CalorAI: The Food Calorie Estimator

Yifei Wang
Student Number 1008822730
yifeii.wang@mail.utoronto.ca

Eric Miao
Student Number 1008009218
eric.miao@mail.utoronto.ca

Vhea He
Student Number 1009525202
vhea.he@mail.utoronto.ca

Abstract

In the digital age, tracking food consumption has become increasingly important for promoting healthier lifestyles. *CalorAI* addresses this need through a deep learning pipeline that uses convolutional neural networks (CNNs) to recognize food items from meal images and estimate their total caloric content. By automating calorie estimation, *CalorAI* aims to reduce barriers to healthy eating, particularly for individuals unsure of where to begin. Our system is designed to be practical and user-friendly, requiring only an image of a meal to deliver nutritional insights. The open-source codebase is available at GitHub, and our curated dataset can be accessed at Google Drive.

1 Introduction

The global obesity epidemic is one of the most pressing public health challenges of today. According to the World Obesity Atlas 2023 report, 38% of the global population is currently overweight or obese [1], which is a concerning trend that is only projected to get worse. Obesity is heavily associated with a multitude of chronic conditions such as heart disease, diabetes, high blood pressure, liver disease, and cancer [2]. This rising prevalence of weight-related illnesses highlights the importance of tracking food consumption, as it is imperative that people are able to understand the caloric content and nutritional properties of their diets. Traditional diet tracker applications often require their users to input food items and portion sizes manually, which can be tedious and suffer from estimation inaccuracies. In this paper, we present *CalorAI*, a deep learning-based food calorie estimation system that processes meal images to predict their caloric content. This project can serve as a starting point for more advanced diet tracking methods, and help many individuals who aren't sure how to start practicing healthy food habits. *CalorAI* boasts a CNN-based pipeline for food classification and calorie estimation. We hope to take a novel approach to handling multiple food items in a single image, and contribute our open-source implementation for further research and potential real-world applications. This paper will detail the data preprocessing pipeline, the *CalorAI* model architecture and the model's current evaluation results.

2 Background

Tracking caloric intake is an essential step in addressing the global obesity epidemic and managing related health conditions like diabetes. For this reason, calorie estimation has garnered significant attention in recent years. With the widespread use of smartphones and easy access to the web, it should be simple for users to track their diets. However, many existing methods for calorie estimation are usually time-consuming, and suffer from estimation inaccuracies. Traditional diet tracking apps, such as *MyFitnessPal* [3] and similar apps like *Lose It!* or *Yazio* all follow a common design

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

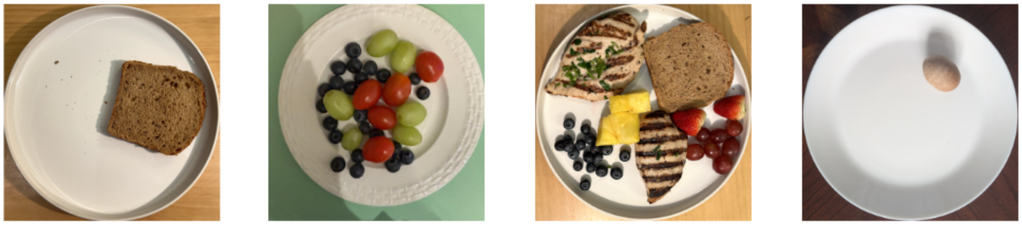

Figure 1: Sample images from our dataset.

framework. These apps require users to manually log food items and portion sizes, which can quickly become tedious and prone to human error. Additionally, they rely heavily on user-generated databases of manually inputted calorie information and usually charge subscription fees, which creates an obstacle for many prospective users.

Many food and calorie related machine learning models have also attempted to explore AI-driven food recognition and calorie estimation. However, these models are either not very helpful for the common user, or are not publicly available. For instance, a paper published by IEEE Xplore looks into using DCNNs for food photo recognition [4]. Although their model was quite successful, with a 78.77% success rate, this type of model does not tell the user much nutritional information, which does not address the diet tracking issue we are tackling. Another paper explores topics much closer to this project, using images to estimate calorie content for dietary assessment [5]. With a 79% correctness within ±40% error, it does of much of what our project aims to accomplish. However, this model is not publicly available for everyday users, nor is a ±40% margin of error the best for accurate estimation.

Applications that are publicly available, such as *CalorieMama* [6], tend to not be the best at providing accurate calorie estimates. Instead of getting the total amount of estimated calories for a given image, CalorieMama focuses on food recognition but can sometimes misidentify items, leading to inaccurate calorie estimates. Additionally, the app only provides caloric information per serving, requiring users to manually calculate the total calorie content by measuring and converting their meals. This is even more time-consuming than traditional calorie tacking apps, and requires the user to perform their own calculations.

In this project, we hope to build a model that provides meaningful calorie estimation with minimal user effort, and is accessible and cost-effective for the general population.

Although there are a vast number of datasets that map food images to food categories and food categories to their calories, we found a lack of public datasets that map food images directly to the calories in the image. This creates an obstacle for our model, as there is no data from which our model can learn *portion estimation* of different types of food. Therefore, we decided to obtain our own dataset for training.

2.1 Data Collection

As our model is designed for everyday users to track their calorie intake, we chose to use cell phone images as our primary data source to simulate real-world usage. To collect these images, three members of our team took photos using mobile phone cameras in various environments. The dimensions of the images were standardized to a 1:1 ratio, and all food was placed on 10-inch diameter dinner plates. We ensured contrast between the plate and the image background so that the meal was clearly distinguishable, which we hypothesized would help the model learn better visual boundaries for food segmentation.

We deliberately selected a variety of common food items that many people likely have access to at home—such as bread, eggs, chicken breast, apples, broccoli, and strawberries. This decision was intentional in order to make the model practical for real-life usage. Less common foods (e.g., jujubes or chives) were included in smaller amounts to add diversity, but the focus remained on broadly available ingredients.

To construct the dataset, the foods were weighed using a kitchen scale, and the portion sizes (in grams) of each food item in each image were manually recorded in a spreadsheet. The total calories for each image were then calculated using a database that maps food items to their calorie-per-gram values [7]. While taking pictures, we aimed to maximize variability and reduce overfitting by capturing each food in different orientations and combinations. This included rotating the plates, adjusting lighting, and combining multiple food items in a single image.

In total, we collected 1,093 images across 26 food categories, including single-food and multi-food compositions. Figure 1 shows some example images from our dataset.

2.2 Data Pre-processing and Analysis

In order to ensure our dataset was ready for training, we applied a number of preprocessing steps. All images were resized to 400×400 pixels, normalized, and converted to tensors. Labels were encoded into a one-hot multi-label vector indicating which food items were present in each image.

One of the strengths of our dataset is its balanced data split. As shown in Table 4 in A, we ensured that each food category was split approximately into 80% training, 10% validation, and 10% test, preserving class-level consistency across all splits. This helps prevent data leakage and ensures a fair evaluation of model generalization. Similarly, Table 5 shows that the number of food types per image is also well-distributed across splits, allowing the model to learn both single-item and multi-item meal scenarios.

However, there are notable imbalances in the number of times each food item appears in the dataset. For example, pineapple and blueberries appear in nearly 200 images, while foods like onion or banana appear in fewer than 25 images. These discrepancies arose due to limitations in manual data collection and the perishability of certain food types, which made it more difficult to gather evenly distributed examples. As a result, the model may be biased toward frequently occurring food items and underperform on rare categories. This imbalance is an important limitation to consider, particularly in a multi-label setting where rare foods might be overshadowed during training.

Additionally, while we strived for visual variety in the image backgrounds and lighting, all data was taken using the same types of plates and similar environments, which may introduce distributional biases. Expanding the dataset in the future to include diverse plate shapes, containers, and real-world scenarios like takeout meals or restaurant settings would be critical for making the model more robust in practical applications.

3 Model Design

3.1 Task Definition

The CalorAIze model solves the following task. Given an input plated food image U_i, the model should identify the j food items contained on the plate, denoted $F_i = (f_1, f_2, ..., f_j)$, and estimate the calories $C_i = (p_1, p_2, ..., p_j)$ of each food item f_j.

3.2 Baseline Testing

We performed baseline testing on various existing image classification models on our custom training and testing data. We determined to test various ResNet architectures due to their scalability and reputation in this field, as well as vision transformer (ViT) architectures due to its attention mechanisms in transformers. Furthermore, these architectures were also tested in the regression part of our model mainly to create a CNN embedding for the FC layers.

Baseline testing was performed with 16GB RAM and Nvidia RTX 3060 Laptop GPU with CUDA 12.5, 6GB VRAM. As such, ResNet101, ResNet200, ViT Medium and ViT Large were not included in the testing due to memory limitations.

The classification baseline results are shown in Table 1, and the regression baseline results are shown in Table 2.

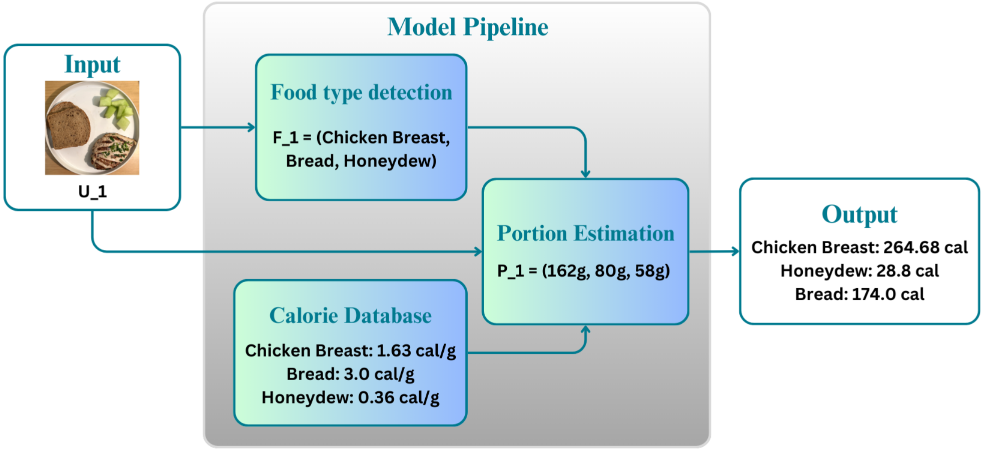

Figure 2: CalorAI Model Architecture

Model Name	Parameters	Test Accuracy (%)
ResNet18	11,189,850	88.779
ResNet26	13,999,450	90.085
ResNet34	21,298,010	88.567
ResNet50	23,561,306	92.449
ViT-Tiny	5,602,394	65.702
ViT-Small 16	21,821,594	69.442
ViT-Small 32	22,540,442	65.314

Table 1: Test accuracy of different models evaluated on the dataset, with corresponding model sizes.

3.3 Model Architecture

As shown in Fig. 2, the CalorAI model consists of three main components: **food type classification**, **portion estimation regression**, and **calorie computation**.

The food classifier is a multi-label classification model based on the ResNet18 architecture [8]. Given a pre-processed input image, the model outputs a sigmoid-activated probability vector indicating the presence of each food class. Each image can contain multiple food items, and therefore the classification task is treated as multi-label rather than multi-class. The final fully connected layer of the pretrained ResNet18 is modified to output a vector of length equal to the number of food categories in the dataset. The model is trained using the 'BCEWithLogitsLoss' function, which is appropriate for multi-label classification.

Although our baseline experiments (Table 1) showed that ResNet50 outperformed ResNet18 in terms of classification accuracy, we ultimately selected ResNet18 for the final model due to practical training constraints. ResNet50 took approximately 1 hour per epoch to train on our available hardware, whereas ResNet18 offered significantly faster training times with only a modest tradeoff in accuracy. This allowed us to iterate and tune the model more effectively within the limited development window.

Food Type Classification

The food classification component of CalorAI is a multi-label image classifier based on the ResNet18 architecture [8], pre-trained on the ImageNet dataset. The model receives a 400×400 RGB image as input and outputs a sigmoid-activated probability vector of length N, where N is the number of unique food classes in our dataset, dynamically determined from the calorie database JSON file.

ResNet18 is a convolutional neural network composed of 18 layers, including 17 convolutional layers and one fully connected (FC) output layer. It uses residual connections that allow the network to bypass certain layers, mitigating the vanishing gradient problem and enabling deeper architectures to

Model Name	Parameters	Training Regression Loss	Test Regression Loss
ResNet18	11,189,850	271.47	249.12
ResNet26	13,999,450	168.74	202.30
ResNet34	21,298,010	235.15	179.79
ResNet50	23,561,306	200.25	192.87
ViT-Tiny	5,602,394	290.51	295.15
ViT-Small 16	21,821,594	327.39	315.69
ViT-Small 32	22,540,442	265.51	266.85

Table 2: Regression loss of different models evaluated on the dataset, with corresponding model sizes.

train effectively. Each residual block contains a sequence of two 3×3 convolutional layers followed by batch normalization and ReLU activations.

In our implementation, we retain all convolutional and residual layers of the pre-trained ResNet18 and replace only the final fully connected layer. The original classification layer, which maps to 1000 ImageNet categories, is replaced with a new linear layer:

```
nn.Linear(512, N)
```

This layer projects the 512-dimensional feature vector output from the final average pooling layer into N logits corresponding to our food classes. Because the task is multi-label classification (an image can contain multiple food items), we apply a sigmoid activation to each logit to obtain independent probabilities:

$$\hat{y}_i = \sigma(z_i) = \frac{1}{1 + e^{-z_i}}, \quad i = 1, 2, ..., N$$

The model is trained using the `BCEWithLogitsLoss` function, which combines a sigmoid activation and binary cross-entropy loss in a numerically stable way. This loss function is appropriate for multi-label settings, where each class label is treated as a separate binary prediction.

We chose ResNet18 for several reasons. While deeper models like ResNet50 demonstrated superior performance during baseline testing (see Table 1), they incurred significantly higher training costs—up to one hour per epoch on our hardware. In contrast, ResNet18 offered a good balance of accuracy and training efficiency, completing each epoch in under 10 minutes. This allowed for faster iteration and experimentation, which was crucial given our project time constraints. Moreover, ResNet18 has a significantly smaller parameter count (11.2M vs. 23.6M for ResNet50), making it more suitable for lightweight applications and potential mobile deployment.

Images were normalized to the range $[-1, 1]$ using mean and standard deviation values of 0.5 for each channel. Labels were represented as binary vectors of shape $(N,)$, with ones indicating the presence of a food item.

This component outputs a vector of probabilities representing the predicted presence of each food category, which is passed into the portion estimation module for subsequent weight prediction.

Portion Estimation Regression

The portion estimator is a CNN-based regression model that takes both the input image and the predicted food categories as input. It predicts the weight of each identified food item (in grams), which is later used to compute total caloric content.

The model uses a pre-trained ResNet34 backbone implemented through the `timm` library. Instead of using the original classification layer, we modify the model to return a 16-dimensional feature vector representing the visual features of the input image. Simultaneously, we project the multi-label food classification vector (with length equal to the number of food classes) into another 16-dimensional vector using a fully connected layer.

The image feature and food vector embeddings are then concatenated into a 32-dimensional vector, which is passed through a series of three fully connected layers with ReLU activations:

$$\texttt{Linear(32, 256)} \to \texttt{ReLU} \to \texttt{Linear(256, 32)} \to \texttt{ReLU} \to \texttt{Linear(32, 26)} \to \texttt{ReLU}$$

The final output is a vector containing the predicted portion size (in grams) for each food category. ReLU is applied to ensure all outputs are non-negative, since portion sizes cannot be negative.

We chose to explicitly condition the portion estimation on the known food categories detected in the previous classification stage. By concatenating the food presence vector with the image embedding, the model focuses on estimating quantities rather than identifying foods, which improves performance in multi-food scenarios.

The model is trained using mean squared error (MSE) loss between the predicted and ground-truth portion sizes. Since the label vector only contains non-zero values for the foods actually present in the image, the model is implicitly penalized only on relevant categories. The final model is trained using the Adam optimizer with a learning rate of 1×10^{-4}, a batch size of 32, and a total of 10 epochs. Model checkpoints are saved along with class metadata to ensure future compatibility with the classifier.

3.4 Special Considerations

Even though it is seen from Table 1 that ResNet50 achieved the highest food type classification accuracy, due to its long training time at around 1 hour per epoch and given our time constraint of 120 hours for the entire project, we determined that using ResNet50 for our food type classification component of the model is unfeasible.

4 Evaluation

4.1 Evaluation Metrics

The evaluation of the CalorAI model consists of two primary components: **food classification** and **portion estimation**. The food classification performance is measured using the sample-wise F1 score, Hamming loss, and exact match ratio. These metrics are chosen to evaluate the model's ability to handle the multi-label nature of the task. The F1 score balances precision and recall across multiple labels, Hamming loss quantifies label-level misclassification, and the exact match ratio indicates how often the model predicts all food items in an image correctly.

For portion estimation evaluation, the model's ability to predict the weight of each detected food item is assessed using multiple error metrics. The Mean Absolute Error (MAE) computes the average absolute difference between the predicted and actual portion sizes, providing an intuitive measure of accuracy. The Root Mean Squared Error (RMSE), which penalizes larger errors more heavily, helps identify significant discrepancies in portion predictions. Additionally, the model's accuracy within a ± 10% range is reported, indicating the proportion of predictions that fall within 10% of the actual portion size. This metric is particularly relevant for practical applications, as small deviations in portion estimation can significantly impact calorie calculations. Together, these evaluation measures ensure that the model provides reliable predictions for food identification and calorie estimation.

4.2 Current Performance

We trained and evaluated the model on two versions of the dataset: the original smaller set (573 images), and a larger expanded version containing 1093 images.

Table 3 compares the results across the two versions. Surprisingly, performance significantly declined in the newer version across all classification metrics.

The performance results across the two dataset versions show that the situation is complicated. For the classification stage, the transition to a larger dataset resulted in a significant drop in performance: the overall F1 score decreased from 0.78 to 0.09, Hamming loss increased nearly fivefold, and exact match ratio fell from 54.39% to 5.5%. These results suggest that the model struggled to generalize when faced with a more diverse and imbalanced dataset. Appendix B shows that most food categories now have an F1 score near zero, with only a few classes (such as raisins) being correctly identified.

Metric	**Smaller Dataset**	**Larger Dataset**
Classification		
F1 Score	0.7815	0.0891
Hamming Loss	0.0243	0.1048
Exact Match Ratio	54.39%	5.50%
Portion Regression		
Mean Absolute Error (MAE)	0.64g	1.81g
Root Mean Squared Error (RMSE)	1.09g	9.52g
Accuracy (±10%)	21.01%	31.03%

Table 3: Comparison of model performance on the smaller and larger datasets. See Appendix B for per-class classification breakdown.

In contrast, the portion regression component showed improvement in a key practical metric: the percentage of predictions within ±10% of ground truth rose from 21.01% to 31.03%, a 10% absolute improvement. Although both MAE and RMSE increased due to greater prediction spread and more complex multi-food examples, the model was better able to estimate reasonable weight values within acceptable error margins. This suggests that the larger dataset helped the regression model learn more generalized patterns for portion size estimation, even if individual predictions became noisier on average.

Overall, while classification performance degraded, the portion estimation became more robust and useful—highlighting a shift in the model's utility under a more complex data regime.

4.3 Improvements in the Model

To recover from the performance degradation observed on the larger dataset, our next steps should include explicitly addressing class imbalance through weighted loss functions or over/under-sampling strategies. Additionally, we may apply label smoothing or focal loss to handle sparse label distributions more effectively in multi-label settings.

Future work should also explore deeper or more flexible backbone architectures (e.g., ResNet50, MobileNetV2) and apply more robust data augmentation techniques to help generalize across label combinations. Finally, it will be important to audit the labeling process for consistency and explore semi-supervised training or label cleaning techniques to reduce noise introduced during manual annotation.

After changing the CNN architecture in the regression stage from ResNet18 to ResNet34, despite a slight $1.17g$ increase in MAE, there was a 10% improvement in the accuracy of the portion regressor, from 21.01% to 31.03%. This improvement in accuracy might be due to an increase in variety within the dataset to reduce bias and overfitting within the model architecture, and is further improved by switching to a CNN architecture with the lowest loss trained on the custom dataset.

4.4 Comparison to Baseline

5 Next Steps

If we were to continue developing this project to prepare it for the user market, there are several areas we would focus on improving and expanding from what we currently have.

First of all, since the current data in the dataset has many preprocessed restrictions and requirements, including the image being a square image and the plate must be a 10" diameter dinner plate, the general public should have no expectations of abiding by the same requirements when using CalorAI. Thus,the most important step is to expand our dataset to include a variety of various variables, namely portion sizes, image scaling, and the tableware.

5.1 Expand Dataset

In order to improve the model's performance and scalability, we need to greatly expand the dataset it will be trained on. Currently, our dataset only includes 26 types of foods with images and labels, which is far from what a program that aims to be able to identify most foods should be trained on. We will need a much larger and diverse set of food images, which may involve collecting more photos, or integrating the model with internet resources and databases. This part will require significant time and resources, but is also crucial to improving the model's quality.

5.2 Improve Model Performance

With a larger and more diverse dataset, we would aim for improving the performance of the model. Although our current model shows potential, it doesn't perform at ideal levels to be used by the general public. This can be improved by training on better data. To do this, we plan to experiment with alternative pre-trained models, such as MobileNetV2 or even more transformer based models, like other, more robust ViT models. We could also manipulate the database with more advanced data augmentation techniques, such as random cropping, rotation, or colour adjustments. With a greater image database to work with, diversifying the types of images available is very important for more generalization.

While the portion regression model benefited from the larger dataset and achieved better ±10% accuracy, the classification model suffered significantly. This indicates that future improvements should focus primarily on classification. Possible next steps include: rebalancing the dataset or applying class-weighted loss functions to handle skewed class distributions, enhancing the classifier architecture (e.g., using ResNet50 or MobileNetV2) to handle more complex visual patterns, increasing the number of training epochs or using learning rate scheduling to better fit the larger dataset, and applying strong data augmentation and label smoothing to prevent overfitting to dominant classes and reduce noise.

For the portion regressor, additional improvements could be made by incorporating contextual cues (e.g., object sizes, depth estimation) or multi-view images, but current results already indicate a positive trend.

5.3 Prototype and User Interface

For the final product, we also plan to build a user-friendly interface that uses the CalorAI model. The application will be available for mobile use, and should allow users to upload images of their meals or take photos directly using their camera. The system will process the image, estimate the calorie content, and display both the detected food items and their corresponding calorie values. Users will also have the option of recording meals into daily logs, which compiles total calories consumed throughout the day. This feature will enable users to track their long-term eating habits effectively.

References

[1] Author(s) Unknown. Study on ai-based food recognition and calorie estimation. *PubMed Central*, 2023. Accessed: 2025-02-27.

[2] Mayo Clinic Staff. Obesity - symptoms and causes, 2024. Accessed: 2025-02-27.

[3] American Academy of Family Physicians. Myfitnesspal: An app review. *Family Practice Management*, 22(2):31–32, 2015.

[4] Yoshiyuki Kawano Keiji Yanai. Food image recognition using deep convolutional network with pre-training and fine-tuning. *IEEE Xplore*, 2015.

[5] Kiyoharu Aizawa Tatsuya Miyazaki, Gamhewage C. de Silva. Image-based calorie content estimation for dietary assessment. *IEEE Xplore*, 2011.

[6] CalorieMama. Caloriemama - ai-based food calorie estimation, 2025. Accessed: 2025-03-09.

[7] Kaggle Community. Calorie estimation discussion thread, 2021. Accessed: 2025-03-08.

[8] MathWorks. *ResNet-18*, 2024. Deep Learning Toolbox.

A Appendix A: Data Class and Distribution

Food Category	Total	Train	Validation	Test
Pineapple	188	151 (80%)	18 (10%)	19 (10%)
Blueberries	182	146 (80%)	20 (11%)	16 (9%)
Strawberries	166	135 (81%)	16 (10%)	15 (9%)
Chicken Breast	159	129 (81%)	15 (9%)	15 (9%)
Cantaloupe	156	126 (81%)	16 (10%)	14 (9%)
Egg	102	84 (82%)	9 (9%)	9 (9%)
Bread	98	77 (79%)	10 (10%)	11 (11%)
Grapes	93	74 (80%)	10 (11%)	9 (10%)
Cherry Tomato	91	75 (82%)	8 (9%)	8 (9%)
Mushrooms	90	71 (79%)	11 (12%)	8 (9%)
Jujube	86	65 (76%)	10 (12%)	11 (13%)
Broccoli	83	70 (84%)	8 (10%)	5 (6%)
Honeydew	81	64 (79%)	9 (11%)	8 (10%)
Cauliflower	80	65 (81%)	10 (13%)	5 (6%)
Raisins	75	61 (81%)	6 (8%)	8 (11%)
Sweet Potato	65	52 (80%)	7 (11%)	6 (9%)
Garlic	64	51 (80%)	5 (8%)	8 (13%)
Apple	51	39 (76%)	7 (14%)	5 (10%)
Carrot	50	40 (80%)	5 (10%)	5 (10%)
Clementine	41	31 (76%)	5 (12%)	5 (12%)
Pear	33	27 (82%)	2 (6%)	4 (12%)
Chives	30	23 (77%)	4 (13%)	3 (10%)
Orange	23	20 (87%)	2 (9%)	1 (4%)
Banana	21	16 (76%)	3 (14%)	2 (10%)
Potato	20	18 (90%)	0 (0%)	2 (10%)
Onion	14	11 (79%)	2 (14%)	1 (7%)
Total	**2432**	**1949 (80%)**	**270 (11%)**	**213 (9%)**

Table 4: Distribution of food categories across dataset splits.

Number of Food Types	Total	Train	Validation	Test
1	587	467 (80%)	56 (10%)	64 (11%)
2	212	172 (81%)	21 (10%)	19 (9%)
3	147	120 (82%)	16 (11%)	11 (7%)
4	82	63 (77%)	10 (12%)	9 (11%)
5+	65	53 (82%)	6 (9%)	6 (9%)
Total	**1093**	**895 (82%)**	**109 (10%)**	**109 (10%)**

Table 5: Image count and percentage by number of food types.

B Appendix B: Per-Class Classification Scores

The following table summarizes the F1 score of the classification model on each food class, trained on the larger dataset.

Food Category	**F1 Score**	**Food Category**	**F1 Score**
Bread	0.0000	Grapes	0.0000
Cherry Tomato	0.1667	Chicken Breast	0.3846
Cantaloupe	0.0000	Strawberries	0.2759
Blueberries	0.0000	Sweet Potato	0.0000
Egg	0.0000	Broccoli	0.0000
Apple	0.0000	Carrot	0.0000
Honeydew	0.0000	Clementine	0.0000
Pineapple	0.1429	Garlic	0.0000
Pear	0.0000	Chives	0.0000
Cauliflower	0.0000	Jujube	0.0000
Orange	0.0000	Banana	0.0000
Potato	0.0000	Raisins	1.0000
Mushrooms	0.0000	Onion	0.0000

Table 6: F1 scores per food category. Most underrepresented classes perform poorly.