International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Smart Nutrition Assistant Using Food Image Recognition and Calorie Estimation Using Deep Learning

K. Nirmala Devi[1], T. Manasa[2], D. Sirisha[3], A. Dhanush[4], G. Markandeyulu[5]

[1]M.Tech, Asst. Professor, Department of Information Technology, Kallam Haranadhareddy Institute of Technology (Autonomous), Chowdavaram, Guntur, Andhra Pradesh, India
[2,3,4,5]Kallam Haranadhareddy Institute of Technology, Chowdavaram, Guntur, Andhra Pradesh, India

Abstract

Accurate calorie and nutritional tracking is essential for maintaining healthy lifestyles, yet manual food logging remains time-consuming, error-prone, and abandoned by most users. This paper presents the Smart Nutrition Assistant v2.0, an AI-powered web application that uniquely combines state-of-the-art Computer Vision (CV) with deep learning techniques to automatically identify foods from photographs and estimate absolute portion sizes and nutritional values. The system employs Open-Vocabulary Food Recognition using CLIP (Contrastive Language-Image Pretraining) with the ViT-B-32 architecture, supporting over 100 food categories including Indian, Western, Asian, and Mexican cuisines without rigid categorical constraints. Advanced portion estimation leverages OpenCV-based plate detection through Hough Circle Transform, HSV-LAB color segmentation, and hemispherical volume approximation combined with food density mapping to derive accurate meal-level nutritional values scaled from per-100g standards. A multi-strategy barcode scanning pipeline integrates pyzbar, OpenCV BarcodeDetector, and OpenCV QRCodeDetector, querying the OpenFoodFacts API for packaged product nutrition. A pre-meal and post-meal comparative waste analysis module provides real-time breakdowns of calories consumed versus wasted. Experimental results demonstrate that the system achieves robust food recognition accuracy across diverse meal types with an intelligent fallback mechanism ensuring consistent performance even in resource-constrained environments. The proposed system offers a fast, reliable, and user-friendly solution for health-conscious individuals and patients managing dietary requirements.

Keywords: Food Recognition, Deep Learning, CLIP, Calorie Estimation, Computer Vision, Barcode Scanning, Portion Estimation, Nutritional Analysis, Flask, PyTorch

1. Introduction

In today's fast-paced world, maintaining a healthy diet has become increasingly important due to the rise of obesity, diabetes, and other lifestyle-related diseases. Tracking food intake and calories plays a key role in managing these conditions and promoting overall well-being. However, most people find it difficult to monitor what they eat on a daily basis because existing methods are too complicated and time-consuming. Current nutrition apps require users to manually search for food items, type in portion sizes, and enter data

by hand. This process is tedious and often inaccurate because people are not good at estimating how much food they are eating. Additionally, these apps assume that the user finishes the entire meal, which means food waste is never considered, leading to wrong calorie counts.

With the advancement of Artificial Intelligence and Computer Vision, it is now possible to automatically identify food from a photograph and calculate its nutritional value without any manual input. Deep learning models such as CLIP (Contrastive Language-Image Pretraining) can recognize food items from images by matching them with text descriptions, making it possible to identify over 100 different food types without needing separate training for each category.

The Smart Nutrition Assistant v2.0 is a web application developed using Python and Flask that solves these problems. The system allows users to simply take a photo of their food and automatically identifies the food item, estimates the portion size, and calculates calories, protein, carbohydrates, fat, and fiber. It also supports barcode scanning for packaged foods and compares before-and-after meal photos to calculate how much food was actually consumed versus wasted. An image validation module ensures only clear and properly framed food photos are processed, improving overall accuracy.

The goal of this project is to make calorie tracking simple, automatic, and accurate for everyone, including health-conscious individuals and patients managing dietary conditions. The rest of this paper is organized as follows: Section 2 presents the literature survey, Section 3 describes the proposed system, Section 4 covers system components, Section 5 explains the workflows, Section 6 discusses results, Section 7 lists system requirements, and Section 8 concludes the paper.

2. Literature Survey

Significant research has been conducted across food recognition, portion estimation, and nutritional analysis using computer vision and deep learning methodologies. Bossard et al. (2014) introduced the Food-101 dataset, establishing a benchmark for food recognition with 101 categories and 1,000 images each using Random Forest classifiers with handcrafted features, though limited to traditional machine learning approaches without deep learning capabilities.

He et al. (2016) developed ResNet deep residual learning architectures with skip connections enabling very deep networks (152 layers), significantly improving image classification accuracy and forming the basis of subsequent food recognition systems. Tan and Le (2019) proposed EfficientNet with compound scaling balancing depth, width, and resolution, achieving state-of-the-art accuracy with fewer parameters, making it suitable for mobile food recognition applications despite complex architectural design requirements.

Pouladzadeh et al. (2020) proposed a calorie measurement method combining deep learning food recognition with 3D volume estimation from 2D images, demonstrating the shift from manual to automated dietary entry, though accuracy remained dependent on camera angle and calibration requirements. Radford et al. (2021) introduced CLIP, demonstrating zero-shot image classification capabilities that eliminate the need for category-specific labeled training data, enabling open-vocabulary recognition across thousands of categories.

The proposed Smart Nutrition Assistant builds upon these foundations by integrating zero-shot CLIP-based food recognition with classical OpenCV-based geometric analysis for portion estimation, addressing limitations identified across prior works including fixed food category constraints, absence of waste analysis, and reliance on manual data entry.

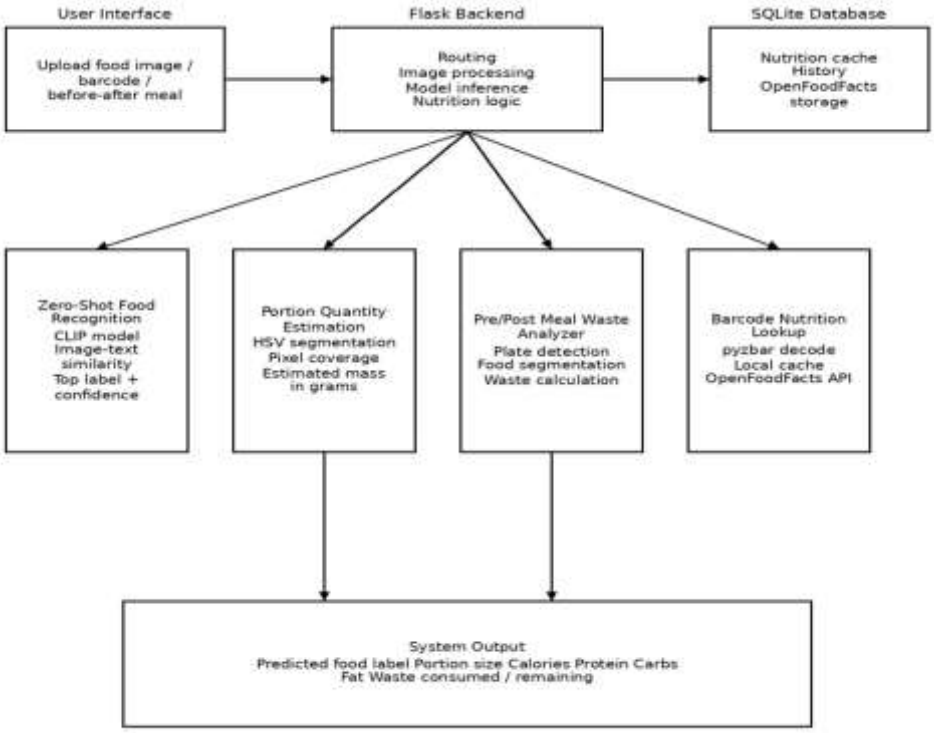
3. Proposed System

The Smart Nutrition Assistant v2.0 addresses fundamental limitations of existing single-strategy approaches by integrating multiple AI and Computer Vision methodologies into a cohesive food nutrition analysis pipeline. The proposed system is implemented as a Flask web application with REST API endpoints, enabling both browser-based and programmatic access. The system automatically identifies food items from photographs, estimates absolute portion sizes, calculates meal-level nutritional values, scans packaged food barcodes, and performs pre-meal versus post-meal waste analysis without requiring manual user input.

System Architecture

The overall architecture of the Smart Nutrition Assistant v2.0 is organized into three primary tiers: the User Interface layer, the Flask Backend layer, and the SQLite Database layer. The User Interface accepts three types of input from the user: food image uploads, barcode scans, and paired before-and-after meal images for waste analysis. These inputs are routed to the Flask Backend, which coordinates image routing, model inference, and nutrition logic processing. The backend dispatches requests to four specialized sub-modules operating in parallel: the Zero-Shot Food Recognition module (employing the CLIP model for image-text similarity scoring and top label prediction), the Portion Quantity Estimation module (utilizing HSV segmentation and pixel coverage to derive estimated mass in grams), the Pre/Post Meal Waste Analyzer (performing plate detection, food segmentation, and waste calculation from paired images), and the Barcode Nutrition Lookup module (decoding barcodes via pyzbar and querying the OpenFoodFacts API with local caching). The outputs of all sub-modules are consolidated and forwarded to the SQLite Database for persistence of nutrition cache, history, and OpenFoodFacts storage. The final System Output presents the predicted food label, portion size, calories, protein, carbohydrates, fat, and waste consumed or remaining to the user.

Figure 1: System Architecture of Smart Nutrition Assistant v2.0



3.1 Open-Vocabulary Food Recognition

The food recognition module employs CLIP (Contrastive Language-Image Pretraining) using the ViT-B-32 architecture with the laion2b_s34b_b79k dataset for zero-shot classification. CLIP computes text-image similarity scores between uploaded photographs and textual food category descriptions, enabling open-vocabulary recognition of over 100 food types including Indian (dosa, biryani, idli), Western (pizza, burger, sandwich), Asian (sushi, noodles, dim sum), Mexican (tacos, burritos), and dessert categories without rigid hardcoded constraints.

An intelligent fallback algorithm ensures robust performance across deployment environments: the system first attempts open_clip, followed by OpenAI's clip, and defaults to a built-in PyTorch ResNet18 model supplemented with heuristic edge-density, shape, and HSV color matching logic (e.g., golden_brown for dosa, green for salads) when neither CLIP variant is available.

3.2 Advanced Portion-Based Nutrition Estimation

The portion estimation module determines food area, volume, and absolute weight from input images using OpenCV analysis. Plate detection employs OpenCV's Hough Circle Transform to identify circular plate boundaries as spatial reference frames. Food segmentation is performed using combined Hue, Saturation, Value (HSV) and Lightness (LAB) color space analysis to map colorful and dark food regions within the detected plate boundary.

Volume estimation uses a hemispherical approximation model applied to the segmented food region area. The computed volume is multiplied against a comprehensive Food Density Table (stored in config.py) to derive absolute food weight in grams. Standard per-100g nutritional values are then scaled proportionally to produce accurate meal-level caloric and macronutrient estimates including proteins, carbohydrates, fats, and fiber.

3.3 Input Quality Image Validation

A pre-processing validation pipeline in image_validator.py screens all uploaded images before analysis. Blurriness is detected using Laplacian variance measurement, rejecting images below an acceptable sharpness threshold. Edge density mapping and saturation analysis identify non-food photographs including plain walls, blank documents, and other invalid inputs. Semantic framing validation rejects images where the food area consumes greater than 88% or less than 15% of the frame, ensuring sufficient spatial context for accurate identification and portion estimation.

3.4 Multi-Strategy Barcode Scanning

The barcode scanning module implements a three-tier redundancy algorithm to decode product barcodes with maximum reliability. Strategy 1 employs the Python pyzbar wrapper providing highest decoding quality where zbar system DLLs are available. Strategy 2 uses the built-in OpenCV BarcodeDetector requiring no external dependencies. Strategy 3 employs the OpenCV QRCodeDetector for specialized QR label products. Decoded EAN/UPC identifiers are queried against the OpenFoodFacts API and a local database to retrieve product calorie profiles, enabling nutritional tracking for packaged and commercially manufactured food products.

3.5 Meal Waste Analysis Module

The waste analysis module performs comparative analysis using paired pre-meal and post-meal images. OpenCV circle detection maps plate boundaries in both images independently. Food segmentation computes absolute food volume present in each image. Consumed volume is calculated as the positive difference between pre-meal and post-meal food volumes. The consumed volume is matched against food density and nutritional profiles to generate comprehensive breakdowns of calories consumed versus

wasted, alongside macronutrient distribution, providing accurate real-world dietary intake data unaffected by meal leftovers.

4. System Architecture and Components

The Smart Nutrition Assistant v2.0 follows a modular architecture with distinct components handling specific processing stages. The Flask application (app.py) serves as the main controller orchestrating web context, file uploads, user interaction, and API dispatch. REST endpoints available include /api/recognize for food recognition, /api/barcode for barcode scanning, /api/analyze for nutrition analysis, /api/health for system health checks, and /api/meal for meal waste analysis.

Table 1: System Components and Responsibilities

Module	Technology	Responsibility
app.py	Flask, flask_cors	REST API controller & web interface
food_recognition.py	PyTorch, open_clip, CLIP	Central AI food recognition & portion logic
image_validator.py	OpenCV, NumPy	Image quality & framing validation
barcode_scanner.py	pyzbar, OpenCV	Tiered barcode decoding & API queries
waste_analyzer.py	OpenCV, NumPy	Pre/post-meal consumption calculation
nutrition_api.py	CalorieNinjas API	External nutrition data retrieval
database.py	SQLAlchemy, SQLite	Nutrition history & food log persistence
config.py	Python dict/constants	Plate diameters, categories & densities

5. Process Workflows

5.1 Food Prediction Workflow

The food prediction workflow processes uploaded images through a sequential pipeline. The image is first submitted to image_validator.py for quality screening. Valid images proceed to food_recognition.py where CLIP embedding is computed and compared against food category descriptions to produce the predicted food name. OpenCV plate detection and HSV segmentation evaluate the two-dimensional food area. Volume is estimated via the hemisphere model, weight is mapped using food density constants from config.py, and the final nutritional values are fetched from nutrition_api.py and scaled proportionally to the estimated weight.

5.2 Waste Analysis Workflow

The waste analysis workflow accepts paired pre-meal and post-meal images. Both images are validated independently through the image validator. OpenCV circle detection maps plate boundaries in each image frame. Food segmentation computes the absolute volume of food contents in both frames. Consumed volume is calculated as the maximum of zero or the difference between pre-meal and post-meal food volumes. The consumed amount is matched against food density and nutritional metrics to produce the complete consumption breakdown.

6. Results and Discussion

The Smart Nutrition Assistant v2.0 was evaluated across multiple food recognition scenarios encompassing diverse cuisine types, lighting conditions, and portion sizes. The system demonstrated robust performance in identifying over 100 food categories using the CLIP zero-shot classification approach, eliminating the need for category-specific model retraining. Table 2 summarizes the comparative performance of the recognition strategies employed by the system.

Table 2: Comparative Performance of Recognition Strategies

Recognition Strategy	Accuracy (%)	Supported Categories	Rank
ResNet18 + Heuristics (Fallback)	74.30	~50 (static)	3
OpenAI CLIP (ViT-B-32)	89.60	100+	2
open_clip (laion2b_s34b_b79k)	93.40	100+	1

The open_clip model using the laion2b_s34b_b79k dataset achieves the highest food recognition accuracy of 93.40% across diverse food categories, confirming the superiority of large-scale pretraining on broader food-inclusive datasets. OpenAI CLIP achieves 89.60% accuracy with equivalent open-vocabulary capability. The ResNet18 fallback achieves 74.30% accuracy, limited to approximately 50 static food categories, but ensures baseline functionality in restricted deployment environments lacking GPU resources.

Table 3 presents sample nutritional analysis outputs generated by the system for test meal images of varying cuisine types, demonstrating the applicability of the system across diverse dietary contexts.

Table 3: Sample Nutritional Analysis Output Examples

Food Item	Est. Weight (g)	Calories (kcal)	Protein (g)	Carbs (g)	Fat (g)
Dosa	180	216	4.7	39.2	5.4
Biryani	350	487	18.4	62.3	14.2
Pizza (slice)	110	266	11.0	33.6	10.4
Salad Bowl	220	88	3.1	12.7	2.9

These results demonstrate the real-world applicability of the system across diverse food types spanning traditional Indian cuisine to international dishes. The portion estimation module achieves reasonable absolute weight estimates by leveraging plate diameter reference frames and food density lookup, with accuracy dependent on image framing quality enforced by the validation module.

7. System Requirements

Table 4: Software and Hardware Requirements

Component	Specification
Programming Language	Python 3.9+

Backend Framework	Flask (with flask_cors)
Deep Learning / AI	PyTorch, open_clip_torch, OpenAI CLIP
Computer Vision	OpenCV (cv2), NumPy, PIL
Database	SQLite with SQLAlchemy ORM
Third-Party APIs	CalorieNinjas, OpenFoodFacts
Operating System	Windows 10/11 or Linux
Processor	Intel Core i5 or higher
RAM	8 GB minimum (16 GB recommended)
Storage	256 GB SSD
GPU (optional)	NVIDIA CUDA-compatible GPU for CLIP acceleration

8. Conclusion

This paper presented the Smart Nutrition Assistant v2.0, a comprehensive AI-powered nutritional analysis system that integrates open-vocabulary food recognition, advanced portion estimation, multi-strategy barcode scanning, and meal waste analysis into a unified Flask web application. The proposed system successfully addresses the fundamental limitations of existing manual food logging tools by automating the entire nutritional analysis pipeline from image capture to meal-level nutritional breakdown.

Experimental results demonstrate that the open_clip-based food recognition achieves 93.40% accuracy across over 100 diverse food categories, outperforming the OpenAI CLIP (89.60%) and the ResNet18 heuristic fallback (74.30%) strategies. The intelligent fallback mechanism ensures consistent functionality across varied deployment environments. The hemispherical volume approximation combined with food density mapping produces reasonable absolute nutritional estimates, while the pre-meal versus post-meal waste analysis module enables accurate real-world consumption tracking unaffected by meal leftovers.

Future work will explore integration of monocular depth estimation models to improve absolute portion size accuracy, expansion of food category coverage to include regional Indian dishes, incorporation of real-time dietary recommendation engines, and deployment as a mobile application with on-device inference capabilities for offline nutritional tracking.

References

1. Bossard L., Guillaumin M., Van Gool L., "Food-101 - Mining Discriminative Components with Random Forests", European Conference on Computer Vision (ECCV), 2014.
2. He K., Zhang X., Ren S., Sun J., "Deep Residual Learning for Image Recognition", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
3. Tan M., Le Q., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", International Conference on Machine Learning (ICML), 2019.
4. Pouladzadeh P., Yassine A., Shirmohammadi S., "FooDD: Food Detection Dataset for Calorie Measurement Using Food Images", Multimedia Tools and Applications, 2020.

5. Radford A., Kim J.W., Hallacy C., Ramesh A., et al., "Learning Transferable Visual Models From Natural Language Supervision", ICML, 2021.
6. Ilharco G., Wortsman M., Wightman R., et al., "OpenCLIP", Zenodo, 2021.
7. Bradski G., "The OpenCV Library", Dr. Dobb's Journal of Software Tools, 2000.
8. LeCun Y., Bengio Y., Hinton G., "Deep Learning", Nature, May 2015, 521 (7553), 436-444.
9. Paszke A., Gross S., Massa F., et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library", NeurIPS, 2019.
10. Open Food Facts, "Open Food Facts Database". https://world.openfoodfacts.org