Personalized Nutritional Recommendations from Food Images Using a Multimodal GUI

Chase Clemence & Tyler Blue

2025-12-02

Table of contents

1 Abstract

Accurate dieting requires manual input from the individual, requiring them to carefully measure the portion, macros, and quality of their food. Recent developments in the deep learning space attempt to automate this process, leveraging convolutional neural network (CNN) architectures to predict food types with semantic image segmentation as well as food volume using pixel distances from the camera. However, no one has accurately solved this issue yet. We add the body of research by building a multimodal graphical user interface (GUI) that leverages a CNN to predict portion sizes and macros as well as a LLM wrapper that gives personalized dietary advice based on the food's content. The CNN is trained on the MM-Food-100K dataset - a state of the art food dataset containing information about ingredients used, portion size, nutritional profile, cooking method, and more. The developed CNN used ResNet18 pretrained weights, a regression head with a hidden layer of 512, and output layer of 5 to estimate portion size, calories, fat, carbohydrates, and protein given a picture of food. The LLM wrapper leverages OpenAI's GPT-4o mini model to generate personalized dietary advice based on food macros, physical characteristics, and dietary goals. This is integrated into a simple GUI that consolodates the process into a simple, easy to use platform. With this GUI, users are able to get real-time improvements to their diet, helping them achieve their goals faster.

2 Introduction

Automated food recognition and nutritional estimation are emerging research areas with potential to substantially improve public health. Today, accurately measuring dietary macronutrients still requires individuals to manually weigh ingredients, look up nutritional values, and log foods by hand. This process is time-consuming, error-prone, and often frustrating, causing many people to abandon nutrition tracking altogether. If automated solutions become widely accessible, dietary monitoring could become far more efficient, lowering the barrier to healthy eating and enabling more consistent long-term adherence. Deep learning — particularly convolutional neural networks (CNNs) — provide a promising pathway toward this goal. CNNs excel at extracting latent visual features from images, giving them the capacity to identify dishes, estimate portion sizes, and infer nutritional content directly from pixel information. When paired with an intuitive user interface, such systems could offer real-time macro predictions from a single food photograph, drastically simplifying the nutrition-tracking workflow. Motivated by these opportunities, our project aims to develop a comprehensive, Python-based graphical user interface (GUI) capable of predicting portion size along with protein, fat, and carbohydrate content using a CNN trained on images of food. The GUI also integrates a large language model (LLM) to provide personalized feedback based on the predicted macros. Together, these components form a prototype system for fast, automated nutritional assessment.

3 Related Work

Automatic food recognition is a challenging problem because food images lack consistent geometry or structure. Unlike objects such as cars or people, which have predictable shape and orientation, food items are highly varied and exhibit non-uniform size, scale, etc. Furthermore, estimating nutritional attributes from a single RGB image increases the challenge as portion size, ingredients, and density are not readily observable from a single image.

Early progress in this area began with the introduction of the Food-101 dataset by Bossard et al. (2014), which provided the first large-scale benchmark dataset for food classification, containing 101,000 images across 101 classes. Their work introduced a Random Forest–based part-mining approach and achieved 50.76% classification accuracy, which significantly outperformed all traditional feature-based models of the time, but still fell short of CNN-based approaches. This dataset established a foundation for further deep learning driven research in the field. (Bossard, Guillaumin, and Van Gool 2014)

Following the introduction of Food-101, Meyers et al. (2015) included the problem of nutritional estimate along with classification. They introduced several curated datasets, including an extended version of Food-101, Food-201. They trained CNN models to estimate calories, fat, protein, and carbohydrates from a single image in both a restaurant and general food setting. While their models achieved moderate accuracy, the authors emphasized the difficulty of inferring nutrient content without richer metadata, particularly portion size or ingredient composition. (Myers et al. 2015)

To address these limitations, Thames et al. (2021) introduced the Nutrition5k dataset, a high-quality dataset of 5,000 cafeteria meals with rich metadata to include precise measurements of mass, calories, fat, carbohydrate, and protein. Their setup used RGB and depth (RGB-D) images, which enabled more accurate volume estimation. This proved to be a strong driving factor in estimating caloric density. Their CNN-based models outperformed expert human nutritionists when evaluating generic cafeteria meals, which demonstrated the value of richer metadata. (Thames et al. 2021)

Most recently, Dong et al. (2025) expanded the scale of food analysis datasets with MM-Food-100k. This dataset consists of 100,000 images containing JSON structured annotations including ingredients, portion estimates, and nutritional profiles. Compared with earlier datasets, MM-Food-100K offers rich metadata while greatly expanding the quantity of images making it well-suited for our use case. This dataset serves as the foundation for our project. (Dong et al. 2025)

4 Methods

4.1 Dataset and Preprocessing

The models were trained on MM-Food-100K, a large-scale dataset consisting of 100,000 high-quality images spanning a wide variety of food categories. In addition to the photos themselves, the dataset includes rich metadata such as nutritional information, portion size, cooking method, and ingredient descriptions. Prior research indicates that models trained on MM-Food-100K consistently achieve higher performance on both regression and classification tasks compared to models trained on smaller or less diverse food datasets. The original schema of the dataset is shown below:

![Schema of MM-Food-100k-dataset, from Dong et al. (2025)[1]].(assets/original_schema.png){fig-align="center" width="80%" "}

Several preprocessing steps were required before model development. Numerous metadata fields were removed due to irrelevance to the project's prediction targets. Dish names were normalized to ensure consistent labeling across the dataset, and the nutritional profile was parsed so that protein, fat, and carbohydrate content each occupied its own column. Because the dataset provides image URLs rather than raw image files, we downloaded all images locally via HTTP requests. Given storage and compute constraints, we further reduced the dataset by selecting the 300 most frequently occurring food categories and retaining 50 images per category, resulting in a final working dataset of 15,000 images. We then used a conventional 80/20 train–validation split to analyze the initial feasability of the model and then a 70/15/15 train-validation-test split to train, evaluate, and test.

4.2 CNN Architecture

Our models' architecture builds upon ResNet-18, a well-studied convolutional neural network with pretrained weights derived from ImageNet, a dataset containing more than one million labeled images. These pretrained weights provide rich, general-purpose visual representations—including edges, textures, shapes, and color gradients—giving the model a strong starting point for fine-tuning on food imagery. Leveraging this prior knowledge significantly accelerates training and reduces computational cost. The final model uses a hidden dimension of 512 followed by a linear output predicting portion mass and macronutrient values. Although dropout is commonly used to reduce overfitting, we elected to keep it at 0.0 for the final model. During the initial experimentation, the validation loss tracked the training loss throughout all epochs and early stopping was not hit. We initially incorporated a dish-ID embedding layer to encode contextual information about the food category; however, this component is not utilized in inference to ensure that the GUI could generalize to unseen foods without requiring a predefined dish label.

[1] Dong et al. (2025)

4.3 Training Procedure

We trained three CNNs with differing optimization strategies to assess how freezing or unfreezing the backbone would affect the output of the model. All models were based on same ResNet-18 backbone and regression head, however, we varied whether the pretrained backbone was frozen or not and the learning rate.

Model 1: Frozen Backbone The first model served as a baseline for our approach and all pretrained layers of Resnet-18 were frozen. Model 1 was trained for 15 epochs using the ADAM optimizer and an intial learning rate of 1e-4. We also implemented a scheduler that reduced the learning rate by a factor of 0.1 whenever validation loss failed to improve, along with early stopping with patience = 3 to prevent unecessary training. Despite the stability of this model, it was limited in its ability to adapt to food imagery.

Model 2: Unfrozen Backbone The second model relaxed the constraints of the baseline and unfroze the entire ResNet-18 backbone. Model 2 was trained for 20 epochs using the ADAM optimizer and a more cautious learning rate of 3e-5. The scheduler and early stopping of Model 1 were also implemented in Model 2. Although unfrozen backbones typically improve feature learning, Model 2 exhibited severe overfitting and validation loss collapse.

Model 3: The third model corrects Model 2's overfitting. The same 70/15/15 split was used and Model 3 was trained for 20 epochs using the ADAM optimizer and a restored learning rate of 1e-4. The scheduler and early stopping were also implemented.

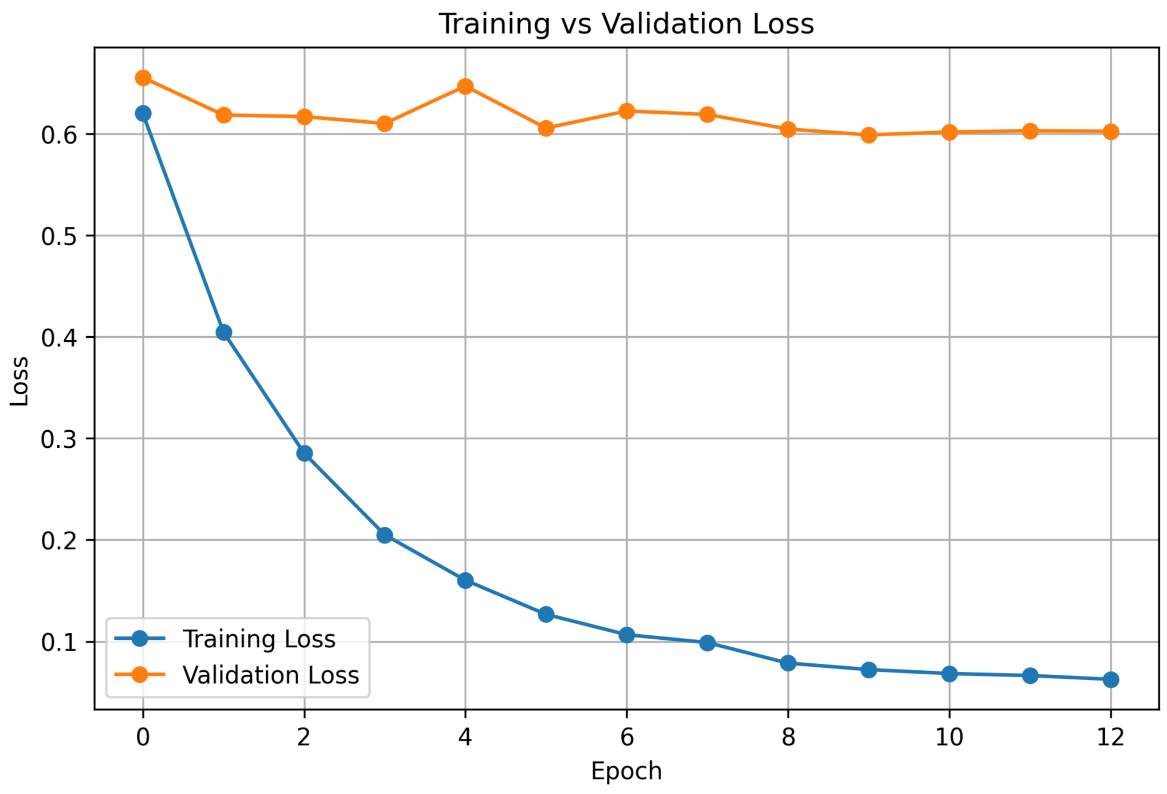
5 Results

5.1 Loss Curve Comparison

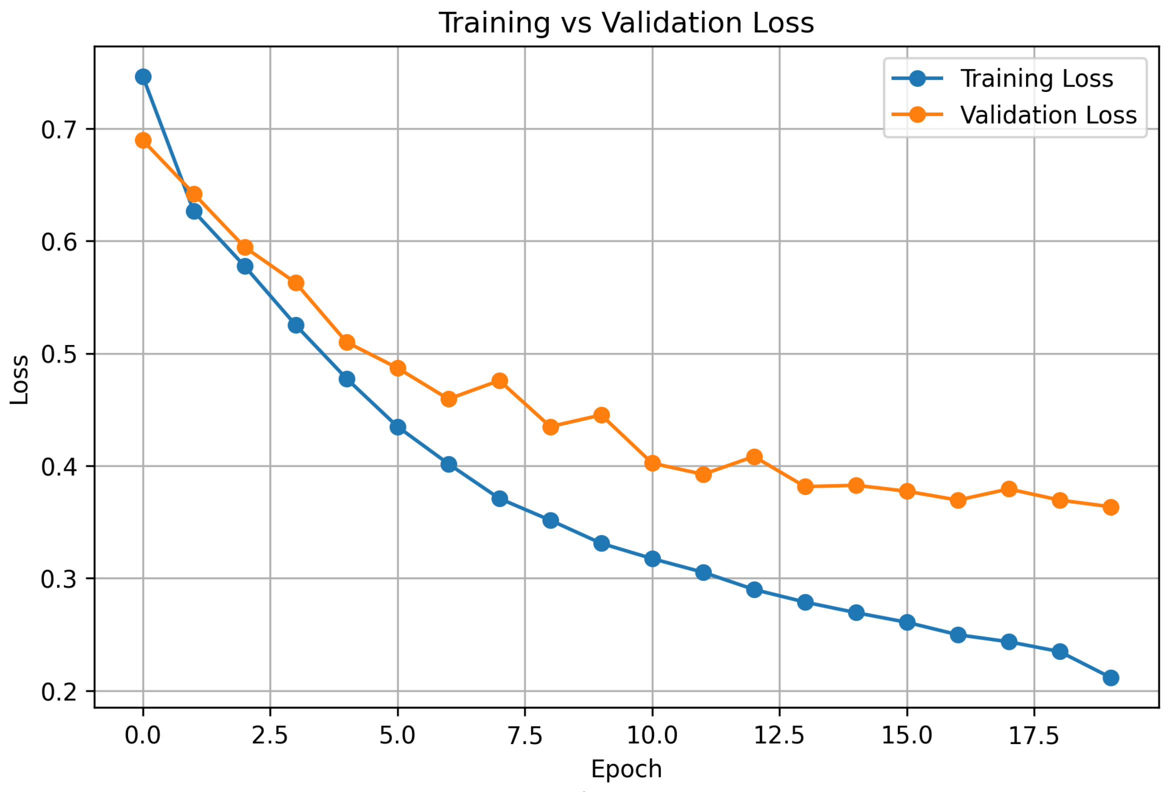
Figure A shows the training and validation loss curves for Model 2. The training loss curve is relatively smooth and suggests stable optimization. However, validation loss begins to plateau extremely early, which indicates that the model is overfitting severly.

Figure B shows the training and validation loss curves for Model 3. The loss curves are steadily decreasing towards 20 epochs. Early stopping was not hit for this model, but it appears that validation loss is starting to plateau.

Comparing the two models, the unfrozen backbone severly overfit while the frozen backbone demonstrated incremental improvements. This was due to the entire model being unfrozen, when the last layer should have been unfrozen first. This resulted in the pre-trained weights of ResNet-18 to be overwritten, even with a less aggressive learning rate.



(a) Model 2 training vs validation loss



(b) Model 3 training vs validation loss

5.2 Model Evaluation

To assess performance, Model 2 and 3 were evaluated on the 15% test split. Metrics were computed after denormalizing predictions and returning them to their original units. Performance was measured using Mean Absolute Error (MAE) and Mean Squared Error (MSE) and can be found in Table 1 below. Model 2 achieved an MAE of 163.15 and and MSE of 66865.51 on the test set, while Model 3 achieved an MAE of 166.48 and MSE of 70923.63. This would indicate that the unfrozen Model 2 is the higher performer. However, this is not the case because the variables have very different scales and per output MAE and RMSE are better evaluation metrics in this instance.

Table 1: **Overall Model Performance on Test Set**

Metric	Model 2 (Unfrozen)	Model 3 (Frozen)
Overall MAE	163.15	166.48
Overall MSE	66862.51	70923.64

Per output MAE and RMSE were computed for both models and can be found in Table 2 below. Model 3 outperforms Model 2 in every output even though it has a higher overall MAE than Model 2. This can be attribute to large absolute error in portion or calories dominating the aggregated MAE, even when the model performs consistently better at the individual output level. For this task, target level MAE is the more informative metric for overall model performance.

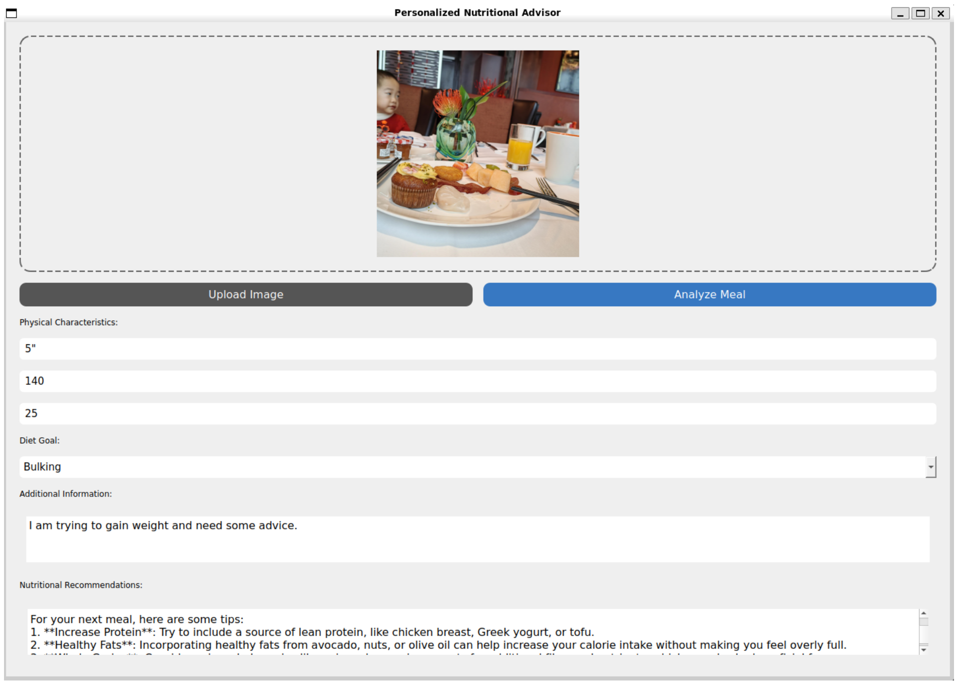
Table 2: **Per Output Model Performance on Test Set**

Metric / Target	Model 2 (Unfrozen)	Model 3 (Frozen)
Portion (g) MAE	82.90	75.16
Portion RMSE	136.63	118.33
Calories MAE	99.48	81.27
Calories RMSE	163.52	138.07
Protein (g) MAE	6.86	5.87
Protein RMSE	11.17	9.51
Fat (g) MAE	5.68	5.08
Fat RMSE	9.99	9.49
Carbs (g) MAE	14.21	10.37
Carbs RMSE	21.69	15.23

In terms of real-world deployment, the model results are not quite there yet. An MAE of 75 grams and 81 calories is significant when compounded across multiple meals, especially for those attempting to lose weight. Additionally, protein, fat, and carbohydrate MAE would compound as additional meals are accounted for. At best, this model could be utilized as a general idea of a person's dietary spread.

6 GUI System Design

The GUI was implemented in Python using the PyQt5 framework to provide an intuitive, user-friendly front end for real-time nutritional prediction. Upon launching the application, the user is presented with a large image display panel, input fields for height, weight, and age, a dietary-goal selector, and an optional free-text box for additional context. The interface includes two primary controls: an Upload Image button for selecting a local food photograph and an Analyze Meal button that triggers the prediction pipeline. Once an image is selected, it is previewed within the GUI, and the regressor model is loaded onto either a CPU or GPU depending on hardware availability. When the user initiates analysis, the application applies the same transformation pipeline used during training, converts the input image to a PyTorch tensor, and performs inference in evaluation mode to generate predicted portion weight and macronutrient values. From there, the portion weight and macronutrient values are fed into the LLM wrapper that generates dietary advice based on the information given in the display panel. The output formatted into a human-readable report and displayed within a dedicated, read-only panel. The layout is organized using a combination of horizontal and vertical PyQt5 containers, and custom stylesheets are applied throughout to ensure a polished, modern visual aesthetic. Overall, the GUI tightly integrates the trained CNN model with a responsive interface, enabling seamless meal analysis without requiring the user to manually interact with the underlying code or model artifacts.



Physical Characteristics:

5"

140

25

Diet Goal:

Bulking

Additional Information:

I am trying to gain weight and need some advice.

Nutritional Recommendations:

For your next meal, here are some tips:
1. **Increase Protein**: Try to include a source of lean protein, like chicken breast, Greek yogurt, or tofu.
2. **Healthy Fats**: Incorporating healthy fats from avocado, nuts, or olive oil can help increase your calorie intake without making you feel overly full.

Figure 2: Prototype GUI

6.1 LLM Wrapper

To complement the nutritional estimation, we developed a large language model (LLM) wrapper that transforms model outputs into personalized feedback. The wrapper creates a structured prompt from the predicted portion and macronutrient values, user-provided characteristics such as height, weight, and age, and the user's dietary goal. This prompt is submitted to the OpenAI API using the GPT-4o-mini model to provide reliable results while saving computation costs.

The LLM is explicitly instructed to produce supportive, actionable guidance. Specifically, the prompt directs the model to: - offer friendly, encouraging feedback - identify at least one positive aspect of the meal - suggest one or more areas for improvement - explain how the meal aligns with the user-stated dietary goal - provide simple suggestions for future meals rather than full recipes

This approach enables the user to contextualize the output of our nutrition estimation model and gain actionable advice based on their personal information and goals.

7 Discussion & Future Work

This project highlights both the promise and difficulty of estimating nutritional information from a single food image. Although CNNs are great at extracting visual features, this becomes extremely challending in the domain of general foods, where there is high variability. During our EDA, we observed many inconsistencies within the MM-Food-100k dataset, including stock images, bulk food trays, extreme close-ups, and even images containing people. Even with increased validation by the Dong et al., it seemed like the metadata attached to the image did not meaningfully correspond to what the image was showing. Mismatches such as this introduce noise in the training process and limit the model's ability to learn reliable feature mappings.

Despite these challenge we believe that our approach holds significant potential for improving public health, especially if it can be deployed in accessible tools such as our prototype application. Based on our results, there are several directions that future work could go.

1. Expand and refine the dataset Our project curated a subset of the top 300 classes of food with 50 images each, resulting in a full dataset of 15k images. A full-scale system would benefit from leveraging the information available from the entire dataset. The original dataset contains 19,288 classes, which is unwieldy for classification. Considering this, clustering dish embeddings could reduce the number of classses while preserving the dish integrity.

2. Conduct additional hyperparameter and architectural tuning Our training pipeline was limited by computational resources, limiting our exploration of hyperparamters. Future iterations of testing could use ResNet-34 or ResNet-50 as the model backbone, potentially expanding the relationships the model is able to learn from the images.

3. Move towards deployable applications The prototype GUI effectively demonstrates that model output can be integrated with LLM-generated nutritional guidance. A natural next step, after improving the model, would be to build a mobile-friendly application and conduct user studies to evaluate the practical impact of our approach.

This approach and future steps offer a robust and practical nutrition assistance systems that has the potential to improve public health for vast amounts of people.

References

Bossard, Lukas, Matthieu Guillaumin, and Luc Van Gool. 2014. "Food-101 – Mining Discriminative Components with Random Forests." In *Computer Vision – ECCV 2014*, edited by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, 8694:446–61. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-10599-4_29.

Dong, Yi, Yusuke Muraoka, Scott Shi, and Yi Zhang. 2025. "MM-Food-100K: A 100,000-Sample Multimodal Food Intelligence Dataset with Verifiable Provenance." arXiv. https://doi.org/10.48550/arXiv.2508.10429.

Myers, Austin, Nick Johnston, Vivek Rathod, Anoop Korattikara, Alex Gorban, Nathan Silberman, Sergio Guadarrama, George Papandreou, Jonathan Huang, and Kevin Murphy. 2015. "Im2Calories: Towards an Automated Mobile Vision Food Diary." In *2015 IEEE International Conference on Computer Vision (ICCV)*, 1233–41. Santiago, Chile: IEEE. https://doi.org/10.1109/ICCV.2015.146.

Thames, Quin, Arjun Karpur, Wade Norris, Fangting Xia, Liviu Panait, Tobias Weyand, and Jack Sim. 2021. "Nutrition5k: Towards Automatic Nutritional Understanding of Generic Food." In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8899–8907. Nashville, TN, USA: IEEE. https://doi.org/10.1109/CVPR46437.2021.00879.