portions remain uneaten, resulting in inadequate nutrient intake and considerable waste. Existing monitoring approaches rely on labor-intensive manual methods or automated systems that focus solely on served meals, overlooking actual consumption patterns and food waste. Moreover, these systems fail to take into account care-specific challenges, such as blended or mixed foods. This work presents LeftoVision, a computer vision-based pipeline designed to address both nutrient intake and food waste estimation through the analysis of paired images of meal plates before and after consumption. The system is trained and evaluated on a custom dataset collected by the Flanders Research Institute for Agriculture, Fisheries and Food (ILVO), containing images of meals served in residential care centers before and after consumption. The standard pipeline uses semantic segmentation to identify and isolate food components, followed by weight estimation to quantify the amounts. Nutrient intake is calculated by querying a food composition database using the identified items and their estimated weights. All information is stored in the Obelisk Core data platform and visualized via an interactive dashboard, providing healthcare professionals insights into residents' nutrient intake patterns, waste trends, and menu optimization opportunities. We evaluated three pipeline configurations. The standard approach, discussed above, demonstrated superior performance (per component: Mean Absolute Error: 22 g, Root Mean Square Error: 40 g, Concordance Correlation Coefficient: 0.53). The region-of-interest variant aimed at enhancing weight estimation through plate isolation and the dual-model approach for weight estimation distinguishes between untouched and partially consumed meals. LeftoVision establishes a foundation for nutrition and waste monitoring in residential care settings, facilitating data-driven interventions that simultaneously improve resident health outcomes and reduce food waste.

Index Terms—Nutrient Intake Monitoring, Food Waste Estimation, Food Weight Estimation, Semantic Segmentation

I. INTRODUCTION

Malnutrition represents a significant health challenge in residential care settings, particularly affecting elderly populations. Research, conducted in Belgium, revealed that 63 % of residential care center residents are either malnourished or at risk, with even higher rates among those with dementia (82 %) or depression (79 %) [1].

Concurrent with this nutritional crisis is the problem of food waste in healthcare facilities. Belgian residential care centers discard approximately 15,240 tons of food annually, representing 25-40 % of all food served. This results in annual

financial losses of approximately €25,000 per facility [2] and indicates inefficiencies in food management practices within these institutions.

To address these interconnected challenges, and as part of the broader Food INtake Ecosystem (FINE) project, which aims to create a data ecosystem for nutrition and health-care [3], we have developed an automated computer vision pipeline called LeftoVision. This system estimates both food intake and waste by analyzing just two images per meal per resident—one before and one after consumption—thereby introducing minimal overhead for practical deployment. Stakeholder meetings conducted during the FINE project identified automation and minimal operational burden as essential requirements for implementation in care environments, factors that directly informed LeftoVision's two-image design. Our approach uses semantic segmentation to identify individual food components, followed by a weight estimation model that quantifies the served and leftover portions. By calculating the difference between the served and the remaining quantities, the system determines the actual nutrient intake and measures food waste.

To optimize the automated food monitoring capabilities, we developed three distinct pipeline variants, each differing in their approach to image preprocessing or weight estimation. These variants, detailed in Section III, allow us to explore optimal configurations. This work represents a significant advancement in automated nutrition monitoring for vulnerable populations, with potential applications that extend beyond residential care to other institutional settings where nutrition monitoring and food waste reduction are priorities.

The remainder of this paper is structured as follows: Section II presents related work in food nutrient intake estimation systems and discusses the research gaps we aim to address. Section III details our methodology, discussing the different components in the pipelines. Section IV provides an evaluation of our system's performance. Finally, Section V concludes with a summary of our contributions, and Section VI briefly discusses directions for future research.

II. RELATED WORK

Computer vision-based food nutrient intake estimation has evolved around two main paradigms: multi-stage pipelines and end-to-end models [4]. This section reviews both approaches and identifies key limitations that this work seeks to address.

A. Multi-Stage Architectures

Multi-stage approaches typically consist of three sequential tasks: food segmentation and classification, weight estimation (with possibly volume estimation as an intermediate step), and nutrient calculation [4].

The initial segmentation stage has seen advances, with Transformer-based models like BEiT v2 [5] and Ingrid-SAM [6] achieving mean Intersection over Union (mIoU) scores of 49.4 % and 48.8 % respectively on the FoodSeg103 dataset [7]. However, segmentation accuracy remains limited and will be even more challenging in environments such as residential care centers, where food presentations often include mixed or blended textures that eliminate visual boundaries and distinctive features critical for segmentation algorithms, inevitably reducing mIoU performance.

For volume estimation, approaches can be categorized as stereo-based, model-based, deep learning-based, depth camera-based, or perspective transformation-based [8], [9]. Lo et al. [10] achieved a notable Mean Absolute Percentage Error (MAPE) of 4.6 % using deep learning approaches for 9 food categories, while Dehais et al. [11] reported 8.2 % MAPE using stereo-based techniques across 45 categories.

Direct weight estimation methods broadly fall into two categories: pixel-area approximations, which assume a direct proportionality between the pixel area of a food component and its weight, and more sophisticated learned regression models which exploit more features than only the pixel count. Konstantakopoulos et al. [12] achieved a MAPE of 3.7 % using a regression-based pipeline across 226 food categories, outperforming the simpler pixel-based techniques which typically yield 15 % error or higher [13], [14]. It should be noted that this error metric reflects the accuracy of their entire multi-stage pipeline. However, these comparisons warrant caution, as the models were trained and evaluated on different datasets, limiting direct comparability.

B. End-to-End Architectures

End-to-end methods bypass intermediate stages by directly predicting nutrient values from input images using deep neural networks [4]. While these models can reduce cumulative error, they often require large annotated datasets.

Several studies have explored this approach for calorie estimation, with Thames et al. [15] achieving a 14.7 % MAPE using a Convolutional Neural Network (CNN), and Ruede et al. [16] reporting 9.9 % MAPE. Expanding on this, Han et al. [17] incorporated RGB-D fusion to simultaneously estimate calories, mass, and nutrient content, reaching 17.8 % MAPE. Other notable contributions include Lu et al. [18], who used multi-task learning (MTL) with CNNs for volume estimation (19.1 % MAPE), and Fang et al. [19] using GANs for energy estimation (10.9 % MAPE). These results, while informative, cannot be directly compared due to the use of different datasets across studies.

C. Research Gaps

Three critical limitations persist in the literature:

- 1) **Limited Standardized Datasets:** The field lacks comprehensive, publicly available datasets that include weight or nutritional annotations. Notable exceptions include Nutrition5k [15] and ECUSTFD [20], although these are not widely adopted in studies.
- 2) **Incomplete Consumption Tracking:** Available datasets typically contain only images of served portions without corresponding leftover images, preventing the development of systems that can track actual consumption through before-after comparisons or measure food waste accurately.
- 3) **Residential Care Challenges:** No publicly accessible datasets represent the unique food characteristics in residential care settings, such as modified textures (pureed, blended) and mixed compositions, hindering the development of solutions targeted to these environments.

Unlike previous systems limited to tracking only served portions, our system offers comprehensive monitoring throughout the entire meal cycle, specifically designed for the challenges of residential care.

III. METHODOLOGY

This section outlines the architectural design and system development of LeftoVision. We adopted a multi-stage architecture due to the limited training data available, its flexibility for future extensions, and its facilitation of domain knowledge integration at each processing step. The system is trained and evaluated on a custom dataset collected by the Flanders Research Institute for Agriculture, Fisheries and Food (ILVO), containing images of meals served in residential care centers before and after consumption.

A. System Architecture Overview

LeftoVision uses a modular processing pipeline that transforms plate images into nutritional data. The system takes images of food plates (before and after eating) as input. The core pipeline involves several stages: semantic segmentation to identify and isolate food components, weight estimation for each component, a menu lookup to identify the food item name, and querying a nutrient database to determine nutritional content based on the name and estimated weight. The system stores both served and leftover weights in the Obelisk Core data platform, along with their corresponding nutritional values. The dashboard then calculates eaten nutrients (served minus leftover) and food waste (leftover weight) to visualize food waste patterns and resident nutritional status.

We implemented three pipeline variations to explore different approaches to the problem:

- 1) **Standard Pipeline:** This is the default implementation, processing the entire plate image through all previously discussed stages. The architecture of this pipeline is visualized in Fig. 1. This figure shows an image of a served plate, but the system can also analyze images of cleared plates.
- 2) **ROI-Enhanced Pipeline:** This variant includes a Region of Interest (ROI) cropping step before weight estimation.

It identifies the plate area to focus processing on the actual food components, potentially improving weight estimation accuracy. Note that this cropping is applied after the initial segmentation module, as segmentation performance often benefits from the broader context of the full image rather than a cropped region [21].

- 3) Dual Model Pipeline: This variant uses two separate weight estimation models: one specifically trained for plates before eating and another for plates after eating. This addresses the different visual characteristics of untouched versus partially consumed food to potentially improve accuracy across both scenarios.

The system is implemented in Python 3.11.11 using standard computer vision and deep learning libraries such as OpenCV 4.11.0 [22] and PyTorch 2.6.0 [23].

B. Segmentation Module

The segmentation module is the first step in our pipeline, aiming to accurately delineate the boundaries between the food components, classify each item into one of five classes (starch, vegetables, meat/fish, sauce, and other sauce) and generate segmented images for downstream weight estimation.

We evaluated several state-of-the-art semantic segmentation models, including DeepLabV3, DeepLabV3+, and SegFormer, with various backbones such as ResNet, ResNeXt, MobileNet, and MiTB. The results of the various segmentation architectures and backbones on the validation set are listed in Table I. Based on this evaluation across several performance metrics and computational efficiency, DeepLabV3+ with a ResNeXt-50 32x4d backbone was selected.

TABLE I: Segmentation performance comparison of architectures and backbones on the validation set.

Architecture	Backbone	Parameters (M)	mIoU	mDice	mF1
DeepLabV3	ResNeXt-50 32x4d	39.11	0.45	0.55	0.55
DeepLabV3+	ResNeXt-50 32x4d	26.15	0.48	0.58	0.55
SegFormer	ResNeXt-50 32x4d	24.31	0.40	0.49	0.49
DeepLabV3+	ResNet-18	12.33	0.42	0.51	0.51
DeepLabV3+	ResNet-34	22.44	0.41	0.49	0.49
DeepLabV3+	ResNet-50	26.68	0.40	0.50	0.50
DeepLabV3+	ResNet-101	45.67	0.49	0.59	0.60
DeepLabV3+	MobileNet	4.38	0.40	0.49	0.49
DeepLabV3+	MiTb2	25.35	0.45	0.55	0.55

Note: mIoU, mDice, and mF1 refer to micro-averaged metrics computed across all classes.

The preprocessing pipeline included white balance correction using the gray-world assumption to improve consistency across varying lighting conditions, resizing to a resolution of 512×512 , and normalization using ImageNet parameters. To enhance model generalization with limited training data, we implemented standard transformations (rotations, flips, brightness/contrast adjustments) and a synthetic plate composition method. This method extracted food components from existing images using ground-truth masks and placed them on empty plates to create new compositions, effectively quadrupling the size of the training dataset.

Training was conducted using class-weighted focal loss, chosen to address class imbalance. We set the class weights

to $[1, 1, 1, 1, 4, 4]$ because there are fewer images of sauce and other sauce. The weights correspond to background, starch, vegetables, meat/fish, sauce, and other sauce, respectively. Techniques such as gradient accumulation and mixed precision training enhanced computational efficiency. Early stopping was used to prevent overfitting by monitoring validation loss. The specific hyperparameters used are detailed in Table II. The final model was trained on the combined training and validation datasets. This approach maximizes the use of available labeled data while avoiding leakage.

The output of this stage consists of individual segmented images for each food component, with non-component pixels set to black, as black typically does not appear in food items.

TABLE II: Hyperparameters for the segmentation model (DeepLabV3+).

Category	Hyperparameter	Value
Model Architecture	Backbone	ResNeXt-50 32x4d
	Encoder Weights	ImageNet
Optimization	Loss Function	Focal Loss
	Class Weights	$[1, 1, 1, 1, 4, 4]$
	Optimizer	AdamW
	Weight Decay	10^{-4}
Learning Rate	Initial LR	10^{-4}
	Scheduler	CosineAnnealingLR
	$T_{\max}$	10
	$\eta_{\min}$	10^{-6}
Training	Batch Size	32
	Gradient Accumulation Steps	2
	Mixed Precision	Enabled
Convergence	Early Stopping Patience	10
	Early Stopping $\delta_{\min}$	10^{-5}
	Best Epoch	61

C. Region of Interest Extraction

In the ROI-enhanced pipeline, the plate region is extracted prior to weight estimation. Given the consistently circular plates, the Hough Circle Transform from OpenCV is used for detection. Before circle detection, a series of preprocessing steps are applied to enhance the edges of the plate and reduce noise. These steps include grayscale conversion (specifically using the chroma blue channel of the YCrCb space), histogram equalization, adaptive gamma correction, Contrast Limited Adaptive Histogram Equalization (CLAHE), Gaussian blur, adaptive thresholding, morphological operations (closing and dilation), Canny edge detection, and again dilation.

The HoughCircles function is applied with experimentally optimized parameters for plate detection. After detection, bounding boxes are computed around the detected circles, and a margin of 100 pixels is added to ensure the entire plate is captured when cutting out the region from the segmented images. If no circles are detected, the original image is passed without modification.

D. Weight Estimation Module

The segmented images, obtained after segmentation and optional ROI extraction, are fed into the weight estimation module to determine the weight of each food component. For

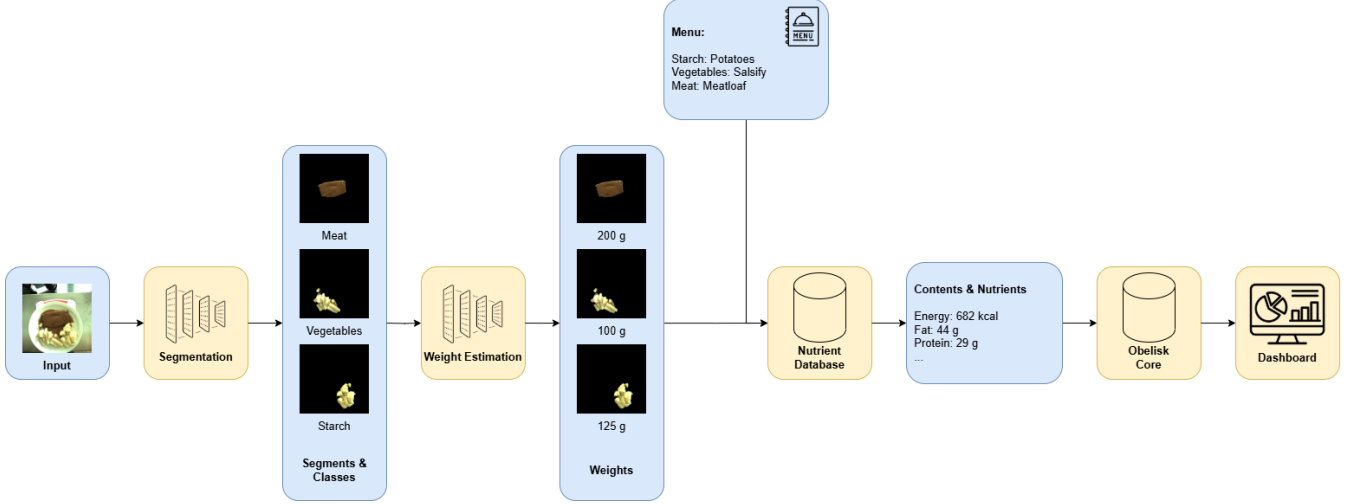


Fig. 1: High-level system architecture of the standard pipeline.

this task, we explored two distinct approaches: a simple pixel-based method and a deep learning approach.

We initially explored a pixel-based weight estimation method, establishing a linear relationship between food component pixel count and weight using category-specific regression. However, this approach proved limited due to variations in food density, as well as insufficient feature representation, resulting in weak performance with an RMSE of 51.71 g.

Motivated by these limitations, we explored deep learning techniques using CNN architectures. We evaluated several state-of-the-art CNNs, including ResNet-50, ResNet-101, AlexNet, DenseNet, GoogLeNet, and Wide ResNet-50, adapted for regression. A key enhancement was the integration of food category information as an additional input, using a multi-input architecture that combined visual features with a one-hot encoded representation of the category. Based on comparative assessment, ResNet-50 was selected as the most promising model, providing an optimal balance between performance and computational efficiency. The results of the different architectures on the validation set, after course hyperparameter tuning, are summarized in Table III.

TABLE III: Weight estimation performance comparison of architectures on the validation set (coarse tuning).

Architecture	Parameters (M)	RMSE (g)	MAE (g)	CCC
ResNet-50	25.56	46.76	33.47	0.29
ResNet-101	44.59	59.63	41.52	0.17
AlexNet	61.10	50.17	34.03	0.15
DenseNet	7.98	80.89	64.09	0.03
Wide ResNet-50	68.88	51.95	34.31	0.28
GoogLeNet	6.62	81.27	63.85	0.02

The input data for the weight estimation module were preprocessed. This involved applying the same white balance correction as in the segmentation module, resizing the segmented images to 224×224 pixels, and normalization using the ImageNet parameters. To improve generalization with

limited training data, data augmentation techniques, such as random horizontal and vertical flips, were applied. Complete retraining of all network layers was performed during training, as this approach was found to be superior to transfer learning, resulting in substantially lower RMSE (42.62 g vs. 78.37 g).

We again used early stopping based on validation loss to prevent overfitting. Hyperparameter tuning resulted in the optimal configuration detailed in Table IV. The final model was trained on the combined training and validation datasets.

For the dual-model approach, separate ResNet-50 models for served and post-consumption plates were similarly trained. At inference time, model selection can be based on the ratio of non-black pixels to black pixels in the image or a temporal sequence assumption. Their hyperparameters are also detailed in Table IV. The final models were trained on the merged training and validation datasets.

The output of this module is the estimated weight for each individual food component.

TABLE IV: Hyperparameters for weight estimation models.

Category	Hyperparameter	General Model	Cleared Plates	Served Plates
Model Architecture	Backbone	ResNet-50	ResNet-50	ResNet-50
Optimization	Loss Function	RMSE	RMSE	RMSE
	Optimizer	Adam	Adam	Adam
Learning Rate	Initial LR	5×10^{-4}	5×10^{-4}	10^{-3}
	Scheduler	StepLR	StepLR	StepLR
	Step Size	5	4	4
	Gamma	0.1	0.1	0.1
Training	Batch Size	128	128	128
Convergence	Early Stopping Patience	5	12	12
	Early Stopping $\delta_{\min}$	10^{-5}	10^{-5}	10^{-5}
	Best Epoch	12	21	28
	Warm-up Epochs	6	6	6

E. Nutrient Database Integration

To convert the food component classification and weight estimation into meaningful nutritional information, LeftoVision interfaces with the USDA Food and Nutrient Database for

Dietary Studies (FNDDS) [24], selected for its generalizability despite limitations in capturing specific items or preparation methods.

Our system first maps identified components to menu items (stored in JSON format) and then queries FNDDS using the item name. The nutritional values are then calculated by scaling the database values based on the estimated weight and the reference weights from the database.

F. Data Querying and Storage

After the computer vision pipeline has determined the nutrients and weights of served meals and leftovers, this information is stored in Obelisk Core, a robust service platform for managing large data volumes [25]. Data are organized using metrics consisting of the weight and the nutrient content, and labels including the resident ID and the component ID for efficient filtering. Data ingestion is handled via Obelisk Core’s REST API. Each implemented query function has a retry mechanism to ensure reliability, supports pagination and sends parallel request to efficiently handle large datasets over multiple days.

G. User Interface: Dashboard

The dashboard, built with Plotly Dash [26], visualizes the system’s output and serves as the primary interface for personnel. The user interface dashboard presents three primary analytical perspectives:

- 1) Resident Nutrition: Monitoring nutritional intake from population-level overview to detailed individual analysis. A part of this tab is visualized in Fig. 2.
- 2) Food Waste Analysis: Identifying waste patterns through temporal and categorical analysis.
- 3) Menu Performance Insights: Supporting optimization decisions by highlighting successful and problematic menu items.

This dashboard enables residential care staff to make evidence-based decisions for improving nutritional outcomes and reducing food waste.

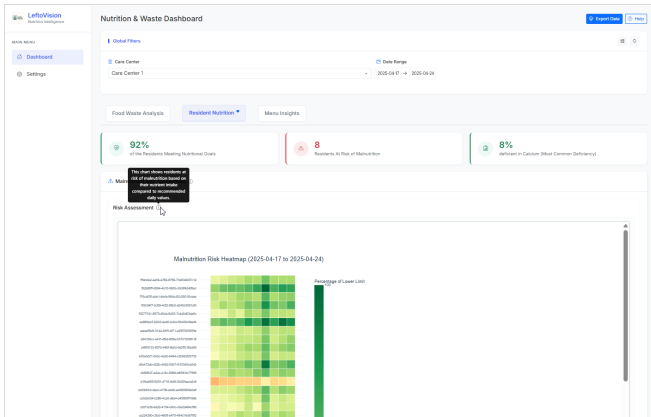


Fig. 2: Screenshot of the Resident Nutrition tab in the dashboard.

IV. EVALUATION AND RESULTS

Following the description of the system’s implementation in the preceding section, this section provides an evaluation of the developed approach.

A. Evaluation Framework

Our evaluation uses a hierarchical approach examining individual components and the complete system’s performance on an institution-based data split strategy to assess real-world generalization. At the component level, segmentation and weight estimation modules were evaluated independently using ground-truth inputs. For segmentation performance, we use IoU, Dice coefficient, Precision, Recall, and F1-score. For weight estimation, the accuracy was evaluated per food component, using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Concordance Correlation Coefficient (CCC). For the end-to-end evaluation, the three distinct pipelines were analyzed in their ability to estimate food weights from raw images. These evaluations were also conducted at the level of individual food components, comparing predicted and ground-truth weights using MAE, RMSE, and CCC, and benchmarking against a naive mean-based predictor. Computational performance metrics, including execution time, peak CPU/GPU memory usage, and CPU/GPU utilization, were measured.

B. Component-level Evaluation

All models are trained and evaluated on the IDLab GPULab infrastructure using an NVIDIA GeForce RTX 4090 GPU with 23 GB memory and 2 CPUs.

1) *Segmentation Module*: On the validation set, the model achieved a micro-averaged Dice coefficient of 0.61. The test set performance is presented in Table V, showing a micro-averaged Dice coefficient of 0.53 and a notable performance gap compared to validation likely due to the center-based data split. Error analysis using a confusion matrix (Fig. 3) provides deeper insight into classification behavior. Common misclassifications include food components being incorrectly predicted as background. The model also shows confusion between regular sauce and meat/fish, as well as between other sauce and starch. These errors are likely due to inconsistent labeling. Additionally, vegetables emerged as the worst-performing class.

TABLE V: Segmentation performance metrics on test set.

Class	IoU	Dice	Precision	Recall	F1-Score
Starch	0.61	0.76	0.73	0.79	0.76
Vegetables	0.30	0.45	0.66	0.35	0.45
Meat/Fish	0.43	0.59	0.61	0.58	0.59
Sauce	0.17	0.27	0.29	0.28	0.27
Other Sauce	0.10	0.14	0.25	0.10	0.14
Micro-average	0.43	0.53	0.59	0.51	0.53

2) *Weight Estimation Module*: On the validation set, the general model achieved an RMSE of 42.62 g and MAE of 36.33 g. Table VI presents the quantitative evaluation on the test set. The general model approach (46.93 g RMSE)

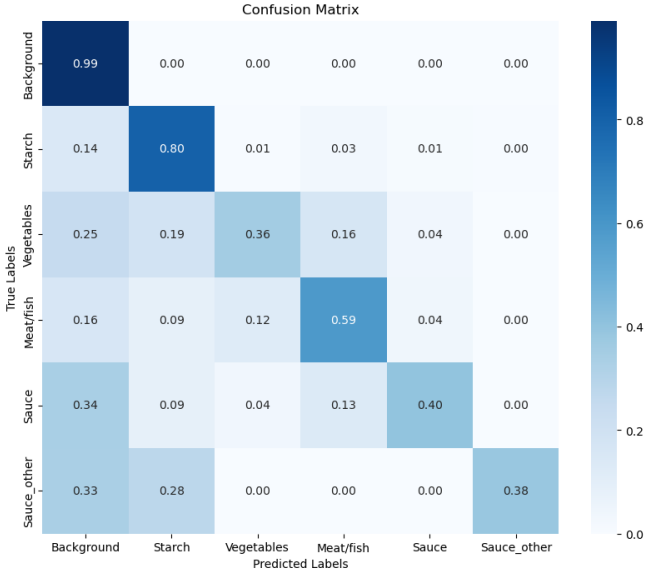


Fig. 3: Confusion matrix for the segmentation model on the test set.

outperformed the specialized models for serving (52.71 g RMSE) and clearing phases (56.76 g RMSE), suggesting potential overfitting of the specialized models. The error analysis revealed regression toward the mean and poor performance for the sauce categories.

TABLE VI: Evaluation metrics for weight estimation models on test set.

Model Type	RMSE (g)	MAE (g)	CCC
General model	46.93	32.88	0.44
Specialized - Serving phase	52.71	39.34	0.26
Specialized - Clearing phase	56.76	41.03	0.15

C. End-to-End System Evaluation

End-to-end evaluation compared the three pipeline variants on the test set with a mean-based baseline (Table VII). This baseline model predicts a constant value—the average weight from the training dataset—for all test samples, serving as a simple benchmark to assess the added value of more complex models. The standard pipeline demonstrated the best performance (MAE = 21.98 g, RMSE = 40.45 g, CCC = 0.53), showing a 24.8 % improvement in MAE over the baseline. The ROI-enhanced and dual model variants showed slightly worse results. The error analysis for the standard pipeline (Fig. 4) revealed systematic patterns, including classifying absent components as present and failing to detect present components. The cumulative error analysis (Fig. 5) indicated that 50 % of predictions fell within ± 8.8 g of ground truth, demonstrating acceptable accuracy for nutritional monitoring applications.

TABLE VII: End-to-end performance comparison of pipeline variants.

Pipeline	MAE (g)	RMSE (g)	CCC
Baseline	29.24	41.27	0.34
Standard	21.98	40.45	0.53
ROI-Enhanced	24.69	47.23	0.35
Dual Model - Ratio	28.85	48.09	0.47
Dual Model - Ground Truth	30.31	50.14	0.45

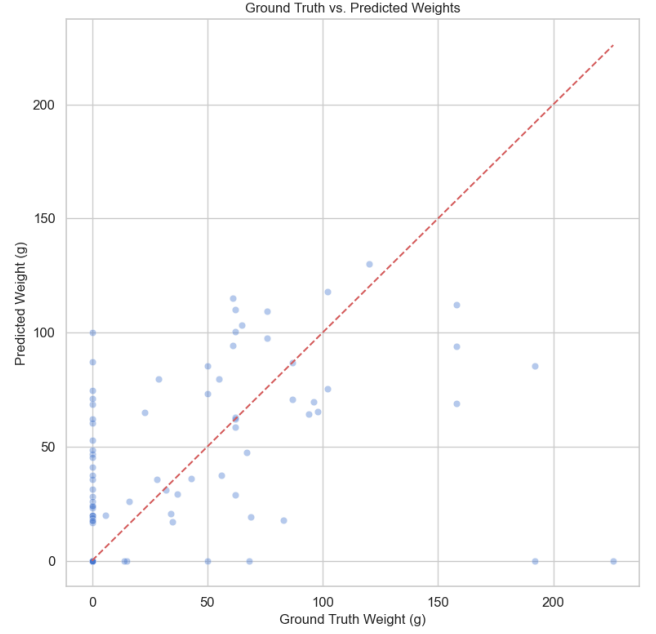


Fig. 4: Ground-truth weights versus predictions.

D. Computational Efficiency

System performance was evaluated by measuring execution times across all pipeline stages on a laptop equipped with an Intel Core i7 vPro (8th generation) CPU and an NVIDIA Quadro P600 GPU. As shown in Table VIII, nutrient database fetching emerged as the most time-consuming stage of the pipeline. The system demonstrated significant efficiency gains through batch processing, achieving 1.20 seconds per image in a 32-image batch compared to 3.69 seconds for single-image processing. This corresponds to processing meals for approximately 25 residents per minute in batch mode.

Resource utilization analysis revealed moderate hardware demands, with GPU utilization peaking at only 43.17 % during image processing tasks. CPU utilization peaked at 144.71 % during batch processing image loading, demonstrating efficient multi-threading capabilities. Memory consumption remained reasonable throughout operation, with a peak GPU RAM usage of 2.95 GB during batch segmentation. These results indicate that LeftoVision can be deployed effectively without requiring extreme or high-end hardware configurations.

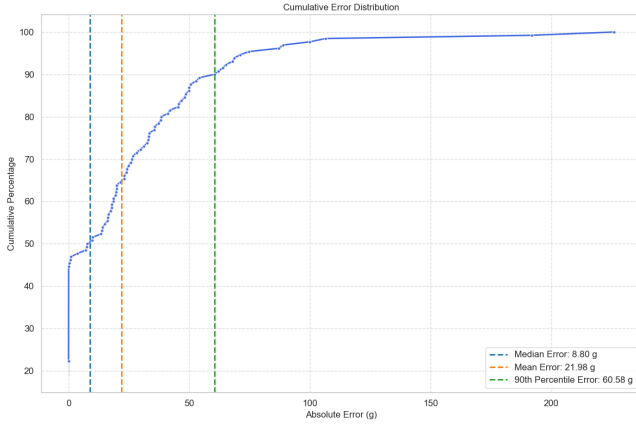


Fig. 5: Cumulative error analysis showing that 50 % of predictions fall within ± 8.8 g of ground truth, while 90 % fall within ± 60.58 g.

TABLE VIII: Mean wall clock time per pipeline stage (batch and single image processing).

Pipeline Stage	32 Images (s)	1 Image (s)	Avg. Per Image (Batch) (s)
Image Loading	5.84 ± 1.74	0.17 ± 0.06	0.18
Segmentation Model	5.94 ± 0.78	0.21 ± 0.17	0.19
Segment Extraction	0.21 ± 0.03	0.02 ± 0.01	0.01
Weight Estimation	4.69 ± 0.59	0.18 ± 0.04	0.15
Nutrient Database Fetching	20.98 ± 37.95	2.87 ± 0.49	0.66
Nutrient Processing	0.00 ± 0.00	0.00 ± 0.00	0.00
Obelisk Data Ingestion	0.65 ± 1.50	0.24 ± 0.16	0.02
Total Pipeline	38.31	3.69	1.20

V. CONCLUSION

To address the interconnected challenges of resident malnutrition and significant food waste prevalent in residential care facilities, we developed LeftoVision, a nutrient intake monitoring system. The system analyzes pre- and post-consumption food images through a multi-stage pipeline that identifies food components via semantic segmentation, estimates component weights, performs menu lookups, and calculates nutritional content before storing the data in the Obelisk Core platform for dashboard visualization.

Our evaluation demonstrates that LeftoVision achieves an MAE of 21.98 g for food weight estimation, enabling reliable tracking of even small dietary changes. Furthermore, its processing speed of 25 resident meals per minute demonstrates practical scalability for deployment in larger facilities.

Beyond technical performance, LeftoVision addresses two critical gaps identified in the literature. First, unlike existing solutions that predominantly monitor served food without corresponding consumption data, our system captures both pre- and post-consumption images to enable true intake tracking through before-after analysis that quantifies actual nutritional intake and food waste simultaneously. Second, LeftoVision is specifically designed and validated for residential care environments, addressing the challenges of modified textures, mixed compositions, and smaller portion variations that existing solutions fail to accommodate.

This research establishes a foundation for improved clinical nutrition management, and more sustainable food practices within residential care environments.

VI. FUTURE WORK

Building upon the foundation established by LeftoVision, several promising directions exist to further enhance the system’s accuracy, robustness, and practical utility in residential care settings.

Future research should explore more advanced model architectures, such as developing unified models for combined segmentation and weight estimation, or implementing specialized weight estimation models tailored to distinct food categories for improved accuracy. Another promising enhancement involves allowing the input of food item names and their corresponding densities into the weight estimation models.

Addressing current data limitations represents another critical advancement opportunity. This includes investigating synthetic data generation techniques to expand training datasets and integrating 3D depth sensor information to provide spatial context that could improve weight prediction accuracy beyond the current 2D image-based approach.

Enhancing system robustness for real-world deployment involves implementing uncertainty quantification mechanisms to identify and flag predictions with lower confidence levels. Furthermore, developing transfer learning and domain adaptation methodologies will enable efficient system deployment across diverse care facilities while minimizing the need for extensive site-specific data collection and retraining.

Pursuing these avenues will contribute to evolving LeftoVision into an even more reliable tool for combating malnutrition and reducing food waste in healthcare environments.

REFERENCES

- [1] M. F. J. Vandewoude, J. P. van Wijngaarden, L. De Maesschalck, Y. C. Luiking, and A. Van Gossom, “The prevalence and health burden of malnutrition in Belgian older people in the community or residing in nursing homes: results of the NutriAction II study,” **Aging Clin. Exp. Res.**, vol. 31, pp. 175–183, 2019.
- [2] Foodlab Brugge, “Voedselverlies in de zorgsector: Hoe kan je het voorkomen?,” [Online]. Available: https://foodwin.org/wp-content/uploads/2021/03/20180924_PU_brochureA5VoedselVerliesZorgweb-1-1.pdf [Accessed: Oct. 10, 2024].
- [3] imec, “FINE: Food INtake Ecosystem,” [Online]. Available: <https://www.imec.be/nl/vlaamse-innovatiemotor/impactdomeinen/smart-health/fine>. [Accessed: May 15, 2025].
- [4] W. Wang, W. Min, T. Li, X. Dong, H. Li, and S. Jiang, “A review on vision-based analysis for automatic dietary assessment,” *Trends in Food Science & Technology*, vol. 122, pp. 223–237, 2022, doi: 10.1016/j.tifs.2022.02.017.
- [5] G. Sinha, K. Parmar, H. Azimi, A. Tai, Y. Chen, A. Wong, and P. Xi, “Transferring Knowledge for Food Image Segmentation using Transformers and Convolutions,” *arXiv preprint, arXiv:2306.09203*, 2023. [Online]. Available: <https://arxiv.org/abs/2306.09203>
- [6] L. Chen, B. Wang, and J. Zhang, “IngredSAM: Open-world food ingredient segmentation via a single image prompt,” *J. Imaging*, vol. 10, no. 12, Art. no. 305, 2024, doi: 10.3390/jimaging10120305. [Online]. Available: <https://www.mdpi.com/2313-433X/10/12/305>
- [7] X. Wu, X. Fu, Y. Liu, E.-P. Lim, S. C. H. Hoi, and Q. Sun, “A large-scale benchmark for food image segmentation,” in *Proc. 29th ACM Int. Conf. Multimedia*, Chengdu, China, 2021, pp. 506–515.

- [8] F. P. W. Lo, Y. Sun, J. Qiu, and B. Lo, "Image-based food classification and volume estimation for dietary assessment: A review," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 7, pp. 1926–1939, Jul. 2020, doi: 10.1109/JBHI.2020.2987943.
- [9] J. Sultana, B. Md. Ahmed, M. M. Masud, A. K. O. Huq, M. E. Ali, and M. Naznin, "A study on food value estimation from images: Taxonomies, datasets, and techniques," *IEEE Access*, vol. 11, pp. 45910–45935, 2023, doi: 10.1109/ACCESS.2023.3274475.
- [10] F. P.-W. Lo, Y. Sun, J. Qiu, and B. Lo, "A novel vision-based approach for dietary assessment using deep learning view synthesis," in *Proc. 2019 IEEE 16th Int. Conf. Wearable and Implantable Body Sensor Netw. (BSN)*, 2019, pp. 1–4.
- [11] J. Dehais, M. Anthimopoulos, S. Shevchik, and S. Mougiakakou, "Two-view 3D reconstruction for food volume estimation," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 1090–1099, 2016.
- [12] F. S. Konstantakopoulos, E. I. Georga, and D. I. Fotiadis, "A novel approach to estimate the weight of food items based on features extracted from an image using boosting algorithms," *Scientific Reports*, vol. 13, no. 1, p. 21040, 2023.
- [13] G. Ciocca, P. Napoletano, and R. Schettini, "Food recognition and leftover estimation for daily diet monitoring," in *New Trends in Image Analysis and Processing-ICIAP 2015 Workshops*, Genoa, Italy, Sep. 2015, pp. 334–341.
- [14] Y. A. Sari, S. Adinugroho, J. M. Maligan, E. N. Candra, F. Utaminigrum, and N. Nur'Aini, "Leftovers Food Recognition using Deep Neural Network and Regression Approach for Objective Visual Analysis Estimation," in *Proc. 2021 4th Int. Conf. Computer and Informatics Engineering (IC2IE)*, pp. 24–29, 2021, doi: 10.1109/IC2IE53219.2021.9649045.
- [15] Q. Thames, A. Karpur, W. Norris, F. Xia, L. Panait, T. Weyand, and J. Sim, "Nutrition5k: Towards Automatic Nutritional Understanding of Generic Food," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, Jun. 2021, pp. 8903–8911.
- [16] R. Ruede, V. Heusser, L. Frank, A. Roitberg, M. Haurilet, and R. Stiefelhagen, "Multi-Task Learning for Calorie Prediction on a Novel Large-Scale Recipe Dataset Enriched with Nutritional Information," in *Proc. 25th Int. Conf. Pattern Recognition (ICPR)*, 2021, pp. 4001–4008, doi: 10.1109/ICPR48806.2021.9412839.
- [17] Y. Han, Q. Cheng, W. Wu, and Z. Huang, "DPF-Nutrition: Food nutrition estimation via depth prediction and fusion," *Foods*, vol. 12, no. 23, p. 4293, 2023.
- [18] Y. Lu, D. Allegra, M. Anthimopoulos, F. Stanco, G. M. Farinella, and S. Mougiakakou, "A multi-task learning approach for meal assessment," in *Proc. Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management (CEA/MADiMa '18)*, Stockholm, Sweden, 2018, pp. 46–52, doi: 10.1145/3230519.3230593.
- [19] S. Fang, Z. Shao, R. Mao, C. Fu, E. J. Delp, F. Zhu, D. A. Kerr, and C. J. Boushey, "Single-view food portion estimation: Learning image-to-energy mappings using generative adversarial networks," in *Proc. 2018 25th IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, 2018, pp. 251–255, doi: 10.1109/ICIP.2018.8451461.
- [20] Y. Liang and J. Li, "Computer vision-based food calorie estimation: dataset, method, and experiment," *CoRR*, vol. abs/1705.07632, 2017. [Online]. Available: <http://arxiv.org/abs/1705.07632>
- [21] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint, arXiv:1706.05587*, 2017.
- [22] G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.
- [23] A. Paszke, "PyTorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.
- [24] J. B. Montville, J. K. C. Ahuja, C. L. Martin, K. Y. Heendeniya, G. Omolewa-Tomobi, L. C. Steinfeldt, J. Anand, M. E. Adler, R. P. LaComb, and A. Moshfegh, "USDA food and nutrient database for dietary studies (FNDDS), 5.0," *Procedia Food Science*, vol. 2, pp. 99–112, 2013.
- [25] imec IDLab, Obelisk Core Documentation, 2025. [Online]. Available: <https://obelisk.pages.ilabt.imec.be/obelisk-core/obelisk-core.html>
- [26] Plotly Technologies Inc., Collaborative Data Science. Montréal, QC: Plotly Technologies Inc., 2015. [Online]. Available: <https://plot.ly>

Contents

Acknowledgment	ii
Abstract	iv
Extended Abstract	v
List of Figures	xviii
List of Tables	xx
List of Acronyms	xxii
1 Introduction	1
1.1 Problem Statement and Motivation	1
1.1.1 Prevalence and Impact of Malnutrition in Residential Care Centers	1
1.1.2 Food Waste Challenges in Residential Care Centers	2
1.1.3 Limitations of Current Nutrient Monitoring Methods	2
1.1.4 Need for a Combined Food Intake and Waste Monitoring System	2
1.2 Objectives of the Research	2
1.2.1 Functional Requirements	3
1.2.2 Non-Functional Requirements	3
1.3 Overview of the Proposed Approach	4
1.4 Thesis Structure	4
2 Fundamentals of Computer Vision	6
2.1 Core Computer Vision Tasks Relevant to Food Analysis	6
2.1.1 Image Classification	6
2.1.2 Image Segmentation	7
2.1.3 Image Regression	10
2.2 Commonly Used Technologies for Food Segmentation and Weight Estimation	11
2.2.1 Convolutional Neural Networks	13

2.2.2	Beyond Convolutional Neural Networks: Long Short-Term Memory Networks and Vision Trans-	16
	formers	
2.2.3	Probabilistic Deep Learning	18
2.2.4	Specialized Segmentation Techniques	19
3	Literature Review	22
3.1	Architectures for Food Nutrient Intake Estimation	22
3.1.1	Multi-Stage Architectures	23
3.1.2	End-to-End Architectures	23
3.2	Public Datasets for Food Image Analysis	24
3.2.1	Food Segmentation Datasets	25
3.2.2	Weight, Volume and Nutrient Estimation Datasets	25
3.3	Key Components of a Multi-Stage Architecture	26
3.3.1	Food Segmentation Techniques	27
3.3.2	Food Volume and Weight Estimation Methods	28
3.3.3	Nutrient Calculation from Food Consumption Data	31
3.4	End-to-End Architectures	31
3.5	Existing Hardware and Software Solutions	32
3.5.1	Hardware Solutions	33
3.5.2	Software Solutions	33
3.6	Research Landscape in Food Image Analysis	35
3.6.1	Leading Research Groups and Universities	35
3.6.2	Primary Publication Venues	36
3.7	Challenges in Image-Based Food Nutrient Estimation for Residential Care Settings	37
3.8	Gaps in Existing Systems	38
3.8.1	Limited Standardized Datasets	38
3.8.2	Lack of Food Analysis in Residential Care Settings	38
3.8.3	Lack of End-to-End Consumption Tracking	39
3.8.4	Limited Focus on Small Portion and Leftover Estimation	39
3.8.5	Lack of Transparency in Commercial Solutions	39
3.9	Summary of the Literature Review	39
4	Data Collection and Dataset Preparation	41
4.1	Data Collection Methodology	41
4.2	Dataset Composition and Structure	43
4.2.1	Tabular Data Architecture	43
4.2.2	Multimodal Image Data	44

4.2.3	Annotations	44
4.3	Exploratory Data Analysis	45
4.3.1	Initial Dataset Characteristics	45
4.3.2	Distributional Analysis	45
4.3.3	Image Quality Assessment	49
4.3.4	Characteristics of the Dataset After Quality Filtering	51
4.3.5	Analysis of Segmentation Annotations	52
4.3.6	Potential Challenges	53
4.4	Data Splitting Strategy	54
4.4.1	Data Splitting Approach	54
4.4.2	Institutional Allocation	55
4.5	Recommendations for Future Data Collection	56
4.5.1	Data Acquisition Enhancements	56
4.5.2	Annotation Methodology Refinements	57
5	Architectural Design and System Development	58
5.1	System Architecture Overview	58
5.1.1	Core Pipeline Design	58
5.1.2	Used Technologies	59
5.1.3	Implementation Architecture	60
5.1.4	Pipeline Variants	61
5.2	Food Segmentation Module	63
5.2.1	Implementation Framework	63
5.2.2	Model Architecture Selection	64
5.2.3	Input Data Preprocessing and Augmentation Strategy	64
5.2.4	Training Methodology	66
5.2.5	Extracting Segments	68
5.3	Region of Interest Extraction	68
5.3.1	Image Preprocessing	70
5.3.2	Hough Circle Detection	71
5.3.3	Region Extraction	72
5.4	Weight Estimation Module	72
5.4.1	Implementation Framework	73
5.4.2	Model Architecture Selection	73
5.4.3	Input Data Preprocessing and Augmentation Strategy	76
5.4.4	Training Methodology	76
5.4.5	Dual-Model Approach for Different Plate States	77

5.5	Nutrient Database Integration	79
5.5.1	Database Selection and Considerations	80
5.5.2	Nutrient Retrieval Pipeline	80
5.5.3	Nutritional Parameters	81
5.5.4	Current Limitations and Future Improvements	81
5.6	Data Querying and Storage	82
5.6.1	Data Organization	82
5.6.2	Data Ingestion	83
5.6.3	Data Querying	84
5.7	User Interface: Dashboard	87
5.7.1	Technology Selection	87
5.7.2	Dashboard Design	88
5.7.3	Resident Nutrition Monitoring Interface	88
5.7.4	Food Waste Analysis Interface	90
5.7.5	Menu Insights Interface	91
6	Evaluation and Results	93
6.1	Evaluation Framework	93
6.1.1	Evaluation Strategy and Metrics	93
6.1.2	Computational Performance Measurement Methodology	94
6.1.3	Dataset Organization	95
6.2	Component-level Evaluation	95
6.2.1	Segmentation Module	95
6.2.2	Weight Estimation Module	99
6.3	End-to-End System Evaluation	106
6.3.1	Comparative Results	106
6.3.2	Prediction Error Analysis	109
6.3.3	Computational Performance Analysis	110
6.4	Conclusion	115
6.4.1	Key Findings	115
6.4.2	Limitations	116
	Conclusion	119
	Future Work	121
	Sustainability and Ethical Reflection	123
	Contribution to the United Nations Sustainable Development Goals	123

Ethical Considerations	123
References	126
Appendices	134
Use of Generative AI	135
Code Availability	136

List of Figures

2.1	Comparison of semantic, instance, and panoptic image segmentation [1].	8
2.2	Taxonomy of technologies and concepts often used in food image analysis.	12
2.3	Typical architecture of a CNN [2].	14
2.4	Visualization of a residual connection in a neural network [3].	14
2.5	Inception Module used in GoogLeNet [4].	15
2.6	Architecture of an LSTM network [5].	17
2.7	Visualization of the Transformer architecture [6].	17
2.8	U-Net architecture [7].	20
2.9	Comparison of standard and dilated convolutions [8].	20
3.1	Taxonomy of sensors used for food portion size estimation [9].	23
3.2	Multi-stage architecture for image sensor-based food nutrient intake estimation.	24
3.3	End-to-end architecture for image sensor-based food nutrient intake estimation.	24
4.1	Mobile data acquisition system used for dataset capture.	42
4.2	Bidirectional image capture process during food serving (a) and plate clearing (b).	42
4.3	ERD of the dataset database schema.	43
4.4	Frequency distribution of image counts per meal moment.	46
4.5	Frequency distribution of image counts per residential care center.	47
4.6	Frequency distribution of image counts per food category.	47
4.7	Frequency distribution of image counts per meal phase.	48
4.8	Weight distribution per residential care center (violin plot).	48
4.9	Examples of images with red overlay artifacts.	50
4.10	Examples of incomplete plate captures.	50
4.11	Images showing illumination and imaging variability across residential care centers (continued on next page).	50
4.12	Examples of images with over- and underexposure issues.	52
4.13	Examples of segmentation annotations.	53
4.14	Example of the issues with the segmentation annotations.	53

5.1	High-level architecture of the LeftoVision standard pipeline.	59
5.2	Execution flow of the pipeline.	60
5.3	High-level architecture of the ROI-enhanced pipeline.	61
5.4	High-level architecture of the dual weight estimation pipeline.	62
5.5	Examples of segmentation predictions from FoodSAM.	64
5.6	Examples of the synthetic data augmentation.	66
5.7	Training and validation loss curves for the segmentation model.	68
5.8	Preprocessing pipeline for ROI extraction (plate detection).	69
5.9	Examples of cropped plate images produced by the ROI extraction module.	72
5.10	Pixel count versus actual weight with regression line for each food category.	75
5.11	Training and validation loss curves for the general weight estimation model.	77
5.12	Training and validation loss curves for the specialized serving and clearing weight estimation models.	79
5.13	Example JSON representation of a meal menu.	81
5.14	Screenshot of the Resident Nutrition tab in the LeftoVision dashboard.	89
5.15	Screenshot of the Food Waste Analysis tab in the LeftoVision dashboard.	91
5.16	Screenshot of the Menu Insights tab in the LeftoVision dashboard.	92
6.1	Confusion matrix for the segmentation module on the test set.	98
6.2	Actual versus predicted weights for each food category on the test set (general model).	104
6.3	Weight estimation error distribution per food category on the test set (general model).	105
6.4	Density comparison of actual versus predicted weights per food category on the test set (general model).	106
6.5	Actual versus predicted weights for the standard pipeline on the test set.	110
6.6	Distribution of weight estimation errors for the standard pipeline on the test set.	111
6.7	Cumulative distribution of absolute weight estimation errors for the standard pipeline on the test set.	111
6.8	Examples of segmentation predictions on the test set with error visualization.	118

List of Tables

3.1	Datasets for food segmentation.	25
3.2	Comparison of state-of-the-art food segmentation techniques on FoodSeg103.	27
3.3	Comparison of state-of-the-art food segmentation techniques on UEC-FOODPIX Complete.	28
3.4	Advantages and disadvantages of food volume estimation approaches [10, 11].	29
3.5	Comparison of approaches for food volume estimation.	30
3.6	Comparison of approaches for direct food weight estimation.	31
3.7	State-of-the-art end-to-end approaches for food nutrient estimation or related tasks.	32
3.8	Overview of commercial food waste management systems.	34
3.9	Overview of nutrient intake applications.	34
4.1	Initial dataset characteristics.	45
4.2	Statistical summary of weight measurements per residential care center.	49
4.3	Overview of the dataset composition after quality filtering.	52
4.4	Segmented image distribution across data partitions per food category.	56
5.1	Hyperparameters for the segmentation model (DeepLabV3+).	69
5.2	Performance of the pixel-based weight estimation method on the test set.	74
5.3	Hyperparameters used for weight estimation models.	78
5.4	List of tracked nutrients and their units.	82
5.5	List of metrics stored in the Obelisk Core data platform.	83
5.6	Summary of implemented Obelisk Core query methods.	85
5.7	Comparative analysis of Grafana and Plotly Dash for dashboard implementation.	87
6.1	Segmentation performance comparison of architectures and backbones on the validation set.	96
6.2	Impact of input resolution on segmentation performance on the validation set.	96
6.3	Impact of synthetic data augmentation on segmentation performance on the validation set.	96
6.4	Impact of different class weights on segmentation performance on the validation set.	97
6.5	Segmentation performance metrics on the test set per food category.	97
6.6	Summary of computational performance metrics for the segmentation model.	100
6.7	Weight estimation performance comparison of architectures on the validation set (course tuning).	100

6.8	Comparison of transfer learning versus training from scratch for weight estimation on the validation set (fine tuning).	101
6.9	Impact of input resolution on weight estimation performance on the validation set (fine tuning). .	101
6.10	Weight estimation performance metrics on the validation set for general and specialized models. .	102
6.11	Weight estimation performance metrics on the test set for general and specialized models. . . .	103
6.12	Computational performance metrics for general and specialized weight estimation models. . . .	105
6.13	End-to-end performance comparison of the pipeline variants on the test set.	107
6.14	Mean wall clock time per pipeline stage (batch and single image processing).	112
6.15	Performance of Obelisk Core data query methods.	113
6.16	Performance of dashboard component rendering times.	113
6.17	CPU and GPU utilization per pipeline stage.	114
6.18	Peak CPU and GPU memory consumption per pipeline stage.	114

List of Acronyms

AI	Artificial Intelligence.
AP	Average Precision.
API	Application Programming Interface.
ASPP	Atrous Spatial Pyramid Pooling.
BNN	Bayesian Neural Network.
CCC	Concordance Correlation Coefficient.
CLAHE	Contrast Limited Adaptive Histogram Equalization.
CNN	Convolutional Neural Network.
CPU	Central Processing Unit.
CVPR	Computer Vision and Pattern Recognition.
DenseNet	Densely Connected Convolutional Network.
EDA	Exploratory Data Analysis.
ERD	Entity Relationship Diagram.
FCN	Fully Convolutional Network.
FINE	Food INtake Ecosystem.
FLOPS	Floating Point Operations Per Second.
FNDDS	Food and Nutrient Database for Dietary Studies.
GAN	Generative Adversarial Network.
GPU	Graphics Processing Unit.
gRPC	Remote Procedure Calls.
i.i.d.	independently and identically distributed.
ICME	International Conference on Multimedia and Expo.
ICPR	International Conference on Pattern Recognition.
IEEE	Institute of Electrical and Electronics Engineers.
ILVO	Instituut voor Landbouw-, Visserij- en Voedingsonderzoek.
IoU	Intersection over Union.
JSON	JavaScript Object Notation.

KPI	Key Performance Indicator.
LSTM	Long Short-Term Memory.
MAE	Mean Absolute Error.
MAP	Mean Average Precision.
MAPE	Mean Absolute Percentage Error.
MPA	Mean Pixel Accuracy.
MSE	Mean Squared Error.
MTL	Multi-Task Learning.
NeRF	Neural Radiance Fields.
NNDB	National Nutrient Database.
PA	Pixel Accuracy.
PNG	Portable Network Graphics.
RAM	Random Access Memory.
ReLU	Rectified Linear Unit.
ResNet	Residual Network.
REST	REpresentational State Transfer.
RGB	Red Green Blue.
RMSE	Root Mean Squared Error.
RNN	Recurrent Neural Network.
ROI	Region of Interest.
RSQL	RESTful Service Query Language.
SaaS	Software as a Service.
SAM	Segment Anything Model.
SDG	Sustainable Development Goal.
SETR	Segmentation Transformer.
Std Dev	Standard Deviation.
TIFF	Tagged Image File Format.
UN	United Nations.

ViT Vision Transformer.

WASP Waterfall Atrous Spatial Pooling.

YAML YAML Ain't Markup Language.

1

Introduction

1.1 Problem Statement and Motivation

Malnutrition, defined as an imbalance of nutrients through excess or deficiency, represents a critical issue within residential care settings, significantly affecting health outcomes and quality of life for elderly residents. Despite the efforts of care centers to provide nutritionally adequate meals, food is often not consumed entirely, resulting in unmet nutritional requirements and considerable waste. This presents a dual challenge: addressing the nutritional deficiencies of residents while also managing resource inefficiency.

1.1.1 Prevalence and Impact of Malnutrition in Residential Care Centers

Research consistently highlights the prevalence of malnutrition among elderly individuals in care settings. A comprehensive Belgian study involving 3,299 older adults living in community dwellings and residential care centers found that 12 % were classified as malnourished, and 44 % were at risk of malnutrition. The situation in residential care centers was particularly alarming, with 63 % of residents either malnourished or at risk. Vulnerable groups, such as individuals with dementia and depression, exhibited even more concerning rates, with 82 % of those with dementia and 79 % of those with depression affected by malnutrition or at risk [12].

The consequences of malnutrition extend beyond immediate health concerns. Inadequate nutrition compromises the immune system, increasing susceptibility to infections and exacerbating pre-existing conditions. Malnutrition also leads to complications such as electrolyte imbalances, anemia, and chronic fatigue. These issues ultimately contribute to declining functional capacity, with malnourished elderly individuals experiencing reduced physical strength and cognitive function, heightened frailty, increased falls, and higher fracture risk [13].

These health impacts translate to significant healthcare system burdens, as malnourished patients typically require longer hospital stays and more intensive care, generating substantial additional costs [13].

1 Introduction

1.1.2 Food Waste Challenges in Residential Care Centers

Parallel to malnutrition concerns, food waste represents a substantial environmental and economic challenge. According to the Vlaams Ketenplatform Voedselverlies, approximately 15,240 tons of food are wasted annually in Belgium's healthcare sector, making it one of the main contributors to food waste [14].

A 2017 pilot project in residential care centers and hospitals in Bruges revealed that food waste accounted for 25 % to 40 % of the total food served. The economic impact is considerable—residential care centers faced losses of around €25,000 per year from wasted food alone [14]. These findings underscore the need for more efficient food management strategies to reduce waste and mitigate both financial and environmental impacts.

1.1.3 Limitations of Current Nutrient Monitoring Methods

Traditional food intake monitoring in elderly care relies on manual methods such as food diaries and meal weighing. Although weighing offers precision, these approaches are labor-intensive, error-prone, and impractical for routine implementation [15].

Furthermore, several automated nutrient intake monitoring systems exist, but few focus on accurate leftover assessment, and none are specifically tailored to elderly populations in institutional settings. Most focus predominantly on served meals, overlooking the unique challenges of leftover estimation—a crucial component not only for precise nutrient intake evaluation but also for food waste estimation. These gaps will be discussed in detail in Section 3.8.

1.1.4 Need for a Combined Food Intake and Waste Monitoring System

The significant prevalence of malnutrition, the considerable scale of food waste, and the inherent limitations of current monitoring methods collectively underscore the need for technological innovation. This research, conducted as part of the Food INTake Ecosystem (FINE) project [16], responds to these challenges. Through stakeholder meetings, the FINE project identified key implementation requirements for care environments: solutions must be automated and impose minimal operational burden on staff to be practical in real-world care settings.

1.2 Objectives of the Research

Building on the identified needs, this thesis aims to develop an automated tool for malnutrition detection and food waste reduction in elderly care centers, while simultaneously reducing the workload of nursing staff.

To address these challenges, we will develop a proof-of-concept system that uses computer vision techniques to estimate food waste and quantify nutrient intake, with a primary focus on data captured in residential care

1 Introduction

centers. The system requires only two images per resident—one before and one after the meal—minimizing time investment from care staff. These images are automatically analyzed to estimate the food actually consumed and the food wasted. Although the system will initially be trained on data from residential care facilities, its design will allow for adaptation to other institutional settings, should it be trained on appropriate data. In addition, a user-friendly dashboard will allow nursing staff to easily interpret results.

1.2.1 Functional Requirements

The functional requirements of the system are:

- **Nutrient Intake Tracking:** The system must quantify the nutrient intake per resident based on images, taking into account the served meal and the leftovers.
- **Food Waste Estimation:** The system must estimate the food waste, thus determining the amount of food that was served but not consumed.
- **User-Friendly Dashboard:** The system must provide an intuitive dashboard for nursing staff to access food waste, menu insights, and nutrient intake data.

1.2.2 Non-Functional Requirements

The non-functional requirements of the system are:

- **Accuracy:** The system must provide weight estimations with a Mean Absolute Error (MAE) below 29 g per component, which represents the baseline error achieved by simply predicting the average weight from our dataset. Furthermore, a typical meal has an average total weight of approximately 400 g to 600 g and is generally composed of three main components: vegetables, protein, and a starchy element. An MAE of 29 g per component corresponds to an error of about 15-22 % relative to the weight of each individual component.
- **Usability:** The dashboard must be intuitive, requiring minimal training for nursing staff to use effectively.
- **Efficiency:** The system should reduce the time and effort nursing staff spend on tracking food intake and waste, allowing them to focus more on patient care.
- **Processing Time:** The system must be able to process data within 4 hours after the meal, as real-time processing is not required due to the gradual development of malnutrition, making periodic assessments sufficient for monitoring nutritional status.

1.3 Overview of the Proposed Approach

To address these requirements, we propose LeftoVision, a computer vision system monitoring food intake and waste via paired pre- and post-meal images. LeftoVision follows a multi-stage processing pipeline:

1. **Food Segmentation:** Using a segmentation model, LeftoVision identifies and separates individual food components on each plate.
2. **Weight Estimation:** For each food component, the system estimates its weight based on visual characteristics using deep learning models.
3. **Menu Identification:** The system matches the detected food items with the facility's menu to get the name of the food component.
4. **Nutritional Retrieval:** LeftoVision first retrieves nutritional information from a food composition database based on the identified food item names. It then combines this data with the estimated weight of each component to calculate the nutritional content consumed or wasted.
5. **Data Storage:** The nutritional data are stored persistently to enable historical analysis and retrieval.
6. **Data Visualization:** Results are presented through a dashboard interface that highlights malnutrition risks, menu insights and food waste statistics.

As subsequent chapters explore the theoretical foundations, this overview provides context for understanding how these elements contribute to the final implementation described in Chapter 5.

1.4 Thesis Structure

The remainder of this thesis is organized into five additional chapters that progressively build from background information to the development and evaluation of our proposed solution.

Chapter 2 explores the fundamental concepts of computer vision relevant to food analysis, including image classification, segmentation, and regression techniques. This foundation chapter also examines commonly used technologies for food segmentation and weight estimation to establish the technical groundwork for our approach.

Chapter 3 presents a comprehensive literature review of existing food nutrient estimation systems, analyzing relevant datasets, key architectural components, and available solutions while identifying critical research gaps that our work addresses.

Chapter 4 details the data collection methodology used in this research, providing exploratory analysis of the dataset, explaining the data splitting strategy implemented, and suggesting potential improvements for future

1 Introduction

data collection efforts.

Chapter 5 describes the system's architectural design and the implementation of its various components, including the segmentation model, the region extraction technique, the weight estimation models, the nutrient database integration, the data management system, and the dashboard interface.

Chapter 6 evaluates the system performance through rigorous component-level and end-to-end assessment, analyzing the results to demonstrate the efficacy of our approach.

The thesis concludes with a summary of contributions, directions for future research, and ethical reflections on the societal impact of our work and its alignment with broader sustainability goals.

2

Fundamentals of Computer Vision

This chapter presents an overview of fundamental computer vision techniques, providing the necessary background to support the understanding of the subsequent chapters. Computer vision, a subfield of computer science, focuses on enabling machines to interpret and analyze visual information from their environment in a manner that mimics human perception. The chapter first outlines the core computer vision tasks relevant to food analysis, including a description of each task and the evaluation metrics commonly used to assess performance. Then it examines frequently adopted techniques, with particular attention to those implemented in this study or in related research. Readers seeking a more comprehensive treatment of these topics are referred to the cited literature.

2.1 Core Computer Vision Tasks Relevant to Food Analysis

This section discusses three computer vision tasks relevant to this work: classification, segmentation, and regression. Each task is introduced with a brief explanation of its purpose and significance in the context of food analysis. Additionally, commonly used evaluation metrics are presented to provide insight into performance assessment. While many other computer vision tasks exist, this thesis focuses only on these three as they are directly applicable to the food analysis system being developed.

2.1.1 Image Classification

Image classification is a fundamental task in computer vision that involves assigning a predefined category to an entire image based on its visual content. It serves as the basis for more advanced tasks such as object detection and image segmentation. In food analysis, classification is commonly used to identify different types of food items or assess food quality. In this thesis, classification is applied to distinguish between various food items and is integrated into the segmentation model.

2 Fundamentals of Computer Vision

Common evaluation metrics for image classification:

- **Accuracy:** The proportion of correctly classified images out of all images.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

- **Precision:** The proportion of true positive predictions among all positive predictions. Precision reflects the model's ability to avoid falsely labeling negative instances as positive, i.e., it measures the accuracy of positive predictions.

$$\text{Precision} = \frac{TP}{TP + FP}$$

where TP represents the number of true positives and FP is the number of false positives.

- **Recall:** The proportion of true positive predictions among all actual positives. Recall indicates the model's ability to identify all relevant instances, i.e., it measures how well the model detects positive cases.

$$\text{Recall} = \frac{TP}{TP + FN}$$

where FN denotes the number of false negatives.

- **F1-score:** The harmonic mean of precision and recall, balancing both metrics.

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Top-k Accuracy:** The proportion of times the true class is within the top-k predicted classes.

$$\text{Top-k Accuracy} = \frac{\text{Number of correct top-k predictions}}{\text{Total number of predictions}}$$

2.1.2 Image Segmentation

Image segmentation is a pixel-level classification task that partitions an image into meaningful regions or segments. It plays a crucial role in applications requiring precise localization and boundary delineation of objects. There are three primary types of image segmentation: semantic segmentation, instance segmentation, and panoptic segmentation.

- Semantic segmentation assigns each pixel to a specific category but does not differentiate between individual objects within the same category. For example, all potatoes in an image would be labeled as 'potato' without distinguishing between separate instances.
- Instance segmentation extends semantic segmentation by identifying and differentiating individual objects of the same class, assigning each a unique label. Unlike semantic segmentation, it excludes the background from consideration.

2 Fundamentals of Computer Vision

- Panoptic segmentation combines both approaches, providing a comprehensive view of the scene by simultaneously classifying pixels into semantic categories and distinguishing individual object instances.

Figure 2.1 illustrates the differences between these segmentation techniques. In this thesis, image segmentation is used to accurately localize food items on plates.

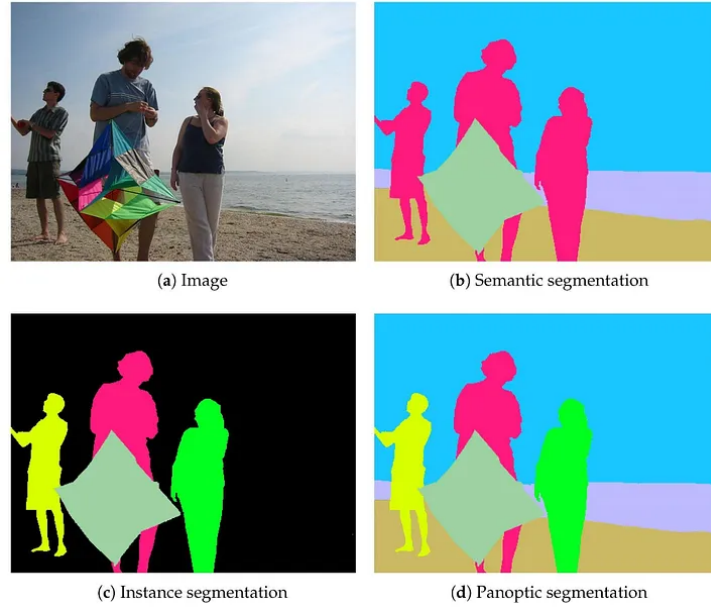


Figure 2.1: Comparison of semantic, instance, and panoptic image segmentation [1].

Image segmentation performance is typically assessed using various quantitative metrics that evaluate different aspects of model prediction accuracy. These metrics provide complementary insights into segmentation quality, helping researchers understand model strengths and limitations.

First, we will discuss the overlap-based metrics. Overlap-based metrics assess the spatial alignment and similarity between predicted and ground truth segmentation regions. These metrics are used to evaluate a model's ability to capture the spatial extent of objects.

- **Intersection over Union (IoU):** Measures the overlap between predicted (A) and ground truth (B) regions by comparing their intersection to their union. It indicates segmentation accuracy, with values closer to 1 representing more precise segmentation.

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|}$$

- **Mean Intersection over Union (mIoU):** Calculates the average IoU across all classes in a multi-class segmentation task. There are two common approaches:

2 Fundamentals of Computer Vision

1. **Macro-Average mIoU:** Calculates the IoU for each class independently and then averages these values, giving equal weight to each class regardless of its frequency.

$$\text{Macro-Average mIoU} = \frac{1}{N} \sum_{i=1}^N \frac{|A_i \cap B_i|}{|A_i \cup B_i|} = \frac{1}{N} \sum_{i=1}^N \frac{TP_i}{TP_i + FP_i + FN_i}$$

2. **Micro-Average mIoU:** Aggregates the contributions of all classes to compute the average metric, giving equal weight to each sample classification.

$$\text{Micro-Average mIoU} = \frac{\sum_{i=1}^N |A_i \cap B_i|}{\sum_{i=1}^N |A_i \cup B_i|} = \frac{\sum_{i=1}^N TP_i}{\sum_{i=1}^N (TP_i + FP_i + FN_i)}$$

where N is the number of classes, A_i and B_i are the predicted and ground truth masks for class i .

- **Dice Coefficient:** An alternative overlap metric that emphasizes the size of the overlap relative to the total area of both regions. Unlike IoU, it is less severe in penalizing under- and over-segmentation.

$$\text{Dice Coefficient} = \frac{2|A \cap B|}{|A| + |B|}$$

Next to the overlap-based metrics there are also pixel-based metrics. Pixel-based metrics evaluate segmentation accuracy at the individual pixel level. These metrics are simple to compute but can be sensitive to class imbalance and do not capture spatial relationships as effectively as overlap-based metrics.

- **Pixel Accuracy (PA):** Calculates the proportion of pixels correctly classified across the entire image. Simple but potentially misleading in datasets with uneven class distributions.

$$\text{Pixel Accuracy} = \frac{\text{Number of correctly classified pixels}}{\text{Total number of pixels in the image}}$$

- **Mean Pixel Accuracy (MPA):** Computes the average of pixel-wise accuracies for each class independently. By giving equal weight to each class, regardless of how frequently it appears, MPA mitigates the bias inherent in PA, which would otherwise over-represent dominant classes.

$$\text{MPA} = \frac{1}{N} \sum_{i=1}^N \frac{\text{Number of correctly classified pixels of class } i}{\text{Total number of pixels of class } i}$$

As a third kind of metrics, we have precision- and recall-based metrics. These provide a more nuanced evaluation of segmentation performance by considering the trade-off between detection accuracy and completeness.

- **Average Precision (AP):** Integrates the precision-recall curve to create a single score that captures model performance across different confidence thresholds. Particularly useful for evaluating the quality of instance segmentation.

$$AP_i = \int_0^1 \text{Precision}(\text{Recall} = r) dr$$

2 Fundamentals of Computer Vision

- **Mean Average Precision (MAP):** Averages the AP across all classes N , providing a comprehensive assessment of multi-class segmentation performance.

$$\text{MAP} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i$$

Finally, we have boundary metrics, which focus specifically on the accuracy of segmentation edges, providing insights into how precisely the model delineates object boundaries. Unlike standard overlap-based metrics, these focus solely on the quality of the predicted outline, involving boundary precision and boundary recall.

- **Boundary F1-score:** Evaluates the accuracy of segmentation boundaries by calculating the harmonic mean of boundary precision and recall. Crucial for applications requiring precise object edge detection.

$$\text{Boundary F1-score} = 2 \cdot \frac{\text{Boundary Precision} \cdot \text{Boundary Recall}}{\text{Boundary Precision} + \text{Boundary Recall}}$$

2.1.3 Image Regression

Image regression is a broad computer vision technique that goes beyond traditional classification by predicting continuous numerical values from visual data. In food analysis, this approach enables precise estimations such as measuring food portion sizes, quantifying leftover food weight, or determining nutritional content from image characteristics. In this thesis, the weight estimation task is an image regression task.

Common evaluation metrics for image regression:

- **Mean Absolute Error (MAE):** Calculates the average absolute difference between predicted and actual values.

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

where y_i is the actual (ground truth) value for the (i)-th instance and $\hat{y}_i$ is the predicted value for the (i)-th instance.

- **Mean Squared Error (MSE):** Measures the average squared difference between predicted and actual values, heavily penalizing larger errors.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- **Root Mean Squared Error (RMSE):** Represents the square root of the MSE, thereby expressing the error in the same units as the original data. RMSE facilitates a more intuitive interpretation of the prediction error while preserving the penalization of large deviations inherent in MSE.

$$\text{RMSE} = \sqrt{\text{MSE}}$$

2 Fundamentals of Computer Vision

- **Mean Absolute Percentage Error (MAPE):** Represents the average percentage difference between predicted and actual values, providing a percentage-based error metric.

$$\text{MAPE} = \frac{1}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\%$$

- **R-squared (R^2):** Represents the proportion of variance explained by the regression model.

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

- **Concordance Correlation Coefficient (CCC):** Measures both precision and accuracy to determine how well the predicted values agree with the actual values, assessing both their correlation and their deviation from the 45-degree line of perfect agreement.

$$\text{CCC} = \frac{2\rho\sigma_y\sigma_{\hat{y}}}{\sigma_y^2 + \sigma_{\hat{y}}^2 + (\mu_y - \mu_{\hat{y}})^2}$$

where ρ is the Pearson correlation coefficient between predicted and actual values, σ_y and $\sigma_{\hat{y}}$ are the standard deviations, and μ_y and $\mu_{\hat{y}}$ are the means of the actual and predicted values, respectively. CCC ranges from -1 to 1, with 1 indicating perfect agreement. Unlike R^2 , CCC accounts for both systematic bias and random error [17].

2.2 Commonly Used Technologies for Food Segmentation and Weight Estimation

This section presents key technologies essential for food image analysis, structured into four principal categories: feature extraction architectures, sequential & contextual understanding, uncertainty handling, and semantic segmentation specialists. These categories encompass techniques relevant to both related research and this thesis.

Figure 2.2 illustrates the taxonomy of deep learning architectures used in food analysis. Feature extraction relies on Convolutional Neural Networks (CNNs) as foundational components, with enhancements from residual architectures like ResNet and DenseNet, which improve depth through skip connections. Efficient designs such as EfficientNet and MobileNet optimize computational resources while maintaining accuracy.

2 Fundamentals of Computer Vision

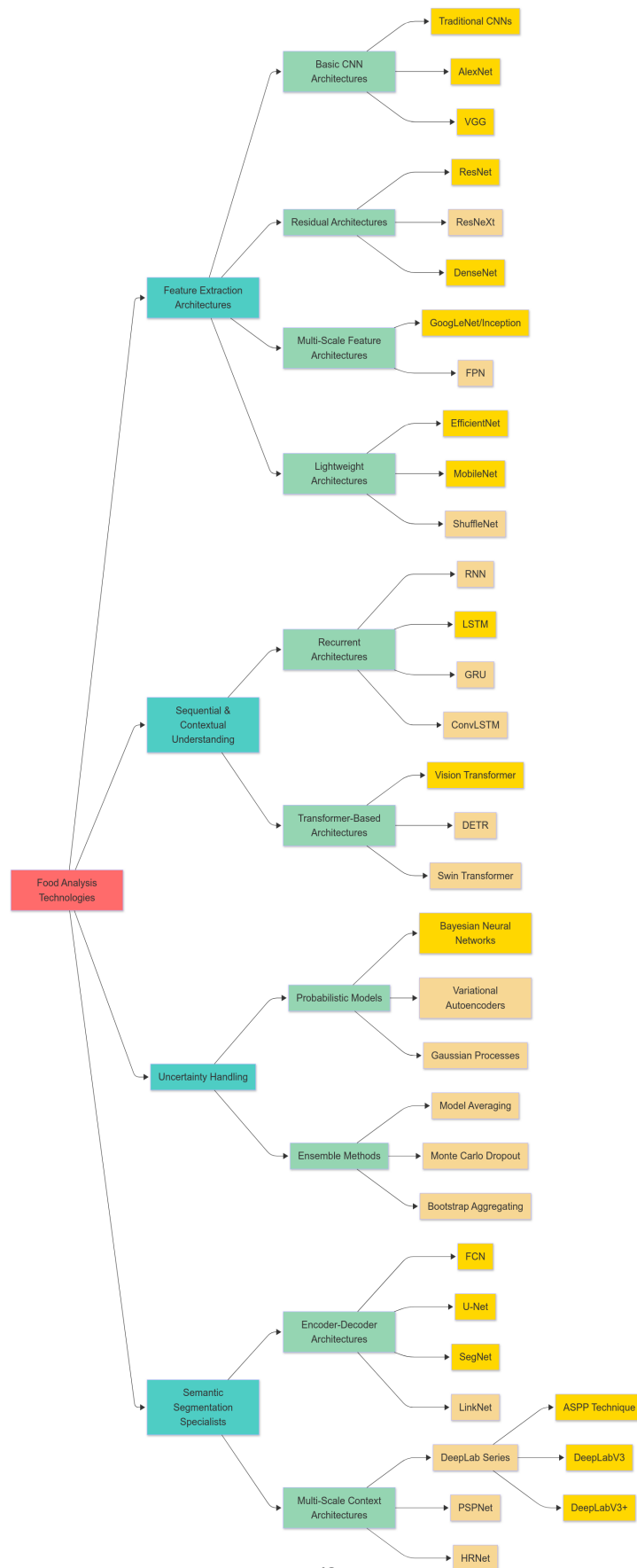


Figure 2.2: Taxonomy of technologies and concepts often used in food image analysis.

2 Fundamentals of Computer Vision

For sequential and contextual understanding, Long Short-Term Memory (LSTM) networks and Vision Transformers (ViTs) capture spatial dependencies in complex food scenes. Uncertainty quantification is addressed through probabilistic deep learning techniques, such as Bayesian Neural Networks (BNNs), which enable the model to account for uncertainty in its predictions.

Semantic segmentation plays a key role in delineating individual food items. Architectures such as Fully Convolutional Networks (FCN), U-Net, and SegNet enable pixel-wise classification, while the DeepLab series, leveraging Atrous Spatial Pyramid Pooling (ASPP), enhances contextual awareness for complex food images.

The yellow components in the taxonomy represent architectures which are either used in this work or in the related work. The following sections examine these technologies in detail, focusing on their architectural principles.

2.2.1 Convolutional Neural Networks

The foundation of modern visual analysis lies in CNNs, which have revolutionized computer vision through their ability to automatically learn hierarchical visual features. This section explores both the traditional CNN architecture and its modern variants, focusing on innovations that have made deep learning practical for complex visual tasks.

Traditional CNN Architecture

CNNs emerged from the pioneering work of LeCun et al. [18]. The key innovation of CNNs lies in their use of local receptive fields, shared weights, and spatial subsampling, design choices that dramatically reduce the number of parameters compared to fully connected networks while preserving sensitivity to visual features regardless of their position in the image.

The modern CNN architecture typically consists of the following layers and is visualized in Figure 2.3:

- Convolutional layers that apply learned filters across the input space.
- Pooling layers that reduce spatial dimensions while retaining dominant features.
- Activation functions (such as ReLU, Sigmoid, Tanh, and Leaky ReLU) that introduce nonlinearity.
- Fully connected layers that perform final classification or regression.

CNNs can be used for both classification and regression tasks and can also be used in more complex segmentation architectures as backbone.

2 Fundamentals of Computer Vision

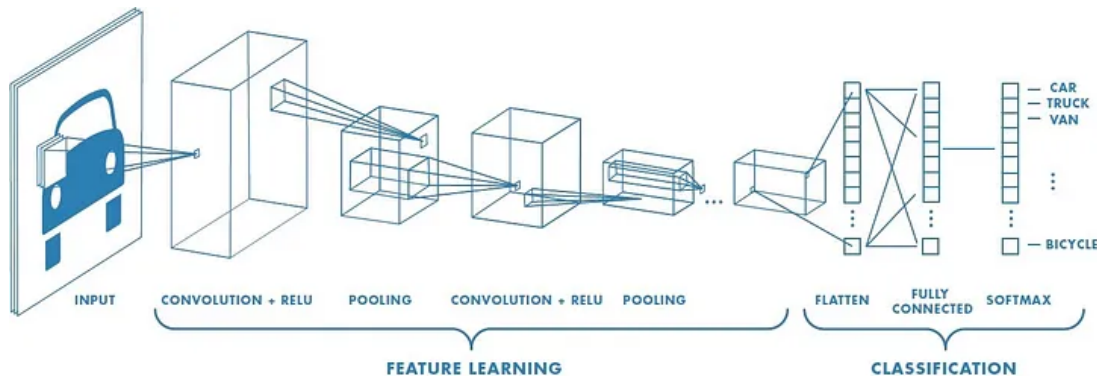


Figure 2.3: Typical architecture of a CNN [2].

Advanced CNN Architectures

Over the years several specialized CNN architectures have been developed to enhance the performance across various computer vision tasks. The following paragraphs highlight key CNN variants.

AlexNet was one of the pioneering CNN architectures that demonstrated the feasibility of deep networks for large-scale image classification tasks. It uses multiple convolutional layers, max-pooling layers and fully connected layers, combined with the ReLU activation function. AlexNet played a critical role in advancing the use of GPUs for deep learning and won the ImageNet challenge in 2012, establishing CNNs as the leading approach for image classification [19].

ResNet addresses the problem of vanishing gradients in very deep networks by introducing residual connections, which allow gradients to flow through identity mappings. This structure, visualized in Figure 2.4, enables the training of extremely deep networks, such as ResNet-50 or ResNet-101, without degradation in performance. ResNet has shown remarkable performance in image classification tasks, as it learns robust feature representations through deeper layers [20].

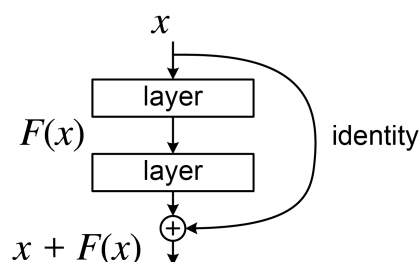


Figure 2.4: Visualization of a residual connection in a neural network [3].

GoogLeNet introduced the Inception module, which allows for multi-scale feature extraction by applying convolutions of different kernel sizes in parallel. The Inception module is visualized in Figure 2.5. This architecture is

2 Fundamentals of Computer Vision

much deeper than AlexNet, but more efficient in terms of parameter usage, making it a highly scalable network. Inception networks are particularly effective in image classification tasks due to their ability to capture both local and global features in an image [4]. Another popular inception network is Inception-V3 [21].

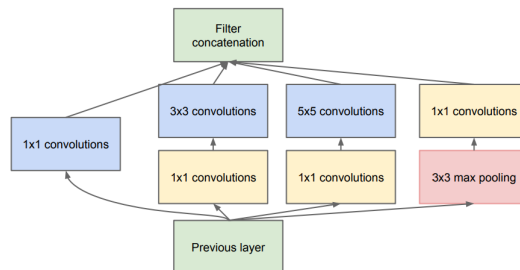


Figure 2.5: Inception Module used in GoogLeNet [4].

DenseNet introduces a novel connectivity pattern where each layer is connected to every other layer in a feed-forward fashion. This dense connectivity promotes feature reuse, allowing layers to access the outputs of all previous layers, which mitigates the vanishing gradient problem and reduces the number of parameters required. DenseNet also improves computational efficiency by enabling the network to learn more compact representations. It has shown strong performance across tasks such as image classification and medical image segmentation, particularly where efficient learning of detailed feature maps is crucial [22].

EfficientNet is a family of models that balances accuracy and computational cost by systematically scaling the network's depth, width and resolution using a compound scaling method. This method ensures that all three dimensions are scaled in a balanced manner, allowing EfficientNet to achieve state-of-the-art performance while using fewer resources compared to traditional models. EfficientNet is particularly effective for tasks that require high accuracy with limited computational power, such as mobile and edge applications in object detection, image classification and other resource-constrained environments [23]. Similarly designed for resource-constrained environments, MobileNet models achieve efficiency primarily through the use of depthwise separable convolutions. This factorization of standard convolutions significantly reduces computational cost and model size while maintaining competitive accuracy [24].

These advanced CNN architectures have significantly improved the performance and efficiency of computer vision models through innovative concepts. While these examples showcase key advancements, it's important to recognize that numerous other sophisticated CNN architectures exist, each contributing to the progress in visual intelligence.

2 Fundamentals of Computer Vision

2.2.2 Beyond Convolutional Neural Networks: Long Short-Term Memory Networks and Vision Transformers

Beyond CNNs, LSTM networks and ViTs offer alternative approaches to image analysis by deviating from the convolutional paradigm. LSTM networks capture sequential dependencies and have been adapted for spatial and temporal modeling in visual tasks. ViTs, leveraging self-attention mechanisms, enable global feature extraction without the constraints of local receptive fields. This section examines their architectures.

Long Short-Term Memory Networks

LSTM networks, introduced by Hochreiter and Schmidhuber in 1997 [25], were originally designed to address the vanishing gradient problem in Recurrent Neural Networks (RNNs). While primarily used for sequential data, LSTMs have been adapted for image analysis by capturing spatial dependencies across image patches or modeling temporal changes in video sequences.

A key innovation of LSTMs is the memory cell architecture, illustrated in Figure 2.6, which incorporates explicit gating mechanisms:

- Forget gate: Determines which spatial or temporal features should be discarded.
- Input gate: Regulates the integration of new features into the memory cell.
- Output gate: Controls the extracted feature representations for further processing.

These gates, implemented with sigmoid activation functions, enable the network to selectively retain information over many time steps or patches, effectively creating a form of differentiable memory.

Transformer Architectures

Transformers, introduced by Vaswani et al. in 2017 [6], marked a paradigm shift from recurrent and convolutional approaches to sequence modeling. Originally developed for natural language processing, their success has led to significant advancements in computer vision through adaptations such as the ViT [26].

The core innovation of Transformers is the self-attention mechanism, which enables parallel computation and models long-range dependencies without the sequential constraints of RNNs. This mechanism computes weighted relationships between all positions in a sequence, allowing global feature interactions. Mathematically, given queries Q , keys K , and values V , the scaled dot-product attention is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

2 Fundamentals of Computer Vision

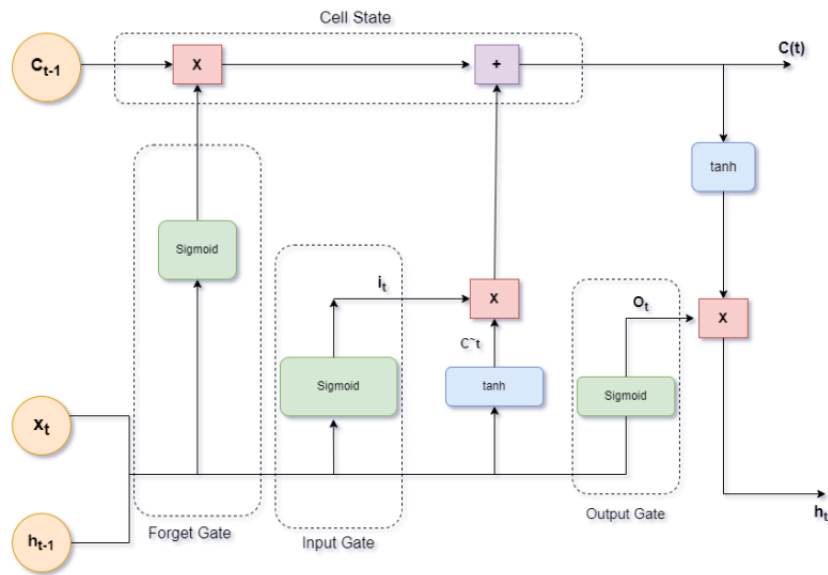


Figure 2.6: Architecture of an LSTM network [5].

where d_k represents the dimension of the keys. A visualization of the Transformer architecture is provided in Figure 2.7.

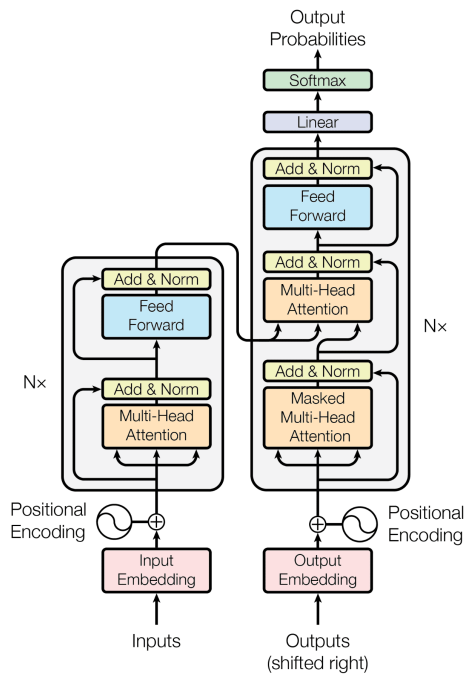


Figure 2.7: Visualization of the Transformer architecture [6].

2 Fundamentals of Computer Vision

The ViT extends this framework to image processing by transforming an image into a sequence of patch embeddings. The process consists of the following steps:

1. Splitting the input image into fixed-size patches.
2. Linearly embedding each patch into a feature vector.
3. Adding positional embeddings to retain spatial information.
4. Processing the resulting sequence with a standard Transformer encoder.

Unlike CNNs, which extract features through local operations and build spatial understanding gradually, ViTs process all image patches at once using global self-attention, enabling them to model complex spatial relationships from the very first layer.

2.2.3 Probabilistic Deep Learning

Real-world visual analysis must contend with inherent uncertainties arising from occlusions, lighting variations, and ambiguous cases. Traditional deep learning models provide point estimates without quantifying uncertainty, which can be problematic in critical applications. This section explores Bayesian learning as a probabilistic approach that explicitly models and quantifies uncertainty.

BNNs enhance standard neural networks by treating weights as random variables drawn from probability distributions rather than fixed values. This probabilistic formulation enables the model to express uncertainty in its predictions [27].

In a Bayesian framework, the objective is to compute the posterior distribution over the network weights given the observed data:

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)}$$

where w represents the network weights, D is the training data, $p(D|w)$ is the likelihood, $p(w)$ is the prior, and $p(D)$ is the evidence.

For making a prediction y^* on a new input x^* , we integrate over all possible weight values:

$$p(y^*|x^*, D) = \int p(y^*|x^*, w)p(w|D)dw$$

2 Fundamentals of Computer Vision

The motivation for BNNs arises from the limitations of deterministic networks, which include:

- Susceptibility to overfitting, particularly with limited training data.
- Lack of uncertainty quantification in predictions.
- Vulnerability to adversarial examples and out-of-distribution data.

By offering uncertainty quantification through probabilistic reasoning, BNNs enhance reliability. This can be valuable in our application, as it allows nursing staff to intervene and manually correct weight estimations when the model indicates uncertainty.

2.2.4 Specialized Segmentation Techniques

This section explores key architectures specifically designed for semantic segmentation tasks. Semantic segmentation requires pixel-level predictions while preserving global contextual awareness.

Fully Convolutional Networks

FCNs, introduced by Long et al. in 2015 [28], marked a paradigm shift in dense prediction tasks like semantic segmentation. Traditional CNNs typically end with fully connected layers that discard spatial information, making them suitable for image-level classification but unsuitable for tasks requiring pixel-wise output. FCNs addressed this limitation by transforming the standard classification architecture into a fully convolutional network. This involves replacing the fully connected layers with convolutional layers (specifically, 1×1 convolutions) to preserve spatial resolution and enable the network to output feature maps that retain spatial dimensions. However, since the intermediate feature maps in CNNs are often reduced in resolution by operations like pooling, FCNs employ upsampling using transposed convolutional layers (sometimes referred to as "deconvolutional layers"). This upsampling step restores the spatial resolution of the final feature maps to match the original input image size, thereby generating spatially consistent, pixel-wise segmentation maps.

FCNs have inspired several architectural variants that address specific limitations:

- U-Net [29] introduced a symmetric encoder-decoder structure with extensive skip connections. This design preserves spatial information more effectively than standard FCNs by directly linking corresponding layers in the contracting and expanding paths, enabling precise localization while maintaining contextual awareness. The U-Net architecture is visualized in Figure 2.8.
- SegNet [30] refined the encoder-decoder paradigm by introducing pooling indices. Instead of learning to upsample, SegNet reuses encoder pooling indices to guide the upsampling process, reducing model complexity while preserving boundary details.

2 Fundamentals of Computer Vision

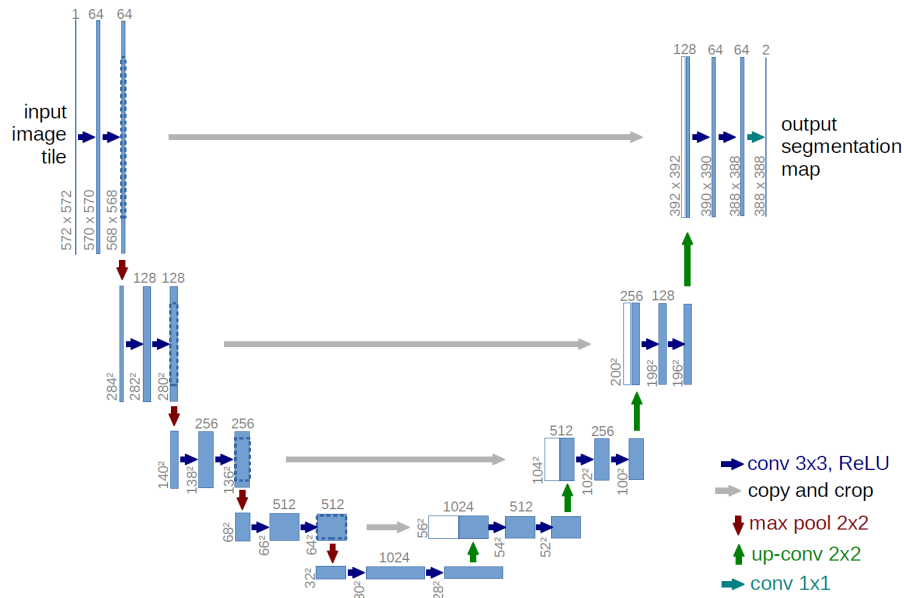


Figure 2.8: U-Net architecture [7].

Atrous Spatial Pyramid Pooling

Traditional segmentation architectures faced a significant trade-off: achieving both a broad contextual understanding and fine-grained spatial details. Deeper networks, which use pooling layers to capture context effectively, often lose the precise spatial information needed for accurate boundaries. Conversely, models designed to maintain high spatial resolution typically have limited receptive fields, preventing them from incorporating sufficient surrounding context. ASPP, introduced in DeepLab [31], enhances segmentation by effectively addressing this trade-off. It achieves this by using dilated (atrous) convolutions, which expand the receptive field of the filters without increasing the number of parameters or reducing the spatial resolution of the feature maps. The difference between standard and dilated convolutions is visualized in Figure 2.9. By applying multiple parallel dilated convolutions with different dilation rates to the input feature map, ASPP effectively integrates information across various scales, capturing multi-scale context efficiently while preserving spatial resolution.

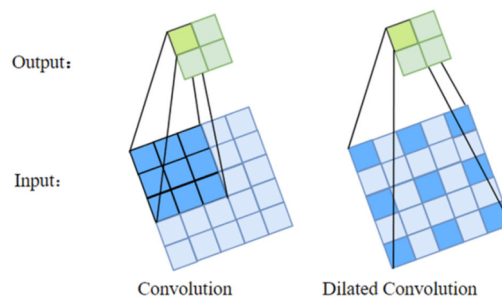


Figure 2.9: Comparison of standard and dilated convolutions [8].

2 Fundamentals of Computer Vision

Subsequent refinements of DeepLab further enhanced its effectiveness:

- DeepLabV3 [32] improved ASPP by incorporating batch normalization and global context through image-level features, enhancing both local and global segmentation accuracy.
- DeepLabV3+ [33] introduced a decoder module for sharper object boundaries and leveraged the Xception backbone for more efficient feature extraction.

These advancements made DeepLab architectures particularly effective for applications requiring precise boundary delineation while maintaining computational efficiency.

3

Literature Review

This chapter presents a comprehensive review of existing research on food nutrient intake and food waste estimation. It first explores the diverse sensor technologies used in this domain, before focusing particularly on imaging sensor-based approaches due to their accuracy and feasibility in elderly care settings. Section 3.1 examines architectures for food nutrient intake estimation, contrasting multi-stage and end-to-end approaches. Section 3.2 analyzes public datasets used for training and evaluating models. The subsequent section, Section 3.3, provides a detailed analysis of individual components within a multi-stage architecture, including segmentation and regression techniques. In Section 3.4, we discuss existing solutions which implement the end-to-end architecture. Then Section 3.5 explores the existing hardware and software solutions. Thereafter, we determine the pioneering research groups and important publication venues in Section 3.6. The chapter concludes with critical analysis in Section 3.7, which addresses key challenges in food nutrient estimation, followed by Section 3.8, highlighting existing gaps in the literature. Finally, Section 3.9 synthesizes key insights from the literature review.

3.1 Architectures for Food Nutrient Intake Estimation

Sensor-based technologies have emerged to automate nutrient intake tracking, offering improved accuracy and reduced manual effort in estimating food portion size and nutrient content. Figure 3.1 illustrates the spectrum of sensor-based methods, including strain sensors, acoustic sensors, motion sensors, imaging sensors and weighing sensors. The first method involves strain sensors, which measure chewing activity using piezoelectric devices. Another approach uses acoustic sensors that detect swallowing and chewing sounds through microphones. A third method uses motion sensors to track hand and head movements during meals, relying on accelerometers and gyroscopes. An often used type of sensors is imaging sensors, such as mobile phones and depth cameras. These offer a visual way to capture food portions. Lastly, weighing sensors provide direct measurements of food weight before and after consumption using scales or smart plates. By integrating these diverse technologies, caregivers can collect precise, automated data on food intake [9].

This thesis focuses on imaging sensor-based approaches for two primary reasons: (1) they offer superior accuracy compared to other sensor types [9], and (2) they minimize burden on both residents and caregivers in elderly care

3 Literature Review

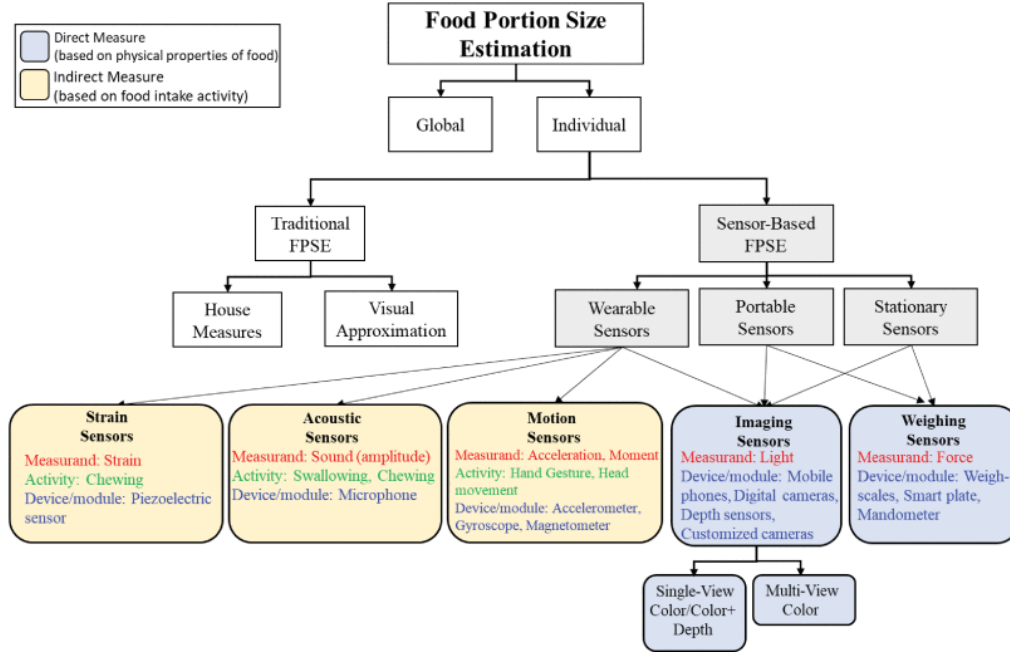


Figure 3.1: Taxonomy of sensors used for food portion size estimation [9].

settings by eliminating the need for physical sensor interaction or wearable devices. These advantages make imaging sensors particularly suitable for monitoring in residential care environments.

Two primary architectures are used for food nutrient intake estimation: multi-stage and end-to-end [34].

3.1.1 Multi-Stage Architectures

The multi-stage architecture, illustrated in Figure 3.2, involves a sequential pipeline approach. The process begins with the segmentation of food images to isolate individual food items. These isolated regions are then classified into predefined food categories, establishing both the location and type of each food component on the plate. Following classification, the system estimates the weight of each food item, potentially using volume estimation as an intermediate step, before calculating nutritional values through reference to a food composition database [34].

3.1.2 End-to-End Architectures

In contrast, the end-to-end architecture streamlines the process by directly inputting image data into a single deep learning network to derive nutrient values. The architecture is visualized in Figure 3.3. Recent advancements in deep learning have enabled the development of these end-to-end solutions, which eliminate the need for multiple components or stages. This approach reduces the complexity of the system and can offer improved accuracy in practical applications as the error does not propagate and accumulate in the pipeline. Additionally, in multi-stage

3 Literature Review

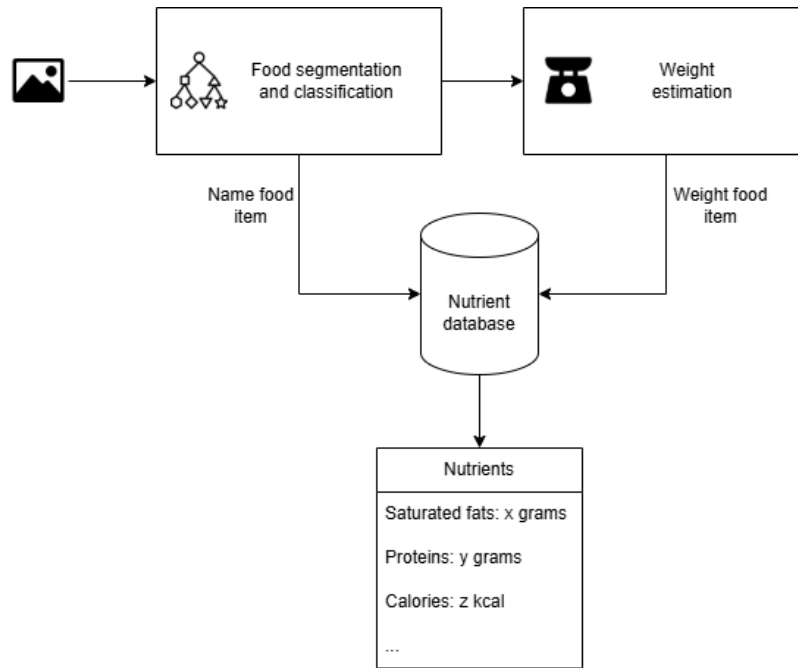


Figure 3.2: Multi-stage architecture for image sensor-based food nutrient intake estimation.

solutions each component has its own input and outputs, meaning that potentially helpful information cannot be passed on to later stages [34]. While this end-to-end approach can offer improved accuracy, a significant difficulty is that a large amount of data with the necessary labels must be available to train the model. This is because end-to-end architectures tackle a more complex task, necessitating more complex models. However, these complex models are more prone to overfitting, and therefore require a larger dataset to ensure proper generalization.

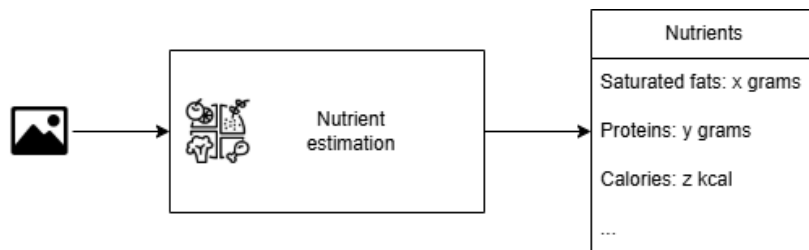


Figure 3.3: End-to-end architecture for image sensor-based food nutrient intake estimation.

3.2 Public Datasets for Food Image Analysis

Before delving into the details of the different architectures used for food nutrient intake estimation and its components, it is essential to first consider the datasets required for training and evaluating these models. High-quality and diverse datasets are crucial for developing reliable models, as they provide the necessary data for segmentation, volume estimation and weight estimation tasks. In this section, we explore the most relevant

3 Literature Review

datasets available in the domain of food image analysis per task. To the best of our knowledge, there do not exist datasets specifically captured within a residential care setting, which often exhibits distinct characteristics. Therefore, we focus our discussion on general food datasets.

3.2.1 Food Segmentation Datasets

Table 3.1 presents an overview of prominent datasets used in food segmentation research, highlighting their key characteristics. These datasets vary in scale, category diversity and annotation type, factors crucial for training and evaluating robust segmentation models.

Specifically, UNIMIB2016 [35] provides a relatively smaller dataset of 1,027 images with instance-level annotations across 73 food categories and includes tray images, which is unique among the datasets listed. FoodSeg103 [36] focuses on Western cuisine, offering 7,118 images with semantic masks for 103 categories. Expanding upon FoodSeg103, FoodSeg154 [36] incorporates Asian cuisine, increasing the category count to 154 and the image count to 9,490.

Segmented UEC Food-100 [37] and UEC-FoodPix Complete [38] are both extensions of the original UEC Food-100 dataset [39], providing enhanced annotations. Segmented UEC Food-100 offers 12,740 images with instance-level segmentation for 100 categories, while UEC-FoodPix Complete provides 9,000 images with semantic masks for the same 100 categories.

Finally, Food Recognition 2022 v2.0 [40] stands out with its large scale, comprising 43,962 images and 498 categories, all with semantic mask annotations, representing the most diverse dataset listed. This dataset allows for the training of models capable of recognizing a wide variety of food items.

Table 3.1: Datasets for food segmentation.

Dataset Name	Segmentation Type	#Images	#Categories
UNIMIB2016 [35]	Instance	1,027	73
FoodSeg103 [36]	Semantic	7,118	103
FoodSeg154 [36]	Semantic	9,490	154
Segmented UEC Food-100 [37]	Instance	12,740	100
UEC-FoodPix Complete [38]	Semantic	9,000	100
Food Recognition 2022 v2.0 [40]	Semantic	43,962	498

3.2.2 Weight, Volume and Nutrient Estimation Datasets

Despite the growing interest in automated nutrient intake tracking, there is a notable absence of widely accepted benchmark datasets for food weight, volume, or nutrient estimation. This lack of standardized resources is likely

3 Literature Review

attributable to the inherent complexity and significant time investment required for collecting and annotating such detailed food data. Consequently, most studies rely on custom datasets tailored to specific methodologies and experimental conditions, which significantly hinders rigorous benchmarking and comparative evaluation across different approaches and systems.

While no universally adopted benchmark dataset exists, several notable datasets have been developed within the research community. These can broadly be categorized by data type:

- **2D image datasets:** Among datasets based on 2D images, Nutrition5k [41] is a prominent example, offering 5,006 images with detailed per-ingredient annotations including mass and macronutrient composition (fat, protein, and carbohydrates). Another relevant dataset is ECUSTFD [42], which comprises 2,978 images annotated with both volume and mass for individual food items.
- **3D model datasets:** More recently, datasets featuring 3D food models have been introduced. NutritionVerse-3D [43] includes a collection of 105 high-resolution 3D food models, each annotated with weight, nutritional information, and food names. Addressing the need for comprehensive nutritional data in 3D resources, MetaFood3D [44], released in September 2024 and revised in December 2024, provides 743 meticulously scanned 3D food objects across 131 categories, offering detailed nutrition information, weight, and food codes. Although promising for 3D food modeling and portion estimation, its recent release means it has yet to gain widespread adoption as a standard resource.

Despite their contributions, existing datasets present limitations for real-world food intake analysis. Specifically, 3D model datasets like NutritionVerse-3D and MetaFood3D typically focus on individual food items rather than plated meals, a common scenario in intake tracking. Furthermore, many existing datasets for weight, volume, and nutrient estimation capture only food before consumption. This often results in images containing larger, uneaten portions, which may represent a simpler estimation task compared to analyzing leftovers or partially consumed meals where portions are smaller and potentially more irregular.

3.3 Key Components of a Multi-Stage Architecture

As previously outlined, the multi-stage architecture for food nutrient intake estimation is a modular system, where the overall process is broken down into several distinct components or stages. The efficacy of this multi-stage approach hinges on the accuracy and robustness of each constituent component. In this section, we delve into the state-of-the-art solutions for each of these components, examining their contributions to the overall system and identifying areas for potential improvement.

3 Literature Review

3.3.1 Food Segmentation Techniques

Accurate segmentation plays a critical role in the precise identification and isolation of distinct food items within an image. This capability is essential for accurately assessing both consumed and leftover food portions, thereby facilitating reliable estimations of food waste and nutrient intake.

Tables 3.2 and 3.3 present a comparative analysis of food segmentation techniques evaluated on the FoodSeg103 and UEC-FOODPIX Complete datasets, respectively. These tables underscore the evolution and efficacy of various methodologies in the domain of food image segmentation.

Table 3.2 clearly demonstrates the dominance of Transformer-based architectures. The Semantic Segmentation Transformer (SETR) [45] integrates FCNs with Transformer models, achieving an mIoU of 45.1 %. The Swin Transformer [46], a hierarchical vision Transformer using shifted windows, attains an mIoU of 41.6 %. ReLeM-SeTR [36] combines LSTM networks with Transformers, yielding an mIoU of 43.9 %. Notably, FOODSAM [47], a framework that uses the Segment Anything Model (SAM) [48], achieves an mIoU of 46.4 %. BEiT v2 [49], another Transformer-based model, reports an mIoU of 49.4 %, while IngedSAM [50], an approach extending SAM’s capabilities for open-world food ingredient segmentation, achieves an mIoU of 48.8 %.

Table 3.2: Comparison of state-of-the-art food segmentation techniques on FoodSeg103.

Method	Year	Technique	mIoU (%)
SETR [45]	2020	FCN + Transformer	45.1
Swin-Transformer [46]	2021	Transformer	41.6
ReLeM-SeTR (LSTM) [36]	2021	LSTM + Transformer	43.9
FOODSAM [47]	2021	Transformer	46.4
BEiT v2 [49]	2023	Transformer	49.4
IngedSAM [50]	2024	Transformer	48.8

Table 3.3 highlights methods evaluated on the UEC-FOODPIX Complete dataset. GourmetNet [51], which integrates ResNet with a Waterfall Atrous Spatial Pooling (WASP) module, achieves an mIoU of 62.9 %. FOODSAM [47] attains an mIoU of 66.1 %. The table also includes two Bayesian methods that quantify model uncertainty: Bayesian-DeepLabv3+ [52], based on ResNet, reports an mIoU of 64.2 %, and BayesianGourmetNet [52] achieves an mIoU of 66.2 %.

In conclusion, our comparative analysis of these state-of-the-art techniques reveals a clear trend toward integrating Transformer-based architectures, indicating a paradigm shift from conventional convolutional approaches to more holistic models capable of addressing complex segmentation challenges. Despite these advancements, performance remains suboptimal, likely due to the challenges discussed in Section 3.7.

3 Literature Review

Table 3.3: Comparison of state-of-the-art food segmentation techniques on UEC-FOODPIX Complete.

Method	Year	Technique	mIoU (%)
GourmetNet [51]	2021	ResNet + WASP	62.9
FOODSAM [47]	2021	Transformer	66.1
BayesianDeepLabv3+ [52]	2022	Bayesian + ResNet	64.2
BayesianGourmetNet [52]	2022	Bayesian + ResNet + WASP	66.2

3.3.2 Food Volume and Weight Estimation Methods

The next step after segmenting the images is determining the amount of food present. Directly estimating weight from images alone is often difficult due to the lack of direct visual cues for mass. Therefore, estimating the food's volume can be used as an intermediate step, as food weight and volume are closely related. The weight W of food can be derived from its volume V using the following formula:

$$W = \rho \times V$$

where ρ is the density of the food item. The FAO/INFOODS Density Database [53] can be used as a reference for obtaining density values of various food items. However, this approach also has limitations, as not all foods are covered by the database. Furthermore, the density of food items can vary significantly based on cooking methods, such as baking, frying, or boiling, which are not always accounted for in standard databases.

In the following paragraphs, we discuss both volume and weight estimation.

Volume Estimation

We adopt the classification from Lo et al. (2020) [10], organizing the food volume estimation techniques into the following categories:

- Stereo-based approach: Uses multiple frames to reconstruct 3D structures of food by finding pixel correspondences and reprojecting them to world coordinates using extrinsic parameters.
- Model-based approach: Relies on pre-built shape templates or mathematical models to estimate the volume of food objects.
- Deep learning approach: Uses deep neural networks to infer depth maps from single RGB images.
- Depth camera-based approach: Leverages sensors for direct 3D measurements of object scale.

3 Literature Review

- Perspective transformation approach: Involves estimating volume from a single image using perspective transformation, which creates a bird's-eye view and provides rough volume estimates.

Table 3.4 summarizes the advantages and disadvantages of these different approaches [10, 11]. This comparison highlights that selecting the appropriate method depends on balancing accuracy requirements against computational resource constraints and implementation practicality in real-world applications.

Table 3.4: Advantages and disadvantages of food volume estimation approaches [10, 11].

Technique	Pros	Cons
Stereo-based	Can reconstruct 3D shape from multiple views Suitable for irregular shapes	Requires multiple calibrated cameras
Model-based	Accurate for items with well-defined, known 3D models	Limited to objects with available models Fails for irregular or deformed foods
Deep learning	Potential to learn complex mappings from 2D appearance to volume Can handle appearance variations	Requires extensive labeled data for training Training and inference can be computationally expensive
Depth camera-based	Directly captures depth and 3D shape information Simpler processing than stereo	Requires dedicated hardware (depth camera)
Perspective transformation	Can estimate area on a planar surface from a single image with a reference object	Cannot estimate 3D volume directly for objects with significant height/irregular shape Requires a reference object and proper calibration

Table 3.5 presents a comparison of various state-of-the-art techniques for estimating food volume. The table includes key details such as the authors, year of publication, method used, number of food categories analyzed, and the MAPE metric.

A thorough examination of the techniques presented in Table 3.5 reveals several key insights. First, no single technique has emerged as universally superior in food volume estimation. In recent years, there has been a shift

3 Literature Review

towards deep learning models, which demonstrate significant potential due to their reliance on 2D images while still maintaining competitive performance. However, a notable limitation in many of these models is the small number of food categories on which they are trained and tested. This restricts their generalizability and practical application in diverse real-world environments. Expanding the range of food categories remains a critical challenge for future research. Additionally, it is difficult to make a direct comparison between these methods, as the datasets used in the various studies present different challenges, such as varying lighting conditions, occlusions, and other environmental factors. This highlights the need for standardized datasets.

Table 3.5: Comparison of approaches for food volume estimation.

Authors	Year	Technique	#Categories	MAPE (%)
Dehais et al. [54]	2016	Stereo-based	45	8.2
EAH Akpa et al. [55]	2017	Perspective transformation	15	6.8
Gao et al. [56]	2018	Stereo-based	3	18.6
Lo et al. [57]	2018	Deep learning	8	6.9
Lo et al. [58]	2019	Deep learning	9	4.6
Vinod et al. [59]	2024	Model-based	12	14.0

Weight Estimation

In this section, we dive into methods that directly estimate food weight without calculating volume as an intermediate step. There are two primary approaches to this task.

The first approach, the pixel-area mapping method, establishes a direct relationship between the pixel area occupied by the food component in the image and its corresponding weight. This method uses a simple linear mapping function that converts pixel counts to weight estimations. The underlying assumption is that the area covered by food in an image is proportional to its mass.

The second approach uses feature-based regression models. These models are trained on datasets that include image features and corresponding weight labels. By learning the complex relationships between image characteristics and weight, these models can predict the weight of unseen food items. This approach allows for more nuanced estimations, taking into account factors beyond the simple pixel count.

Table 3.6 provides a concise comparison of various techniques for estimating food weight directly, as applied to segmented food items within multi-stage pipelines. The results presented in the table reflect the accuracy of the entire multi-stage pipeline, rather than solely the weight estimation component. Analysis of the table reveals a notable performance difference between pixel-area mapping methods and feature-based regression models. While pixel-area mapping methods, such as those by Ciocca et al. (2015) [60] and Sari et al. (2021) [61], report a

3 Literature Review

MAPE of 15.0 %, the regression model developed by Konstantakopoulos et al. (2023) [62] achieves a significantly lower MAPE of 3.7 %. This comparison suggests that feature-based regression models hold significant potential for improving overall pipeline accuracy in weight estimation. When comparing these results with volume-based estimation techniques, performance appears similar, implying that an intermediate volume estimation step might not be essential for achieving better overall weight estimation results. Crucially, interpreting these comparisons must be done with caution, as the models were trained and evaluated on different datasets, which limits direct comparability.

Table 3.6: Comparison of approaches for direct food weight estimation.

Authors	Year	Technique	# Categories	MAPE (%)
Ciocca et al. [60]	2015	Pixel based	15	15.0
Sari et al. [61]	2021	Pixel based	33	15.0
Konstantakopoulos et al. [62]	2023	Feature based	226	3.7

3.3.3 Nutrient Calculation from Food Consumption Data

With the food type and weight successfully identified and estimated, nutritional databases linking food classifications to predefined nutrient values are queried. This process allows the system to determine the meal's nutritional composition, such as macronutrients, vitamins, and minerals.

The USDA National Nutrient Database (NNDB) is frequently used for the conversion of food types and their respective weights into a detailed list of macronutrients and micronutrients. This database provides nutrient information for over 7,900 food items across 146 components, including vitamins, minerals, amino acids and fatty acids [63]. However, a notable limitation of the NNDB is its emphasis on raw foods rather than prepared meals. For instance, while it provides the caloric content of raw ingredients such as 500 grams of beef, it does not account for the caloric changes that occur during cooking, such as in a prepared burger. For more accurate analysis of cooked meals, the USDA Food and Nutrient Database for Dietary Studies (FNDDS), which is derived from NNDB, is often preferred [64]. Nonetheless, the calorie content of cooked foods can vary significantly based on the method of preparation (e.g. grilling versus frying). This variability presents a challenge in developing highly precise systems for estimating nutrient intake.

3.4 End-to-End Architectures

After the discussion of the multi-stage architecture, the focus shifts to the end-to-end architecture, which has gained prominence with the advancement of deep learning. In contrast to the multi-stage approach, where the

3 Literature Review

task is divided into several stages, the end-to-end architecture applies food images directly to a single neural network for nutrient estimation [34].

Table 3.7, an extended and adapted version of Wang et al. (2022) [34], provides an overview of the most important end-to-end approaches for food nutrient estimation or related tasks. A key trend is the increasing use of Multi-Task Learning (MTL), which allows simultaneous handling of multiple related tasks, such as food recognition, segmentation and volume estimation. This approach enhances generalization by leveraging shared representations across tasks, which is particularly useful in dietary assessment where tasks are interconnected. Furthermore, most of these methods are based on CNNs, which serve as the backbone for extracting meaningful features from food images.

Table 3.7: State-of-the-art end-to-end approaches for food nutrient estimation or related tasks.

Authors	Year	Target	Technique	MAPE (%)
Lu et al. [65]	2018	Volume	MTL-based CNN	19.1
Fang et al. [66]	2018	Energy	GAN	10.9
Ege & Yanai [67]	2018	Calorie	MTL-based CNN	36.1
Ruede et al. [68]	2021	Calorie	MTL-based CNN	9.9
Thames et al. [41]	2021	Calorie and nutrient	MTL-based CNN	14.7
Shao et al. [69]	2023	Calorie and nutrient	RGB-D fusion	18.5
Han et al. [70]	2023	Calorie, mass and nutrient	RGB-D fusion	17.8

End-to-end methods have shown results similar to multi-stage architectures. However, a direct comparison between the two is difficult, as many studies tend to select easier cases or images for evaluation.

3.5 Existing Hardware and Software Solutions

The rapid advancement of technology has enabled the development of various hardware solutions and applications aimed at estimating food weight and determining its nutrients, but also on estimating the food waste. This section provides an overview of the existing hardware solutions and applications that allow users to estimate their food weight, nutrient intake or food waste. By exploring these existing tools, we can gain valuable insights into their functionalities, strengths and limitations.

3 Literature Review

3.5.1 Hardware Solutions

In this section, we will discuss both open-source and commercial hardware solutions for food waste management. While hardware solutions can effectively address the tracking and measurement of food waste, there are currently no hardware solutions designed to measure nutrient intake.

Open-source Hardware Solutions

While some food waste management systems are available for applications in urban environments, such as waste bins in cities or offices, there is a noticeable gap in the availability of open-source solutions specifically designed for the restaurant setting, particularly for commercial or institutional kitchens.

One notable solution designed for a restaurant setting is the Smart eNose Food Waste Management System. This system utilizes an Arduino UNO microcontroller to manage data from MQ4 and MQ135 sensors, which are capable of detecting gases such as methane (CH₄), carbon dioxide (CO₂), and ammonia (NH₃) emitted by different food items. It is also equipped with a 5 kg strain gauge load cell sensor to measure the weight of food waste. A decision tree algorithm is used to predict the type of food based on the detected gas emissions. The system has demonstrated 92.65 % accuracy in predicting food types and measuring food waste, offering valuable insights that can aid in reducing food waste in both homes and restaurants [71].

Commercial Hardware Solutions

While open-source hardware solutions for food waste management are limited, several commercial systems are designed to measure food waste, providing valuable insights into disposal practices. These systems typically combine cameras and scales, using Artificial Intelligence (AI) to automatically detect the type of food. Their primary aim is to help businesses reduce food waste through data-driven insights.

An overview of the most notable commercial systems for food waste measurement is presented in Table 3.8. A common characteristic across all these commercial systems is that they provide dashboards with visualizations of waste data. However, they share significant limitations: none focus on measuring nutrient intake at the individual level (a significant gap for understanding personal food consumption), and there is a lack of transparency regarding measurement accuracy, raising questions about data reliability.

3.5.2 Software Solutions

In the domain of food nutrient intake estimation, a number of applications have been developed to assist users in tracking their diet and monitoring their nutritional intake. We will discuss both the mobile applications and the desktop applications.

3 Literature Review

Table 3.8: Overview of commercial food waste management systems.

System	Setup
Kitro [72]	Camera, scale, lighting, and AI; detects waste after disposal
Winnow [73]	Camera, scale, LED display, lighting, and AI; detects waste after disposal
Orbisk [74]	Camera, scale, LED display, lighting, and AI; detects waste before disposal
Positive Carbon [75]	Sensor-based monitoring system
Leanpath [76]	Camera, scale, LED display, and AI
Lumitics Insight [77]	Camera, scale

Mobile Applications

There exists an abundant number of food and nutrient tracking applications available today. Notable examples include YAZIO, Lifesum, Cronometer, among others. These applications are designed to help users log their food consumption and monitor the nutritional content of their diets. They generally achieve this by allowing users to record their meals, estimate portion sizes, and subsequently analyze the nutritional breakdown of their food.

Nutrient intake apps use various data collection methods: manual input from databases, camera-based image analysis, or a combination of both. Manual input is detailed but time-consuming, while camera-based methods offer convenience but vary in accuracy. Hybrid approaches aim to balance these factors.

Table 3.9 provides an overview of a selection of nutrient intake applications, directly found via the Apple App Store using the keywords "food tracker" and "nutrient tracker" and outlines the input method used by each application. It is important to note that this table represents a limited subset of available applications, and a more comprehensive search would reveal a much larger number of similar tools.

Table 3.9: Overview of nutrient intake applications.

Application	Input Method
MyFitnessPal: Calorie tracker	Manual or Camera
YAZIO	Manual or Camera
Lifesum	Manual or Camera
Fantastic AI Food Calorie Scanner	Hybrid
Calorie Counter - EasyFit	Manual or Camera
Calorie Counter: Cronometer	Hybrid
Nutrition Tracker: Foodnoms	Manual or Camera
Calorie Counter - MyNetDiary	Manual or Camera

3 Literature Review

Despite their usefulness for the general population, these tools face limitations when applied in specialized settings. In environments such as residential care centers for elderly individuals, the reliance on manual input can be a significant drawback. Many elderly residents may face challenges such as cognitive impairments or unfamiliarity with technology, making the process of logging food intake difficult.

Moreover, none of the current applications are designed to integrate seamlessly with the structured, communal eating environments found in care facilities, where food is often served in standardized portions. In addition, these apps focus only on tracking food intake. For communal eating environments, it is also interesting to track global food waste. In this way, portion sizes and meals can be better tailored to patients.

Non-mobile Applications

While mobile applications are widely used for tracking food nutrient intake, many of these applications also have non-mobile versions available, such as desktop or browser-based alternatives. Examples include MyFitnessPal: Calorie Tracker, Calorie Counter: Cronometer, Lifesum, and MyNetDiary.

Furthermore, the hardware solutions discussed in Section 3.5.1 incorporate a software component, typically in the form of a dashboard. This software enables the visualization of insights, facilitating data interpretation and user interaction with the system.

3.6 Research Landscape in Food Image Analysis

This section provides an overview of the major research institutions and publication venues that have shaped the field of computational food image analysis. Understanding the key players in this domain offers valuable context for the current state of research and highlights the centers where significant advancements have originated.

3.6.1 Leading Research Groups and Universities

Several universities and research groups have emerged as leaders in advancing food image analysis techniques. Based on our analysis of the literature, the following institutions stand out for their significant contributions.

University of Electro-Communications Tokyo

The University of Electro-Communications Tokyo, particularly through researchers like professor Yanai K. and colleagues, has made substantial contributions to food image segmentation and recognition. Their development of the UEC-FoodPix Complete dataset [38] represents a major contribution to the field. Their work spans over a decade, with a focus on food image recognition [39, 67].

3 Literature Review

University of Milano-Bicocca

The Imaging and Vision research group at the University of Milano-Bicocca, with researchers like Ciocca G. and colleagues, has made important contributions to food recognition systems and datasets. Their work on leftover estimation for diet monitoring [60] and their development of a new food recognition dataset [35] have advanced practical applications in dietary assessment.

Imperial College London

Imperial College London's research group, led by Lo F. and Sun Y., has pioneered several key advancements in food image analysis, particularly focusing on food volume and nutritional intake estimation. Their work on food volume estimation for dietary assessment [57] and food portion size estimation using wearable cameras [56] has been widely recognized for its practical applications in automated dietary monitoring.

Purdue University

Purdue University's Computer Vision team has pioneered advanced methods for food portion estimation. Their research includes innovative approaches such as 3D object scaling [59] and image-to-energy mappings using GANs [66]. Their recent collaboration with the University of Waterloo on the MetaFood3D dataset [44] represents a significant advancement in 3D food modeling with nutritional information.

University of Waterloo

The Computer Vision group at the University of Waterloo has emerged as a leader in 3D food modeling and nutritional estimation. Their development of NutritionVerse-3D [43] and their collaboration on MetaFood3D [44] demonstrate their expertise in creating sophisticated datasets for nutritional intake estimation. Their work on knowledge transfer for food image segmentation using transformers [49] showcases their innovative approaches.

3.6.2 Primary Publication Venues

Research in food image analysis is primarily published in conferences and journals that focus on computer vision, multimedia processing, and health informatics. These venues provide a platform for presenting state-of-the-art advancements, fostering academic collaboration, and shaping the future directions of the field. It's worth noting that the interdisciplinary nature of this research means contributions are distributed across numerous specialized and general-purpose venues, with no single dominant outlet.

3 Literature Review

Conferences

The following conferences serve as key outlets for cutting-edge research in food image analysis:

- IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): A leading conference in computer vision, covering a wide range of topics including image analysis and pattern recognition.
- IEEE International Conference on Multimedia and Expo (ICME): Focuses on multimedia technologies and applications, encompassing areas like image processing and multimedia content analysis.
- International Conference on Pattern Recognition (ICPR): Dedicated to all aspects of pattern recognition, including image analysis, computer vision, and machine learning.

Journals

The following journals are prominent publication channels for food image analysis research:

- IEEE Transactions on Multimedia: Publishes research on multimedia technologies, including content analysis, processing, and retrieval.
- IEEE Journal of Biomedical and Health Informatics: Covers developments in biomedical and health informatics, with topics such as medical image analysis and health-related data processing.
- Nutrients: An open-access journal focusing on human nutrition, including studies on dietary assessment and nutritional analysis methodologies.

3.7 Challenges in Image-Based Food Nutrient Estimation for Residential Care Settings

Accurately estimating food nutrients from images presents several significant technical challenges that impact system performance. When developing a food nutrient estimation system for residential care centers, the following challenges must be considered:

- **Food Appearance Variability:** Food items exhibit high intra-class variability but low inter-class variability. The same dish may appear completely different depending on preparation methods, while different foods can look similar.
- **Mixed and Occluded Foods:** Real-world meals often contain overlapping or mixed food items, creating significant segmentation challenges. As observed in the segmentation techniques reviewed, even state-of-the-art models achieve relatively modest mIoU scores, highlighting the difficulty of this task.

3 Literature Review

- **Portion Size Estimation:** Small leftover portions are particularly difficult to assess, as minor absolute errors become proportionally larger when measuring small quantities. In addition, they have less visual cues.
- **Implementation in Elderly Care:** Residents often require blended foods due to swallowing difficulties, making visual identification more challenging. The original texture and form of ingredients are fundamentally altered by blending. This removes the distinct visual cues deep learning models rely on for recognition.

Addressing these challenges requires improved methods, better datasets, and understanding of specific constraints in elderly care environments to develop systems providing meaningful dietary insights.

3.8 Gaps in Existing Systems

Our comprehensive review has identified several gaps in current food nutrient estimation systems that represent opportunities for future research.

3.8.1 Limited Standardized Datasets

A significant challenge in advancing weight and volume estimation for food intake analysis, particularly in specialized care settings, is the scarcity of dedicated datasets. Currently, most research in this area relies on custom-collected datasets that are often small-scale and lack the diversity needed for robust algorithm development and evaluation. This reliance on non-standardized collections creates several limitations. Firstly, it makes direct comparison between different proposed algorithms difficult, as each is typically trained and tested on unique data with varying characteristics. Additionally, the lack of publicly available, well-annotated datasets hinders reproducibility and slows down progress in the field. Researchers often spend considerable effort on data collection and annotation, diverting resources from algorithm development and refinement.

3.8.2 Lack of Food Analysis in Residential Care Settings

Current systems predominantly focus on conventional food presentations and fail to address the unique challenges of food analysis in residential care settings. In these environments, food is frequently served in modified forms (pureed or blended) to accommodate residents with swallowing difficulties. These modified textures significantly complicate the segmentation and classification tasks, as the visual characteristics that algorithms typically rely on are often absent.

3 Literature Review

3.8.3 Lack of End-to-End Consumption Tracking

Current approaches typically focus either on initial food portions, but rarely take into account the leftovers. A comprehensive understanding of nutritional intake requires analyzing both the initial serving and the remaining food after the meal. This gap is especially problematic in healthcare settings, where precise monitoring of patients' nutritional intake is vital for proper care and intervention. Systems that can effectively track the entire consumption cycle—from initial serving to leftover analysis—would provide more accurate assessments of actual nutrient intake and could significantly improve nutritional monitoring in clinical environments.

3.8.4 Limited Focus on Small Portion and Leftover Estimation

Achieving true end-to-end tracking necessitates the ability to accurately assess not only the initial large portions but also the smaller portions and leftovers remaining after a meal. Although, most research has focused on estimating nutrients in standard meal portions rather than small leftovers. The precision required to accurately assess the nutrient content of small food remnants presents unique challenges that remain largely unaddressed.

3.8.5 Lack of Transparency in Commercial Solutions

As discussed in Section 3.5, commercial solutions for food waste management lack transparency regarding their measurement accuracy and methodology. Without this information, it is difficult to assess their reliability for critical applications.

3.9 Summary of the Literature Review

This literature review has examined the state of the art in food nutrient intake estimation using image-based methods, with particular attention to applications in residential care settings. By analyzing architectural approaches, dataset characteristics, technical performance, and implementation challenges, we have identified both significant advances and critical limitations in current systems.

The field has seen significant progress in both multi-stage and end-to-end architectures for food analysis. Multi-stage approaches separate the tasks of segmentation, classification, and weight estimation, while end-to-end models directly predict nutritional content from images. Consequently, each approach comes with its own set of advantages and disadvantages.

Food segmentation has progressed significantly with models like BEiT v2 and IncredSAM achieving mIoU scores above 45% on benchmark datasets. However, these scores indicate that accurate food segmentation remains challenging, especially with mixed or modified foods common in care settings.

3 Literature Review

Weight and volume estimation methods show promising results but still exhibit error rates between 4-15%, with limited standardized datasets hindering comparative evaluation.

Several critical gaps in current research were identified:

- **Data Scarcity:** Lack of standardized datasets for food weight and volume analysis.
- **Setting-Specific Limitations:** No solutions tailored to the unique demands of residential care settings.
- **Incomplete Tracking:** Absence of end-to-end consumption tracking systems, taking into account the actual consumed food.
- **Precision Gaps:** Limited focus on accurate estimation of small portions and leftovers.
- **Transparency Issues:** Opacity in the measurement accuracy of commercial solutions.

To address these critical gaps, our research introduces several novel contributions to the field:

- Development of a nutrient tracking system optimized for residential care settings.
- Creation of an end-to-end consumption tracking system integrating served food and leftover assessment.
- Development of algorithms which work on small portions and leftovers.

4

Data Collection and Dataset Preparation

This chapter examines the dataset underpinning this thesis, a critical component of deep learning solutions where data quality and quantity directly impact model performance. The absence of publicly available datasets specifically focusing on food images from residential care centers with weight annotations necessitated the creation of a custom dataset. We begin with an exposition of the data collection methodology, followed by a detailed characterization of the dataset structure encompassing both tabular and image data. Exploratory data analysis provides insights into dataset characteristics and identifies potential challenges. Finally, we present the data splitting strategy adopted for model training, validation, and testing, and conclude with proposed improvements for future data collection efforts.

Note: Throughout this chapter, residential care centers are referred to by numbers rather than by name to maintain privacy.

4.1 Data Collection Methodology

The custom dataset was created by the Flanders Research Institute for Agriculture, Fisheries and Food (ILVO). The data acquisition system comprised a mobile cart equipped with dual imaging capabilities (2D RGB and 3D cameras), a precision scale, and a touch screen interface for data entry (resident information, food category, food weight, etc.), as illustrated in Figure 4.1. Data collection was performed collaboratively by ILVO personnel and healthcare staff from participating residential care centers.

The resulting dataset consists of two components:

- **Tabular data:** Stored in a PostgreSQL database, containing information about care centers, residents, meals, and weight annotations.
- **Image data:** Comprising 2D and 3D captures of meals before and after consumption.

4 Data Collection and Dataset Preparation

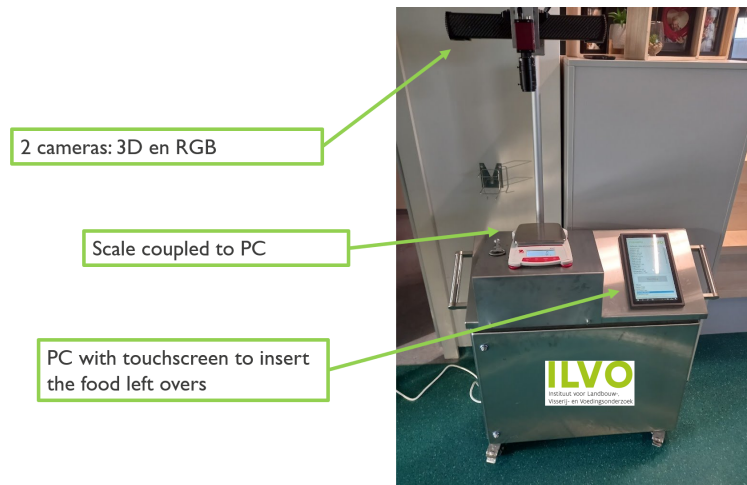


Figure 4.1: Mobile data acquisition system used for dataset capture.

To ensure accurate data capturing, a systematic data capture protocol was implemented. For each meal, images were captured at two critical phases:

1. **Serving phase:** The process began with an empty plate; components were added one by one, and data was captured after each item was placed on the plate until the full meal was assembled.
2. **Clearing phase:** Starting with the plate showing leftovers, the process involved removing remaining components one by one, with data captured after each item was taken off, until the plate was empty.

This bidirectional capture approach is illustrated in Figure 4.2.



(a) Sequential image capture during the progressive food serving process.



(b) Sequential image capture during the progressive plate clearing process.

Figure 4.2: Bidirectional image capture process during food serving (a) and plate clearing (b).

4 Data Collection and Dataset Preparation

4.2 Dataset Composition and Structure

4.2.1 Tabular Data Architecture

The tabular database encompasses data from residential care centers and their departments located across Flanders, Belgium. Figure 4.3 presents the Entity Relationship Diagram (ERD) of the database schema, revealing the complex relationships between various entities within the dataset.

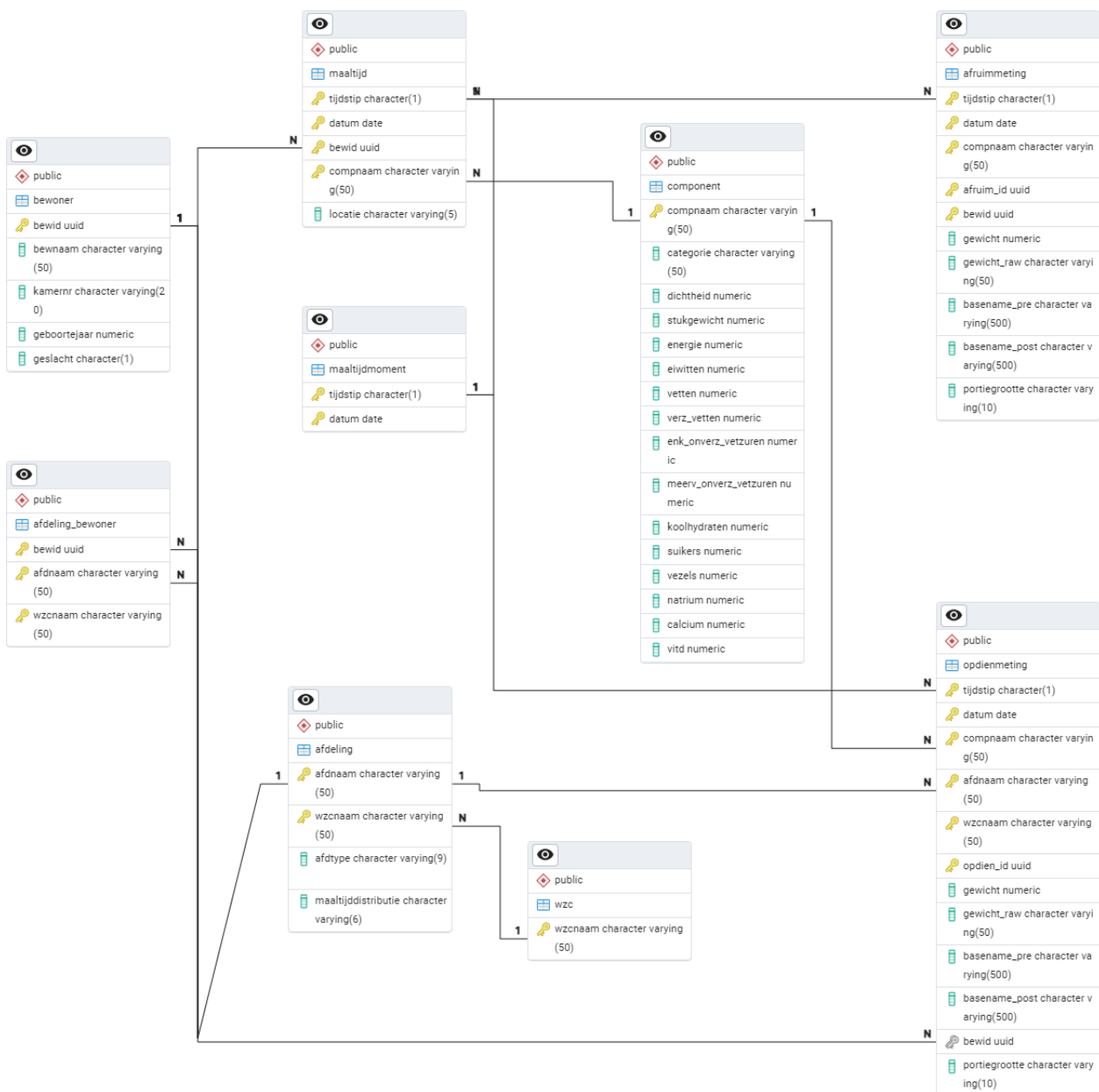


Figure 4.3: ERD of the dataset database schema.

4 Data Collection and Dataset Preparation

The key entities in the database architecture include:

- **Residential care center (*wzc*)**: Contains the identifying information for each participating care facility.
- **Department (*afdeling*)**: Represents subdivisions within each residential care center.
- **Meal (*maaltijd*)**: Central entity that stores comprehensive information about meals provided to residents, linked to the resident (*bewoner*) table via a foreign key.
- **Resident (*bewoner*)**: Contains anonymized resident demographic information.
- **Dish component (*component*)**: Captures the detailed nutritional composition of each food component, enabling nutrient intake calculations.
- **Serving measurement (*opdienmeting*)**: Contains standardized portion information applicable across all residents.
- **Clearing measurement (*afruimmeting*)**: Records individualized post-consumption measurements.

4.2.2 Multimodal Image Data

The image data, which constitutes the principal input for the computer vision system proposed in this thesis, encompasses multiple representational modalities for each plate capture. Each acquisition includes:

- **PLY File**: 3D geometric information storing plate and food surfaces as interpolated point clouds.
- **Interpolated Points Visualization**: PNG image visualizing the interpolated points.
- **Depth Map**: Pixel-wise depth information enabling volumetric analysis.
- **Normal Maps**: Three orthogonal maps (X, Y, Z axes) capturing surface orientation.
- **Point Cloud Maps**: Three point cloud maps representing different coordinate systems.
- **Texture Maps**: High-resolution surface texture details in PNG and TIFF formats.
- **RGB Image**: Standard 2D color image.

4.2.3 Annotations

The images were systematically annotated with semantic segmentation masks for six potential classes: meat, starch, vegetables, sauce, other sauce, and dessert. These polygon-based annotations were created by the ILVO team and stored in plain text format. While the semantic segmentation classes operate at food category level rather than individual food-item level, this is not a limitation, as the system can exploit menu data to link these

4 Data Collection and Dataset Preparation

broad categories to specific menu items, enabling precise identification and analysis of individual food components. Weight data were recorded during the serving and clearing process using a precision scale and subsequently stored in a PostgreSQL database.

4.3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) provides insights into dataset characteristics and potential limitations, informing our approach to building robust segmentation and weight estimation models. Our focus will be on the image data, as these will serve as the primary input for our system.

4.3.1 Initial Dataset Characteristics

The original dataset consists of images captured across eight residential care centers comprising fourteen departments. Data collection spans four meal moments (breakfast, lunch, soup, and dessert). All images have a resolution of 2048×2048 pixels. The statistics of the unfiltered dataset are visualized in Table 4.1.

Table 4.1: Initial dataset characteristics.

Characteristic	Value
Residential care centers	8
Departments	14
Food component classes	6
Meal moments	4
Images	3,947
Segmentation annotations	1,833
Weight annotations	1,294
Image resolution	2048×2048

4.3.2 Distributional Analysis

Understanding the distributional properties across various dimensions reveals potential biases and informs appropriate training and evaluation strategies. In this analysis we only take into account the images with weight labels.

4 Data Collection and Dataset Preparation

Meal Moment Distribution

The dataset encompasses four distinct meal occasions. As evident in Figure 4.4, lunch images substantially outnumber other meal moments. This pronounced imbalance led to our strategic decision to focus model training and evaluation exclusively on lunch images to prevent potential confusion from other meal moments. From now on, we only take into account the images of the lunch moment.

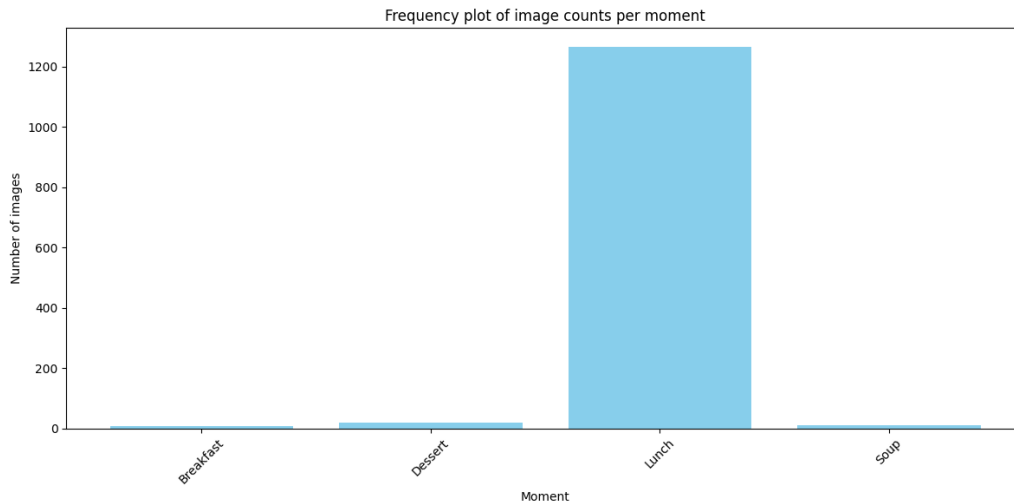


Figure 4.4: Frequency distribution of image counts per meal moment.

Institutional Distribution

The distribution of images across residential care centers demonstrates significant imbalance, as shown in Figure 4.5, with care center 5 and care center 2 contributing disproportionately more samples than other centers. This institutional skew necessitates careful consideration in data partitioning and when interpreting model generalization performance across different residential care contexts.

Food Category Distribution

The dataset contains six food categories, with dessert items excluded from our analysis to maintain focus on the primary lunch components. Figure 4.6 illustrates the frequency distribution across the remaining five categories.

Starch and meat/fish components appear most frequently, followed by vegetables, while sauce categories are significantly underrepresented. This class imbalance has implications for model training and performance expectations, particularly for the underrepresented sauce classes.

4 Data Collection and Dataset Preparation

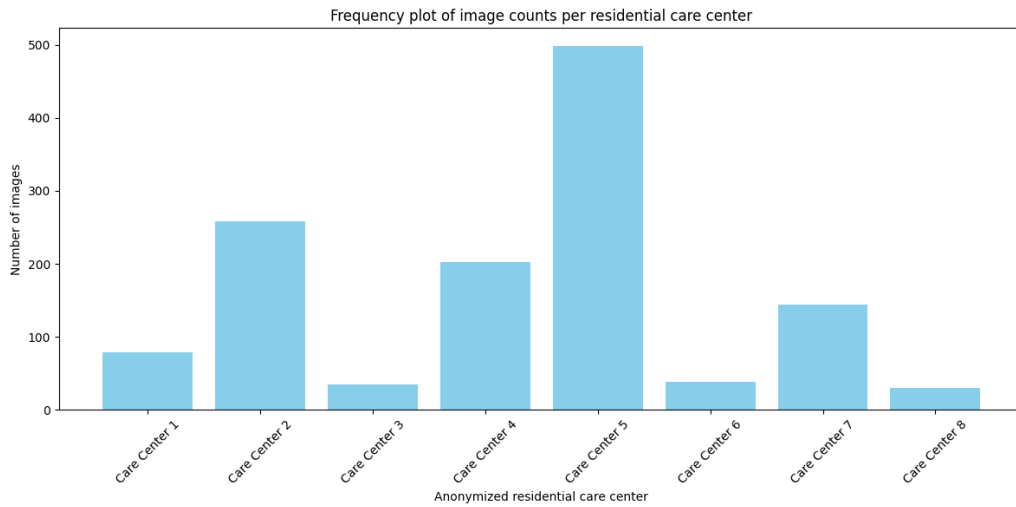


Figure 4.5: Frequency distribution of image counts per residential care center.

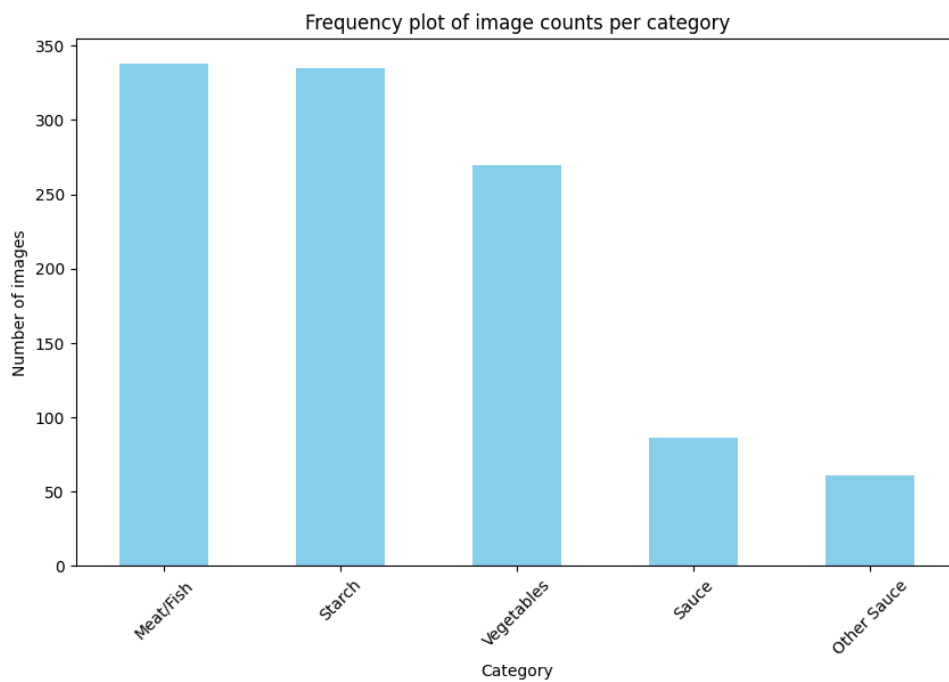


Figure 4.6: Frequency distribution of image counts per food category.

Meal Phase Distribution

Figure 4.7 illustrates the distribution of images across the two capture phases: serving (before consumption) and clearing (after consumption).

The higher proportion of clearing phase images is methodologically sound, as a single serving image typically corresponds to multiple clearing images representing various consumption patterns. Clearing phase images

4 Data Collection and Dataset Preparation

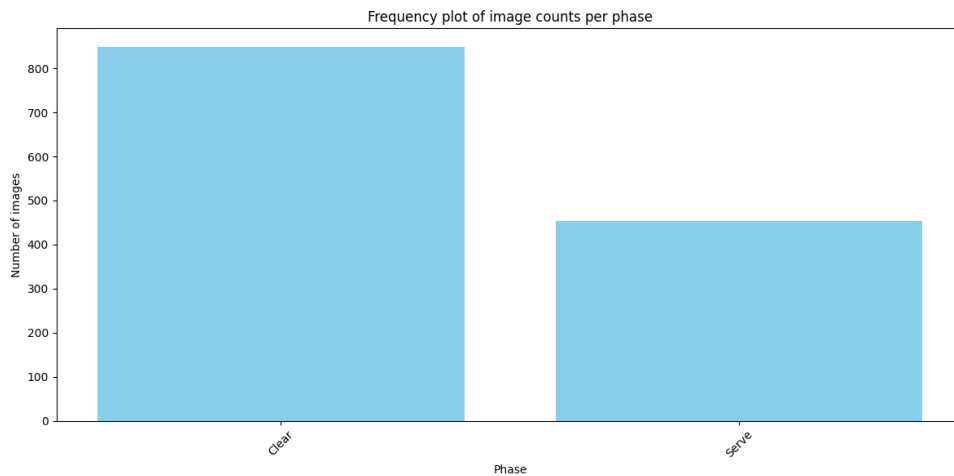


Figure 4.7: Frequency distribution of image counts per meal phase.

present inherently more complex analytical challenges due to food mixing, partial consumption, and altered visual appearances—precisely the scenarios where more images can be useful to learn the features.

Weight Distribution Analysis

Component weight distributions demonstrate significant inter-institutional variation, as visualized in Figure 4.8. Although median weights cluster around 75 g, the density shapes, spreads, and presence of outliers vary notably across centers. These differences may reduce generalizability.

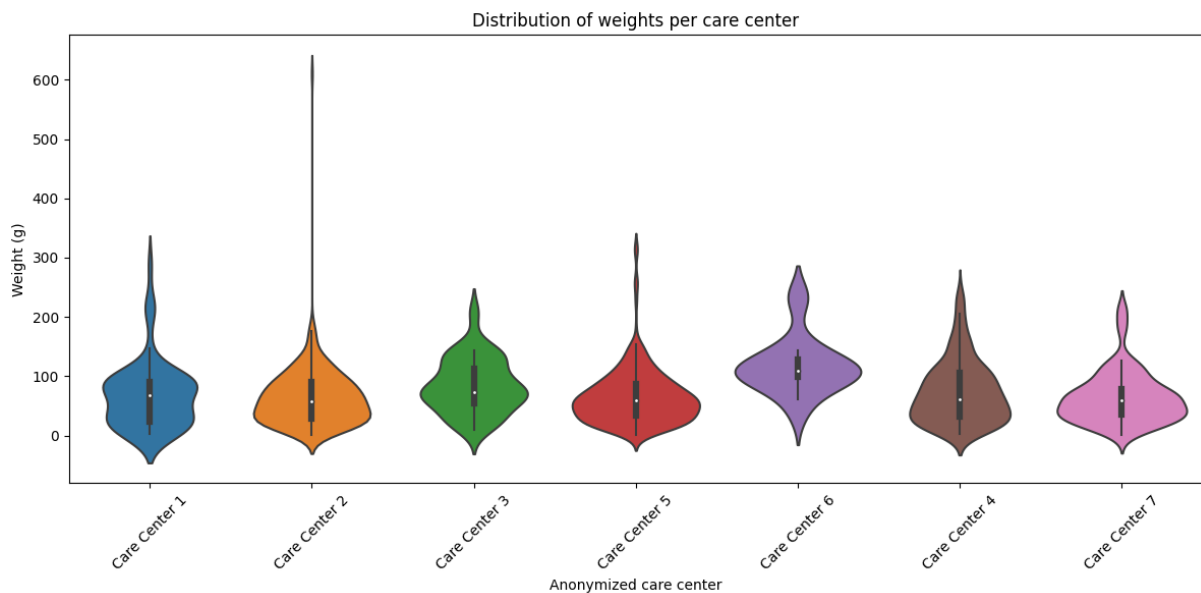


Figure 4.8: Weight distribution per residential care center (violin plot).

4 Data Collection and Dataset Preparation

This variability stems from institutional differences in portion sizing policies, the varying ratio of served to cleared plate images, and demographic factors. Elderly residents and those with dementia typically consume smaller portions than younger individuals, further contributing to the weight distribution differences across facilities. Key statistical measures of the weight distributions are summarized in Table 4.2.

Table 4.2: Statistical summary of weight measurements per residential care center.

Residential Care Center	Median (g)	Mean (g)	Std Dev (g)	Min (g)	Max (g)
Care Center 1	87.5	85.0	61.4	3.0	288.0
Care Center 2	71.0	71.8	45.1	1.0	610.0
Care Center 3	73.0	84.8	45.3	10.0	207.0
Care Center 4	76.0	85.0	54.1	3.0	244.0
Care Center 5	66.0	74.3	48.7	2.0	314.0
Care Center 6	110.0	122.4	46.8	38.0	233.0
Care Center 7	62.0	69.6	45.2	1.0	214.0

4.3.3 Image Quality Assessment

Image quality significantly impacts the performance of computer vision models. Therefore, we conducted a systematic analysis and filtering of the images. This evaluation revealed several issues requiring attention:

- **Red overlay artifacts:** Some images contain a red overlay. These images were removed from the dataset as they would negatively impact model performance and the model should not be able to deal with these low quality images. Some of the images are visualized in Figure 4.9 as example.
- **Partial plate visibility:** In some of the images, the plate is not entirely visible within the frame. These partial views challenge our circular Region of Interest (ROI) detection algorithm, which relies on complete plate boundaries. Some of the images are visualized in Figure 4.10.
- **Lighting inconsistencies:** For each residential care center we have visualized 5 random images in Figure 4.11. We can see that the quality and lighting conditions of the images captured in different residential care centers differs significantly. These differences create domain shifts between centers, potentially limiting the transferability of learned features across environments.
- **Over- and underexposure:** Some images are over- or underexposed, which means that some (parts of) components are invisible. This can affect the performance of the segmentation model and therefore the entire system. Some examples are visualized in Figure 4.12.
- **Duplicate images:** Duplicates were removed to prevent overfitting.

4 Data Collection and Dataset Preparation

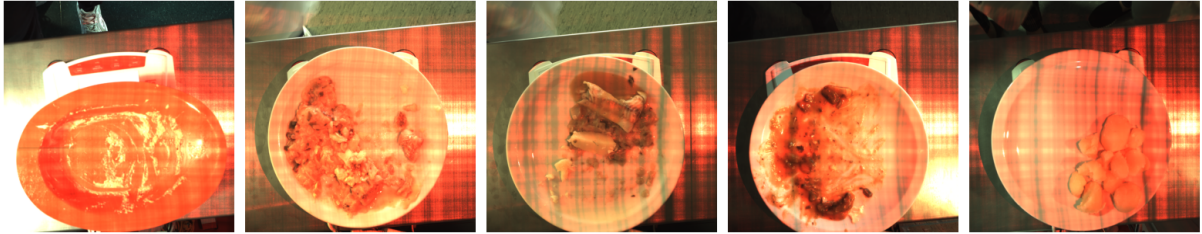


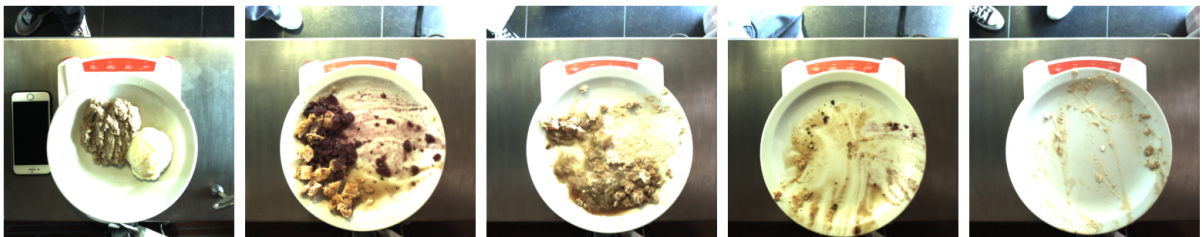
Figure 4.9: Examples of images with red overlay artifacts.



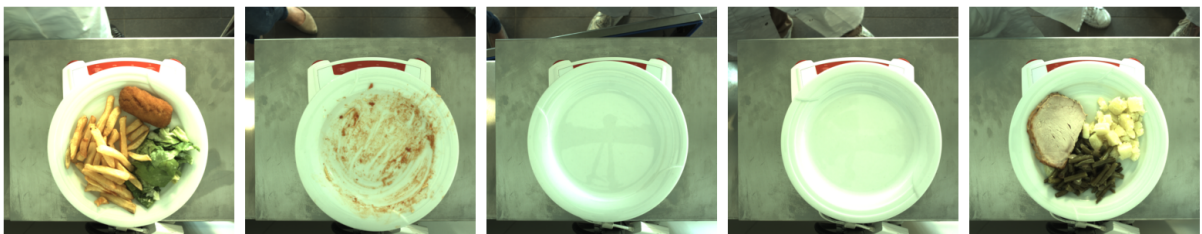
Figure 4.10: Examples of incomplete plate captures.



(a) Images captured in care center 1.



(b) Images captured in care center 2.



(c) Images captured in care center 3.

Figure 4.11: Images showing illumination and imaging variability across residential care centers (continued on next page).

4 Data Collection and Dataset Preparation



(d) Images captured in care center 4.



(e) Images captured in care center 5.



(f) Images captured in care center 6.



(g) Images captured in care center 7.

Figure 4.11 (continued): Images showing illumination and imaging variability across residential care centers.

4.3.4 Characteristics of the Dataset After Quality Filtering

Following the systematic image quality assessment and filtering process discussed in the previous sections, we now detail the characteristics of the resulting dataset used in this thesis.

4 Data Collection and Dataset Preparation



(a) Examples of overexposure affecting component visibility.



(b) Examples of underexposure affecting component visibility.

Figure 4.12: Examples of images with over- and underexposure issues.

Table 4.3 presents detailed statistics per care center after the quality filtering. "Segmented images" refer to images where only one component is visible and other pixels are masked, providing specialized training data for the regression model. "Unique food components" refer to food items served exclusively in one care center. Data for residential care centers 6 and 8 are not included: the images of care center 6 were excluded due to a red overlay, while the images of care center 8 lacked any segmentation annotations.

Table 4.3: Overview of the dataset composition after quality filtering.

Residential care Center	#Images	#Segmented Images	#Food Components	#Unique Food Components
Care Center 1	67	71	11	3
Care Center 2	230	342	42	25
Care Center 3	39	51	9	0
Care Center 4	268	234	29	11
Care Center 5	398	473	33	12
Care Center 7	175	203	32	17
Total	1,177	1,374	156	68

4.3.5 Analysis of Segmentation Annotations

We evaluated segmentation annotation quality through systematic random sampling and visual inspection, with representative examples shown in Figure 4.13.

4 Data Collection and Dataset Preparation

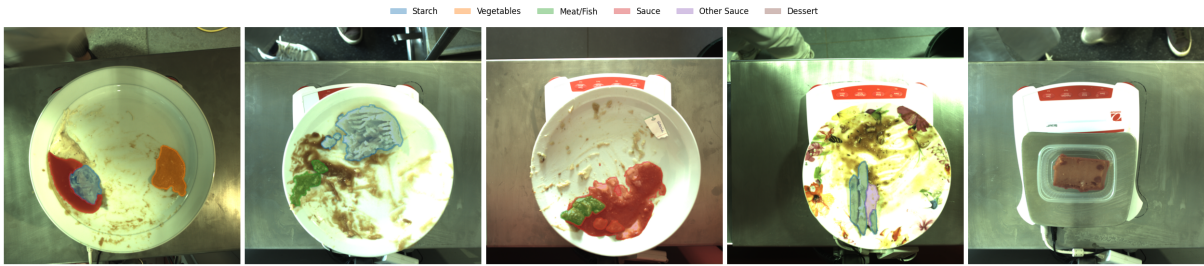


Figure 4.13: Examples of segmentation annotations.

Our quality assessment revealed two primary annotation challenges. The first observation is that the annotation strategy needs to be refined. When one component overlaps another (particularly sauce covering another component), annotation strategies were inconsistent. In some cases, only the visible component was labeled; in others, the underlying component was prioritized. This inconsistency may confuse the segmentation model. The second remark is that sauce is not always (precisely) annotated. For the purpose of this application, this high precision is not necessary but sometimes labeling sauce and sometimes not can lead to confusion for the model. Both problems occur in Figure 4.14.



Figure 4.14: Example of the issues with the segmentation annotations.

These annotation issues reflect the inherent challenges in creating ground truth for overlapping food components and suggest the potential value of implementing a multi-label annotation approach, as discussed in Section 4.5.2.

4.3.6 Potential Challenges

Based on our EDA, we identified several challenges and limitations that may impact model development and performance:

4 Data Collection and Dataset Preparation

- **Class imbalance:** The uneven distribution of food components may lead to degraded recognition performance for underrepresented classes, particularly sauces. This can be partially mitigated through class-weighted loss functions during model training.
- **Inter-institutional variability:** Different residential care centers use varying plate types, food preparation methods, and imaging conditions. We partially address this through data preprocessing techniques based on the gray-world assumption to normalize color distributions.
- **Ambiguous classification taxonomy:** The distinction between "sauce" and "other sauce" categories lacks clear definition, potentially confusing both human annotators and machine learning models.
- **Single-label pixel assignment:** The current annotation scheme assigns each pixel to exactly one class, inadequately representing overlapping food items. This limitation affects our ability to precisely segment and measure components that frequently overlap, such as sauces on other foods.

These identified limitations directly inform our methodology, including data preprocessing strategies, model architecture choices, and evaluation metrics, as described in Chapter 5 and 6.

4.4 Data Splitting Strategy

A critical aspect of this study is determining the allocation of residential care centers across the training, validation, and test sets. Ensuring an appropriate data split is essential, as each subset should ideally be independently and identically distributed (i.i.d.) relative to the others and to unseen data.

Each care center exhibits distinct data patterns, making data samples more similar within a single center than across different centers. Consequently, if data from the same center were to appear in multiple splits, it would introduce data leakage and artificially inflate validation and test performance, resulting in overoptimistic assessments of generalization capability. To prevent this, we maintain institutional integrity by assigning each residential care center exclusively to a single partition.

While this approach means the data are not perfectly identically distributed across splits, it aligns with the system's intended purpose: to generalize effectively to new residential care environments. By ensuring that each center is entirely contained within a single split, the model is more rigorously evaluated on its true ability to adapt to unseen institutional contexts.

4.4.1 Data Splitting Approach

The goal is to achieve a 70-15-15 % distribution on the segmented images across the three splits while maintaining generalization and preventing data leakage. The selection of residential care centers for each split is guided by

4 Data Collection and Dataset Preparation

two primary considerations:

1. **Food Component Diversity:** The training set should be composed of residential care centers that contain a wide variety of food components. This design choice ensures the model is exposed to diverse food items during training, enhancing its ability to recognize and classify a broader range of food components during inference.
2. **Weight Distribution Coverage:** In addition to food diversity, the training set should also include residential care centers with a broad range of food portion sizes and weight distributions. This is particularly important for the task of weight estimation, as the model should be capable of predicting food weights across varying portion sizes. By ensuring that the training set encompasses residential care centers with both small and large portion sizes, the model learns to estimate weights accurately, regardless of the serving size, thus improving its robustness in real-world settings where portion sizes can vary significantly.

4.4.2 Institutional Allocation

The specific allocation of residential care centers to the training, validation, and test sets was determined by applying the primary considerations to the observed data characteristics. The number of images, the number of segmented images, the number of food components and the number of unique food components are summarized in Table 4.3. The weight ranges per residential care center are visualized in a violin plot in Figure 4.8. Based on this table and figure, we have decided to allocate the centers as follows.

The training set includes images from residential care centers 1, 2, and 5. These centers were selected because they contain the largest number of segmented images and the highest diversity of unique food components, ensuring the model learns a broad range of variations. They also provide a wide range of weights for training.

The validation set consists of images from care centers 3 and 7. Care center 7, in particular, contains a significant number of unique food components, which may initially impact validation performance. However, before the evaluation on the test set, the model will be retrained on both the training and validation sets, allowing it to learn from these additional variations.

Finally, the test set consists exclusively of images from care center 4, providing an unseen dataset to assess the model's generalization performance.

The distribution of segmented images across food categories in each partition is summarized in Table 4.4.

4 Data Collection and Dataset Preparation

Table 4.4: Segmented image distribution across data partitions per food category.

Category	#Training	#Validation	#Testing
Starch	249	67	86
Vegetables	232	69	54
Meat and Fish	279	93	76
Sauce	70	18	15
Other Sauce	56	7	3
Total	886	254	234

4.5 Recommendations for Future Data Collection

Based on our dataset analysis and system development experience, we propose several enhancements to address current limitations in data quality, annotation methodology, and collection protocols. These recommendations aim to improve system efficacy for food monitoring in residential care environments.

4.5.1 Data Acquisition Enhancements

Standardized Illumination Protocol

Inter-institutional lighting variability presents a significant challenge to model generalization. We recommend implementing a standardized illumination apparatus with controlled color temperature and consistent luminance to normalize acquisition parameters. This would mitigate exposure issues and reduce domain shift between centers, enhancing feature detection reliability.

Geometric Positioning Standardization

Inconsistent plate positioning impacts the ROI identification. We propose a defined positioning protocol with visual alignment guides to ensure consistent plate registration. A positioning template with concentric guides for standard plate diameters would facilitate consistent central positioning.

Stratified Data Collection Strategy

While our current dataset provides valuable training material, a systematic expansion using stratified sampling principles would address several identified limitations:

- **Extended weight distribution coverage:** Implementation of a stratified sampling approach across pre-defined weight intervals (e.g., 25 g increments from 0-250 g per component) would ensure comprehensive

4 Data Collection and Dataset Preparation

representation across the full spectrum of clinically relevant portion sizes. This would particularly enhance model performance at distribution extremes.

- **Balanced institutional representation:** Addressing the current inter-center data imbalance through targeted supplementary data collection from underrepresented facilities would mitigate potential institutional bias in the model.
- **Compositional diversity enhancement:** Systematic inclusion of varied food preparations and food items would improve generalization. Particular emphasis should be placed on underrepresented components such as sauces, which demonstrated lower detection performance in our current implementation.

4.5.2 Annotation Methodology Refinements

Multi-label Semantic Segmentation Framework

The current single-label pixel assignment method inadequately represents overlapping food components, particularly when sauce covers other items. Implementing a multi-label annotation scheme would allow pixels to simultaneously belong to multiple component classes, better representing the natural layering of food items. While this would require modifications to both the annotation process and model architecture, it would provide a more accurate representation of the physical reality on the plate.

Refined Component Classification

The distinction between "sauce" and "other sauce" is vague. Merging these into a unified "sauce" category or establishing clearer definitional boundaries between them would reduce annotation ambiguity. Alternatively, a hierarchical labeling approach could be implemented, with primary categories (meat, vegetables, starch, sauce) and subcategories that provide additional granularity when needed.

5

Architectural Design and System Development

In this chapter, we present LeftoVision, our proposed solution to monitor food intake and food waste automatically in residential care facilities. Building upon the requirements identified in previous chapters, we first outline the high-level system architecture. We then proceed to discuss the technical implementation details of each module and explaining how they interact to form a cohesive system.

5.1 System Architecture Overview

This section presents the high-level architecture of LeftoVision, designed to monitor food consumption, prevent malnutrition, and reduce food waste in residential care facilities. In Chapter 3, we examined two predominant architectural approaches: end-to-end and multi-stage architectures.

After careful consideration, we adopted a multi-stage architecture rather than an end-to-end approach for several reasons. First, the limited availability of training data makes it challenging to train an effective end-to-end model. Second, this architecture provides flexibility for future extensions and improvements, as components can be upgraded independently without affecting the entire system. Finally, a stage-based approach facilitates more effective integration of domain knowledge at each processing step.

5.1.1 Core Pipeline Design

LeftoVision follows a modular processing pipeline that transforms plate images into nutritional data, stores it, and visualizes it in a dashboard user interface. Figure 5.1 illustrates the different modules in this standard pipeline architecture. The blue components are inputs or intermediate results (data). The yellow components are the implemented modules.

The system takes an image of a food plate as input. While Figure 5.1 shows a plate before eating, the pipeline processes both before-meal and after-meal images. Next, a semantic segmentation model identifies individual food components on the plate, creating separate images for each component with black backgrounds. This

5 Architectural Design and System Development

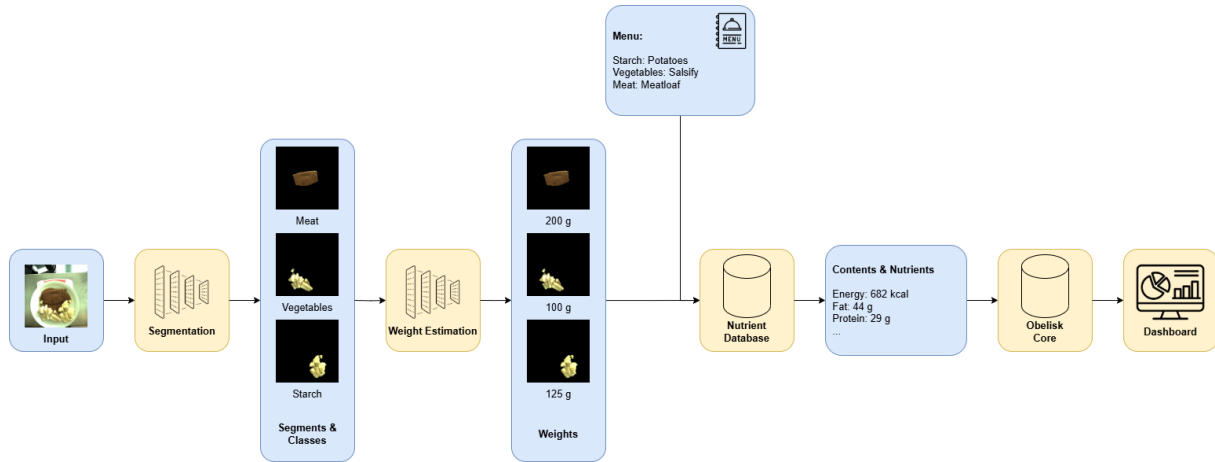


Figure 5.1: High-level architecture of the LeftoVision standard pipeline.

background choice is intentional, as black typically does not appear in food items, preventing model confusion. Each segmented component image is processed by a weight estimation model that determines the component's mass. Next, the system performs a menu lookup to identify the exact component name, then queries a nutrient database to determine the nutritional content based on the component's name and estimated weight. The final outputs, nutritional data for each food component (both before and after eating), are stored in the Obelisk Core data platform. By querying the data from the Obelisk Core data platform and doing some processing, a dashboard user interface can be generated, displaying information about the food waste, the status of malnutrition of the residents, and the menu insights.

5.1.2 Used Technologies

LeftoVision is implemented in Python 3.11.11 and relies on several key libraries, selected for their robustness, community support, and specific capabilities:

- OpenCV 4.11.0 [78] and PIL 9.4.0 [79]: For image processing and manipulation, chosen for their comprehensive set of image processing functions and efficiency in handling large volumes of images.
- PyTorch 2.6.0 [80] and TorchVision 0.21.0 [81]: For implementing deep learning models, selected for flexibility in model architecture.
- Segmentation Models Pytorch 0.3.3 [82]: For leveraging existing semantic segmentation architectures and backbones, reducing development time while maintaining high performance.
- NumPy 2.2.3 [83] and Pandas 2.2.3 [84]: For numerical operations and data management, essential for efficient processing of nutritional information.
- Matplotlib 3.10.1 [85] and Seaborn 0.12.2 [86]: For visualization and analysis during development.

5 Architectural Design and System Development

- Dash Plotly 3.0.0 [87]: For dashboard prototyping, selected for its balance of capabilities and ease of implementation.

5.1.3 Implementation Architecture

This section describes the architectural design principles that guide the system's organization.

The system uses a hybrid architecture combining modular components with centralized orchestration:

- **Component Modules:** Each pipeline component has well-defined responsibilities, facilitating maintenance and future enhancements.
- **Centralized Execution:** A main orchestration layer coordinates the overall execution flow, loading trained models and ensuring proper data transfer between components.

The interaction between the different components and the orchestrator is visualized in Figure 5.2.

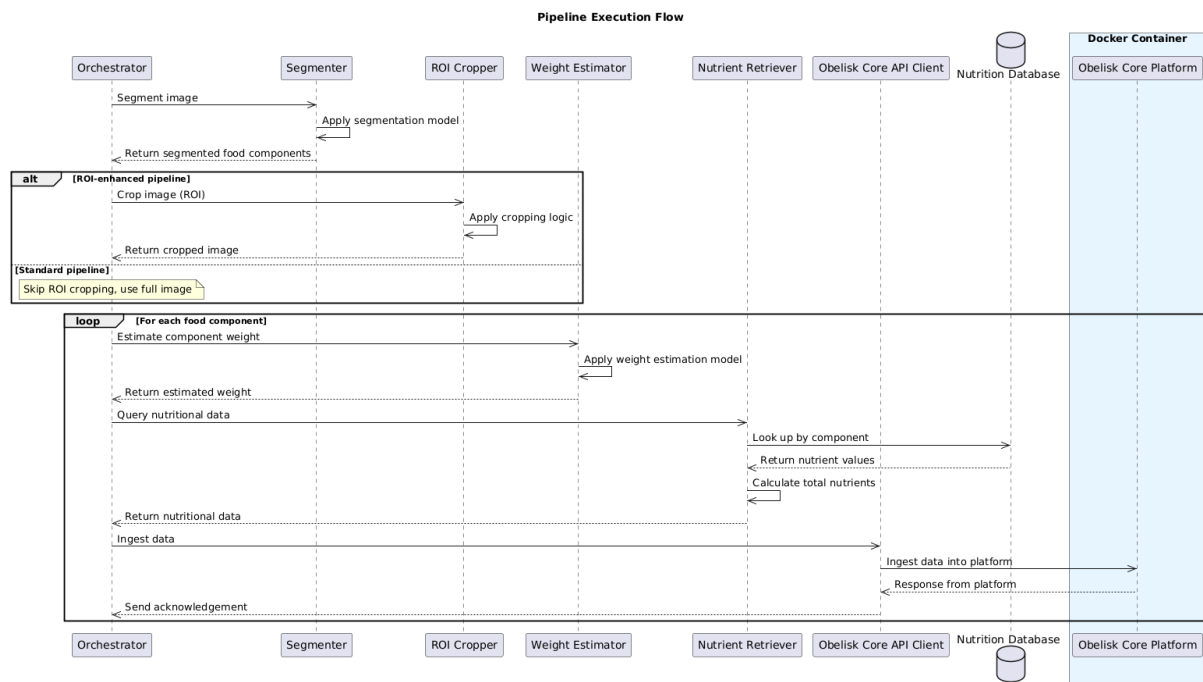


Figure 5.2: Execution flow of the pipeline.

This approach balances modular development with efficient pipeline execution. The modular architecture is designed to be easily adaptable to REST-API or gRPC wrappers, enabling a smooth transition to a microservice architecture for production environments.

5 Architectural Design and System Development

5.1.4 Pipeline Variants

To explore different approaches to the food intake monitoring problem, we implemented three variations of the processing pipeline. We will discuss these in more detail in the following sections. Afterwards, we will discuss the design patterns which enhance the flexibility and maintainability of our implementation, allowing for easy extension and adaptation of the pipeline architecture to different experimental setups.

Standard Pipeline

The default implementation, as described in Section 5.1.1, which processes the entire plate image directly through all stages of the pipeline.

ROI-Enhanced Pipeline

This variant adds an ROI cropping step before weight estimation, as shown in Figure 5.3. The ROI cropper identifies the plate area, potentially improving weight estimation accuracy by focusing on the actual food components and eliminating noise from background elements. The ROI cropper is placed before the weight estimation module and not before the segmentation module as most segmentation models are designed to use global context to make accurate predictions. Cropping the image first would remove potentially valuable contextual information from surrounding regions that helps the model understand what it is seeing [32].

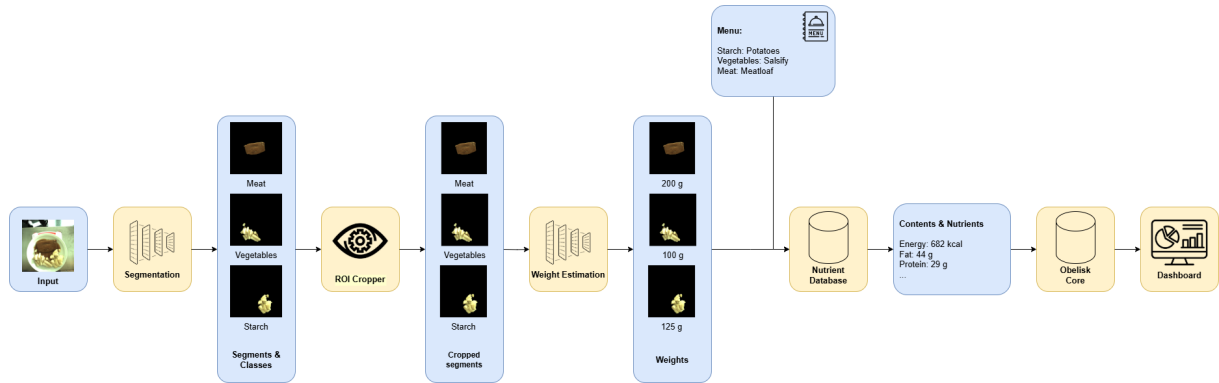


Figure 5.3: High-level architecture of the ROI-enhanced pipeline.

Dual Model Pipeline

This variant uses two separate weight estimation models, one specialized for plates before eating and another for plates after eating, as illustrated in Figure 5.4. This approach acknowledges the different visual characteristics of untouched versus partially consumed food and may improve estimation accuracy across both scenarios. For model selection at inference time, we implemented two strategies: (1) a method based on the ratio of non-black pixels compared to the number of black pixels, and (2) a temporal sequence assumption where the first captured

5 Architectural Design and System Development

image is classified as pre-consumption and the second as post-consumption. We expect the second approach to give a higher accuracy.

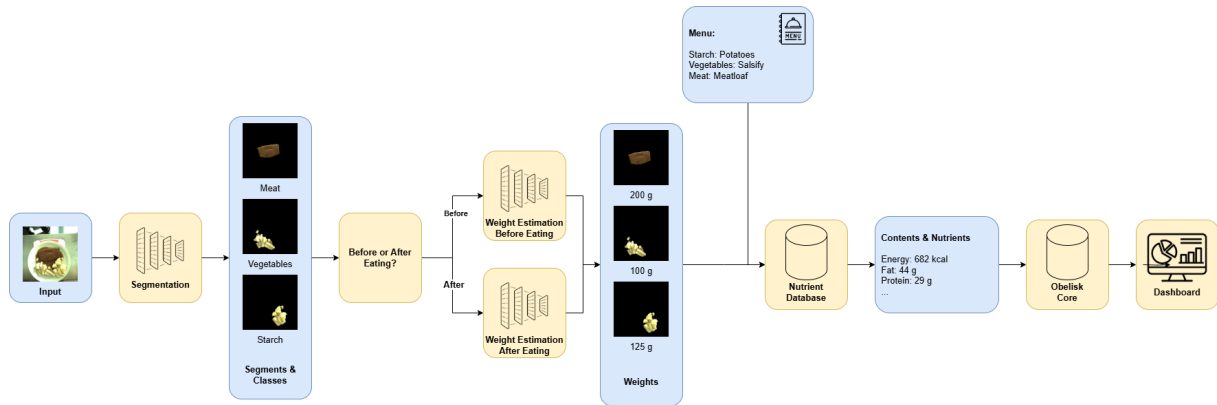


Figure 5.4: High-level architecture of the dual weight estimation pipeline.

Design Patterns

The system architecture incorporates several object-oriented design patterns that enhance flexibility and maintainability. These patterns support the diverse pipeline variants while maintaining a consistent interface and minimizing code duplication across implementations.

The template method pattern is implemented through a base pipeline class that defines the fundamental structure and common operations for all pipeline variants. This abstract base class establishes the core processing sequence—image segmentation, segment extraction, and weight estimation—while providing default implementations that can be inherited or overridden by specialized pipelines. For example, all pipelines perform image segmentation in a similar manner, but the ROI-Enhanced pipeline extends this process with additional region identification steps. This approach standardizes the pipeline workflow while enabling customization where needed.

A factory method pattern handles the creation of appropriate pipeline objects based on configuration parameters. The system includes a dedicated pipeline factory that examines the requested configuration and instantiates the corresponding pipeline implementation—standard, ROI-enhanced, or dual model. This encapsulates the instantiation logic and provides a clean interface for client code to obtain the desired pipeline variant without needing to understand implementation details.

The strategy pattern enables the interchangeability of different processing algorithms through concrete pipeline classes that implement the same interface but use different strategies. Each pipeline variant represents a distinct strategy for processing that can be selected at runtime.

5 Architectural Design and System Development

5.2 Food Segmentation Module

The foundation of LeftoVision's food intake monitoring system is the segmentation module, which serves as the critical first step in our pipeline. The segmentation module's primary objectives are: (1) to accurately delineate boundaries between different food components, (2) to correctly classify each food item according to the five pre-defined categories, and (3) to generate segmented images for downstream weight estimation. The challenge lies in achieving these objectives for complex meal compositions where food items may share similar visual characteristics or overlap. The precision of all downstream processes depends directly on the quality of this initial segmentation.

In Section 2.1.2, we have discussed the three possible segmentation approaches. After careful consideration of the various approaches, we selected semantic segmentation for this task. While instance segmentation and panoptic segmentation offer the ability to distinguish between different instances of the same class, this level of differentiation is unnecessary for our application and would introduce additional computational overhead without providing tangible benefits for nutritional analysis. Semantic segmentation provides the optimal balance between accuracy and efficiency for our specific requirements.

In this section, we document our systematic journey toward developing an optimal segmentation solution—from model selection to implementation details and training methodology—highlighting key design decisions that enabled us to overcome the challenges of food segmentation in residential care settings.

5.2.1 Implementation Framework

To streamline the development process, we developed our segmentation module within a Jupyter notebook with externalized configuration parameters. All experimental hyperparameters, preprocessing settings, and model configurations were centralized in YAML configuration files, allowing for systematic experimentation and precise tracking of different model variations.

This configuration-driven approach facilitated rapid iteration across architectural variants while maintaining reproducibility of results. The framework automatically logged all experimental parameters alongside corresponding performance metrics, creating a comprehensive record of our development progression. This methodology proved particularly valuable during hyperparameter optimization, enabling efficient comparison between different experimental configurations without code modifications.

5 Architectural Design and System Development

5.2.2 Model Architecture Selection

Architecture and Backbone Evaluation

To determine the most suitable architecture and backbone combination for our food segmentation task, we conducted a comprehensive evaluation of several state-of-the-art semantic segmentation models. Our assessment included architectures such as DeepLabV3, DeepLabV3+, and SegFormer paired with various backbones including ResNet (18, 34, 50, 101), ResNeXt-50 32x4d, MobileNet, and MiTB2, which are trained on ImageNet. Rather than implementing the model architectures ourselves, we used the existing implementation from the Segmentation Models Pytorch library [82] to ensure development efficiency.

Based on experimental performance results presented in Table 6.1, we selected DeepLabV3+ as our architecture. For the backbone, we chose ResNeXt-50 32x4d. This combination delivers an optimal balance between segmentation accuracy and computational efficiency for our specific use case.

Alternative Approaches Considered

Prior to selecting DeepLabV3+, we evaluated several alternative approaches, including the recent segment anything model. Specifically, we tested FoodSAM [47], a specialized version of SAM trained on food data, on our dataset. Figure 5.5 shows some predictions that exhibited poor boundary precision and inadequate classification.



Figure 5.5: Examples of segmentation predictions from FoodSAM.

We also considered retraining SAM 2 developed by Meta [88], but ultimately decided against this approach. SAM 2 generates masks through an iterative prompt-based process, which introduces computational overhead and latency. Although real-time processing is not a hard requirement for our application, this model appeared to be excessive for our needs.

5.2.3 Input Data Preprocessing and Augmentation Strategy

Effective preprocessing and data augmentation are critical components in developing robust deep learning models for semantic segmentation. By implementing these techniques, we aim to enhance model generalization

5 Architectural Design and System Development

and performance despite the inherent variability in food presentation. This section details our methodology for preparing input data and artificially expanding our training dataset.

Preprocessing Pipeline

To ensure consistent and optimal input conditions for our segmentation model, our preprocessing pipeline consists of several components:

- **White Balance Correction:** Based on the gray-world assumption, white balance correction was performed to mitigate color casts caused by different lighting environments. This method operates under the principle that the average reflectance in a scene is achromatic, which helps to bring consistency to the image colors.
- **Resizing:** Although the original images in our dataset have a resolution of 2048×2048 pixels, we standardized the input size to 512×512 pixels. This decision was informed by comparative testing of the model's performance across various input resolutions, as detailed in Table 6.2. The selected resolution represents an optimal trade-off between accuracy and processing efficiency.
- **Normalization:** Using channel means and standard deviations matching those of the ImageNet dataset to align with the pretrained backbone.

Data Augmentation Strategies

To enhance model generalization despite limited training data availability, we implemented both conventional and innovative augmentation techniques:

- **Standard Transformations:** Including random rotations of up to ± 30 degrees, random horizontal and vertical flips each with a 50 % probability of application, and random brightness and contrast adjustments limited to ± 10 % variation from the original image.
- **Synthetic Plate Compositions:** To substantially expand our training dataset, we developed a more sophisticated augmentation method: creating synthetic plate compositions. This process involves extracting food components from existing images using their ground truth masks. These components are then randomly selected, scaled, and rotated before being placed onto images of empty plates to form new, synthetic meal arrangements. For every synthetic image generated, the corresponding ground truth segmentation mask is created simultaneously.

Some synthetically generated images are visualized in Figure 5.6. While this approach theoretically allows for the creation of an almost infinitely large dataset through various combinations, in our implementation, we effectively quadrupled the size of our training dataset by creating variations not present in the original data collection. However, despite this significant increase in training examples, this method did not yield substantial performance

5 Architectural Design and System Development

improvements as measured by our evaluation metrics. The comparative impact of the augmentation strategy on model performance is presented in Table 6.3. A possible reason for this limited impact is that the lighting differences between the extracted food components and the empty plates were not always realistically modeled. This finding suggests that for food segmentation tasks, the quality and realism of training data may be more important than sheer quantity, particularly when synthetic composition methods struggle to accurately replicate real-world lighting interactions.



Figure 5.6: Examples of the synthetic data augmentation.

5.2.4 Training Methodology

With preprocessing and augmentation strategies in place, we focused on designing an optimal training regimen. This required careful hyperparameter selection and monitoring to ensure model convergence while preventing overfitting. Additionally, we have implemented a few techniques to enhance both the performance and computational efficiency. After the hyperparameter selection we have retrained the model by using the selected hyperparameters on both the training and validation set.

Training Process

The training process incorporated several advanced techniques to enhance both performance and computational efficiency:

- **Focal Loss:** Selected to address class imbalance in the images, as most of the pixels belong to the background. This imbalance means that a model which always predicts "background" would achieve a quite good performance for some metrics. The Focal Loss function is defined as:

$$\mathcal{L}_{focal} = \frac{1}{N} \sum_{i=1}^N \alpha_i (1 - p_t^i)^\gamma \mathcal{L}_{CE}^i \quad (5.1)$$

where p_t^i is the model's estimated probability for the target class of sample i , γ is the focusing parameter that reduces the loss contribution from easy examples, α_i represents the class-specific weight factors, $\mathcal{L}_{CE}^i$ is the cross-entropy loss for sample i , and N is the total number of samples. To counteract the imbalance of the sauce categories, we added custom class weights to give these components larger

5 Architectural Design and System Development

weights. Specifically, we used weights of $[1, 1, 1, 1, 4, 4]$ corresponding to background, starch, vegetables, meat/fish, sauce, and other sauce, respectively. This approach makes sense given the limited number of images containing these elements, and empirically it provides better performance, as clearly demonstrated in Table 6.4.

- **Gradient Accumulation:** Used to enable training with larger effective batch sizes than would fit into GPU memory. This technique works by computing the gradients for smaller mini-batches sequentially and accumulating them over several steps, performing a single weight update only after the gradients corresponding to the desired large effective batch size have been summed. This process mimics the optimization step of training with a true large batch, helping improve training stability and convergence when memory is a constraint.
- **Mixed Precision Training:** Uses half-precision format for faster computation while maintaining numerical stability.
- **CosineAnnealingLR:** Provided cyclical learning rate adjustments to escape local minima and improve convergence.
- **Early Stopping:** Prevented overfitting by monitoring validation loss and stopping training when no improvement of minimum 10^{-5} was observed for 10 consecutive epochs.

Figure 5.7 illustrates the training and validation loss curves throughout the training period. The initial steep decline in loss reflects the model's rapid learning of basic segmentation patterns. After approximately 30 epochs, the validation loss began to plateau, with only minor fluctuations. Early stopping was applied, resulting in a final training duration of 71 epochs. This process helped prevent overfitting and ensured the model reached its optimal performance point. Importantly, the gap between the training and validation loss remained reasonably small throughout this phase, suggesting that overfitting was not severe. It is worth noting that a significant gap, however, does not solely indicate overfitting but can also be symptomatic of data-related issues.

Hyperparameter Optimization

We conducted extensive hyperparameter tuning to maximize the model's segmentation performance. Table 5.1 presents the final configuration used for training, representing the optimal settings identified through systematic experimentation. For any methods or components not explicitly detailed in the table, we used the default settings provided by the corresponding PyTorch implementation.

Final Model Training

After identifying the optimal hyperparameters through our experimental validation process, we proceeded to train the final model using the combined training and validation datasets. This approach maximizes the use of

5 Architectural Design and System Development

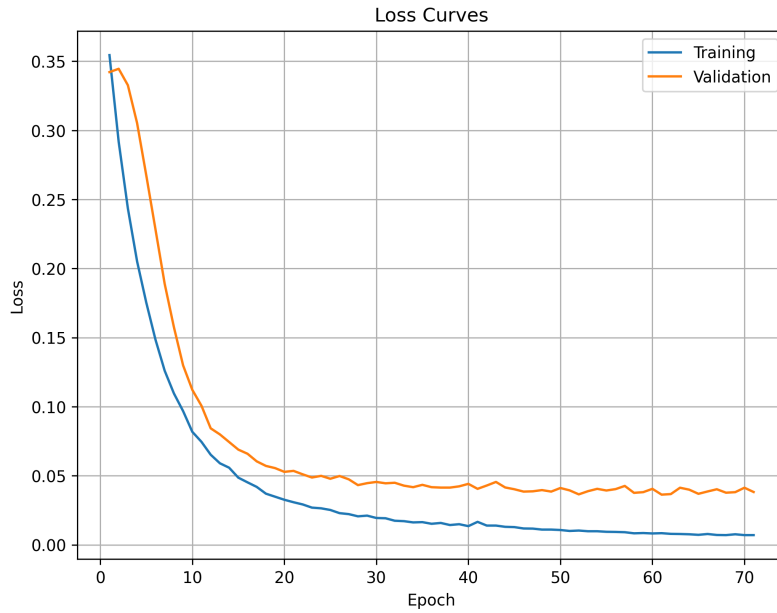


Figure 5.7: Training and validation loss curves for the segmentation model.

available labeled data while maintaining the hyperparameter configuration that demonstrated optimal performance during validation. Importantly, by first determining the optimal hyperparameters using only the validation set performance (completely separate from the test set), we ensure there is no data leakage into our evaluation process. The final model was trained for the same number of epochs (61) as determined by our early stopping criterion in the validation experiments. The results of this model are discussed in Section 6.2.1.

5.2.5 Extracting Segments

The masks created by the segmentation model are used to extract segments. Per present component a segmented image is created, such that only the pixels of that component are visible and the rest of the pixels are black. After any optional cropping, these images are passed to the weight estimation module.

5.3 Region of Interest Extraction

In the case of the ROI-enhanced pipeline, the ROI is first determined and cut-out before moving on to the weight estimation module. In this application, the ROI corresponds to the plate.

Given that the plates in the dataset are consistently circular, the Hough Circle Transform from OpenCV is used to detect the circles representing the plates. Prior to applying the circle detection algorithm, a series of preprocessing steps are used to reduce noise and enhance the plate's edges, thus ensuring accurate circle detection. A visual representation of the preprocessing pipeline is provided in Figure 5.8.

5 Architectural Design and System Development

Table 5.1: Hyperparameters for the segmentation model (DeepLabV3+).

Category	Hyperparameter	Value
Model Architecture	Backbone	ResNeXt-50 32x4d
	Encoder Weights	ImageNet
Optimization	Loss Function	Focal Loss
	Class Weights	[1, 1, 1, 1, 4, 4]
	Optimizer	AdamW
	Weight Decay	10^{-4}
Learning Rate	Initial LR	10^{-4}
	Scheduler	CosineAnnealingLR
	$T_{\max}$	10
	$\eta_{\min}$	10^{-6}
Training	Batch Size	32
	Gradient Accumulation Steps	2
	Mixed Precision	Enabled
Convergence	Early Stopping Patience	10
	Early Stopping $\delta_{\min}$	10^{-5}
	Best Epoch	61

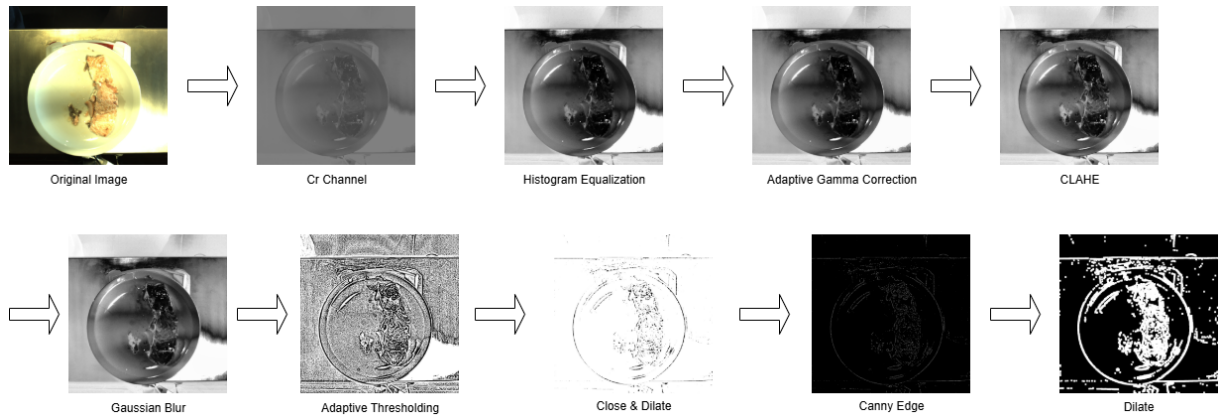


Figure 5.8: Preprocessing pipeline for ROI extraction (plate detection).

5 Architectural Design and System Development

5.3.1 Image Preprocessing

The objective of image preprocessing is to improve the quality of the input image such that the detection of circular objects, specifically the plates, is more effective. The preprocessing pipeline comprises several steps aimed at enhancing image contrast, reducing noise, and emphasizing the plate boundaries. The following paragraphs describe the steps involved in preprocessing the image.

Grayscale Conversion

First, we convert the input image to grayscale to simplify the processing and computational complexity. This step removes color information and focuses on intensity values, allowing for more efficient edge detection. Specifically, we extract the chroma blue channel from the YCrCb color space, which offers enhanced contrast for plate detection.

Histogram Equalization

Histogram equalization is applied to improve the image's global contrast by redistributing the pixel intensity values across the entire available range. This technique enhances both dark and light regions, improving the overall visibility of plate boundaries. The `equalizeHist` function in OpenCV is used in the implementation and provides a global enhancement that serves as a foundation for subsequent local contrast adjustments.

Adaptive Gamma Correction

Adaptive gamma correction is performed to adjust the brightness of dark regions while maintaining well-exposed brighter areas. The gamma value is adjusted locally based on the average intensity of each region. This helps to enhance the visibility of plates under varying lighting conditions, particularly in shadows or areas with uneven illumination.

Contrast Limited Adaptive Histogram Equalization (CLAHE)

While the initial histogram equalization provides global contrast enhancement, CLAHE is applied to refine local contrast details. Unlike global histogram equalization, CLAHE operates on small regions (tiles), preventing excessive amplification of noise in uniform areas. The `createCLAHE` function is used with a clip limit of 3.0 and a tile grid size of (8, 8). The parameters are selected through experimentation.

Gaussian Blur

A Gaussian blur is applied to reduce high-frequency noise and smooth out the image, which helps avoid interference with edge detection. The kernel size used for this operation is (5, 5), applied via the OpenCV `GaussianBlur` function.

5 Architectural Design and System Development

Adaptive Thresholding

Adaptive thresholding converts the image into a binary format, where the edges and boundaries of the plates are highlighted. The threshold is calculated based on the local neighborhood of each pixel, which makes the method more robust to varying lighting conditions. The OpenCV function `adaptiveThreshold` is used, with a block size of 45 and a constant C value of 2, parameters determined through extensive testing to optimize plate boundary detection.

Morphological Operations

Morphological operations, specifically closing and dilation, are applied to strengthen the structure of the binary image. These operations help close small gaps in the detected edges and reinforce the plate boundaries. The `morphologyEx` function with a kernel size of (3, 3) is used for this purpose.

Canny Edge Detection

Canny edge detection is used to identify strong edges in the image, which are crucial for the effective detection of circles. The Canny edge detector is applied with a lower threshold of 100 and an upper threshold of 200, based on experimentation to highlight plate edges and suppress noise.

Dilation

After the canny edge detection, dilation is performed to strengthen and connect nearby edge pixels, improving the continuity of detected edges. This final preprocessing step produces edge maps that serve as input to the Hough Circle Transform.

5.3.2 Hough Circle Detection

After the preprocessing steps, the Hough Circle Transform is applied to detect the plates in the image. The `HoughCircles` function is used with the following parameters, each experimentally optimized for our specific plate detection task:

- `dp`: 1.3 — The inverse ratio of the accumulator resolution to the image resolution, providing an optimal balance between accuracy and efficiency.
- `minDist`: 50 pixels — The minimum distance between the centers of detected circles, preventing multiple detections of the same plate.
- `param1`: 75 — The higher threshold for the internal Canny edge detector, refining edge detection quality.

5 Architectural Design and System Development

- `param2`: 0.0005 — The accumulator threshold for detecting circle centers, balanced to improve detection accuracy.
- `minRadius`: 715 pixels — The minimum radius of the detected circles, matching the typical size of plates in the dataset.
- `maxRadius`: 770 pixels — The maximum radius of the detected circles, accommodating variability in plate sizes.

5.3.3 Region Extraction

After detecting the circles, bounding regions are computed around each detected circle. The region is then cut-out on the segmented images. To ensure the entire plate is captured, a margin of 100 pixels is added around each region. This margin compensates for cases where the detected circle may be slightly smaller than the actual plate, ensuring the full plate is included in the cropped area. Figure 5.9 demonstrates examples of cropped plate images produced by our ROI extraction module. If no circles are detected, the original image is passed to the pipeline without modification.



Figure 5.9: Examples of cropped plate images produced by the ROI extraction module.

5.4 Weight Estimation Module

After applying the segmentation, and optionally the ROI extraction, the segmented images are passed to the weight estimation module. The weight estimation module's primary objective is to accurately determine the weight of each food component. The challenge lies in establishing a reliable relationship between visual features and actual food mass, particularly given variations in density, shape, and portion size across different food categories. The accuracy of nutritional calculations depends directly on the precision of these weight estimations.

In this section, we document our systematic approach toward developing an optimal weight estimation solution—from baseline methods and architectural evaluation to implementation details and training methodology—highlighting key design decisions that enabled us to overcome the challenges of food weight estimation in residential care settings.

5 Architectural Design and System Development

5.4.1 Implementation Framework

Similar to our segmentation module, we developed the weight estimation module using a structured approach with externalized configuration parameters. All experimental hyperparameters, preprocessing settings, and model configurations were centralized in configuration files, allowing for systematic experimentation and precise tracking of different model variations.

5.4.2 Model Architecture Selection

Architecture Evaluation

To determine the most suitable architecture for our food weight estimation task, we conducted a comprehensive evaluation of several state-of-the-art CNN models. Our assessment included architectures such as ResNet-50, ResNet-101, AlexNet, DenseNet, GoogLeNet, and Wide ResNet. Each model was adapted to perform regression rather than classification by replacing the final classification layer with regression layers designed to predict continuous weight values. Rather than implementing the model architectures ourselves, we used the existing implementations from the torchvision library to ensure development efficiency [81].

A key enhancement in our approach was the integration of food category information as an additional input to the CNN models. Rather than relying solely on image features, we implemented a multi-input architecture that combined visual features extracted by the CNN with a one-hot encoded representation of the food category. This approach leverages prior knowledge about food types, allowing the model to learn category-specific weight estimation patterns and potentially improve accuracy. It's important to note that if the segmentation module predicts an incorrect category, this error propagates to the weight estimation model.

Based on our comparative evaluation, we selected ResNet-50 as our backbone. This model delivered an optimal balance between estimation accuracy and computational efficiency for our specific use case, as demonstrated in Table 6.7.

Unlike conventional transfer learning approaches that freeze pre-trained layers, we opted for complete retraining of all network layers. This decision was based on experiments showing that full retraining yielded superior performance for our specific task of food weight estimation. The results of a model completely retrained and of a transfer-learned model are shown in Table 6.8. We hypothesize that this improvement stems from the significant domain difference between general object recognition (the original task of pre-trained networks) and food weight estimation.

Alternative Approaches Considered

Prior to selecting a deep learning approach, we developed and evaluated a simpler pixel-based weight estimation method. This baseline approach attempts to establish a direct relationship between pixel count and component

5 Architectural Design and System Development

weight using linear regression:

$$W = a \cdot P + b$$

where:

- W is the estimated weight of the food component.
- P represents the number of pixels occupied by the segmented food item.
- a and b are the regression parameters learned from the training data.

For the implementation we used the scikit-learn library's LinearRegression class with default parameters [89]. We implemented category-specific regression models to account for the significant variation in density across food categories. This decision was motivated by the observation that a potato and a piece of fish occupying the same number of pixels likely differ substantially in weight. The models were trained on the training and validation set and were evaluated on the test set. However, this pixel-based approach faced two main limitations:

1. **Food Density and Shape Variations:** Even within the same category, foods can have varying densities and shapes. Creating individual models for each specific food item (e.g., carrots, peas, potatoes) was considered but rejected due to data limitations. Such fine-grained modeling would require extensive labeled data for each individual food item, which was beyond the scope of our current dataset.
2. **Limited Feature Representation:** This approach relies solely on the number of pixels as a proxy for food mass, without incorporating additional visual or contextual features such as texture and color intensity.

To evaluate the performance of the pixel-based weight estimation approach, we computed the RMSE for each food category on the test set. Table 5.2 summarizes these results.

Table 5.2: Performance of the pixel-based weight estimation method on the test set.

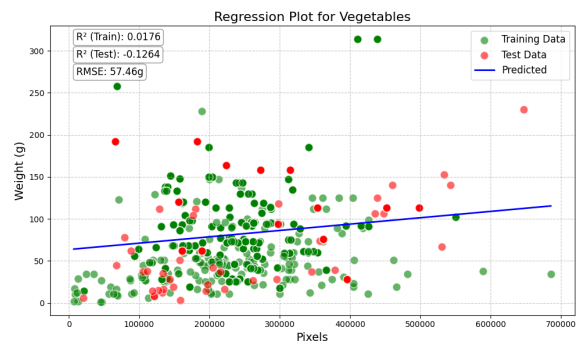
Food Category	RMSE (g)
Starch	43.36
Vegetables	57.46
Meat and fish	57.71
Sauce	52.42
Sauce other	23.43
Average	51.71

5 Architectural Design and System Development

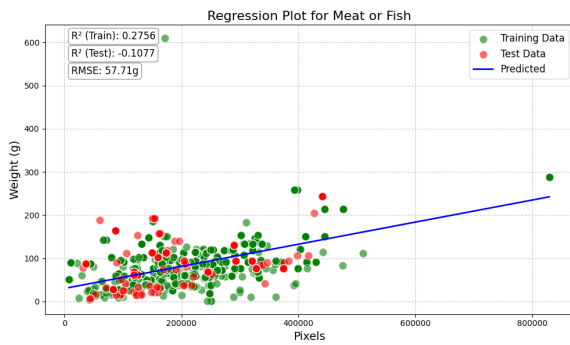
To visualize the relationship between pixel count and weight, we generated regression plots for each food category. Figure 5.10 displays both the training data (green dots), the test data (red dots), and the computed regression line. The substantial dispersion of data points around the regression lines indicates a weak correlation between pixel count and weight for all food categories, confirming the limitations of this approach. This is also confirmed by the low R^2 values. The high RMSE values, particularly for the sauce and vegetables categories, further validate our concerns about the pixel-based approach. These results motivated our exploration of more sophisticated deep learning approaches that could capture complex visual cues beyond simple pixel counts.



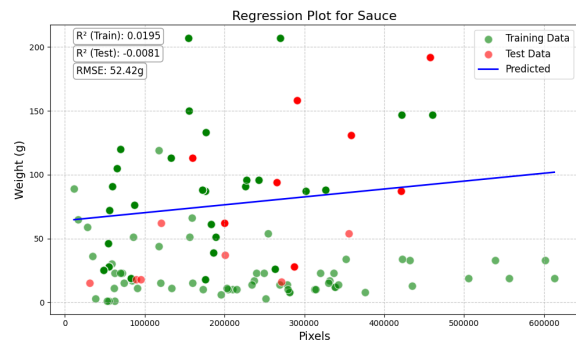
(a) Regression plot for starch.



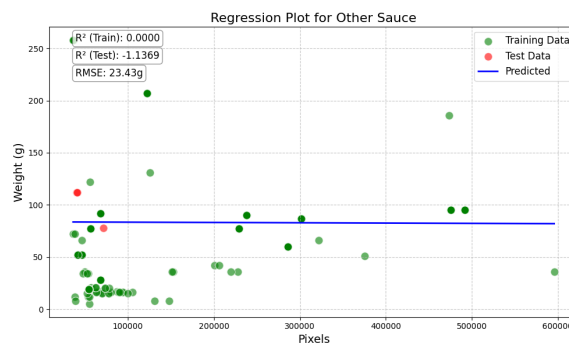
(b) Regression plot for vegetables.



(c) Regression plot for meat and fish.



(d) Regression plot for sauce.



(e) Regression plot for other sauce.

Figure 5.10: Pixel count versus actual weight with regression line for each food category.

5 Architectural Design and System Development

5.4.3 Input Data Preprocessing and Augmentation Strategy

Input Data Preprocessing

To ensure consistent and optimal input conditions for our weight estimation model, our preprocessing pipeline consists of several components:

- **White Balance Correction:** The component-specific images were subjected to white balance correction using the gray-world assumption, consistent with the approach detailed in the segmentation module.
- **Resizing:** The processed images were then resized to a uniform 224×224 pixel resolution, as our experimental analysis (Table 6.9) showed no significant performance improvements from higher resolutions.
- **Normalization:** For training stability, normalization was applied using standard ImageNet normalization parameters.

Augmentation Strategy

To improve model generalization and effectively expand the training dataset, we used data augmentation techniques. The training images were randomly subjected to horizontal and vertical flips, each with a 50 % probability of application. These transformations introduce controlled variations while preserving the essential visual and semantic characteristics of the food components, enabling the model to develop more robust feature recognition capabilities and reducing the risk of overfitting on limited training samples.

5.4.4 Training Methodology

With preprocessing and augmentation strategies in place, we focused on designing an optimal training regimen. This required careful hyperparameter selection and monitoring to ensure model convergence while preventing overfitting.

Training Process

During the training process, we carefully monitored both training and validation loss to ensure proper convergence and prevent overfitting. Figure 5.11 presents these loss curves over the training epochs. The plot demonstrates an initial consistent decrease in both metrics, indicating effective learning. As training progressed, a significant gap emerged between the training and validation loss. However, the validation loss did not increase; instead, it stabilized. This pattern suggested that while the model continued to improve on the training data, the divergence was likely due to data-related issues or characteristics of the dataset rather than classical overfitting.

Early stopping was implemented with monitoring the validation loss. The model achieved its best validation

5 Architectural Design and System Development

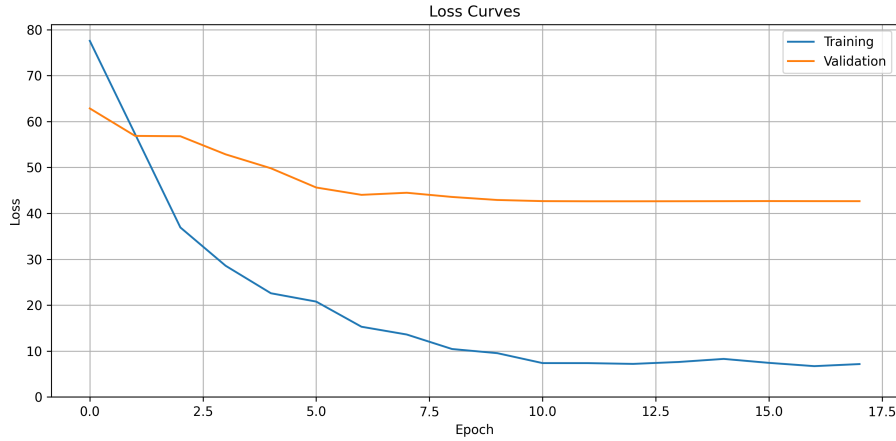


Figure 5.11: Training and validation loss curves for the general weight estimation model.

performance at epoch 12, after which no further improvement was observed. The early stopping mechanism effectively prevented overfitting while ensuring the model converged to its optimal performance point.

Hyperparameter Optimization

We conducted extensive hyperparameter tuning to maximize the model's weight estimation performance. Table 5.3 presents the final configurations used for training our model, representing the optimal settings identified through systematic experimentation on the validation set. For any methods or components not explicitly detailed in the table, we used the default settings provided by the corresponding PyTorch implementation.

Final Model Training

After identifying the optimal hyperparameters through our experimental validation process, we proceeded to train the final model using the combined training and validation datasets. This approach maximizes the use of available labeled data while maintaining the hyperparameter configuration that demonstrated optimal performance during validation. The results of this model are discussed in Section 6.2.2.

5.4.5 Dual-Model Approach for Different Plate States

Motivation and Implementation

Analyzing served (before eating) and cleared (after consumption) plates revealed substantial variance in portion size and visual appearance. This difference suggested a single model would struggle to accurately predict weights due to the distinct visual characteristics of full versus partially consumed plates.

To address this challenge, we developed a dual-model approach that differentiates between the two plate states.

5 Architectural Design and System Development

The dual-model strategy involves training separate neural networks: one specialized for served plates and another for cleared plates. Each model is optimized to handle the unique visual characteristics and estimation challenges of its respective plate state. At inference time, we have two possible approaches to select the appropriate model. The first approach selects the model based on the ratio of non-black pixels to black pixels in the image, using a threshold of 8.3 % which corresponds to the average of the mean ratios for cleared images (5.4 %) and served images (11.2 %). The second approach selects the model based on the ground truth label of plate state. When ingesting the data in our system, we can assume that the first image is of the served plate and the second image is of the plate after eating.

Training Process for Specialized Models

For both the serving and clearing models, we used the same ResNet-50 architecture as our general model. We conducted separate hyperparameter tuning for each model to optimize their performance for their specific use case. The hyperparameters for both specialized models are presented in Table 5.3.

Table 5.3: Hyperparameters used for weight estimation models.

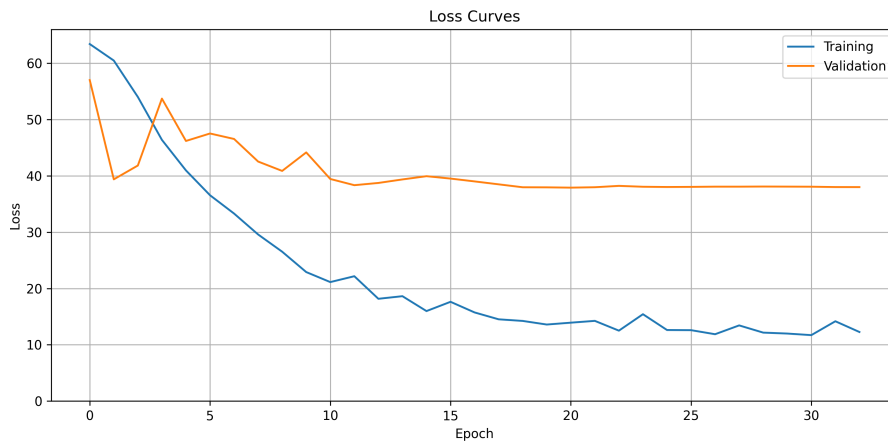
Category	Hyperparameter	General Model	Cleared Plates	Served Plates
Model Architecture	Backbone	ResNet-50	ResNet-50	ResNet-50
Optimization	Loss Function	RMSE	RMSE	RMSE
	Optimizer	Adam	Adam	Adam
Learning Rate	Initial LR	5×10^{-4}	5×10^{-4}	10^{-3}
	Scheduler	StepLR	StepLR	StepLR
	Step Size	5	4	4
	Gamma	0.1	0.1	0.1
Training	Batch Size	128	128	128
Convergence	Early Stopping Patience	5	12	12
	Early Stopping $\delta_{\min}$	10^{-5}	10^{-5}	10^{-5}
	Best Epoch	12	21	28
	Warm-up Epochs	6	6	6

Figure 5.12 visualizes the loss curves for both the serving and clearing models. Both models exhibit good convergence, despite some fluctuations in the early epochs. A substantial gap between training and validation loss is

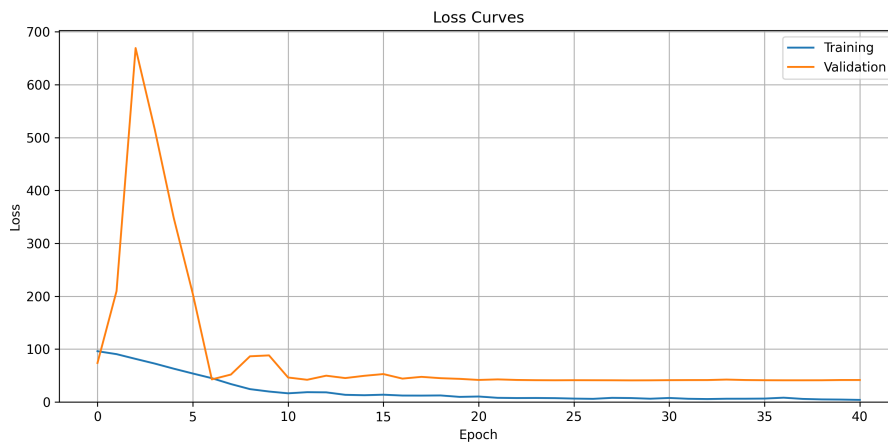
5 Architectural Design and System Development

also evident. As with the general model, this divergence is likely attributable to data-related issues, given that the same dataset was used.

After determining the optimal hyperparameters on the validation set, we retrained both specialized models on the combined training and validation datasets. The performance results of these specialized weight estimation models are provided in Section 6.2.2.



(a) Loss curves of the clearing model.



(b) Loss curves of the serving model.

Figure 5.12: Training and validation loss curves for the specialized serving and clearing weight estimation models.

5.5 Nutrient Database Integration

To convert the food component classification and weight estimation into meaningful nutritional information, LeftoVision interfaces with the USDA FNDDS, which is discussed in Section 3.3.3.

5 Architectural Design and System Development

5.5.1 Database Selection and Considerations

While the database provided by the ILVO contains specific nutrient information for components tracked during the measurement campaign, we opted for FNDDS to enhance the generalizability of our system. This decision facilitates broader application across different food environments and geographical contexts, aligning with our goal of creating a solution adaptable to various residential care settings.

This design choice does present certain limitations that warrant acknowledgment:

- Some region-specific or specialized food components may not be represented in the FNDDS, potentially reducing accuracy for certain cultural or institutional cuisines.
- Preparation and cooking methods can significantly alter nutrient profiles, which may not be fully captured in standardized database values.

Despite these limitations, the comprehensive nature of FNDDS and its regular updates make it a reliable foundation for nutritional analysis in our system.

5.5.2 Nutrient Retrieval Pipeline

The nutrient retrieval works through a systematic four-step process:

1. **Component Identification and Weight Estimation:** The computer vision pipeline processes before/after meal images to identify food components (e.g., starch, vegetables, meat/fish, sauce) and estimates their consumed weight in grams. This step is discussed in the previous sections.
2. **Menu Mapping:** Each identified component is mapped to its corresponding food item in the institutional menu through a lookup function. This requires the menu to be available in a structured JSON format with standardized nomenclature to facilitate matching. An example of the menu format is visualized in Figure 5.13. Per component we need the timestamp, the component ID and the name.
3. **Database Query:** A GET request is sent to the FNDDS API with the food item name as a query parameter. This may return multiple potential matches, from which the system automatically selects the highest-ranked match (first in the returned list) as it represents the closest semantic alignment with the query term.
4. **Nutrient Calculation:** The module calculates the scaled nutrient values based on the consumed portion weight, using the following formula:

$$N_{consumed} = N_{reference} \times \frac{W_{consumed}}{W_{reference}}$$

5 Architectural Design and System Development

Where $N_{consumed}$ is the consumed nutrient value, $N_{reference}$ is the nutrient value per reference quantity, $W_{consumed}$ is the consumed weight, and $W_{reference}$ is the reference quantity specified in the database. This approach accounts for variable reference quantities rather than assuming a standard 100 g baseline, providing more accurate nutritional assessments.

```
1 [
2   {
3     "timestamp": "2025-04-06T12:00:00Z",
4     "component_id": 0,
5     "component_item_name": "Brown Rice"
6   },
7   {
8     "timestamp": "2025-04-06T12:00:00Z",
9     "component_id": 1,
10    "component_item_name": "Steamed Bok Choy"
11  },
12  {
13    "timestamp": "2025-04-06T12:00:00Z",
14    "component_id": 2,
15    "component_item_name": "Teriyaki Chicken"
16  }
17 ]
```

Figure 5.13: Example JSON representation of a meal menu.

5.5.3 Nutritional Parameters

The system currently tracks the nutrients listed in Table 5.4. These nutrients were selected to maintain compatibility with the ILVO database, ensuring consistency in comparative analyses. The system architecture allows for easy expansion to include additional nutrients if required for specific dietary monitoring purposes.

5.5.4 Current Limitations and Future Improvements

Currently, no fallback mechanism is implemented if a food item is not found in the database. This represents an opportunity for future enhancement, as discussed in the Future Work chapter. We chose not to implement a fallback mechanism as it falls outside the main focus of this thesis, which prioritizes accurate segmentation, classification, and weight estimation of food components.

5 Architectural Design and System Development

Table 5.4: List of tracked nutrients and their units.

Nutrient	Unit
Energy	kcal
Protein	g
Carbohydrates	g
Sugars	g
Fat	g
Saturated fat	g
Fiber	g
Sodium	mg
Calcium	mg
Vitamin D	µg

5.6 Data Querying and Storage

After determining the nutrients of both the served meal and the leftovers through the computer vision pipeline, we store this information in the Obelisk Core data platform. Obelisk Core serves as a robust service platform designed for building scalable data-driven applications, providing a comprehensive API for ingesting, querying, and streaming large volumes of data in an organized and secure manner [90]. For the current implementation, we have deployed Obelisk Core locally using Docker containers. However, transitioning to a SaaS implementation would be advantageous if Obelisk Core becomes available as such in the future.

For each component, we store both the nutrient content and weights related to the served meal and the corresponding leftovers. This dual-point data collection strategy allows us to precisely calculate what each resident has consumed and quantify food waste produced by the residential care center. We assume that, for each food component and resident per meal, the first data point corresponds to the nutrient content and weight of the served meal, while the second data point corresponds to the nutrient content and weight of the leftovers.

5.6.1 Data Organization

Before being able to ingest and query data, we need to think out how we will store the data. First we need to determine which data we will store, these are the metrics. Additionally we can store some meta-data which allows us to filter and group the data points, these are called the labels. We create a dataset per residential care center.

5 Architectural Design and System Development

Metrics

Table 5.5 presents the metrics stored within our system. These metric types capture different dimensions of the food analysis process, from basic a weight measurement to detailed nutritional components. This storage structure enables efficient retrieval for both individual patient nutrition tracking and facility-wide food waste monitoring.

Table 5.5: List of metrics stored in the Obelisk Core data platform.

Metric	Type	Description
<code>contents.weight_g</code>	Number	Weight in grams
<code>nutrients.protein_g</code>	Number	Protein content in grams
<code>nutrients.fat_g</code>	Number	Fat content in grams
<code>nutrients.saturated_fat_g</code>	Number	Saturated fat content in grams
<code>nutrients.sugar_g</code>	Number	Sugar content in grams
<code>nutrients.fiber_g</code>	Number	Fiber content in grams
<code>nutrients.calories_kcal</code>	Number	Caloric content in kcal
<code>nutrients.carbohydrates_g</code>	Number	Carbohydrate content in grams
<code>nutrients.sodium_mg</code>	Number	Sodium content in milligrams
<code>nutrients.calcium_mg</code>	Number	Calcium content in milligrams
<code>nutrients.vitamin_d_ug</code>	Number	Vitamin D content in micrograms

Label Structure

To facilitate efficient data querying while mitigating high cardinality issues, we implemented the following label structure:

- Resident ID: Uniquely identifies the resident, enabling efficient retrieval of individual-specific data.
- Component ID: Identifies the specific food category, allowing to know which component the measurements are about.

This structure enables straightforward filtering and aggregation of data at both the individual resident level and the component type level, providing flexibility for various analytical purposes.

5.6.2 Data Ingestion

To ensure reliable data ingestion into Obelisk Core, we implemented a robust ingestion framework with built-in error handling capabilities. Data from the meal analysis system is transmitted to Obelisk Core through its

5 Architectural Design and System Development

REST API. To address potential network failures or service unavailability, we implemented a comprehensive retry mechanism. The implementation adheres to the principles of the decorator design pattern, where the retry functionality encapsulates the base data sending operations without modifying their internal structure. This pattern enhances the system's resilience while maintaining clean separation of concerns in the codebase, resulting in more maintainable software.

This retry mechanism offers several advantages, enhancing robustness through:

- The implementation of exponential backoff to prevent service overload during recovery periods.
- Configurable maximum retry attempts and timeouts, allowing adaptation to diverse operational conditions.

5.6.3 Data Querying

We implemented a set of query methods to support various analytical needs. Each query method follows a consistent pattern while varying the specific filter expressions, fields, ordering, and aggregation methods based on the analytical purpose. Similar to the ingestion pipeline, the query system also implements the retry mechanism using the decorator design pattern.

Query Methods

All query methods construct API requests to the endpoint:

```
{base_url}/datasets/{dataset_id}/data/query
```

The request can include the following parameters:

- `filter`: RESTful Service Query Language (RSQL) expression to narrow down the result set.
- `fields`: Specific data field(s) to retrieve.
- `orderBy`: Field(s) by which to sort results.
- `groupBy`: Field(s) by which to group results.
- `aggregator`: Statistical function to apply to the grouped data.
- `dataType`: Data type of the metric values to query. As all metrics are numbers in our case, this parameter has no use.
- `limit`: The maximum number of items to return per page.

5 Architectural Design and System Development

Table 5.6 provides an overview of the query methods implemented to support the dashboard functionality. Unless otherwise specified, default values are used for the parameters. For the limit parameter, we use a value of 60,000 to ensure comprehensive data retrieval while maintaining optimal performance. For all methods we request all fields, except for the `get_all_resident_ids` method, there we only request the labels.

Table 5.6: Summary of implemented Obelisk Core query methods.

Method	Purpose	Filter Expression
<code>get_all_resident_ids</code>	Get all resident IDs	N/A
<code>get_eaten_weight_between_dates_per_resident</code>	Get the weight of all components per resident in a specific period	<code>labels.resident_id=="{resident_id}" and labels.food_component_id=="{component_id}" and timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric=="contents.weight_g::number"</code>
<code>get_eaten_nutrients_between_dates_per_resident</code>	Get all nutrients of all components per resident in a specific period	<code>labels.resident_id=="{resident_id}" and timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric!="contents.weight_g::number"</code>
<code>get_eaten_nutrients_between_dates_all_residents</code>	Get all nutrients of all components of all residents in a specific period	<code>timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric!="contents.weight_g::number"</code>
<code>get_food_waste_between_dates</code>	Get food waste in a residential care center in a specific period	<code>timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric=="contents.weight_g::number"</code>
<code>get_comprehensive_waste_data</code>	Get information for the food waste KPIs in a specific period	<code>timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric=="contents.weight_g::number"</code>

Continued on next page

5 Architectural Design and System Development

Table 5.6 (continued) – Summary of implemented Obelisk Core query methods.

Method	Purpose	Filter Expression
get_food_waste_breakdown	Get the food waste data per component and per moment in a specific period	timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric=="contents.weight_g::number"
get_favorite_and_worst_menu_items	Get the favorite and least favorite menu items in a specific period	timestamp>="{start_date_iso}" and timestamp<="{end_date_iso}" and metric=="contents.weight_g::number"

Pagination Support

All query methods implement a pagination mechanism to handle large datasets efficiently. We use the template method design pattern to standardize the pagination logic across all query methods while allowing the data processing logic to vary according to each method's specific requirements. Through empirical testing, we determined an optimal page size of 60,000 records. This implementation ensures that no data points are inadvertently dropped when retrieving large result sets.

Parallelization Strategy

To further optimize data retrieval performance, we implemented a parallelization strategy that segments date ranges into multiple sub-ranges, enabling concurrent execution of queries. This approach reduces the overall query time for extensive date ranges. The implementation dynamically determines the optimal number of parallel requests based on the date range size and system capabilities, ensuring efficient resource utilization without overwhelming the server resources.

Data Processing

After retrieving data from the Obelisk Core data platform, each query method applies specific processing logic to transform raw data into insights. For example, we consistently differentiate between the first data point (corresponding to the served meal) and the second data point (corresponding to the leftovers). For food waste analysis, we aggregate all leftover weights, applying appropriate transformations to generate meaningful metrics.

5 Architectural Design and System Development

5.7 User Interface: Dashboard

The final element of the LeftoVision system architecture is the dashboard, which serves as the interface between the system and residential care personnel. This section details the dashboard's design, implementation decisions, and functionality, illustrating how it transforms the nutritional and consumption data into actionable insights.

5.7.1 Technology Selection

The selection of an appropriate dashboard technology presented a critical decision point in the development process. While various frameworks were considered, the final choice was made based on both functional requirements and project constraints. Table 5.7 outlines the comparative analysis between Grafana [91] and Plotly Dash [87], the two primary candidates.

Table 5.7: Comparative analysis of Grafana and Plotly Dash for dashboard implementation.

Feature	Grafana [91]	Plotly Dash [87]
Learning Curve	Steeper learning curve with more complex configuration options	Gentler learning curve for those already familiar with Python and Plotly
Integration with Existing Infrastructure	More challenging Docker setup	Seamless integration with Python ecosystem
Alerting Capabilities	Robust built-in alerting system	Limited alerting capabilities, requires additional implementation
Real-time Update Mechanisms	Native support for real-time monitoring	Requires implementation of callback mechanisms
Development Speed	Slower initial setup but faster dashboard creation once configured	Faster initial prototype but may require more code for complex features
Production Suitability	Extensively adopted in industry production environments with robust security and authentication features	Suitable for controlled production environments, more commonly used in research and prototype contexts

Although Grafana offered superior built-in monitoring capabilities and robust alerting features, Plotly Dash was ultimately selected due to two primary factors: (1) the time constraints of the thesis project, and (2) the seamless integration with our existing Python-based pipeline. This decision prioritized rapid development and integration over long-term operational features. For production environments, a migration to Grafana would be advisable to leverage its superior monitoring capabilities and alerting system.

5 Architectural Design and System Development

5.7.2 Dashboard Design

The dashboard implementation follows a modular architecture with consistent structural patterns across functional areas. This consistency enhances user familiarity and reduces cognitive load when navigating between different analytical perspectives. The architectural design incorporates reactive principles through Dash callbacks, enabling dynamic content updates in response to user interactions without requiring page reloads.

Global Parameter Controls

At the top level, the dashboard presents global parameter controls affecting all visualizations, including:

- Care center selection: Focusing analysis on a specific institutional unit.
- Date range specification: Defining the temporal scope of the displayed data.

These global filters establish consistent parameters across the interface. Parameter modifications trigger coordinated callbacks, updating all visualizations simultaneously to maintain data coherence.

Functional Organization

The dashboard content is organized into three primary functional tabs, each addressing a distinct analytical perspective relevant to residential care facility operations:

1. Resident Nutrition: Focused on nutritional monitoring.
2. Food Waste Analysis: Addressing institutional waste reduction.
3. Menu Performance Insights: Supporting menu optimization decisions.

This functional separation aligns with the different goals of LeftoVision. To provide a comprehensive understanding of the dashboard functionality, a demonstration video is available here ¹. Furthermore, interactive tooltips are provided to enhance the interpretability of the visualizations, offering explanations on how to interpret plots without cluttering the visual display.

5.7.3 Resident Nutrition Monitoring Interface

The Resident Nutrition tab implements a hierarchical information design, progressing from population-level nutritional overview to detailed individual nutritional analysis. This design supports the clinical workflow, where staff first identify at-risk individuals before performing more detailed nutritional assessments. A part of the tab is visualized in Figure 5.14.

¹https://github.com/AxellePenninger/thesis-food-leftovers/blob/master/dashboard_demo.mkv

5 Architectural Design and System Development

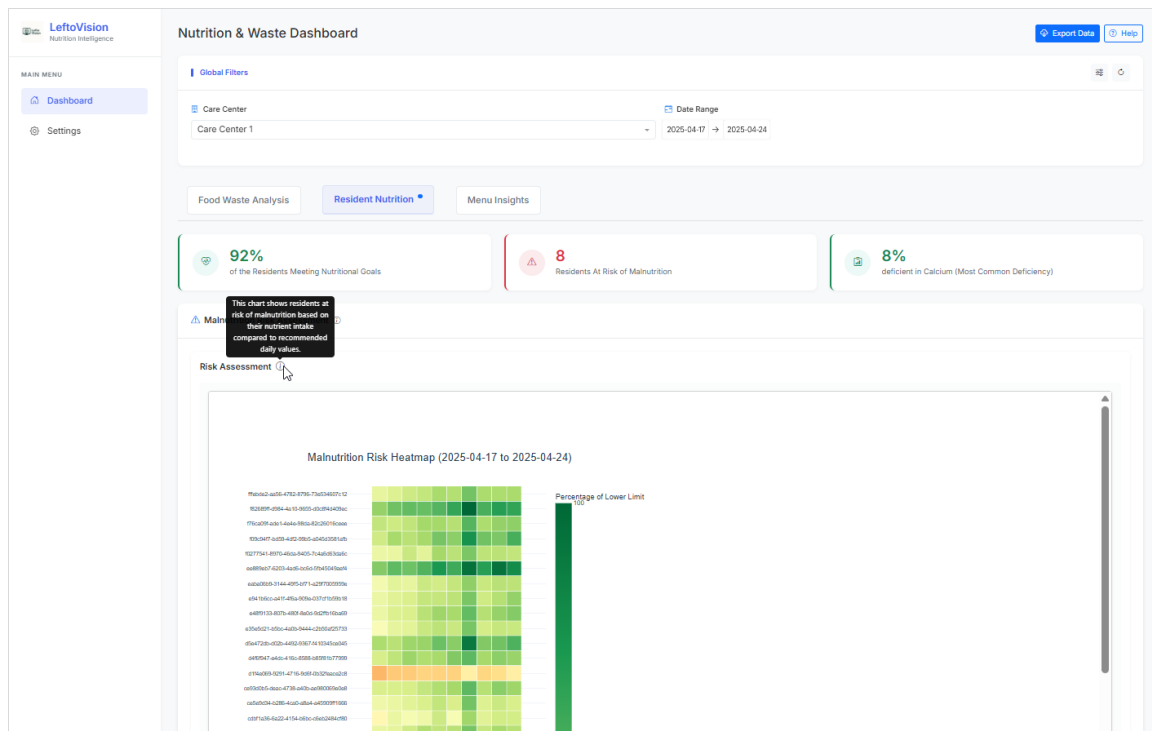


Figure 5.14: Screenshot of the Resident Nutrition tab in the LeftoVision dashboard.

Key Performance Indicators

Three primary numerical indicators provide immediate situational awareness regarding the facility's nutritional status:

1. Nutritional adherence rate: Percentage of residents achieving daily nutrient reference targets.
2. Malnutrition risk count: Number of residents exhibiting consistent nutritional deficiencies.
3. Critical deficiency identifier: The most prevalent nutrient deficiency across the resident population.

Each indicator uses conditional color coding (red-amber-green) to convey severity through pre-attentive visual processing, enabling rapid assessment by healthcare personnel. This color-coding system is applied consistently throughout the dashboard to maintain visual language coherence.

Population Nutritional Assessment

Following the summary KPIs, a comprehensive nutritional heatmap visualizes the entire resident population against the nutritional parameters. This visualization maps individual residents (rows) against their nutrient intake percentages relative to dietary reference values (columns). Color intensity signifies the degree of nutritional adequacy, with deeper red tones indicating more severe deficiencies.

5 Architectural Design and System Development

This visualization leverages heatmap pattern recognition to facilitate rapid identification of:

1. Individual residents experiencing nutritional gaps across multiple nutrients.
2. Specific nutrients consistently deficient across the resident population.

Individual Resident Analysis

The individual analysis section enables a deeper investigation of specific residents selected through a drop-down interface. Upon resident selection, two synchronized visualizations are populated:

1. Consumption Time Series: A dual-representation time series displaying component weights consumed over time, with discrete values represented as points, and a three-day rolling average as a continuous line. This visualization helps to identify changes in the consumption pattern that could indicate the development of health problems.
2. Nutrient Intake Assessment: A comparative bar visualization juxtaposing the resident's actual nutrient intake against reference ranges (establishing both minimum and maximum recommended values). Each nutrient bar uses conditional coloring to indicate deficient, optimal, or excessive intake levels.

5.7.4 Food Waste Analysis Interface

The Food Waste tab presents analytical tools designed to identify waste patterns and support waste reduction initiatives. The interface progresses from aggregate metrics to an increasingly granular analysis of specific waste contributors. A part of the food waste analysis tab is visualized in Figure 5.15.

Key Performance Indicators

Three centralized KPIs establish the context for waste analysis:

1. Aggregate waste volume: Total food waste in kilograms during the selected period.
2. Trend indicator: Directional pattern of waste production (increasing, stable, or decreasing).
3. Consumption efficiency: Average percentage of served food that remains unconsumed.

These indicators are designed to provide immediate feedback on the effectiveness of waste reduction initiatives, with color coding indicating performance relative to the performance of the past weeks.

5 Architectural Design and System Development

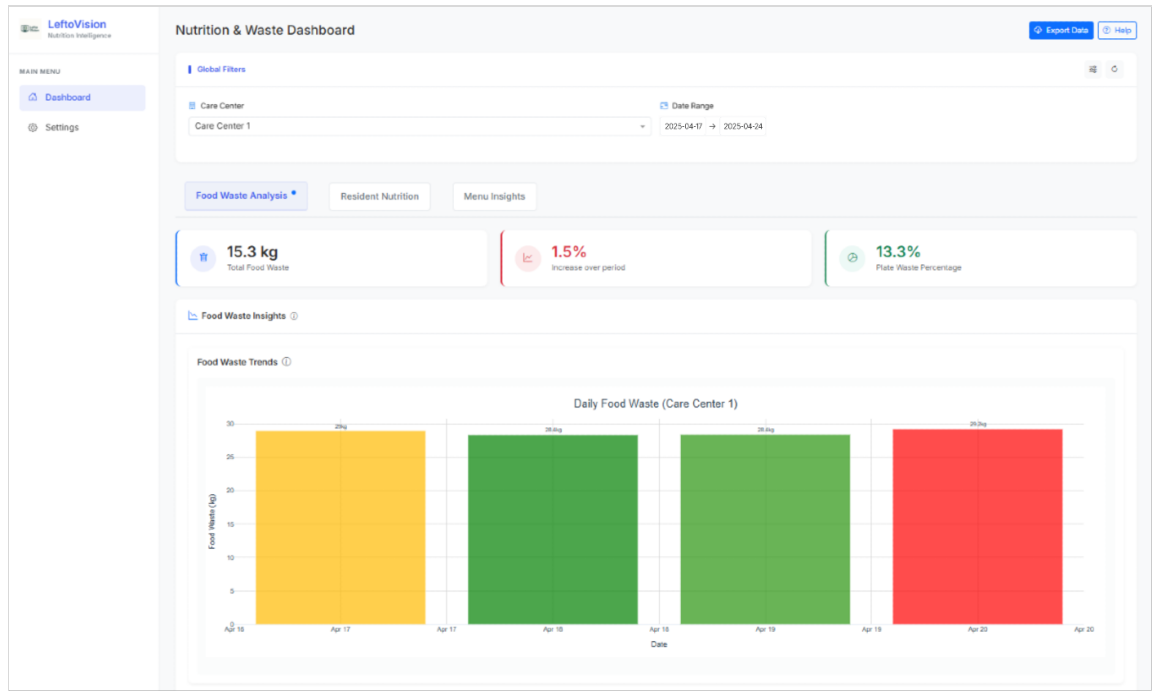


Figure 5.15: Screenshot of the Food Waste Analysis tab in the LeftoVision dashboard.

Temporal and Categorical Waste Analysis

The detailed waste analysis section presents multiple analytical perspectives:

1. **Temporal Analysis:** A configurable bar chart displaying waste distribution across time intervals, with conditional coloring based on the waste of the other days. The x-axis parameter can be modified to analyze different temporal granularities, enabling pattern identification at various scales.
2. **Meal Context Analysis:** A proportional pie chart visualizing waste distribution across meal contexts (breakfast, lunch, dinner), helping to identify specific meal phases that contribute disproportionately to overall waste.
3. **Category Analysis:** A categorical bar chart representing waste volumes by food category, enabling targeted interventions for categories with significant waste due to serving portions exceeding actual consumption.

5.7.5 Menu Insights Interface

The Menu Insights tab provides analytical tools specifically designed to support menu optimization decisions, helping kitchen staff align meal offerings with resident preferences. The menu insights tab is visualized in Figure 5.16.

5 Architectural Design and System Development

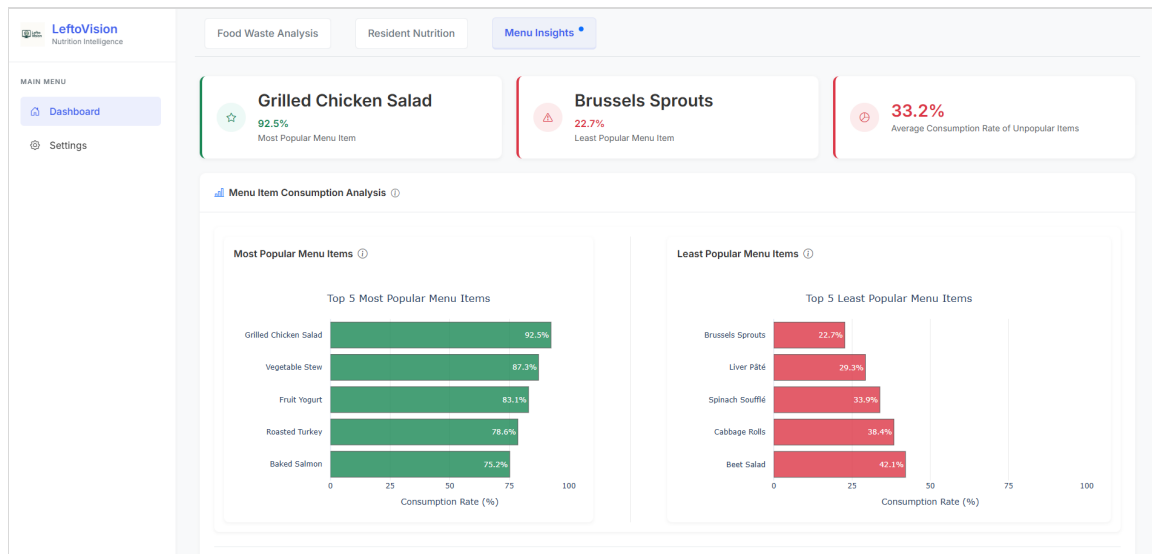


Figure 5.16: Screenshot of the Menu Insights tab in the LeftoVision dashboard.

Key Performance Indicators

The menu performance section begins with three KPIs:

1. Acceptance indicator: The most consumed food item with its average consumption rate.
2. Rejection indicator: The least consumed food item with its average consumption rate.
3. Low performance average: Mean consumption rate across the five least consumed menu items.

These performance metrics spotlight the spectrum of menu item popularity, identifying both standout successes and potential opportunities for recipe refinement or replacement.

Comparative Menu Item Analysis

Two complementary bar visualizations provide deeper insights into the menu performance:

1. Preferred Items Analysis: A ranked visualization of the five most consumed menu items, with consumption rates indicated as percentages.
2. Problematic Items Analysis: A ranked visualization of the five least consumed menu items, highlighting potential candidates for recipe modification or menu rotation.

This comparative analysis supports evidence-based menu planning decisions.

6

Evaluation and Results

This chapter presents a systematic evaluation of LeftoVision's performance in estimating food weights within residential care environments. We establish a comprehensive framework that assesses both individual components and the end-to-end system by testing the three pipeline configurations. The evaluation combines quantitative metrics with qualitative analyses to provide a holistic understanding of system capabilities, while addressing computational efficiency considerations important for practical deployment in residential care settings.

6.1 Evaluation Framework

6.1.1 Evaluation Strategy and Metrics

Our evaluation uses a hierarchical approach that examines both the individual components and the performance of the entire system. At the component level, we independently evaluate the segmentation and weight estimation modules using ground-truth inputs. For the end-to-end assessment, we analyze the three distinct pipelines in their ability to estimate food weights from raw imagery.

For segmentation performance, we use the following complementary metrics:

- **mIoU and Dice coefficient:** These overlap-based metrics capture the spatial agreement between the predicted and ground-truth segmentation masks. We use the Dice coefficient because it is less severe in penalizing under- and over-segmentation. The mIoU is included to facilitate comparison with the existing literature.
- **Precision:** Focuses on the quality of positive predictions.
- **Recall:** Focuses on the completeness of detection.
- **F1-score:** Represents the harmonic mean of precision and recall, offering a balanced evaluation.

These metrics provide a complete assessment of both classification accuracy and region precision. Results are

6 Evaluation and Results

reported globally and per food category to identify potential category-specific biases. In addition, we examine example outputs of the segmentation model to offer a visual understanding of the segmentation quality and capture nuanced performance characteristics, complementing the numerical results.

For examining the weight estimation performance, we use:

- **MAE:** Provides intuitive error interpretation in grams.
- **RMSE:** Emphasizes larger deviations, highlighting problematic predictions.
- **CCC:** Measures agreement between predicted and actual weights, accounting for both systematic bias and random variability.

We deliberately exclude MAPE despite its prevalence in literature, as it disproportionately penalizes errors when true values are small—a common scenario in leftover analysis. Additionally, our application prioritizes absolute rather than relative error.

We supplement numerical metrics with visualization methods, including prediction-vs-actual scatter plots, error distribution histograms, and density comparison plots to provide deeper insights into estimation patterns.

For end-to-end system evaluation, we compare predicted weights against ground truth using MAE, RMSE, and CCC, benchmarking against a mean-based predictor to establish a minimum performance threshold. We supplement numerical metrics with visualization methods, including a prediction-vs-actual scatter plot, an error distribution histogram, and the cumulative distribution of absolute weight estimation errors.

6.1.2 Computational Performance Measurement Methodology

To capture accurate computational performance metrics, we implemented a decorator design pattern that enables systematic monitoring without modifying the underlying implementation. This approach measures:

- **Execution time:** Wall clock time for each main operation.
- **CPU Memory usage:** Peak CPU memory consumption during execution.
- **GPU Memory usage:** Peak GPU memory consumption during execution for components leveraging GPU acceleration.
- **CPU utilization:** Peak CPU load.
- **GPU utilization:** Peak GPU load for components leveraging GPU acceleration.

Performance data were collected across fifteen runs to account for system variability, with both mean and standard deviation reported.

6 Evaluation and Results

6.1.3 Dataset Organization

To evaluate real-world generalization capability, we implement a strict institution-based split strategy. Each residential care center’s images are assigned exclusively to one subset—training, validation, or testing. This approach simulates deployment to unseen environments—a critical consideration for practical applications. Further details on this data splitting approach are provided in Section 4.4.

Each component and the complete pipeline are evaluated on their respective test subsets using configurations identified as optimal during validation, as detailed in Chapter 5. This approach avoids leakage.

6.2 Component-level Evaluation

This section evaluates the performance of the two core components—segmentation and weight estimation—independently before their integration into the complete system. This isolated analysis helps to identify the strengths and limitations of each module.

6.2.1 Segmentation Module

This section presents the systematic evaluation methodology of the segmentation module, beginning with hyperparameter optimization studies that informed our architectural decisions, followed by comprehensive testing on unseen data, qualitative analysis, and an assessment of computational efficiency.

Hyperparameter Optimization Studies

We conducted systematic experiments to determine optimal configurations for the segmentation module, investigating the impact of model architecture, backbone network, input resolution, class-weighted loss function and data augmentation techniques. For each set of experiments detailed in the following tables, the parameter(s) under investigation were varied while all other parameters were held constant.

Table 6.1 compares various segmentation architectures and backbones on the validation set. While DeepLabV3+ with ResNet-101 achieved marginally higher performance, the ResNeXt-50 backbone offered a compelling efficiency-performance balance with 43 % fewer parameters. This significant parameter reduction not only accelerates inference, but potentially enhances generalization by limiting overfitting risk. The consistent performance patterns across architectures—with all models struggling on sauce components while performing well on starch—suggest that these performance differences reflect inherent visual properties of the food categories rather than architecture-specific limitations.

Our investigation of input image resolution (Table 6.2) revealed that the 512×512 pixel configuration provided the optimal balance between performance and computational cost, yielding the highest mIoU (0.48) and micro-

6 Evaluation and Results

Table 6.1: Segmentation performance comparison of architectures and backbones on the validation set.

Architecture	Backbone	Parameters (M)	mIoU	mDice	mPrecision	mRecall	mF1	Class-wise Dice Coefficient				
								Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
DeepLabV3	ResNeXt-50 32x4d	39.11	0.45	0.55	0.60	0.55	0.55	0.60	0.60	0.66	0.10	0.37
DeepLabV3+	ResNeXt-50 32x4d	26.15	0.48	0.58	0.58	0.55	0.55	0.70	0.57	0.67	0.15	0.38
SegFormer	ResNeXt-50 32x4d	24.31	0.40	0.49	0.58	0.48	0.49	0.67	0.52	0.64	0.01	0.11
DeepLabV3+	ResNet-18	12.33	0.42	0.51	0.59	0.51	0.51	0.66	0.60	0.67	0.04	0.10
DeepLabV3+	ResNet-34	22.44	0.41	0.49	0.50	0.49	0.49	0.62	0.62	0.67	0.05	0.00
DeepLabV3+	ResNet-50	26.68	0.40	0.50	0.57	0.47	0.50	0.75	0.47	0.57	0.12	0.11
DeepLabV3+	ResNet-101	45.67	0.49	0.59	0.61	0.60	0.60	0.66	0.63	0.64	0.19	0.46
DeepLabV3+	MobileNet	4.38	0.40	0.49	0.54	0.48	0.49	0.63	0.59	0.61	0.03	0.12
DeepLabV3+	MITB2	25.35	0.45	0.55	0.62	0.53	0.55	0.67	0.56	0.71	0.20	0.18

Note: mIoU, mDice, mPrecision, mRecall, and mF1 refer to micro-averaged metrics computed across all classes.

averaged Dice (0.58). Despite the 1024×1024 resolution offering marginally better micro-averaged recall, these modest gains required a $4 \times$ increase in computational resources while actually reducing mIoU. The 256×256 resolution demonstrated consistently lower performance. The selected mid-resolution configuration thus provides sufficient spatial detail to distinguish food categories while maintaining computational efficiency appropriate for practical deployment.

Table 6.2: Impact of input resolution on segmentation performance on the validation set.

Input Size	FLOPS (B)	mIoU	mDice	mPrecision	mRecall	mF1	Class-wise Dice Coefficient				
							Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
256×256	585.48	0.41	0.51	0.55	0.51	0.51	0.60	0.58	0.56	0.22	0.14
512×512	2341.80	0.48	0.58	0.58	0.55	0.55	0.70	0.57	0.67	0.15	0.38
1024×1024	9367.19	0.46	0.56	0.59	0.58	0.56	0.68	0.59	0.63	0.10	0.36

Note: mIoU, mDice, mPrecision, mRecall, and mF1 refer to micro-averaged metrics computed across all classes.

Synthetic data augmentation (Table 6.3) yielded marginal improvements in overall performance metrics. Class-wise analysis revealed enhanced performance for starch and vegetables, with a slight reduction for meat/fish and sauce. We adopted the augmented model for deployment primarily for its improved generalization capabilities and robustness against visual variations.

Table 6.3: Impact of synthetic data augmentation on segmentation performance on the validation set.

Data	mIoU	mDice	mPrecision	mRecall	mF1	Class-wise Dice Coefficient				
						Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
Original	0.48	0.58	0.58	0.55	0.55	0.70	0.57	0.67	0.15	0.38
Original + Augmented	0.49	0.58	0.61	0.59	0.58	0.72	0.63	0.64	0.13	0.38

Note: mIoU, mDice, mPrecision, mRecall, and mF1 refer to micro-averaged metrics computed across all classes.

To take into account the fact that there are much less images of sauce and other sauce components, we can

6 Evaluation and Results

use a class-weighted loss function where we increase the weight of the sauce and other sauce component. The results of different weights are visualized in Table 6.4. Increasing the weights for underrepresented sauce categories revealed a clear performance tradeoff—modest weights ([1, 1, 1, 1, 4, 4]) significantly improved sauce detection (0.28 vs 0.15 Dice) while maintaining strong performance on majority classes. However, excessive weighting ([1, 1, 1, 1, 8, 8]) began sacrificing overall performance as the model overprioritized sauce detection at the expense of more common food components. For this reason, a weight configuration of [1, 1, 1, 1, 4, 4] is selected, where the last two weights correspond to the sauce and other sauce classes, respectively.

Table 6.4: Impact of different class weights on segmentation performance on the validation set.

Class weights	mIoU	mDice	mPrecision	mRecall	mF1	Class-wise Dice Coefficient				
						Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
[1, 1, 1, 1, 1, 1]	0.48	0.58	0.58	0.55	0.55	0.70	0.57	0.67	0.15	0.38
[1, 1, 1, 1, 2, 2]	0.48	0.59	0.60	0.60	0.59	0.65	0.56	0.66	0.14	0.53
[1, 1, 1, 1, 4, 4]	0.50	0.61	0.64	0.65	0.61	0.65	0.63	0.68	0.28	0.43
[1, 1, 1, 1, 8, 8]	0.46	0.57	0.69	0.60	0.57	0.59	0.54	0.61	0.32	0.38

Note: mIoU, mDice, mPrecision, mRecall, and mF1 refer to micro-averaged metrics computed across all classes.

Test Set Performance

Based on validation results, we retrained the model with the selected hyperparameters for final evaluation. We retrained this configuration on both training and validation sets with synthetic data augmentation. Table 6.5 presents the test set performance across all food categories, providing insights into expected model performance in new care settings.

Table 6.5: Segmentation performance metrics on the test set per food category.

Class	IoU	Dice	Precision	Recall	F1-Score
Starch	0.61	0.76	0.73	0.79	0.76
Vegetables	0.30	0.45	0.66	0.35	0.45
Meat/Fish	0.43	0.59	0.61	0.58	0.59
Sauce	0.17	0.27	0.29	0.28	0.27
Other Sauce	0.10	0.14	0.25	0.10	0.14
Micro-average	0.43	0.53	0.59	0.51	0.53

There is a notable performance gap between the validation and the test set. This discrepancy likely stems from our center-based data split strategy, where food presentation distributions differ between training/validation

6 Evaluation and Results

centers and test centers. Vegetables show a dramatic performance decline. The low recall value (0.35) indicates our model frequently fails to identify vegetable components. This challenge likely stems from the vegetable category's exceptional diversity. With numerous vegetable types, the test centers likely contain many vegetable varieties not seen during training, making them particularly difficult for the model to recognize.

When we compare our model with the state-of-the-art models, discussed in Table 3.2, we can conclude our segmentation model performs slightly worse. Although this is not a fair comparison as the models are trained and evaluated on different datasets.

The confusion matrix in Figure 6.1 provides deeper insight into classification behavior:

- **Background misclassification bias:** A significant share of food pixels (14–34 %) are wrongly classified as background, explaining their poor IoU scores. This may stem from inconsistent or missing labeling of sauces (see Section 4.3.5). These misclassifications often occur near food boundaries, as discussed in the qualitative analysis.
- **Sauce ambiguity:** Sauce is often confused with meat/fish (13 %), while other sauce is frequently misclassified as starch (28 %). This probably results from the inconsistent annotation strategy (see Section 4.3.5).
- **Vegetable misclassification:** Vegetables are frequently misclassified as starch (19 %) and meat/fish (16 %), resulting in the worst performance among all food classes.

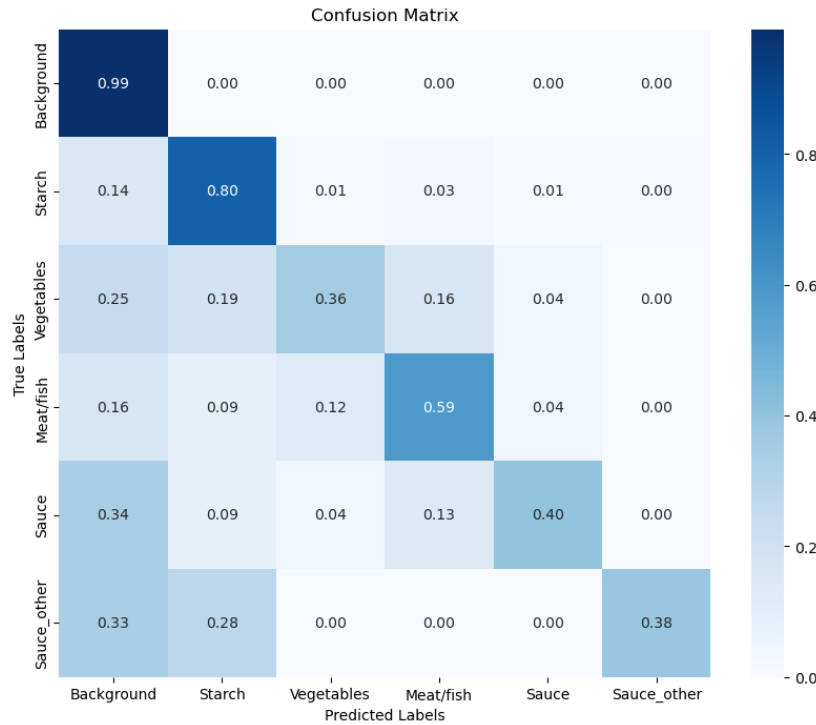


Figure 6.1: Confusion matrix for the segmentation module on the test set.

6 Evaluation and Results

Qualitative Analysis

Figure 6.8 presents qualitative segmentation results across sixteen random samples. Each row displays the original image, ground-truth annotation, model prediction, and an error visualization map. The error visualization uses a color-coded scheme to distinguish between misclassifications between food categories (red), false positives where background is incorrectly identified as food (orange), and false negatives where food regions remain undetected (blue).

Our qualitative analysis reveals heterogeneous performance across different food categories and conditions:

- **Lighting sensitivity:** The model exhibits vulnerability to lighting variations, with samples #3-4 showing misclassifications due to overexposure.
- **Sauce-meat ambiguity:** In samples #5-6, the ground truth contains meat labels while the model predicts sauce. Though not technically incorrect, this discrepancy penalizes metrics due to annotation strategy inconsistencies.
- **Boundary detection challenges:** Error visualizations highlight regions of algorithmic uncertainty, with false negatives and false positives predominating at food boundaries.
- **Misclassifications:** Persistent misclassifications across samples could significantly affect nutritional intake analysis.

These identified challenges inform potential avenues for methodological refinement to enhance segmentation robustness in practical deployment scenarios.

Computational Efficiency

Training and inference were conducted on the IDLab GPU Lab infrastructure using an NVIDIA GeForce RTX 4090 GPU with 23 GB memory and 2 CPUs. Table 6.6 summarizes computational performance, demonstrating efficient inference times (16 ms per single image, 9.4 ms per image in batches). This efficiency enhances the system's practical usability in care facility environments despite not being strictly necessary for post-meal nutritional analysis.

6.2.2 Weight Estimation Module

This section evaluates the weight estimation module, beginning with hyperparameter optimization studies that informed architectural decisions, followed by comprehensive testing on unseen data, error analysis, and computational efficiency assessment. It is important to note that all performance metrics reported herein are calculated on a per-component basis, as the input to the weight estimation module consists of segmented images isolating individual food components rather than whole-plate compositions.

6 Evaluation and Results

Table 6.6: Summary of computational performance metrics for the segmentation model.

Category	Metric	Measured Value
Training Time	Training set only	62 min 29 s
	Training + validation sets	72 min 34 s
Inference Speed	Single image	16 ms
	Batch of 32 images (total)	300 ms
	Per image in batch	9.4 ms
Resource Utilization (Training)	Peak CPU memory	15.9 GB
	Peak GPU memory	18.2 GB
	Peak CPU utilization	88.2 %
	Peak GPU utilization	45.8 %

Hyperparameter Optimization Studies

We conducted systematic experiments to determine optimal configurations for the weight estimation model, investigating model architecture selection, transfer learning, and input resolution.

Table 6.7 compares various architectures on the validation set after course hyperparameter tuning. ResNet-50 achieved the lowest validation RMSE of 46.76 g, outperforming more complex models like AlexNet (50.17 g RMSE) and Wide ResNet-50 (51.95 g RMSE). This suggests that additional parameters in larger architectures may lead to overfitting on this specific task. The relatively lightweight ResNet-50 offers an optimal balance between model capacity and generalization ability for food weight estimation. The relatively lower parameter count of ResNet-50 also offered advantages in terms of training speed and inference efficiency, making it more suitable for potential deployment in residential care facilities where computational resources may be limited.

Table 6.7: Weight estimation performance comparison of architectures on the validation set (course tuning).

Architecture	Parameters (M)	RMSE (g)	MAE (g)	CCC	Class-specific RMSE (g)				
					Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
ResNet-50	25.56	46.76	33.47	0.29	47.27	44.89	65.57	43.37	50.21
ResNet-101	44.59	59.63	41.52	0.17	61.00	42.28	74.67	43.67	62.25
AlexNet	61.10	50.17	34.03	0.15	40.78	34.79	66.16	52.01	51.02
DenseNet	7.98	80.89	64.09	0.03	84.93	64.58	97.92	56.17	74.53
Wide ResNet-50	68.88	51.95	34.31	0.28	46.78	42.82	64.34	50.18	45.95
GoogLeNet	6.62	81.27	63.85	0.02	84.75	64.38	99.37	54.36	76.12

Our examination of transfer learning approaches (Table 6.8) revealed that fine-tuning all layers significantly

6 Evaluation and Results

outperformed the approach of only retraining the classification head (42.62 g vs. 78.37 g RMSE). This substantial difference indicates that domain-specific features are essential for accurate weight estimation from food images, reflecting the specialized nature of this task relative to general image classification. Training from scratch, while showing better performance, often requires more data and computational resources.

Table 6.8: Comparison of transfer learning versus training from scratch for weight estimation on the validation set (fine tuning).

Approach	RMSE (g)	MAE (g)	CCC	Class-specific RMSE (g)				
				Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
From Scratch	42.62	36.33	0.29	44.46	62.96	48.80	40.06	73.37
Pre-trained	78.37	64.59	0.00	76.95	72.58	86.87	70.25	46.13

Our investigation of input resolution effects (Table 6.9) demonstrated that using 224×224 pixels—the standard resolution for pre-trained networks—provides an effective balance between performance and computational cost. While larger input (350×350) showed a slight improvement in performance metrics, this marginal gain comes with significantly higher computational cost (645.72 vs. 261.58 BFLOPS). The smaller resolution (100×100) performed reasonably well but with lower metrics across all categories. Based on this analysis, we selected the 224×224 resolution as the optimal configuration, balancing accuracy with efficiency.

Table 6.9: Impact of input resolution on weight estimation performance on the validation set (fine tuning).

Input Size	BFLOPS	RMSE (g)	MAE (g)	CCC	Class-specific RMSE (g)				
					Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
100×100	67.01	44.51	38.10	0.24	45.00	62.61	51.86	58.50	73.38
224×224	261.58	42.62	36.33	0.29	44.46	62.96	48.80	40.06	73.37
350×350	645.72	42.63	35.27	0.34	41.91	59.57	45.67	38.87	68.73

After extensive hyperparameter optimization, our final weight estimation model achieved the validation metrics shown in Table 6.10. The model demonstrates an RMSE of 42.62 g and MAE of 36.33 g, with a modest CCC of 0.29, indicating room for improvement in predictive power. The class-specific metrics reveal varying performance across food categories, with particularly strong performance for starches and meat/fish, while "other sauce" exhibits the highest error.

We also explored using specialized models for the two meal phases (serving and clearing). When training and evaluating these specialized models, we used ground truth phase labels to assign images to the appropriate model, ensuring that performance differences reflect true capacity to model phase-specific visual characteristics.

6 Evaluation and Results

The results in Table 6.10 demonstrate the effectiveness of this phase-specific specialization approach within the models' target domains. For the serving phase, the specialized model shows notably reduced RMSE for estimating the weight of starches and vegetables. Similarly, the model specialized for the clearing phase achieves significantly lower RMSE values for sauces, meat/fish and vegetables when evaluated on the clearing-phase data. Phase-specific models consistently achieve lower RMSEs compared to the general model, demonstrating accuracy gains from adapting to each phase's distinct visual characteristics.

Table 6.10: Weight estimation performance metrics on the validation set for general and specialized models.

Model Type	RMSE (g)	MAE (g)	CCC	Class-specific RMSE (g)				
				Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
General model	42.62	36.33	0.29	44.46	62.96	48.80	40.06	73.37
Specialized - Serving phase	42.07	31.55	0.30	23.92	40.38	48.93	59.69	49.38
Specialized - Clearing phase	37.91	27.76	0.13	49.63	31.07	33.39	21.42	51.05

Test Set Performance

Based on validation results, we selected the ResNet-50 architecture trained from scratch on 224×224 images with optimized hyperparameters for final evaluation on the test set. We also evaluated the dual-model approach with phase-specific models. Table 6.11 presents the quantitative evaluation on the test set.

Contrary to validation results, the general model approach (46.93 g RMSE) outperformed the specialized models for both serving (52.71 g RMSE) and clearing phases (56.76 g RMSE) on the test set. This performance inversion suggests potential overfitting of the specialized models to phase-specific characteristics present in the validation set but not generalizing well to unseen test data. The visual distinctions between serving and clearing phases may be more nuanced or variable than anticipated, particularly across different residential care centers.

Class-specific metrics show the general model's more consistent cross-category performance. It notably outperforms the specialized models for vegetables and meat/fish, though the clearing phase model excels at sauce estimation. This suggests certain food categories might still benefit from phase-specific modeling despite the general model's overall superiority.

It's worth noting that the "other sauce" category has limited test samples, which explains the absence of metrics for this category in the serving phase model and the high error rates in the other models. This data scarcity highlights a limitation in our evaluation and suggests that more targeted data collection for underrepresented categories could improve future iterations of the system.

Overall, the test results demonstrate the robustness of our general model approach, achieving a CCC of 0.44 which indicates moderate agreement between predicted and actual weights. The higher test RMSE compared to

6 Evaluation and Results

validation (46.93 g vs. 42.62 g) suggests some expected performance degradation when generalizing to new care centers, but the model maintains acceptable accuracy for practical nutritional monitoring applications.

Table 6.11: Weight estimation performance metrics on the test set for general and specialized models.

Model Type	RMSE (g)	MAE (g)	CCC	Class-specific RMSE (g)				
				Starch	Vegetables	Meat/Fish	Sauce	Other Sauce
General model	46.93	32.88	0.44	41.93	55.57	41.92	56.29	73.71
Specialized - Serving phase	52.71	39.34	0.26	43.67	69.72	53.30	47.70	-
Specialized - Clearing phase	56.76	41.03	0.15	57.41	60.49	55.74	14.90	72.34

Error Analysis

We conducted an error analysis to gain deeper insights into the model performance. This section presents visualizations and analyses for the general model, though similar patterns were observed in the specialized models.

Figure 6.2 illustrates the relationship between predicted and actual weights for each food category. The scatter plots reveal that for starch, vegetables, meat/fish and sauce categories, the model demonstrates regression toward the mean—overestimating smaller portions while underestimating larger ones. This suggests the model struggles to capture the full range of weights, instead biasing predictions toward category-specific averages. Notably, the absolute error magnitude increases proportionally with ground truth weight, indicating greater prediction inaccuracy for larger food portions. The other sauce category exhibits severe underestimation across all samples, though the limited amount of test data makes it difficult to draw conclusions for both sauce categories.

The error distribution analysis (Figure 6.3) provides further insights into model behavior. The starch category displays a relatively symmetric distribution centered near its mean error of -3.52 g, with most predictions within ± 50 g of true values—suggesting balanced performance despite considerable variability (Std Dev = 41.78 g). Vegetables and meat/fish categories show negative skew, indicating systematic underestimation. The sauce component exhibits larger variance, reflecting the difficulty in estimating amorphous food components with variable visual appearances. The other sauce category shows exclusive and severe underestimation, likely due to limited training data.

The density comparison in Figure 6.4 compares the distributions of predicted versus actual weights. For most categories, the model produces narrower, more peaked distributions than the ground truth distributions, confirming the previously observed regression toward the mean. The sauce category shows an extremely peaked prediction distribution, indicating the model has defaulted to predicting a narrow range of values rather than capturing the true weight variability. For the other sauce category, there is a complete distributional mismatch—the model consistently predicts values around 25-50 g while actual weights cluster around 100 g.

6 Evaluation and Results

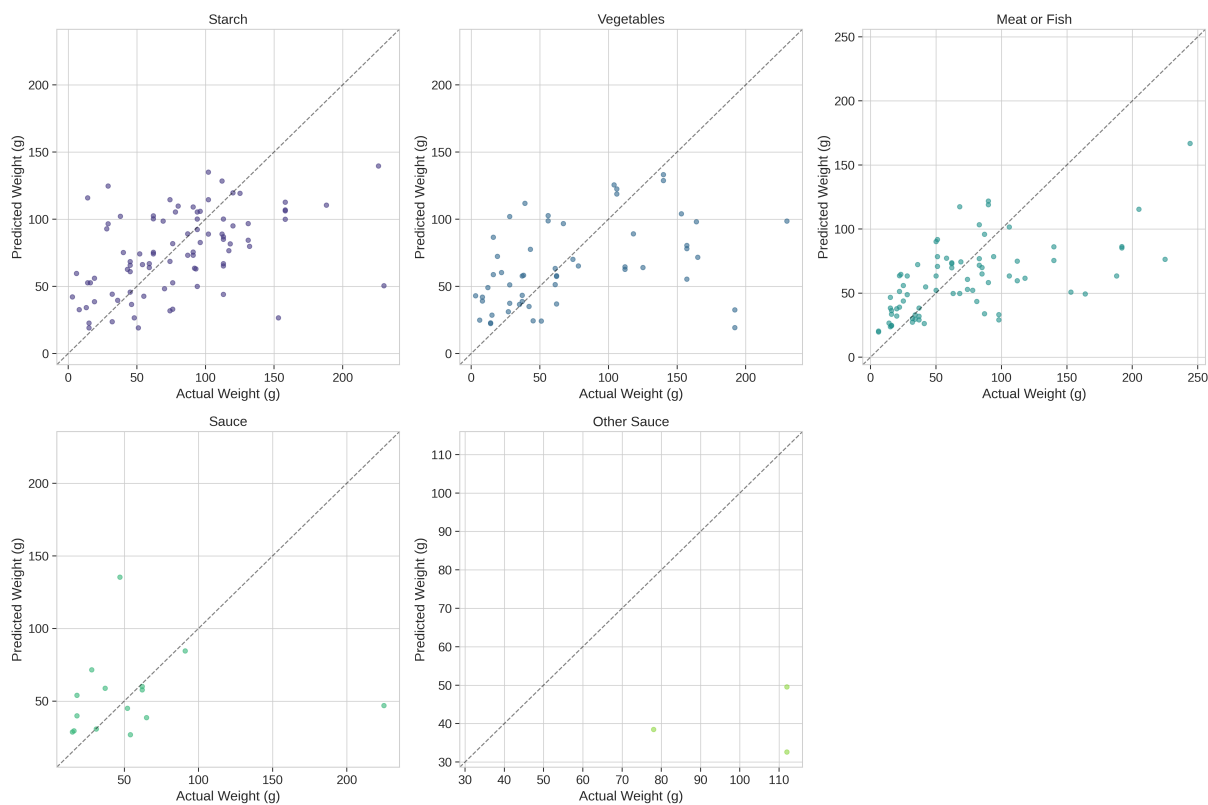


Figure 6.2: Actual versus predicted weights for each food category on the test set (general model).

Overall, the weight estimation module demonstrates moderate performance. The model performs best on well-defined, visually distinct categories like starch and meat/fish, while struggling with components like sauces. The observed regression toward the mean and difficulty with larger portions suggest that improvements could be achieved through balanced training datasets with better representation of weight extremes. Despite these limitations, the current model provides sufficient accuracy for nutritional monitoring applications in residential care environments, particularly when averaging across multiple meals.

Computational Efficiency

Table 6.12 summarizes key computational metrics for our three model configurations: the general model and the specialized serving and clearing models. Training was conducted on the IDLab GPULab infrastructure using an NVIDIA GeForce RTX 4090 GPU with 23 GB memory and 2 CPUs.

6 Evaluation and Results

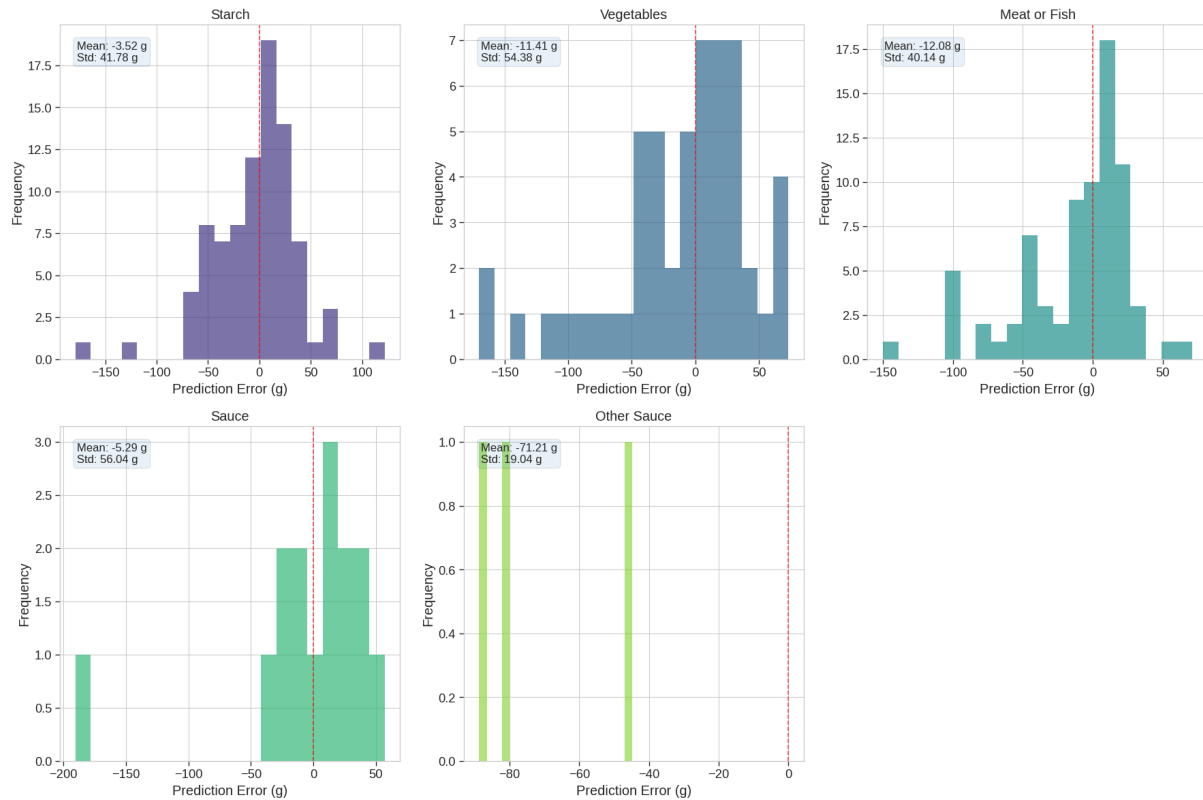


Figure 6.3: Weight estimation error distribution per food category on the test set (general model).

Table 6.12: Computational performance metrics for general and specialized weight estimation models.

Category	Metric	General Model	Serving Model	Clearing Model
Training Time	Training set only	31 min 23 s	25 min 4 s	23 min 16 s
	Training + validation sets	17 min 33 s	11 min 48 s	19 min 41 s
Inference Speed	Single image	14.27 ms	17.63 ms	14.18 ms
	Batch of 128 images (total)	0.06 s	0.06 s	0.14 s
	Per image in batch	0.43 ms	0.43 ms	1.06 ms
Resource Utilization (Training)	Peak CPU memory	3.9 GB	2.4 GB	2.1 GB
	Peak GPU memory	14 GB	14 GB	14 GB
	Peak CPU utilization	100 %	100 %	100 %
	Peak GPU utilization	41 %	60.0 %	40.5 %

The general ResNet-50 model processes images efficiently (14.27 ms per image, 0.43 ms in batch processing), with specialized models showing comparable performance. All models use 14 GB peak GPU memory and share similar resource requirements. These metrics indicate that computational demands would not constrain the practical

6 Evaluation and Results

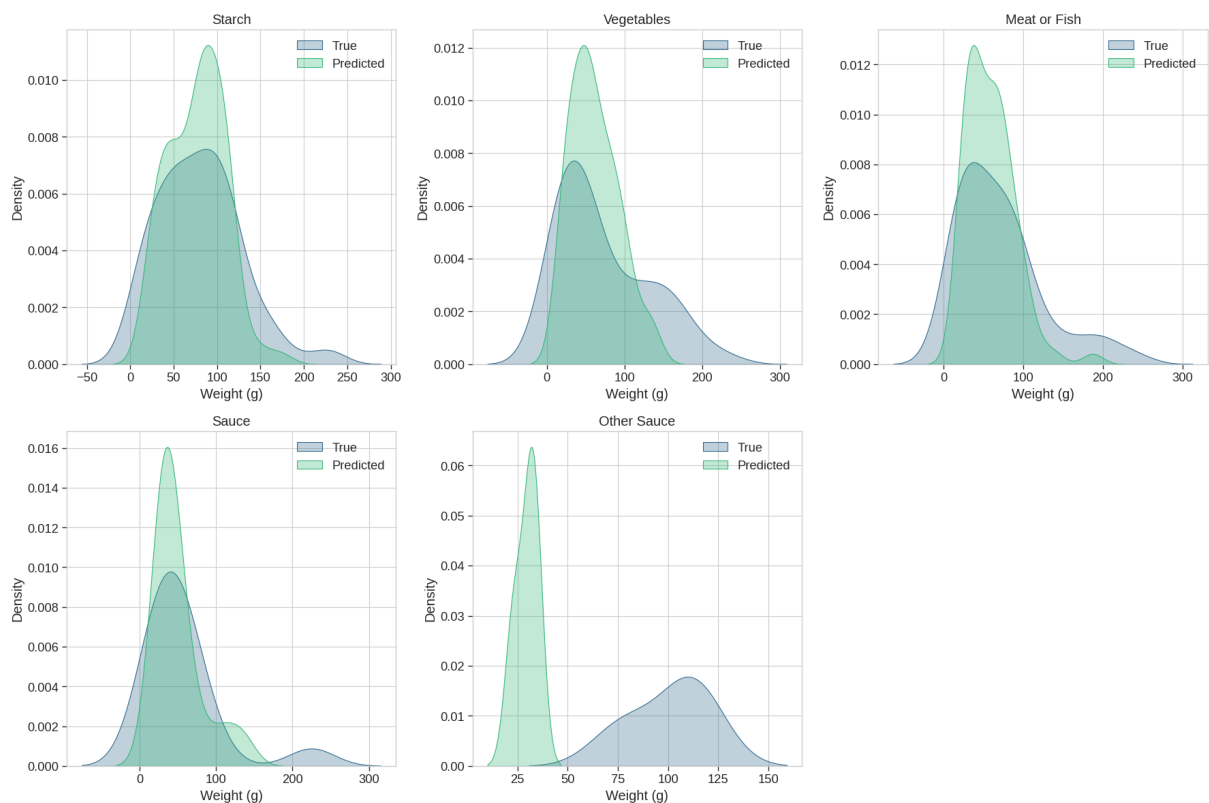


Figure 6.4: Density comparison of actual versus predicted weights per food category on the test set (general model).

deployment in care environments, where timely analysis within hours of mealtime is sufficient.

6.3 End-to-End System Evaluation

While component-level evaluation provides valuable insights into individual modules, the system's true effectiveness must be assessed through an end-to-end evaluation. This section presents an exhaustive analysis of the three pipeline configurations.

6.3.1 Comparative Results

Table 6.13 presents the performance comparison of the three pipeline variants with multiple configurations, across key metrics for both overall weight estimation and per-category assessment. All reported metrics are calculated on a per-component basis. We also include metrics for a baseline pipeline that simply predicts the average weight of the training and validation data.

6 Evaluation and Results

Table 6.13: End-to-end performance comparison of the pipeline variants on the test set.

Pipeline	Category	MAE (g)	RMSE (g)	CCC
Baseline	Starch	48.85	65.10	0.28
	Vegetables	22.32	32.37	0.54
	Meat/Fish	26.85	35.92	0.50
	Sauce	21.72	28.55	0.32
	Other Sauce	26.47	33.56	0.00
	Overall	29.24	41.27	0.34
Standard	Starch	38.39	59.50	0.41
	Vegetables	32.19	49.99	0.09
	Meat/Fish	24.66	38.45	0.52
	Sauce	11.71	23.97	0.49
	Other Sauce	2.97	9.49	0.00
	Overall	21.98	40.45	0.53
ROI-Enhanced	Starch	44.59	65.77	0.26
	Vegetables	32.68	55.75	-0.11
	Meat/Fish	33.4	53.4	0.21
	Sauce	11.35	23.95	0.16
	Other Sauce	1.42	5.12	0.00
	Overall	24.69	47.23	0.35
Dual Model - Ratio	Starch	40.25	61.12	0.41
	Vegetables	48.38	64.02	0.03
	Meat/Fish	32.48	43.65	0.48
	Sauce	19.21	40.89	0.33
	Other Sauce	3.93	12.23	0.00
	Overall	28.85	48.09	0.47
Dual Model - Ground Truth	Starch	42.75	64.32	0.35
	Vegetables	52.31	66.99	0.03
	Meat/Fish	30.97	42.27	0.56
	Sauce	41.75	18.67	0.28
	Other Sauce	6.84	20.29	0.00
	Overall	30.31	50.14	0.45

6 Evaluation and Results

Pipeline-Specific Analysis

The standard pipeline demonstrates the best overall performance (MAE = 21.98 g, RMSE = 40.45 g, CCC = 0.53), showing a 24.8 % improvement in MAE over the baseline. This approach effectively balances precision across different food types without requiring specialized enhancements, making it the most robust general solution in our evaluation.

The ROI-enhanced pipeline (MAE = 24.69 g, RMSE = 47.23 g, CCC = 0.35) performs 15.6 % better than the baseline in overall MAE but shows inconsistent results across categories. The region-specific processing appears beneficial for sauce components but compromises accuracy for the other components.

The dual model variants show mixed results compared to the baseline. The pixel count variant (MAE = 28.85 g, RMSE = 48.09 g, CCC = 0.47) marginally outperforms the baseline by 1.3 % in MAE, while the ground truth variant (MAE = 30.31 g, RMSE = 50.14 g, CCC = 0.45) actually underperforms the baseline by 3.7 % in MAE. Despite using specialized models for pre-consumption and post-consumption plates, these approaches do not improve accuracy as expected. A possible reason is that due to the fact that the models are trained on less data, they overfit more on the validation data. Notably, the simpler pixel count variant outperforms the ground-truth approach, which is really unexpected.

It should be noted that the end-to-end results outperform the individual weight estimation model because the pipeline evaluation includes zero weight components (absent food items) that are correctly identified, effectively reducing error measurements.

Category-Specific Insights

Starch components present significant challenges across all pipelines, with MAE ranging from 38.39 g to 48.85 g. The standard pipeline performs best (MAE = 38.39 g, CCC = 0.41), representing a 21.4 % improvement over the baseline. The difficulty in estimating starch probably stems from the variable density.

Vegetables show unexpected results, with the baseline (MAE = 22.32 g, CCC = 0.54) outperforming all proposed approaches. The standard pipeline (MAE = 32.19 g) shows a 44.2 % higher error than the baseline. This counterintuitive finding suggests either that vegetables in the dataset have relatively consistent portion sizes, making statistical methods effective, or that the visual features of vegetables (irregular shapes, varied colors, large amount of vegetable types) present unique challenges for computer vision algorithms that statistical approaches avoid.

Meat/Fish components are best estimated by the standard pipeline (MAE = 24.66 g, CCC = 0.52), showing an 8.2 % improvement over the baseline. The dual model with ground truth demonstrates similar accuracy with the highest correlation (CCC = 0.56). The relatively consistent density and distinctive visual characteristics likely contribute to this performance.

Sauce components show exceptional estimation accuracy in the ROI-enhanced (MAE = 11.35 g) and standard

6 Evaluation and Results

(MAE = 11.71 g) implementations, with approximately 47 % improvement over the baseline (MAE = 21.72 g). This seems a little odd as the sauce category did not score well in the evaluation of the weight estimation module. The reason for this low MAE score is that the component is often absent and the segmentation module correctly classifies it.

The Other Sauce category shows minimal presence in the dataset (evidenced by zero CCC values across all pipelines), limiting meaningful comparative analysis. All pipelines achieve low MAE for the same reason as in the regular sauce category.

6.3.2 Prediction Error Analysis

To gain deeper insights into the system's performance, we analyzed the distribution and patterns of prediction errors. This section presents and analyzes plots from our standard pipeline—the best performing configuration—though similar trends appear across all tested pipelines.

Figure 6.5 displays a scatter plot comparing predicted versus ground truth weights, revealing systematic patterns in estimation errors. Two notable trends emerge:

1. Components with true weight of 0 g (absent components) but predicted as having substantial weight (vertical line of points at $x = 0$). This error stems from segmentation failures where the system misclassifies regions, falsely detecting absent components.
2. Components with true weights greater than 0 g but predicted as 0 g (horizontal line of points at $y = 0$). These cases represent failures to detect genuinely present food components.

The substantial deviations from the ideal prediction line (diagonal red line) highlight considerable room for improvement in the system's weight estimation capabilities.

Figure 6.6 presents the analysis of the error distribution. This distribution suggests that, although the model attains reasonable accuracy in many cases, underestimations occur more frequently, whereas overestimations, though less common, tend to be more extreme. Examination of Figure 6.5 reveals that these pronounced overestimation outliers correspond directly to instances where the segmentation module fails to detect food components entirely or incorrectly assigns them to incorrect classes.

The cumulative error analysis in Figure 6.7 provides additional insights into the prediction accuracy:

- 50 % of predictions fall within ± 8.80 g of ground truth (median error).
- 90 % of predictions fall within ± 60.58 g of ground truth.
- The mean absolute error of 21.98 g exceeds the median, confirming that there are some serious errors.

6 Evaluation and Results

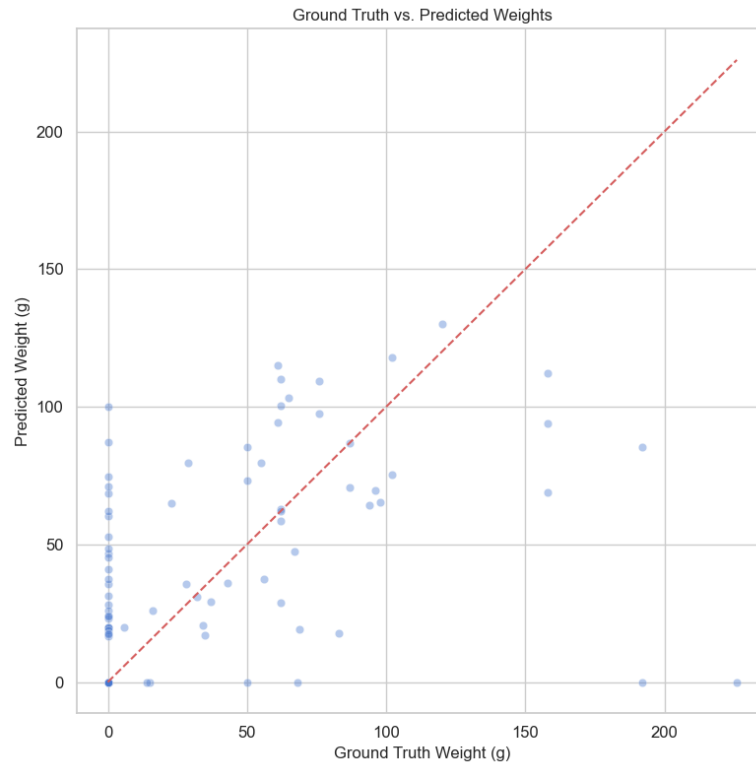


Figure 6.5: Actual versus predicted weights for the standard pipeline on the test set.

The curve's steepest section occurs below 50 g absolute error, indicating most prediction errors are relatively modest, with diminishing improvements beyond this threshold. This analysis offers a practical perspective on system reliability for clinical applications. While the system achieves reasonable accuracy for most predictions, clinicians should exercise caution when interpreting results.

6.3.3 Computational Performance Analysis

Each deep-learning-based system must balance computational efficiency with accuracy to be viable in real-world environments. This section presents an evaluation of our proposed pipeline. These results are for the standard pipeline, but the results of the other variants are similar.

Experimental Setup

Performance assessments were conducted on a laptop equipped with an Intel Core i7 vPro (8th generation) CPU and an NVIDIA Quadro P600 GPU. Although testing on a laptop is not ideal—due to background processes and potential resource contention—it was chosen out of practical necessity given time constraints. However, the results still offer a representative indication of the system's computational behavior. To ensure statistical reliability, fifteen runs were executed for each test scenario: batch processing of 32 images and individual image processing.

6 Evaluation and Results

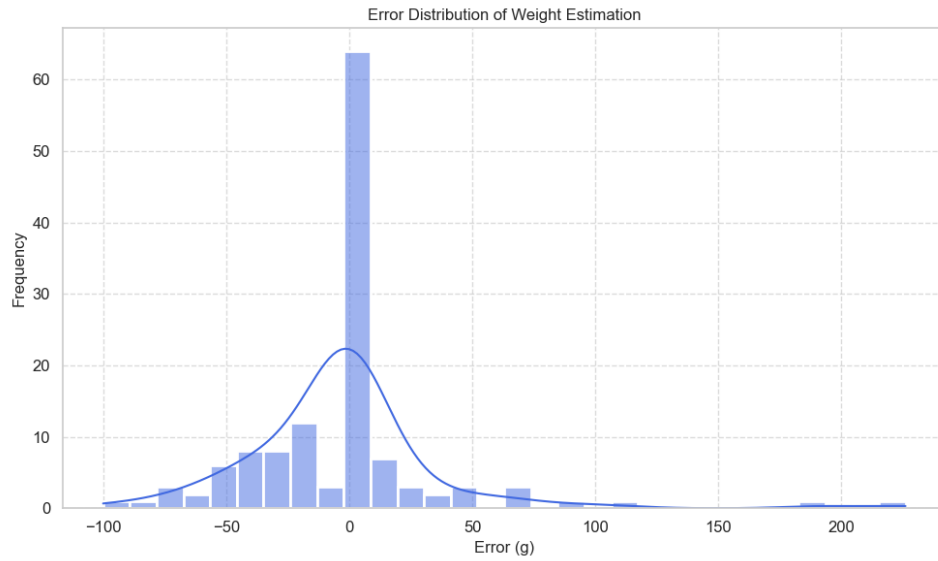


Figure 6.6: Distribution of weight estimation errors for the standard pipeline on the test set.

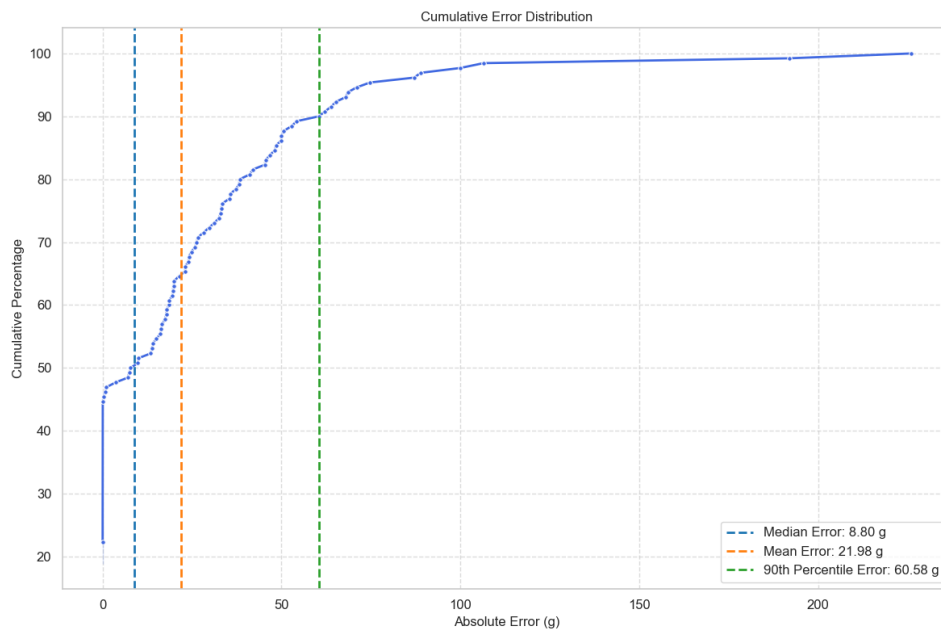


Figure 6.7: Cumulative distribution of absolute weight estimation errors for the standard pipeline on the test set.

Our system architecture comprises seven distinct functional components:

1. **Image Loading:** Reading and preprocessing input images.
2. **Segmentation Model:** Applying the segmentation model to segment and identify food components.
3. **Segment Extraction:** Isolating individual food components.

6 Evaluation and Results

4. **Weight Estimation:** Applying the regression model to approximate the weight of each food component.
5. **Nutrient Database Fetching:** Retrieving nutritional information from an online database.
6. **Nutrient Processing:** Converting the database information to the format used in the Obelisk Core data platform.
7. **Obelisk Core Data Ingestion:** Ingesting the data in the Obelisk Core data platform.

For each component, we monitored execution time (s), CPU utilization (%), GPU utilization (%), peak CPU memory consumption (MB), and peak GPU memory consumption (MB). Additionally, we measured query execution times and dashboard rendering performance.

Execution Time Analysis

Table 6.14 presents the mean wall clock times for each pipeline stage across both test scenarios, along with the calculated per-image processing time for the 32-image case.

Table 6.14: Mean wall clock time per pipeline stage (batch and single image processing).

Pipeline Stage	32 Images (s)	1 Image (s)	Avg. Per Image (Batch) (s)
Image Loading	5.84 ± 1.74	0.17 ± 0.06	0.18
Segmentation Model	5.94 ± 0.78	0.21 ± 0.17	0.19
Segment Extraction	0.21 ± 0.03	0.02 ± 0.01	0.01
Weight Estimation	4.69 ± 0.59	0.18 ± 0.04	0.15
Nutrient Database Fetching	20.98 ± 37.95	2.87 ± 0.49	0.66
Nutrient Processing	0.00 ± 0.00	0.00 ± 0.00	0.00
Obelisk Data Ingestion	0.65 ± 1.50	0.24 ± 0.16	0.02
Total Pipeline	38.31	3.69	1.20

The timing data reveal that nutrient database fetching is the most time-intensive operation. Since this component relies on an external database service rather than our internal implementation, optimization options are limited. A performance enhancement strategy would be the implementation of a caching system to store frequently accessed nutritional data, thereby reducing redundant external database queries and significantly decreasing average response latency.

Processing 32 images concurrently requires approximately 38.31 seconds, whereas a single image takes 3.69 seconds. This represents a substantial efficiency gain when operating in batch mode, with the per-image processing time diminishing to just 1.20 seconds—a 67 % improvement over sequential processing.

6 Evaluation and Results

Table 6.15 presents the combined wall clock times for querying and processing data from the Obelisk Core platform, highlighting performance variations across different analytical functions. Data for three meal moments over 14 days are stored in the platform for 100 residents.

Table 6.15: Performance of Obelisk Core data query methods.

Query Method	Execution Time (ms)
get_eaten_weight_between_dates_per_resident	123.66 ± 58.37
get_eaten_nutrients_between_dates_per_resident	218.96 ± 39.09
get_eaten_nutrients_between_dates_all_residents	7648.11 ± 586.00
get_food_waste_between_dates	1084.75 ± 144.80
get_comprehensive_waste_data	1062.39 ± 103.16
get_food_waste_breakdown	1085.51 ± 112.09
get_favorite_and_worst_menu_items	1780.67 ± 154.31

Resident-specific queries demonstrate exceptional efficiency, with execution times below 250 ms, making them suitable for real-time applications. Conversely, facility-wide nutrient data aggregation requires over 7 seconds on average, identifying a potential bottleneck. Food waste and menu preference queries exhibit moderate latency, positioning them appropriately for periodic reporting rather than continuous monitoring.

Table 6.16 presents the rendering times for dashboard components, encompassing query execution, data processing, and visualization rendering.

Table 6.16: Performance of dashboard component rendering times.

Dashboard Component	Rendering Time (s)
Nutrition KPIs	29.10 ± 5.32
Malnutrition risk chart	26.90 ± 5.79
Nutrient intake chart	0.86 ± 0.74
Resident eaten weight chart	0.92 ± 1.24
Food waste KPIs	6.37 ± 4.60
Food waste per day chart	9.63 ± 5.15
Food waste per component chart	9.60 ± 0.54
Food waste per moment chart	9.63 ± 0.51
Menu insights KPIs	23.26 ± 10.65
Menu insights chart	22.60 ± 7.89

6 Evaluation and Results

The dashboard performance metrics reveal a marked disparity between simple and complex visualizations. Basic charts render in under one second, while nutritional analytics and malnutrition risk assessments require 25-30 seconds. This substantial difference stems from the computational intensity of the data processing and the inherent limitations of the Dash Plotly implementation. Future iterations could benefit from migrating to more performance-optimized visualization frameworks such as Grafana.

Resource Utilization

Table 6.17 presents the CPU and GPU utilization percentages for each stage of the pipeline, comparing performance between batch processing (32 images) and single-image processing. Table 6.18 follows with memory consumption, covering both peak CPU RAM and peak GPU RAM.

Table 6.17: CPU and GPU utilization per pipeline stage.

Pipeline Stage	CPU Utilization (%)		GPU Utilization (%)	
	32 Images	1 Image	32 Images	1 Image
Image Loading	144.71 $\pm$ 92.70	116.18 $\pm$ 54.87	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Segmentation Model	18.37 $\pm$ 15.58	54.35 $\pm$ 58.53	21.63 $\pm$ 15.81	29.70 $\pm$ 8.39
Segment Extraction	18.28 $\pm$ 13.36	92.71 $\pm$ 56.47	20.47 $\pm$ 7.31	0.00 $\pm$ 0.00
Weight Estimation	99.93 $\pm$ 38.47	54.88 $\pm$ 48.57	24.93 $\pm$ 7.19	43.17 $\pm$ 9.92
Nutrient Database Fetching	2.70 $\pm$ 0.73	23.87 $\pm$ 32.74	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Nutrient Processing	11.43 $\pm$ 3.75	14.89 $\pm$ 6.87	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Obelisk Data Ingestion	0.34 $\pm$ 0.21	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00

Table 6.18: Peak CPU and GPU memory consumption per pipeline stage.

Pipeline Stage	CPU RAM (MB)		GPU RAM (MB)	
	32 Images	1 Image	32 Images	1 Image
Image Loading	477.98 $\pm$ 90.28	373.08 $\pm$ 36.94	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Segmentation Model	1163.54 $\pm$ 61.51	298.49 $\pm$ 55.41	3019.39 $\pm$ 424.96	181.92 $\pm$ 11.58
Segment Extraction	1196.08 $\pm$ 58.33	300.23 $\pm$ 54.17	778.12 $\pm$ 73.05	0.00 $\pm$ 0.00
Weight Estimation	846.77 $\pm$ 153.37	266.60 $\pm$ 2.87	724.48 $\pm$ 5.82	154.31 $\pm$ 4.80
Nutrient Database Fetching	650.95 $\pm$ 120.64	251.07 $\pm$ 1.16	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Nutrient Processing	112.13 $\pm$ 43.71	23.12 $\pm$ 76.00	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00
Obelisk Data Ingestion	510.55 $\pm$ 110.57	251.09 $\pm$ 1.14	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00

6 Evaluation and Results

Our analysis of the data presented in Table 6.17 and Table 6.18 reveals several key insights into the system's resource utilization characteristics, notably influenced by the use of multi-threading. CPU utilization frequently exceeds 100% during both batch and single-image processing, particularly in the image loading stage. This elevated utilization is an indicator of successful multi-threading, where multiple CPU cores are actively processing tasks concurrently, leading to aggregate utilization exceeding the capacity of a single core. Other CPU-intensive stages like weight estimation also show significant utilization, approaching 100% in the batch scenario. For stages like segmentation model and segment extraction, single-image processing tends to show higher CPU utilization compared to batch processing, suggesting potential overheads.

GPU utilization shows variability across stages and processing modes. The segmentation model and weight estimation stages show considerable GPU activity in both scenarios, with both having higher utilization during single-image processing compared to batch processing. Conversely, segment extraction shows GPU utilization during batch processing but minimal during single-image processing.

Memory consumption patterns highlight significant differences between batch and single-image processing. Peak CPU RAM usage is consistently higher during batch processing, reflecting the memory required to handle a larger volume of data simultaneously. Peak GPU RAM consumption shows a similar trend, with substantially higher usage during batch processing for GPU-dependent stages like segmentation and weight estimation, as the GPU loads and processes data for the entire batch. The substantial increase in GPU RAM for batch processing underscores the memory demands of parallel GPU operations on larger datasets. The occurrence of $0.00 \text{ MB} \pm 0.00$ in some stages and scenarios indicates that no GPU resources were used.

6.4 Conclusion

The comprehensive evaluation of LeftoVision highlights both its strengths and limitations in the context of automated nutritional monitoring for residential care environments.

6.4.1 Key Findings

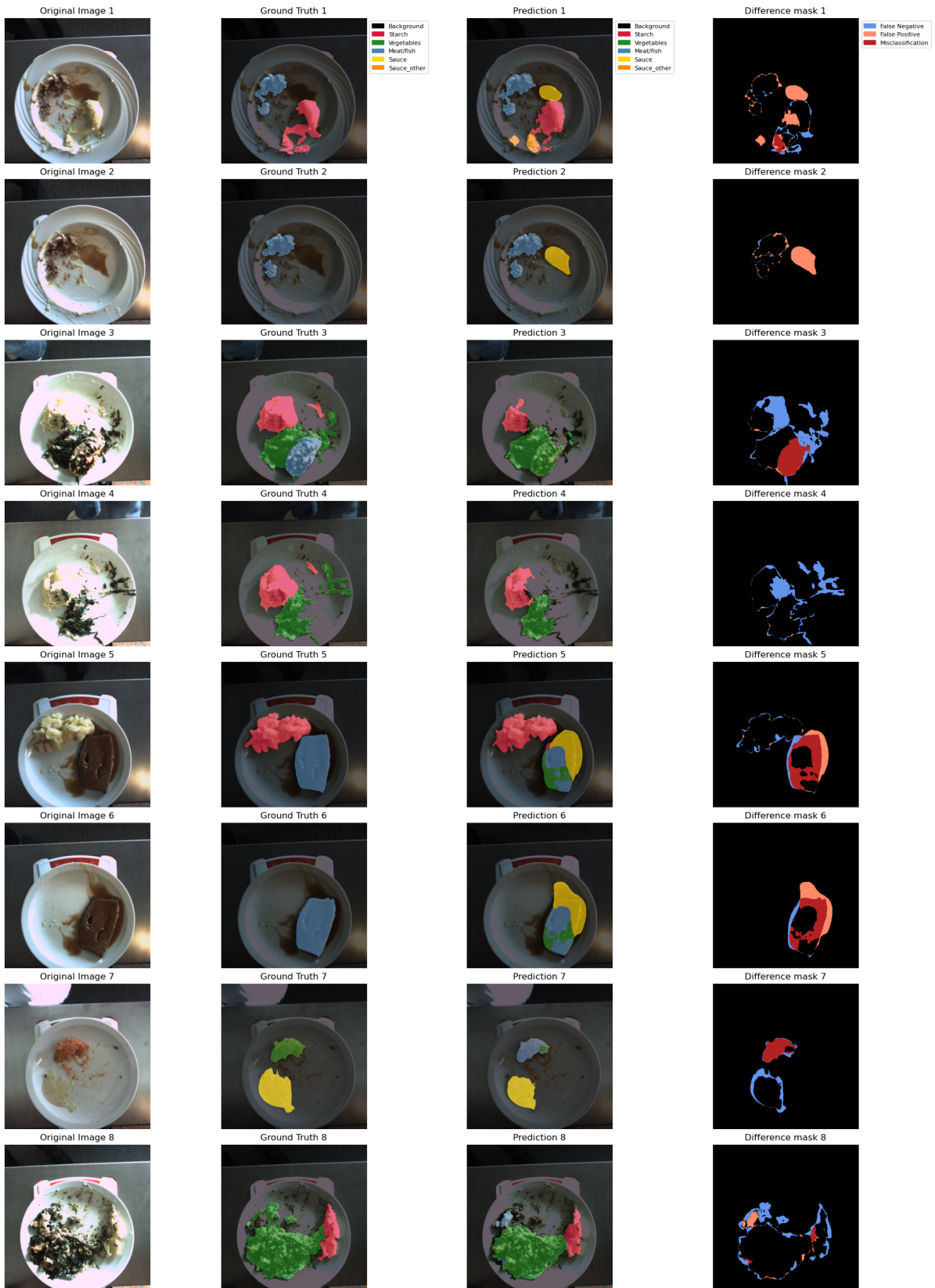
LeftoVision demonstrates notable performance in terms of accuracy, achieving an MAE of 21.98 g using the standard pipeline, which outperforms the baseline implementation. Interestingly, the standard pipeline surpasses more complex models, suggesting that simpler approaches may be more effective due to the potential errors introduced by additional components. The system processes approximately 50 images per minute, thus 25 patients per minute, in batch mode, providing a practical level of computational efficiency suitable for deployment in institutional settings. The adoption of batch processing capabilities has proven to be highly advantageous, improving efficiency by 67 % when compared to sequential processing. The system's architecture is designed to meet the throughput demands of residential care facilities while operating on modest hardware specifications.

6 Evaluation and Results

6.4.2 Limitations

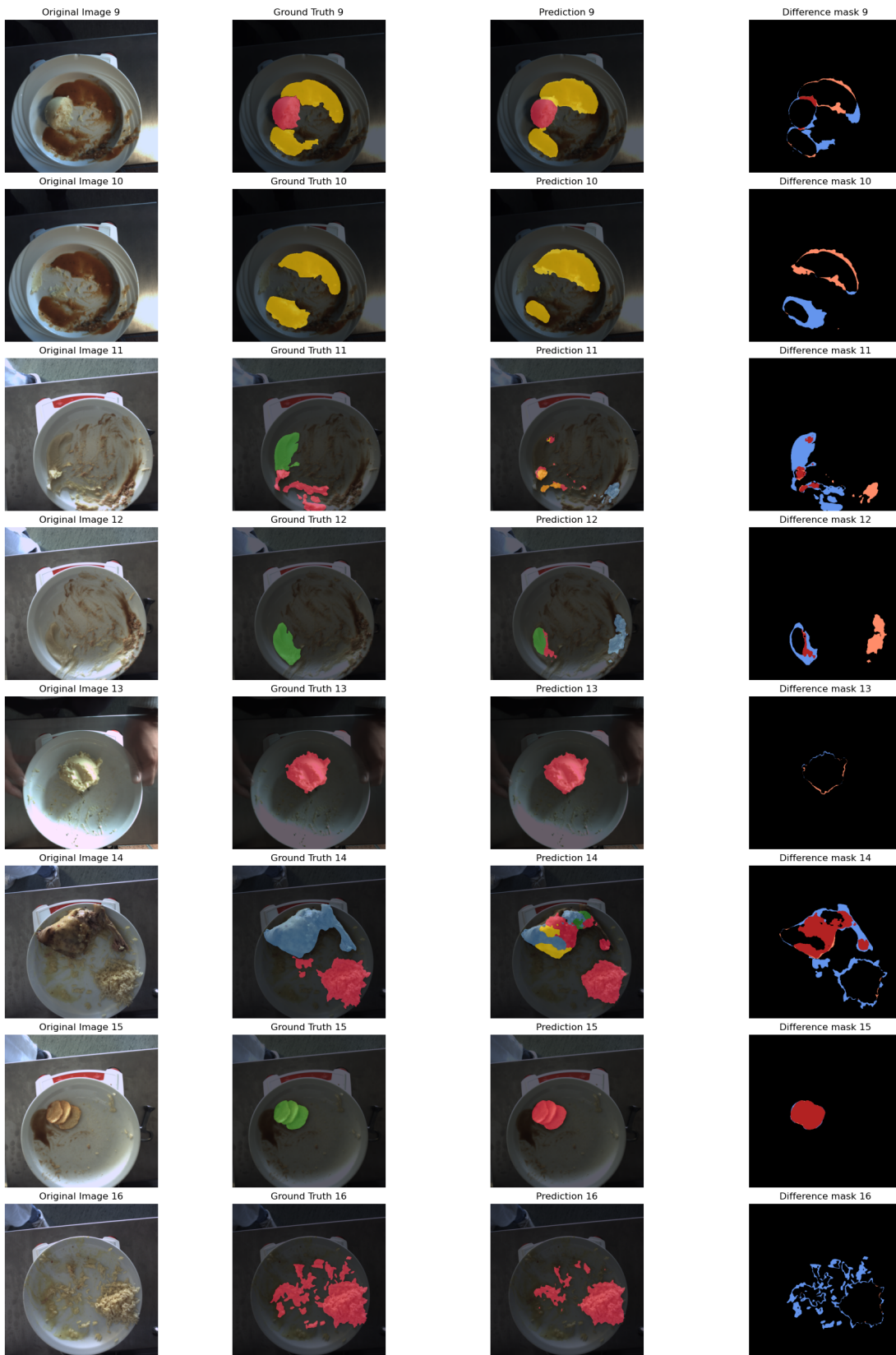
Despite these strengths, several limitations must be addressed. Our evaluation revealed inconsistent performance across different food categories. Components with higher visual and physical variability, particularly vegetables, exhibited poorer estimation accuracy. Furthermore, the discrepancy between validation and test performance suggests distribution shifts between residential care centers, which limits the system's generalizability in real-world settings. The scarcity of images for sauce components resulted in suboptimal performance in this category, which further contributed to the evaluation challenges. The elevated RMSE values, compared to the MAE scores, point to the presence of significant outliers, which could undermine the reliability of nutritional assessments. Additionally, the system's reliance on precise segmentation creates a cascading effect, where errors in segmentation significantly impact the accuracy of weight estimation.

6 Evaluation and Results



(a) Visualization of segmentation predictions (Part 1).

6 Evaluation and Results



(b) Visualization of segmentation predictions (Part 2).

Figure 6.8: Examples of segmentation predictions on the test set with error visualization.

Conclusion

This thesis addressed two interconnected challenges prevalent in residential care facilities: malnutrition among residents and food waste. As highlighted, up to 63 % of the residents in these settings are malnourished or at risk. At the same time, facilities discard between 25 % and 40 % of the served food—representing both an economic burden of approximately €25,000 annually per facility and a considerable environmental impact. Traditional monitoring methods, which are labor-intensive and prone to inaccuracy, are inadequate. Although automated systems exist, they typically only monitor what is served rather than consumed, missing leftover data required for accurate nutrition tracking and waste measurement. Furthermore, existing solutions are often not specifically tailored to the unique dietary complexities of residential care settings.

To bridge these gaps, we developed LeftoVision, a computer vision-based system. LeftoVision provides an end-to-end solution by analyzing images of food plates both before and after consumption. Its multi-stage pipeline uses semantic segmentation to identify individual food components, estimates their weight, performs a menu lookup for precise component identification, and queries a nutrient database to determine nutritional content based on the component and its estimated weight. All pre- and post-consumption data are stored in the Obelisk Core data platform, forming the basis for a dashboard that offers insights into resident nutritional status, food waste patterns, and menu effectiveness.

Our evaluation demonstrates that LeftoVision successfully fulfills both its functional and non-functional requirements. The system quantifies individual nutrient intake by analyzing the difference between served and leftover portions while simultaneously measuring food waste. It achieves an MAE of 21.98 g for food weight estimation, surpassing the target threshold of 29 g. This precision is crucial to obtain reliable nutritional numbers, especially for small portions or when only part of the meal is consumed, helping us to track even small changes in diet. Furthermore, LeftoVision demonstrates practical scalability and computational efficiency, processing meals for approximately 25 residents per minute—a throughput sufficient for deployment in larger residential care facilities. The dashboard is designed to minimize the learning curve for care staff and the entire system is expected to reduce the time required to monitor food compared to manual methods. However, these user-centric benefits warrant validation through real-world operational testing.

Beyond meeting the functional and non-functional requirements, LeftoVision also makes contributions to the field by addressing some of the identified gaps in current research. Unlike comparable systems, it was specifically designed for and tested with actual data from residential care environments, ensuring reliability in these unique settings. Its ability to track consumption from start to finish, including leftovers, is an improvement over systems that only monitor what is served. This methodology, combined with its capacity to quantify smaller amounts, addresses the need for automated nutritional and waste monitoring in elderly care.

By simultaneously addressing malnutrition monitoring and food waste reduction through a single integrated system, LeftoVision represents a paradigm shift in how residential care facilities can approach dietary management.

Conclusion

This research provides a foundation for a practical alternative to current manual and automated systems. Moving forward, LeftoVision has the potential to improve nutritional insights for residents, support clinical decisions, and lead to more efficient food practices in care centers. While no solution is perfect, areas for improvement and future work directions are outlined in the following chapter.

Future Work

This chapter identifies research directions to enhance our computer vision-based food monitoring system for residential care settings. Building upon the foundation established in this thesis, we outline several avenues for future investigation that could improve the accuracy, robustness, and practical utility of the system.

Enhanced Model Architectures

Food items of different categories exhibit substantial variation in their visual-to-mass relationships. Future iterations could implement specialized weight estimation models for distinct food categories, accounting for these variations, and improving accuracy across different food types.

Furthermore, incorporating food-specific contextual information, such as name, typical density, and preparation method, could improve weight estimation accuracy by providing essential cues not evident from appearance alone.

The current implementation uses separate models for semantic segmentation and weight estimation. A compelling direction for future work involves developing a unified architecture that simultaneously performs both tasks or directly predicts the weights of all components without performing segmentation first. This approach could leverage shared feature representations, potentially improving performance by learning inter-dependencies between food segmentation and weight estimation. Such an integration would be particularly beneficial as additional training data become available, enabling more complex architectural designs.

Data Augmentation and Acquisition

To address the scarcity of labeled training data, future work could explore multiple approaches. For example, we can explore diffusion models or GANs to generate new synthetic data. The data will only be useful for the segmentation model.

Beyond synthetic data generation, future iterations could directly incorporate 3D information captured by a 3D camera during inference, establishing more direct correlations between visual appearance and food mass. Volume estimation would serve as an intermediate step between segmentation and weight prediction, reducing errors caused by variations in food density or arrangement.

If 3D integration proves successful, Neural Radiance Fields (NeRF) [92] could be explored to create detailed 3D representations from limited 2D views. This technique could potentially eliminate the need for dedicated 3D cameras in the setup, reducing system complexity while maintaining the benefits of volumetric information.

Expanding beyond standard RGB imaging to include multispectral or hyperspectral imaging could capture food properties invisible to conventional cameras. These technologies could reveal compositional information through

hundreds of narrow spectral bands, potentially improving segmentation by more easily distinguishing between visually similar food items and enhancing weight estimation through additional spectral features.

System Robustness and Deployment Optimization

In healthcare applications such as nutrient intake monitoring, understanding prediction uncertainty is crucial for responsible decision-making. Implementing Bayesian Neural Networks could provide explicit uncertainty quantification by treating model parameters as probability distributions rather than point estimates. This approach would allow the system to express confidence levels in predictions, particularly valuable when encountering unusual food presentations or new menu items. By quantifying uncertainty, the system could flag predictions that require human verification, potentially reducing errors and improving trust.

We have observed a gap between the performance on validation and test sets, indicating limited generalization to new care centers. Although this gap would likely diminish with more diverse training data, transfer learning techniques could help models adapt more quickly to new environments. Domain adaptation methods could be particularly useful for efficiently deploying the system across different care facilities with minimal additional data collection.

Although computational efficiency is not a strict requirement for this application, applying model compression techniques such as pruning and quantization would lower computational demands, thereby enabling deployment on low-power, cost-effective edge devices. This optimization not only facilitates faster performance, but also substantially reduces energy consumption, contributing to more sustainable operation.

To further enhance system robustness, a fallback mechanism can be introduced for handling cases where food components are not found in the primary nutrient database. In such instances, the system could query a secondary database to retrieve the missing nutritional information. This layered approach would reduce the risk of data gaps, and improve the system's reliability.

Conclusion

The enhancements outlined in this chapter represent opportunities to improve our computer vision system for nutritional monitoring in residential care centers. By pursuing advancements in model architectures, data generation strategies, 3D information integration, uncertainty quantification, and deployment optimization, the system could evolve into a more reliable tool for combating malnutrition and reducing food waste. While implementing all these enhancements would require substantial research effort, even incremental improvements in specific areas could yield meaningful benefits in practical settings.

Sustainability and Ethical Reflection

Contribution to the United Nations Sustainable Development Goals

The work conducted in this thesis aligns with multiple United Nations Sustainable Development Goals (UN SDGs) [93]. The following SDGs are directly impacted by the outcomes of this research:

- **SDG 2: Zero Hunger:** Our system addresses malnutrition through systematic tracking of nutrient intake in elderly care centers, ensuring residents receive adequate nutrition.
- **SDG 3: Good Health and Well-being:** The enhanced dietary monitoring enables caregivers to optimize nutritional intake for elderly individuals, leading to improved health outcomes, particularly for those at risk of malnutrition.
- **SDG 12: Responsible Consumption and Production:** By enhancing food waste monitoring and promoting data-driven decision-making, this research aids in reducing unnecessary food disposal. The insights generated can help optimize meal planning, thereby contributing to more responsible consumption patterns within care institutions. On the other hand, while AI-driven solutions offer significant benefits in addressing societal and environmental challenges, it is also crucial to acknowledge their environmental footprint. Training and inference of deep learning models, particularly those requiring extensive computational resources, consumes substantial energy, contributing to carbon emissions. The environmental cost of AI development and deployment must be balanced with its potential to promote sustainability.
- **SDG 9: Industry, Innovation, and Infrastructure:** The implementation of AI-driven solutions in healthcare and food monitoring exemplifies technological innovation that can be extended to other domains. This work underscores the role of research and technology in fostering sustainable infrastructures.

By aligning with these goals, the research conducted in this thesis demonstrates how AI-driven solutions can be leveraged to address global challenges while promoting sustainability and ethical responsibility.

Ethical Considerations

The increasing integration of AI into healthcare and food monitoring raises several ethical considerations, particularly regarding data privacy, model reliability, potential dual-use applications, and environmental impact.

Privacy and Data Anonymization

This thesis involves the collection and processing of food images linked to specific residents. While this connection is necessary for tracking individual dietary intake, it introduces privacy-sensitive aspects that must be addressed

Sustainability and Ethical Reflection

responsibly.

For this research, we used an anonymized database provided by ILVO that allowed us to work with food nutrient data without handling personally identifiable information. The data outputted by the system is stored in the Obelisk Core system and displayed through a dashboard interface.

While security implementation was outside the scope of this thesis, we acknowledge that protecting this health-related data is critical for any production deployment. Future implementations would require proper authentication, and access controls to ensure that sensitive nutritional information remains confidential and accessible only to authorized healthcare providers. This security layer represents an essential next step before any real-world application of the system in residential care environments.

Model Reliability and the Necessity of Human Oversight

Another significant ethical concern is the fallibility of AI models. While deep learning models can achieve high levels of accuracy in food segmentation and portion estimation, they are not infallible. Misclassifications, incorrect segmentations, or inaccurate weight estimations can occur due to variations in lighting and food presentation. Relying solely on AI for decision-making in a healthcare setting could lead to incorrect conclusions about a resident's nutritional intake, potentially affecting their well-being.

To mitigate the risk of erroneous predictions, human oversight remains essential. The system developed in this thesis is intended to support—rather than replace—caregivers and dietitians in elderly care centers. As future work, incorporating uncertainty estimation into the model could further enhance safety and reliability by automatically flagging low-confidence weight predictions. Such a mechanism would proactively alert caregivers when manual verification or adjustment is warranted, thereby strengthening the collaborative interaction between AI systems and human expertise. In this context, the role of AI should be understood as that of an assistive tool that augments efficiency and consistency, while preserving the critical role of professional judgment in nutritional care.

Dual-Use Considerations

Although this research is developed with the primary goal of improving food monitoring and reducing waste in residential care centers, similar AI-driven techniques can be repurposed for alternative applications, including commercial and industrial contexts. Some of these applications could be beneficial, while others might raise ethical concerns:

- **Industrial Food Production:** The ability to estimate nutrient content in meals could be leveraged by food manufacturers and large-scale catering services to ensure consistency in nutrient composition.

Sustainability and Ethical Reflection

- **Automated Dietary Surveillance:** AI-based food tracking could be adapted for corporate or institutional meal monitoring, such as in schools, prisons, or military facilities. If implemented without proper ethical oversight, this could lead to excessive surveillance over individuals' eating habits, raising concerns about personal autonomy and privacy.
- **Insurance and Health Monitoring:** In a more extreme scenario, AI-driven dietary assessment tools could be integrated into health insurance systems, where individuals' food choices might be used to influence premiums or coverage decisions. This could lead to ethical dilemmas regarding food choice autonomy and discrimination based on diet-related health risks.

Environmental Impact

As previously noted, the environmental impact of artificial intelligence is an ethical concern that warrants attention. Both training and inference of deep learning models demand considerable computational resources, leading to increased energy consumption. This creates a paradox: while the system aims to enhance sustainability by reducing food waste, its development and operation contribute to an environmental footprint. Mitigating this requires conscious efforts to adopt energy-efficient model architectures, optimize training procedures, and minimize resource use during inference, thereby aligning the system's implementation with its sustainability goals.

References

- [1] S. Jung, H. Heo, S. Park, S.-U. Jung, and K. Lee, "Benchmarking deep learning models for instance segmentation," *Applied Sciences*, vol. 12, no. 17, p. 8856, 2022.
- [2] Prabhu, "Understanding of convolutional neural network (cnn) — deep learning," Mar 2018. [Online]. Available: <https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>
- [3] Wikipedia contributors, "A residual block in a deep residual network. Here, the residual connection skips two layers." 2025, [Online; Accessed 29-March-2025]. [Online]. Available: https://en.wikipedia.org/wiki/Residual_neural_network
- [4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
- [5] Z. Li, H. Yu, J. Xu, J. Liu, and Y. Mo, "Stock market analysis and prediction using lstm: A case study on technology stocks," *Innovations in Applied Engineering and Technology*, pp. 1–6, 2023.
- [6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: <https://arxiv.org/abs/1505.04597>
- [8] C. Li, Z. Qiu, X. Cao, Z. Chen, H. Gao, and Z. Hua, "Hybrid dilated convolution with multi-scale residual fusion network for hyperspectral image classification," *Micromachines*, vol. 12, no. 5, p. 545, 2021.
- [9] V. B. Raju and E. Sazonov, "A systematic review of sensor-based methodologies for food portion size estimation," *IEEE Sensors Journal*, vol. 21, no. 11, pp. 12 882–12 899, 2021.
- [10] F. P. W. Lo, Y. Sun, J. Qiu, and B. Lo, "Image-based food classification and volume estimation for dietary assessment: A review," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 7, pp. 1926–1939, 2020.
- [11] J. Sultana, B. M. Ahmed, M. M. Masud, A. K. O. Huq, M. E. Ali, and M. Naznin, "A study on food value estimation from images: Taxonomies, datasets, and techniques," *IEEE Access*, vol. 11, pp. 45 910–45 935, 2023.
- [12] M. F. Vandewoude, J. P. van Wijngaarden, L. De Maesschalck, Y. C. Luiking, and A. Van Gossum, "The prevalence and health burden of malnutrition in belgian older people in the community or residing in nursing homes: results of the nutriaction ii study," *Aging clinical and experimental research*, vol. 31, pp. 175–183, 2019.
- [13] D. Maher and C. Eliadi, "Malnutrition in the elderly: an unrecognized health issue," *Journal of Nursing*, 2010.

References

- [14] F. Brugge, "Voedselverlies in de zorgsector: Hoe kan je het voorkomen?" https://foodwin.org/wp-content/uploads/2021/03/20180924_PU_brochureA5VoedselVerliesZorgweb-1-1.pdf, 2017, accessed 10-10-2024.
- [15] S. E. Gariballa and S. Forster, "Dietary intake of older patients in hospital and at home: the validity of patient kept food diaries," *The Journal of Nutrition Health and Aging*, vol. 12, pp. 102–106, 2008.
- [16] imec, "Fine: Food intake ecosystem," <https://www.imec.be/nl/vlaamse-innovatiemotor/impactdomeinen/smart-health/fine>, 2025, accessed: 2025-05-15.
- [17] I. Lawrence and K. Lin, "A concordance correlation coefficient to evaluate reproducibility," *Biometrics*, pp. 255–268, 1989.
- [18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
- [22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [23] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019. [Online]. Available: <http://arxiv.org/abs/1905.11946>
- [24] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
- [25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.
- [27] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

References

- [28] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5–9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.
- [30] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [31] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.
- [32] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.
- [33] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.
- [34] W. Wang, W. Min, T. Li, X. Dong, H. Li, and S. Jiang, "A review on vision-based analysis for automatic dietary assessment," *Trends in Food Science & Technology*, vol. 122, pp. 223–237, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0924224422000656>
- [35] G. Ciocca, P. Napoletano, and R. Schettini, "Food recognition: A new dataset, experiments, and results," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 3, pp. 588–598, 2017.
- [36] X. Wu, X. Fu, Y. Liu, E.-P. Lim, S. C. Hoi, and Q. Sun, "A large-scale benchmark for food image segmentation," in *Proceedings of the 29th ACM international conference on multimedia*, 2021, pp. 506–515.
- [37] E. Battini Sönmez, S. Memiş, B. Arslan, and O. Z. Batur, "The segmented uec food-100 dataset with benchmark experiment on food detection," *Multimedia Systems*, vol. 29, no. 4, pp. 2049–2057, 2023.
- [38] K. Okamoto and K. Yanai, "Uec-foodpix complete: A large-scale food image segmentation dataset," in *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part V*. Springer, 2021, pp. 647–659.
- [39] Y. Matsuda, H. Hoashi, and K. Yanai, "Recognition of multiple-food images by detecting candidate regions," in *2012 IEEE international conference on multimedia and expo*. IEEE, 2012, pp. 25–30.

References

- [40] Alcrowd. (2022) Food recognition 2022. [Online]. Available: <https://www.kaggle.com/datasets/awsaf49/food-recognition-2022-dataset>
- [41] Q. Thames, A. Karpur, W. Norris, F. Xia, L. Panait, T. Weyand, and J. Sim, "Nutrition5k: Towards automatic nutritional understanding of generic food," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 8903–8911.
- [42] Y. Liang and J. Li, "Computer vision-based food calorie estimation: dataset, method, and experiment," *CoRR*, vol. abs/1705.07632, 2017. [Online]. Available: <http://arxiv.org/abs/1705.07632>
- [43] C. en Amy Tai, M. Keller, M. Kerrigan, Y. Chen, S. Nair, P. Xi, and A. Wong, "Nutritionverse-3d: A 3d food model dataset for nutritional intake estimation," 2023. [Online]. Available: <https://arxiv.org/abs/2304.05619>
- [44] Y. Chen, J. He, G. Vinod, S. Raghavan, C. Czarnecki, J. Ma, T. I. Mahmud, B. Coburn, D. Mao, S. Nair, P. Xi, A. Wong, E. Delp, and F. Zhu, "Metafood3d: 3d food dataset with nutrition values," 2024. [Online]. Available: <https://arxiv.org/abs/2409.01966>
- [45] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, and L. Zhang, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 6881–6890.
- [46] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.
- [47] X. Lan, J. Lyu, H. Jiang, K. Dong, Z. Niu, Y. Zhang, and J. Xue, "Foodsam: Any food segmentation," *IEEE Transactions on Multimedia*, 2023.
- [48] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4015–4026.
- [49] G. Sinha, K. Parmar, H. Azimi, A. Tai, Y. Chen, A. Wong, and P. Xi, "Transferring knowledge for food image segmentation using transformers and convolutions," 2023. [Online]. Available: <https://arxiv.org/abs/2306.09203>
- [50] L. Chen, B. Wang, and J. Zhang, "Ingredsam: Open-world food ingredient segmentation via a single image prompt," *Journal of Imaging*, vol. 10, no. 12, 2024. [Online]. Available: <https://www.mdpi.com/2313-433X/10/12/305>
- [51] U. Sharma, B. Artacho, and A. Savakis, "Gourmetnet: Food segmentation using multi-scale waterfall features with spatial and channel attention," *Sensors*, vol. 21, no. 22, p. 7504, 2021.

References

- [52] E. Aguilar, B. Nagarajan, B. Remeseiro, and P. Radeva, "Bayesian deep learning for semantic segmentation of food images," *Computers and Electrical Engineering*, vol. 103, p. 108380, 2022.
- [53] FAO, "Fao/infoods food composition databases." [Online]. Available: <https://www.fao.org/infoods/infoods/tables-and-databases/faoinfoods-databases/en/>
- [54] J. Dehais, M. Anthimopoulos, S. Shevchik, and S. Mougiakakou, "Two-view 3d reconstruction for food volume estimation," *IEEE transactions on multimedia*, vol. 19, no. 5, pp. 1090–1099, 2016.
- [55] E. A. H. Akpa, H. Suwa, Y. Arakawa, and K. Yasumoto, "Smartphone-based food weight and calorie estimation method for effective food journaling," *SICE Journal of Control, Measurement, and System Integration*, vol. 10, no. 5, pp. 360–369, 2017.
- [56] A. Gao, F. P.-W. Lo, and B. Lo, "Food volume estimation for quantifying dietary intake with a wearable camera," in *2018 IEEE 15th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, 2018, pp. 110–113.
- [57] F. P.-W. Lo, Y. Sun, J. Qiu, and B. Lo, "Food volume estimation based on deep learning view synthesis from a single depth map," *Nutrients*, vol. 10, no. 12, p. 2005, 2018.
- [58] —, "A novel vision-based approach for dietary assessment using deep learning view synthesis," in *2019 IEEE 16th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. IEEE, 2019, pp. 1–4.
- [59] G. Vinod, J. He, Z. Shao, and F. Zhu, "Food portion estimation via 3d object scaling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3741–3749.
- [60] G. Ciocca, P. Napoletano, and R. Schettini, "Food recognition and leftover estimation for daily diet monitoring," in *New Trends in Image Analysis and Processing—ICIAP 2015 Workshops: ICIAP 2015 International Workshops, BioFor, CTMR, RHEUMA, ISCA, MADiMa, SBMI, and QoEM, Genoa, Italy, September 7-8, 2015, Proceedings 18*. Springer, 2015, pp. 334–341.
- [61] Y. A. Sari, S. Adinugroho, J. M. Maligan, E. Nadia Candra, F. Utaminingrum, and N. Nur'Aini, "Leftovers food recognition using deep neural network and regression approach for objective visual analysis estimation," in *2021 4th International Conference of Computer and Informatics Engineering (IC2IE)*, 2021, pp. 24–29.
- [62] F. S. Konstantakopoulos, E. I. Georga, and D. I. Fotiadis, "A novel approach to estimate the weight of food items based on features extracted from an image using boosting algorithms," *Scientific Reports*, vol. 13, no. 1, p. 21040, 2023.
- [63] S. Gebhardt, L. Lemar, D. Haytowitz, P. Pehrsson, M. Nickle, B. Showell, R. Thomas, J. Exler, and J. Holden, "Usda

References

- national nutrient database for standard reference, release 21," *United States Department of Agriculture Agricultural Research Service*, 2006.
- [64] J. B. Montville, J. K. Ahuja, C. L. Martin, K. Y. Heendeniya, G. Omolewa-Tomobi, L. C. Steinfeldt, J. Anand, M. E. Adler, R. P. LaComb, and A. Moshfegh, "Usda food and nutrient database for dietary studies (fndds), 5.0," *Procedia Food Science*, vol. 2, pp. 99–112, 2013.
- [65] Y. Lu, D. Allegra, M. Anthimopoulos, F. Stanco, G. M. Farinella, and S. Mougiakakou, "A multi-task learning approach for meal assessment," in *Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management*, ser. CEA/MADiMa '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 46–52. [Online]. Available: <https://doi.org/10.1145/3230519.3230593>
- [66] S. Fang, Z. Shao, R. Mao, C. Fu, E. J. Delp, F. Zhu, D. A. Kerr, and C. J. Boushey, "Single-view food portion estimation: Learning image-to-energy mappings using generative adversarial networks," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 251–255.
- [67] T. Ege and K. Yanai, "Multi-task learning of dish detection and calorie estimation," in *Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management*, ser. CEA/MADiMa '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 53–58. [Online]. Available: <https://doi.org/10.1145/3230519.3230594>
- [68] R. Ruede, V. Heusser, L. Frank, A. Roitberg, M. Haurilet, and R. Stiefelhagen, "Multi-task learning for calorie prediction on a novel large-scale recipe dataset enriched with nutritional information," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 4001–4008.
- [69] W. Shao, W. Min, S. Hou, M. Luo, T. Li, Y. Zheng, and S. Jiang, "Vision-based food nutrition estimation via rgb-d fusion network," *Food Chemistry*, vol. 424, p. 136309, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0308814623009275>
- [70] Y. Han, Q. Cheng, W. Wu, and Z. Huang, "Dpf-nutrition: Food nutrition estimation via depth prediction and fusion," *Foods*, vol. 12, no. 23, p. 4293, 2023.
- [71] S. Gull, I. S. Bajwa, W. Anwar, and R. Rashid, "Smart enose food waste management system," *Journal of Sensors*, vol. 2021, no. 1, p. 9931228, 2021.
- [72] Kitro, "Kitro," 2025, accessed: 2025-03-23. [Online]. Available: <https://www.kitro.ch/>
- [73] Winnow Solutions, "Winnow solutions," 2025, accessed: 2025-03-23. [Online]. Available: <https://www.winnowsolutions.com/>
- [74] Orbisk, "Orbisk," 2025, accessed: 2025-03-23. [Online]. Available: <https://orbisk.com/>

References

- [75] Positive Carbon, "Positive carbon," 2025, accessed: 2025-03-23. [Online]. Available: <https://www.positivecarbon.org/>
- [76] LeanPath, "Leanpath - food waste prevention solutions," 2025, accessed: 2025-03-23. [Online]. Available: <https://www.leanpath.com/>
- [77] Lumitics, "Lumitics - intelligent solutions to reduce food waste," 2025, accessed: 2025-03-23. [Online]. Available: <https://lumitics.com/>
- [78] G. Bradski, "The opencv library," *Dr. Dobbs's Journal of Software Tools*, 2000.
- [79] A. Clark and Contributors, "Pillow (pil fork)," <https://pillow.readthedocs.io/en/stable/>, 2015.
- [80] A. Paszke, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.
- [81] T. maintainers and contributors, "Torchvision: Pytorch's computer vision library," <https://github.com/pytorch/vision>, 2016.
- [82] P. Yakubovskiy, "Segmentation models pytorch," https://github.com/qubvel/segmentation_models.pytorch, 2019.
- [83] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.
- [84] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*. SciPy, 2010, pp. 51–56.
- [85] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
- [86] M. L. Waskom, "Seaborn: Statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021.
- [87] P. T. Inc. (2015) Collaborative data science. Montreal, QC. [Online]. Available: <https://plot.ly>
- [88] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024. [Online]. Available: <https://arxiv.org/abs/2408.00714>
- [89] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

References

- [90] imec IDLab, "Obelisk core documentation," 2025, accessed: 2025-05-08. [Online]. Available: <https://obelisk.pages.ilabt.imec.be/obelisk-core/obelisk-core.html>
- [91] Grafana Labs, "Grafana documentation," 2025, accessed: 2025-05-08. [Online]. Available: <https://grafana.com/docs/>
- [92] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [93] United Nations General Assembly, "The 17 goals," <https://sdgs.un.org/2030agenda>, 2025.
- [94] OpenAI, "Chatgpt," 2025. [Online]. Available: <https://chat.openai.com/>
- [95] Anthropic, "Claude language model," <https://www.anthropic.com/claude>, 2025, accessed: 2025-04-16.
- [96] GitHub, "Github copilot," 2021. [Online]. Available: <https://copilot.github.com/>

Appendices

Attachment A - Use of Generative AI

This appendix details the utilization of generative AI tools throughout the research and writing process of this thesis. Transparency regarding the use of such technologies is crucial for upholding academic integrity.

In this thesis, I have made use of generative AI tools such as ChatGPT [94] and Claude [95] to support various non-sensitive aspects of the writing process. These tools were used across different parts of the thesis, primarily to enhance the clarity, coherence, and structure of the text. Importantly, for every section of the thesis, I first composed an original version myself, ensuring that the core content, structure, and arguments reflected my own insights and intentions. The AI tools were then used to refine these drafts, rephrase complex passages for improved readability, suggest more effective formulations, and provide alternative phrasings that enhanced the flow and academic tone of the writing. Additionally, I used these tools to get ideas of what certain parts should definitely contain. Such as for the extended abstract, I asked how a good extended abstract looks like and what I should write down. In all cases, the final content was critically reviewed and edited by myself to ensure that it represented my ideas and research contributions.

I also leveraged AI tools for technical troubleshooting, including error resolution and software version conflicts, which helped maintain development momentum during challenging phases.

Moreover, I relied on generative AI for preliminary explorations in areas like regression models. For example, when researching available regression models, I used AI tools to get a list of potential models, assisting me in identifying suitable options. Although I supplemented this exploration with my own research, including searches on Google, the AI tools helped by providing a solid starting point, ensuring the correct terminology was applied, and aiding in a deeper understanding of key concepts. An example prompt is: "Can you provide a list of popular regression models used in machine learning, including a brief description of each model and its advantages and disadvantages?"

In adherence to the confidentiality agreement signed, I ensured that no sensitive or confidential data was uploaded to generative AI tools. The document explicitly prohibited the use of such tools with sensitive information, and I followed these guidelines strictly. To further safeguard confidentiality, I also disabled GitHub Copilot [96] during my work on this thesis to ensure that no unintended exposure of sensitive information could occur. This was a precautionary measure to maintain compliance with the ethical and privacy standards set by Ghent University.

By responsibly using AI tools in non-sensitive tasks, I was able to enhance my workflow and improve various aspects of my research, all while adhering to the ethical and privacy guidelines outlined by Ghent University.

Attachment B - Code Availability

The source code used for this thesis is available at this GitHub repository¹.

¹<https://github.com/AxellePenninger/thesis-food-leftovers>