International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 20, No. 1, 2024
© 2024 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

*Research Paper*

# NutriVision AI: An Intelligent Food Recognition and Calorie Estimation System

**Krishna Chaitanya[1],N.Manoj[2],Shivakumar[3],Sneha[4],**

**Dr.P.RAMESH BABU[5],Mrs.V.LAKSHMI SWATHI[6, 5]Assistant Professor[6],Department of CSE(CS)**

**Narasimha Reddy College Of Engineering(Autonomous), Hyderabad, Telangana - 500100.**

**Abstract—**Manual tracking of daily calorie intake is time-consuming, difficult and less likely to be adhered to by individuals with obesity, diabetes, or cardiovascular disease risk. In this paper, we present an automatic pipeline for dietary assessment based on images, which classifies the image content into fruits, vegetables and a plate of food and calculates the calorie content of the food. Instead of using depth sensor or a large fiducial board, the method involves placing a regular Indian coin, 1, 2 or 5 rupees, beside the plate, which allows contour routines in OpenCV to determine the actual density in the real world in pixels per square inch. The captured RGB images are downsampled to 64 × 64 pixels and segmented from the background using iterative GrabCut graph cuts with Gaussian Mixture Models. The segmented regions are then fed into a Convolutional Neural Network (CNN) architecture comprising of ReLU activation functions, 2 × 2 max pooling filters, dense hidden layers and dropout regularization with a rate of 0.2. The number of pixels within each food face is translated into surface area in real world and compared with food nutritional reference tables, to present the per item calorie count and disease-relevant dietary recommendations through a web portal.

Keywords—Calorie estimation, Convolutional Neural Network ( CNN), dietary assessment, food image classification, GrabCut segmentation, OpenCV, reference object calibration.

## I. INTRODUCTION

Over-consumption of calories is now a common public health problem. The estimated overweight population of the adult population (aged 18 years and older) is approximately 1.9 billion [15], with the prevalence of obesity among children and adolescents being on the rise as a result of increased consumption of calorie-dense processed foods. Long-term energy imbalance has been associated with the increased risk of type-2 diabetes, hypertension, stroke, renal failure and some cancers [1, 14].

To manage how much you eat each day, it is important to have an understanding of the number of calories in a meal before you consume it. However, in real life, patients and dietitians stick to written food diaries, printed portion charts or strict meal plans. Those hand-written logs require constant attention, crumble during meals of home-cooked or unpacked produce, and are often under-ambitious when it comes to portion size.

Smartphones now make consumers' pockets packed with high quality cameras, making visual food logging a practical choice. A vision pipeline can extract a meal image, process it to clean and resize, separate the food from the plate, classify the food using a Convolutional Neural Network (CNN), and find the nutritional density of the food [6],[11],[14],[18]. It leverages the Kaggle and open-source GitHub datasets of labeled fruit and vegetable images and meal images to create a web-connected system using CNN classification and coin-calibrated GrabCut area estimation.

## II. LITERATURE SURVEY

The research of automated food recognition can be grouped into three generations: hand-crafted local descriptors and classic classifiers, fuzzy or metaheuristic clustering and deep neural networks. In the mobile space, Kawano and Yanai developed FoodCam [16] based on color histogram, Bag-of-SURF descriptors, as well as using one-vs-rest linear Support Vector Machines (SVMs). Subsequently, they used 32-component Gaussian Mixture Model (GMM) Fisher Vectors compressed using PCA to 1536 dimensions,

reducing the memory overhead without compromising the features' performance for smartphones [17]. Velvizhy et al. [13] took a similar approach with their diabetic patients, which involved clustering Dense SIFT keypoints using K-means followed by a SVM classification of carbohydrate levels.

Portion calibration is as difficult as visual classification. Pouladzadeh et al. [14] requested the users to put a printed calibration card next to each meal, which they then mapped to nutritional tables. To avoid adding extra cards, Kuhad et al. [3] resized the block using deep learning, SVM and hand-finger size to achieve 97% accuracy; however, there is scale drift due to the different size of finger and chopstick among the users. We explored the 3D reconstruction from depth sensors (Kinect) and dual-view planar image in the dataset of ECUSTFD using a 1 RMB coin reference, which achieved $R^2 = 0.95$ and RMSE = 43. It's true, but most consumer phones don't have depth cameras.

In the spirit of the foregoing, several researchers resorted to hybrid clustering and bio-inspired optimization. To achieve 97 % accuracy, Minija et al. [2] fused the texture, color and local neighborhood statistics over stable segments and for 96 % accuracy, they used the Imperialist Competitive Algorithm Deep Belief Network (IpCA-DBN) with Bayesian Fuzzy Clustering (BFC) [7]. Similarly, Samemmanuel et al. [8] used the combination of CSW-WLIFC fuzzy segmentation and Whale Levenberg-Marquardt Neural Network (WLM-NN) for day-to-day images taken of trays, yielding 96.27% standard accuracy.

More recent systems are based directly on the convolutional architecture. Hu and Zhang [1] labeled Chinese and Western dishes using labelImg and then trained a Single Shot MultiBox Detector (SSD) to look up the calorie content of foods in real-time. When testing custom CNNs trained from scratch, Yigit and Ozyildirim [6] found that roughly 2.5 hours of training time per run were required for Food-11 and Food-101. Situju et al. [4] proposed a two-stage multi-task CNN (92.6% accuracy) that was trained to predict food category, calories and salinity at the same time, but the error in calories prediction was higher for the foods with large intra-class variance. To reduce mobile latency and battery drain, Liu et al. [11] offloaded CNN inference to edge servers, while Meyers et al. [18] (Im2Calories), Ege and Yanai [19], and Ciocca et al. [12] tackled multi-item canteen trays. These representative methods are compared with each other in Table I.

**TABLE I**

*Comparative Summary of Existing Food Recognition and Calorie Estimation Methods*

| Author(s) & Year | Methodology / Model | Dataset / Ref. | Accuracy / Metric |
|---|---|---|---|
| Hu & Zhang (2020) [1] | SSD Object Detection + labelImg | Chinese & Western dishes | Real-time dish & calorie ID |
| Situju et al. (2019) [4] | Multi-Task CNN + 2-Stage Transfer | Recipe-annotated corpus | 92.6% accuracy |
| Minija et al. (2019) [7] | Bayesian Fuzzy Clustering + IpCA-DBN | Segmented food images | 96.0% accuracy |
| Samemmanuel (2018) [8] | CSW-WLIFC + WLM-NN classifier | Daily meal tray images | 96.27% standard accuracy |
| Liang & Li (2017) [5] | Faster R-CNN + GrabCut + 3D/planar | ECUSTFD (1 RMB coin ref.) | $R^2 = 0.95$, RMSE = 43 |
| Yigit & Ozyildirim (2018) [6] | DCNN (SGD, NAG, Adam) | Food-11 and Food-101 | Comparable to AlexNet |
| Kuhad et al. (2015) [3] | Deep Learning + SVM + Block Resize | Finger-based distance ref. | 97.0% accuracy |
| Kawano & Yanai (2014) [17] | GrabCut + Fisher Vector + Linear SVM | UEC-FOOD100 (100 classes) | 79.2% top-5 mobile accuracy |

## III. MOTIVATION, SCOPE, AND PROBLEM STATEMENT

### *A. Motivation and Scope*

Numerous current calorie counting prototypes fail under uncontrolled laboratory environments. Hough transform based techniques and two-view point matching require extensive pre-processing and specific camera views, while AR/MR headsets or Kinect sensor are not feasible to use at the dinner table. The project's scope is a web-accessible assessment tool which accepts user submitted 2D photographs of average meals, considers odd shapes and irregular geometries of fruits and vegetables and uses color, texture, shape and pixel size to estimate calories rapidly.

B. Problem Statement

The system is required to perform four interconnected tasks: (1) correct spatial scale (2) perform graph-cut segmentations that separate the food items from the plate and background, (2) classify the fruits, vegetables and meal classes from a single RGB photograph using a CNN trained on data from Kaggle and GitHub, and (4) determine the calorie content by converting segmented pixel counts to calorie estimates using standard nutritional tables.

IV. PROPOSED METHODOLOGY

B. Addressing the Memory Hierarchy with a Primitives Library

The processing pipeline consists of 6 parts: object registration for reference, image acquisition and pre-processing, edge detection and GrabCut segmentation, CNN classification, surface area approximation, and calorie lookup and disease oriented dietary recommendations. The energy densities are retrieved from a paired calorie database by the estimator, and the classifier searches a food database (see Fig. 1).

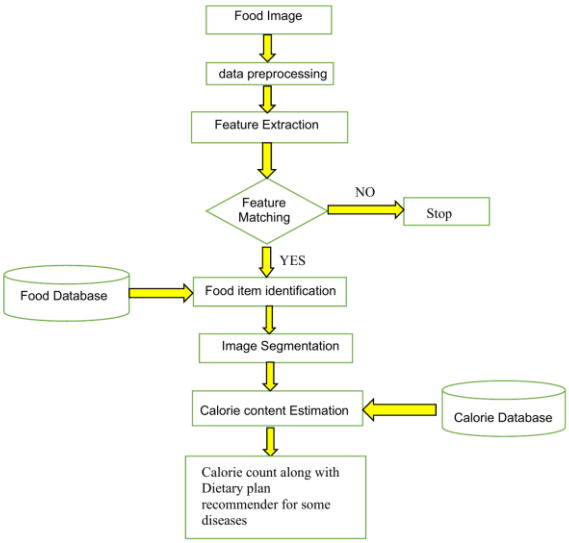


Fig. 1. System flowchart illustrating food recognition, segmentation, and calorie estimation.

### *B. Reference Object Registration*

To remove any scale ambiguity, a standard coin of 1 rupee, 2 rupees or 5 rupees is placed alongside the plate (Fig. 2). Minted coins are not fingers or chopsticks; they all have the same diameter. With the help of two tools, cv2 and imutils, along with scipy, numpy, and argparse, the following openCV pipeline detects the coin contour and saves its pixel area, providing a reliable calibration factor of the number of pixels per number of square inches on the entire image.



Fig. 2. Plate and coin reference object registration for spatial scale calibration.

C . Image acquisition and pre-processing

The RGB photos are uploaded and input into the network as a fixed-sized tensor, 64 x 64 x 3.The RGB photographs are uploaded and given to the network as a fixed size tensor 64 x 64 x 3. Maintaining the spatial resolution of 64 × 64 reduces the number of parameters and training time, as well as making it easier to process images from various Kaggle and GitHub data sets.

D. GrabCut Image Segmentation

We use the GrabCut algorithm [20] to isolate food items in the foreground from the plate. The food region is enclosed in a bounding box by a user or detector, with the pixels outside the box being set as the background, and the pixels inside the box being tentative. GrabCut uses a pair of Gaussian Mixture Models (GMMs) to model the foreground and background color distributions, and constructs an s-t graph, where each pixel is connected to a Source (foreground) node, a Sink (background) node, and the nodes of its surrounding pixels. Low weights are given to neighbor edges on sharp color boundaries. The iterative minimum cut optimization processes the graph until pixel labels converge (Fig. 3), by repeatedly dividing it at the minimum cost.

### *C. Image Acquisition and Pre-processing*



Fig. 3. GrabCut segmentation applied to a fruit bowl (input left, segmented mask right).

### *E. CNN Classification Architecture*

Segmented food crops pass into a custom Convolutional Neural Network organized into an input layer, alternating convolution and pooling blocks, dense hidden layers, and a classification output layer (Fig. 4).

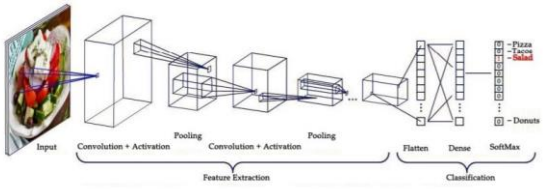


Fig. 4. Custom CNN pipeline showing convolution, pooling, flattening, dense, and classification stages.

1) Input and Convolutional Layers: The network accepts 64×64×3 RGB tensors. For each convolutional stage, it randomly initializes nf 3D filters of size f × f, with stride s and padding p, and applies them to all channels, either by zero-padding the image to maintain the map size ('same' input), or by ignoring all the image padding ('valid' input).

2) ReLU Activation and Max Pooling: Each convolution output has a Rectified Linear Unit (ReLU) applied to it, which helps to prevent vanishing-gradient saturation seen in sigmoid and tanh units, and also introduces non-linearity. The features are then downsampled by 2 × 2 Max Pooling with stride 2 (Fig. 5), and only the maximum activation at each 2 × 2 window is retained, without any trainable weights.

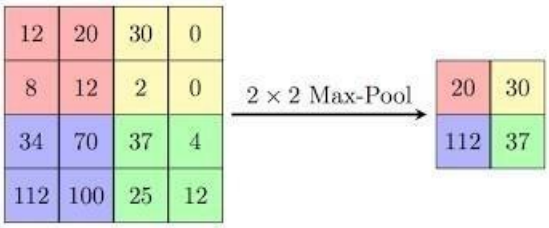


Fig. 5. Downsampling a 4 × 4 feature map to 2 × 2 using 2 × 2 Max Pooling with stride 2.

3) Flattening, Dense Layers, Dropout, and Output: The final pooled volume is flattened into a 1D vector feeding two fully connected blocks: Dense(128, relu) and Dense(50, relu). Dropout at rate p = 0.2 disables 20% of hidden units at random during each training step (Fig. 6) to curb overfitting, and is switched off at test time. The output layer uses Sigmoid (0.5 decision threshold) for binary checks or Softmax for multi-class fruit and vegetable recognition.

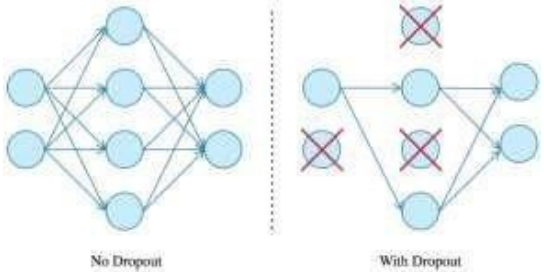


Fig. 6. Standard fully connected network (left) versus Dropout regularization during training (right).

### *F. Surface Area Approximation and Calorie Estimation*

Once classified, the system counts the number of foreground pixels within the GrabCut mask, divides by the pixel density measured from the coin and obtains the food surface area (in square inches). For foods that contain a fixed number of calories per unit (e.g., whole fruits, bread slices, yogurt cup) the pre-determined energy value in the database is returned; foods with variable energy per unit (e.g., mangoes, cookies, pizza) multiply a predetermined calorie per square inch by estimated surface area.

G. Hardware and Software Specifications

Both local development and model training is performed on a 64-bit Windows workstation, with model training performed on a 16 GB NVIDIA Tesla P100 GPU on Google Colab. The full hardware/software environment is given in Table II.

**TABLE II**

*Hardware and Software Requirements Specification*

| Component | Specification / Version |
|---|---|
| Operating System | Windows 10 Pro (64-bit) |
| Processor & Memory | Intel Core i5 CPU, Minimum 4.00 GB RAM |
| Storage | 10 GB+ HDD (SSD recommended) |
| Cloud Training GPU | Google Colab NVIDIA Tesla P100 (16 GB, CUDA) |
| Core Language & IDE | Python 3.9.5 (64-bit), PyCharm Community Ed. |
| DL & Vision Libraries | TensorFlow, Keras, OpenCV, NumPy, SciPy |
| Web Frontend | HTML, CSS, JavaScript (dynamic logging app) |

## V. MATHEMATICAL FORMULATION

The following are formal definitions of the core operations that the classifier, graph-cut segmenter and calorie estimator perform.

1) 2D Discrete Convolution and Output Geometry: If X is an input tensor with size W × H × C, and a 3D filter W_k, the convolved feature map Y is:

$$Y(i, j, k) = \Sigma c\, \Sigma m\, \Sigma n\, X(i \cdot S + m, j \cdot S + n, c) \cdot Wk(m, n, c) + bk \quad (1)$$

Here, stride S and filter k's bias bk are used. The output spatial dimension, O is:

$$O = \lfloor (W - K + 2P) / S \rfloor + 1 \quad (2)$$

2) Non-Linear Activation and Max Pooling: ReLU that zeros out negative activations, 2 × 2 max pooling(F = 2, S = 2) that captures the best response in each window:

$$f(x) = \max(0, x) \quad (3)$$

$$Ypool(i, j, k) = \max(m,n) \in R\, X(m, n, k) \quad (4)$$

3) Classification Output and Loss: If the logits of C classes are denoted by z ∈ ℝC, then Softmax probabilities and categorical cross-entropy loss can be written as:

$$P(y = j \mid z) = \exp(zj) / \Sigma k=1..C \exp(zk) \quad (5)$$

$$L = - \Sigma i=1..C\, yi \log(\hat{y}i) \quad (6)$$

4) GrabCut Energy Minimization: GrabCut [20] degrades the RGB values of the pixels to be either 0 or 1 using the Gibbs energy of data term U and the smoothness term V for the boundary:

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z) \quad (7)$$

5) Calibrated Surface Area and Calorie Estimation: Let Nref be the number of pixels detected by the coins, Aref be the known physical area (in2) of the coins, and Nfood be the number of pixels segmented by the food. The parameters are given as the pixel density ρpx (in2/pixel), food area Afood, and the calories Cfood:

$$\rho px = Nref / Aref \quad (8)$$

$$Afood = Nfood / \rho px \quad (9)$$

$$Cfood = Afood \times Cinch \quad (10)$$

## VI. PHASE-1 RESULTS, DISCUSSION, AND FUTURE SCOPE

During the project phase 1, the literature baseline was established, the hybrid CNN plus GrabCut architecture was selected and the coin based calibration equations were formalised. By using Kinect depth sensors together with an average coin reference, instead of using the area of a bounded box, the hardware implementation is removed, allowing for the processing of irregular contours in fruit and vegetables, which is more difficult to process with a bounded box.

Phase 2 includes experimental model training on Tesla P100 GPU, a confusion-matrix evaluation across fruit/vegetable classes, and deployment on the live web-app. Top-1 and top-5 classification accuracy and calorie mean absolute error (MAE) using weighed ground-truth nutrition tables will be used as measurements for the next level of benchmarking, before personalized diet recommendations are delivered to diabetic and hypertensive users.

## VII. CONCLUSION

We proposed an automatic dietary assessment framework capable of recognizing the fruits and vegetables in 2D images and estimating the calorie content of the plates. The design utilizes a dropout-regularized CNN, GrabCut-based segmentation and OpenCV coin calibration, eliminating the need for human-curated input of food logs without the need for a depth camera. Combined with a web logging platform that offers disease-aware dietary advice, it offers a viable framework for everyday calorie counting.

## REFERENCES

[1] H. Hu, Z. Zhang, and Y. Song, "Image based food calories estimation using various models of machine learning," in Proc. 5th Int. Conf. Mechanical, Control and Computer Engineering (ICMCCE), Dec. 2020, pp. 874–878.

[2] S. J. Minija and W. R. S. Emmanuel, "Food image segmentation using multiple hypothesis for dietary assessment," Int. J. Comput. Appl., 2017.

[3] P. Kuhad, A. Yassine, and S. Shirmohammadi, "Using distance estimation and deep learning to simplify calibration in food calorie measurement," in Proc. IEEE Int. Conf. Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), Jun. 2015, pp. 1–6.

[4] S. F. Situju, H. Takimoto, S. Sato, H. Yamauchi, A. Kanagawa, and A. Lawi, "Food constituent estimation for lifestyle disease prevention by multi-task CNN," Appl. Artif. Intell., vol. 33, no. 8, pp. 732–746, Apr. 2019.

[5] Y. Liang and J. Li, "Computer vision-based food calorie estimation: Dataset, method, and experiment," arXiv preprint arXiv:1705.07632, May 2017.

[6] G. O. Yigit and B. M. Ozyildirim, "Comparison of convolutional neural network models for food image classification," J. Inf. Telecommun., vol. 2, no. 4, pp. 347–357, Mar. 2018.

[7] S. J. Minija and W. R. S. Emmanuel, "Imperialist competitive algorithm-based deep belief network for food recognition and calorie estimation," Soft Comput., Springer-Verlag, Jul. 2019.

[8] W. R. S. Emmanuel and S. J. Minija, "Fuzzy clustering and whale-based neural network to food recognition and calorie estimation for daily dietary assessment," Sadhana, Indian Academy of Sciences, vol. 43, no. 5, Art. 78, May 2018.

[9] H. Pooja and S. A. Madival, "Food recognition and calorie extraction using Bag-of-SURF and spatial pyramid matching methods," Int. J. Comput. Sci. Mobile Comput., vol. 5, no. 5, pp. 387–393, May 2016.

[10] S. Sasano, X.-H. Han, and Y.-W. Chen, "Food recognition by combined bags of color features and texture features," in Proc. 9th Int. Congr. Image and Signal Processing,

BioMedical Engineering and Informatics (CISP-BMEI), Oct. 2016, pp. 815–819.

[11] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, Y. Ma, S. Chen, and P. Hou, "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," IEEE Trans. Services Comput., vol. 11, no. 2, pp. 249–261, Mar./Apr. 2018.

[12] G. Ciocca, P. Napoletano, and R. Schettini, "Food recognition for dietary monitoring: A new dataset, experiments, and results," IEEE J. Biomed. Health Inform., vol. 21, no. 3, pp. 588–598, May 2017.

[13] P. Velvizhy, Pavithra, and A. Kannan, "Automatic food recognition system for diabetic patients," in Proc. 6th Int. Conf. Advanced Computing (ICoAC), Dec. 2014, pp. 329–334.

[14] P. Pouladzadeh, S. Shirmohammadi, and R. Al-Maghrabi, "Measuring calorie and nutrition from food image," IEEE Trans. Instrum. Meas., vol. 63, no. 8, pp. 1947–1956, Aug. 2014.

[15] H. Shiqi, L. Daizhi, and H. Mingxing, Handbook of Obesity, 2nd ed. Baton Rouge, LA, USA: Pennington Biomedical Research Center, 2004.

[16] Y. Kawano and K. Yanai, "Real-time mobile food recognition system," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2013, pp. 1–7.

[17] Y. Kawano and K. Yanai, "FoodCam: A real-time food recognition system on a smartphone," Multimedia Tools Appl., vol. 74, no. 14, pp. 5263–5287, Apr. 2014.

[18] A. Meyers et al., "Im2Calories: Towards an automated mobile vision food diary," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1233–1241.

[19] T. Ege and K. Yanai, "Image-based food calorie estimation using knowledge on food categories, ingredients and cooking directions," in Proc. Thematic Workshops ACM Multimedia, Oct. 2017, pp. 367–375.

[20] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut: Interactive foreground extraction using iterated graph cuts," ACM Trans. Graph., vol. 23, no. 3, pp. 309–314, Aug. 2004.