Crimson Nutrition: A Computer Vision System For Food Classification And Nutritional Information

Dhruvil Joshi
Indiana University Bloomington
joshidh@iu.edu

Arju Singh
Indiana University Bloomington
singarju@iu.edu

1. Introduction

As the world becomes more health-conscious and mindful of dietary choices, there has been a growing interest in understanding nutrition and the food we consume. To explore this intersection practically, we decided to integrate computer vision techniques within the domain of food and nutrition.

Recognizing the broad scope and numerous possibilities within this area, we chose to start with a focused and achievable goal. Through our project, Crimson Nutrition, we aimed to classify images of food items that users capture and upload to our web application. Once identified, our application provides users with the nutritional details associated with that particular food item.

However, it's important to clarify that our goal was not to estimate nutritional information based on the portion sizes appearing in user-submitted images. Instead, we provided nutritional values corresponding to a standardized serving size. Estimating portion sizes accurately from images is a significantly more complex task, extending beyond the scope and timeline of our current project.

In addition, we integrate an open source large language model (LLM) into our system to offer quick and helpful nutritional insights. This feature supports users by providing additional context and information about the nutritional aspects of their food, aligning with our ultimate goal of helping people understand their food choices better.

2. Related Work

Image-based dietary assessment has emerged as a compelling research area within healthcare and artificial intelligence. Tahir and Loo[2] presented a comprehensive survey of food recognition and volume estimation techniques, discussing both handcrafted and deep learning-based approaches. Their work emphasized the challenges of inter-class similarity, intra-class variation, and the lack of standardized food volume datasets.

Amugongo et al.[3] systematically reviewed mobile computer vision-based diet applications and highlighted a significant gap in explainability within deployed systems. They noted that despite promising precision, few systems explained why specific classifications or caloric estimates were made, an important limitation for adoption in clinical settings.

The Food-101 dataset introduced by Bossard et al.[1] has become the benchmark for food classification tasks. It includes a wide range of visually similar food categories, challenging models to develop fine-grained recognition abilities. Techniques such as Convolutional Neural Networks (CNNs) and architectures such as ResNet, Inception, and MobileNet have been employed with notable success.

Furthermore, research has moved beyond classification to include multitask learning, ingredient recognition, and 3D reconstruction for volume estimation. However, most models operate in controlled environments and lack robustness to diverse real-world conditions such as varying lighting, occlusions, or mixed dishes. Our work builds on these foundations, aiming for a practical, web-oriented solution with future potential for multi-class classification and calorie estimation.

3. Methods

Our research aimed to investigate whether computer vision models could accurately classify food items from standard images captured by everyday users. We approached this as a classic image classification problem and structured our methods clearly to ensure reproducibility. Below are detailed descriptions of each step, categorized into clear subsections for ease of understanding and replication.

3.1. Dataset

We utilized the Food101 Dataset consisting of images that span 101 diverse food categories. This dataset was chosen due to its extensive range of food items, allowing robust training and validation.

3.2. Model Selection and Experimentation

To ensure optimal performance and practical usability in a web application, we explored multiple contemporary deep learning models discussed in our course (CSCI-657 Computer Vision), specifically focusing on recent, well-regarded architectures:

3.3. Model Architectures

1. MobileNet-V2
2. ResNet-50
3. Vision Transformer (ViT) (planned but incomplete)

3.4. MobileNet-V2

Initially, MobileNet-V2 was selected for its lightweight architecture, offering the potential for faster inference within our web application.

Implementation Details. The model was implemented using TensorFlow and trained on a Kaggle Kernel (GPU: Nvidia Tesla P100, 30-hour runtime limit). We used an 80% training and 20% validation split. To improve generalization, we applied data augmentation techniques including rotation (±20°), width/height shift (±10%), shearing (0.2), zooming (±10%), and horizontal flipping.

Model Architecture and Training. The base model was MobileNet-V2 pretrained on ImageNet (`include_top=False`, `pooling='avg'`), followed by a custom dense head with softmax activation over 101 classes. The training procedure used the Adam optimizer, categorical crossentropy loss, and accuracy as the main metric. We applied early stopping (`patience=5`, `monitor='val_loss'`, `restore_best_weights=True`) over 7 epochs, with a batch size of 32.

Outcome. The performance of MobileNet-V2 was ultimately unsatisfactory, prompting further experimentation with more robust architectures.

3.5. ResNet-50

Due to the suboptimal results from MobileNet-V2, we transitioned to the more complex and powerful ResNet-50 model, anticipating improved accuracy at the cost of increased computational resources.

Implementation Details. The model was implemented using PyTorch and trained on a Kaggle Kernel (GPU: Nvidia Tesla P100, 30-hour runtime limit) with a 75% training and 25% validation split.

Data Augmentation. We applied several techniques to improve generalization: resizing images to 256×256 (bilinear interpolation), random cropping to 224×224, random horizontal flipping, random rotation (±15°), affine transformations (translation, scaling), color jitter (brightness, contrast, saturation, hue), random grayscale (probability 0.05), and normalization (mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]).

Model Architecture and Training. The base model was ResNet-50 pretrained on ImageNet, with a modified output head consisting of a dropout layer ($p = 0.3$) followed by a fully connected layer for 101 classes. For fine-tuning, we initially froze all layers except `layer4` and the fully connected (`fc`) layer. The training procedure used the CrossEntropyLoss function and Adam optimizer (learning rate 0.001, weight decay 1×10^{-3}), with a StepLR scheduler (step size 5 epochs, gamma 0.2), running over 20 epochs.

Outcome. After approximately 10 hours of training, ResNet-50 delivered significantly better classification results and was selected as the final model for inference in our web application.

3.6. Vision Transformer (ViT) Experimentation

Given the emerging popularity and proven efficacy of Vision Transformers (ViTs) for classification tasks, we briefly explored their potential. However, due to time constraints, this experiment was not completed. The preliminary exploration has motivated future exploration into ViTs, owing to their promising accuracy and adaptability.

3.7. LLM Integration for Nutritional Insights

Beyond classification, we aimed to provide additional, meaningful nutritional insights to users. To achieve this, we integrated an open-source large language model (LLM) to enrich the user experience.

Implementation Details. We used the Groq Python package to interface with the LLM endpoint and experimented with two models:

1. meta-llama/llama-4-scout-17b-16e-instruct (having vision capabilities)
2. llama-3.3-70b-versatile

Prompt Engineering. A system prompt was crafted to instruct the LLM to act as a nutrition expert and generate realistic calorie ranges for standard food servings. The user prompt consisted of the identified food class predicted by the ResNet-50 model.

Workflow. The workflow involved three main steps: (1) obtaining the top-1 predicted food class from ResNet-50, (2) sending the food class as input to the LLM, and (3) parsing the LLM response to extract a calorie range in integer format (e.g., “250 350”), which was then displayed alongside the food classification results.

Outcome. This integration effectively supplemented the computer vision outputs by providing users with actionable nutritional knowledge about their identified food items, ultimately enhancing the application’s overall usefulness.

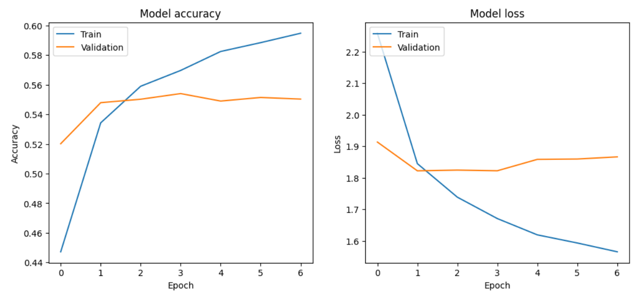

Figure 1. MobileNet-V2 Training Results

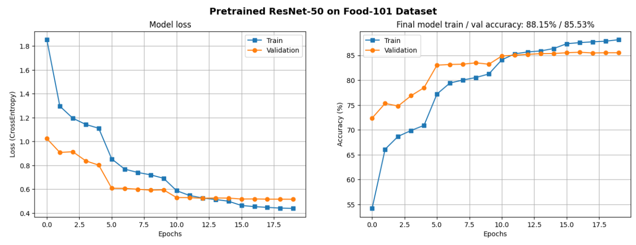

Figure 2. ResNet-50 Training Results

3.8. Web Application

The web application was developed using Flask (Python) as the backend framework, serving a trained ResNet-50 model for efficient food image classification. It accepts uploaded images via HTTP POST, preprocesses them, and returns the top-3 predicted food classes with probabilities, while also leveraging a large language model through the Groq API to estimate calorie ranges for the top prediction. The backend handles API requests and responses in JSON, manages sensitive keys securely with environment variables, and stores uploads in a static directory. On the frontend, we used Flask's Jinja2 template engine.

4. Results

4.1. MobileNet-V2

MobileNet was trained for 7 epochs after which it was stopped due to the early stopping (indicating no increase in validation accuracy for 5 epochs) with patience=5. It doesn't generalize well and so we decided to move with some heavier models such as ResNet-50.

MobileNet-V2	Accuracy
Training	60.2%
Validation	54.79%

Table 1. Accuracy for MobileNet-V2

ResNet-50	Accuracy
Training	88.15%
Validation	85.53%
Test on .h5 set	68.10%

Table 2. Accuracy for ResNet-50

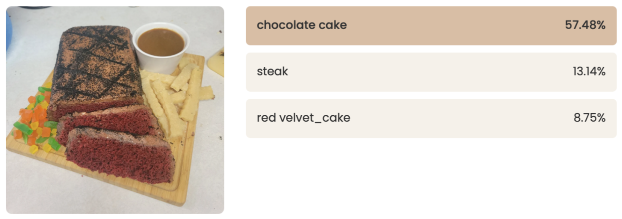

Figure 3. A homemade red velvet cake misclassified as chocolate cake (likely due to dark exterior), then steak (texture), and finally red velvet cake.

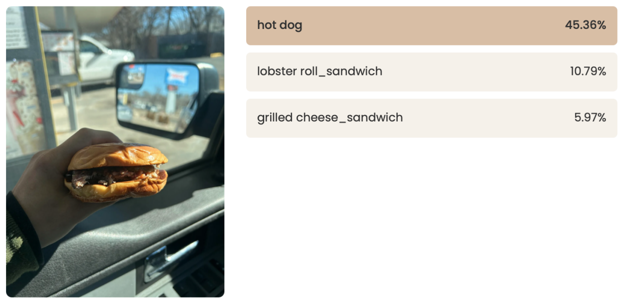

Figure 4. A pulled pork sandwich misclassified as hot dog, but the predictions are reasonable and not entirely off.

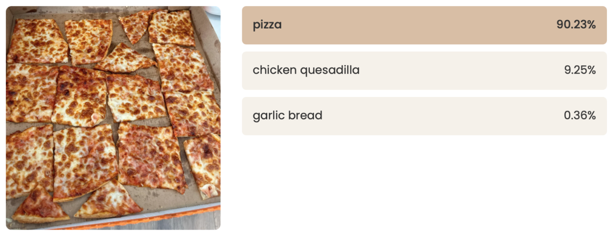

Figure 5. Pizza seems to be correctly classified, in this example the pizza isn't round or standard slice shaped but still model correctly classifies it.

4.2. ResNet-50

Training and validation loss decrease steadily, with validation loss plateauing after around 15 epochs. Our results showed that ResNet-50 achieved strong performance, with a validation accuracy of 85.5%.

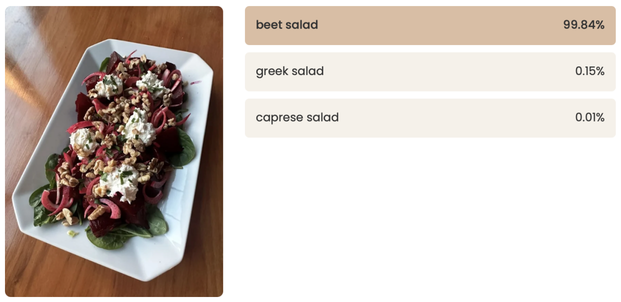

Figure 6. Beet root salad - the salad like features along with the red color could be the clue for the model.

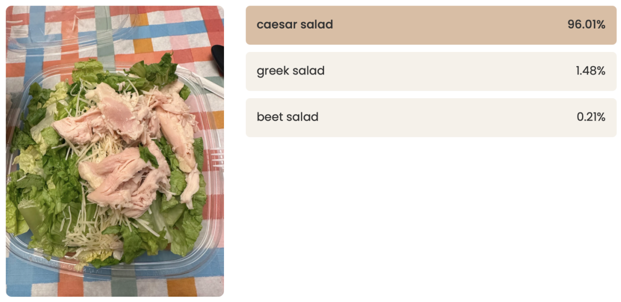

Figure 7. Caesar Salad with raw chicken on top, but the model correctly identifies the salad.

5. Discussion

Our experiments demonstrate that fine tuned ResNet-50 architecture performes well on classifying the Food-101 dataset, achieving the highest accuracy of 85.53 percent. This performance gain likely results from the model's combined ability to capture local features through convolutions and global context via self-attention, which is especially useful for fine-grained visual recognition such as food classification.

Despite the strong quantitative results, some limitations and nuances emerged during qualitative analysis. In particular, the model occasionally misclassified visually ambiguous items - yet these errors were typically reasonable. For example, a homemade red velvet cake was initially misclassified as chocolate cake, likely due to its dark exterior, then as steak (possibly influenced by the texture and lighting), and finally correctly identified as red velvet cake. Similarly, a pulled pork sandwich was misclassified as a hot dog. While technically incorrect, these predictions fall within a semantically similar category and reflect the inherent challenge of distinguishing between visually alike or compositionally complex foods.

This pattern reveals that even when the predictions are incorrect, they are usually close approximations of the true label. This suggests that the model has learned a robust visual feature space, but may benefit from more training or something else to improve it.

Additionally, the current classification pipeline only supports single-label predictions, which is limiting in the case of mixed dishes containing multiple components. Extending the architecture to support multi-label classification is a natural next step and would more accurately reflect real-world food consumption.

Other limitations include sensitivity to image quality - low lighting or poor resolution negatively impacted predictions despite augmentation and the lack of a volume or calorie estimation module. These are critical capabilities for full dietary assessment, and we do plan to explore more about it.

In summary, Crimson Nutrition demonstrates high classification accuracy and strong generalization. Its errors are mostly within reasonable boundaries, indicating a mature model foundation. With extensions in explainability, multi-labeling, and volumetric analysis, the system can evolve into a practical tool for everyday nutritional monitoring.

6. Conclusion and Future Work

Crimson Nutrition presents a practical computer vision solution for food classification and nutritional insight. By leveraging the Food-101 dataset and modern deep learning architectures, we demonstrated that high-accuracy classification of food items from user-submitted images is not only feasible but also robust under good conditions. One of our key findings is that even when the model makes errors, its predictions are often semantically close to the correct label, which speaks to its strong feature understanding.

Looking ahead, there are several promising directions to expand this work:

- Implementing multi-item classification would allow the system to detect and recognize multiple foods in a single image, more accurately reflecting real-life meals.
- Integrating calorie estimation through visual volume analysis or packaging metadata would make the system more personalized and practically useful for nutrition tracking.
- Exploring ingredient-based analysis could help in understanding mixed dishes and offering dietary recommendations for people with allergies or specific nutrition goals.
- Integrate with IU Dining and Hospitality NetNutrition® for real-time food metadata, and menu-based tracking.
- Include underrepresented cuisines from around the world to support international users and travelers with culturally diverse diets.

Crimson Nutrition gives us a good foundational understanding and groundwork in the field of Food and Nutrition. We have many ways to go ahead with future research and are excited to work in improving this tool and deploy it in real world once we feel it's ready to be used.

References

[1] Bossard, Lukas, Matthieu Guillaumin, and Luc Van Gool. ”Food-101 – Mining Discriminative Components with Random Forests.” ECCV, 2014. 1

[2] Tahir, Ghalib, and Chu Kiong Loo. ”A Comprehensive Survey of Image-Based Food Recognition and Volume Estimation.” arXiv preprint arXiv:2106.11776, 2021. 1

[3] Amugongo, Lameck M., et al. ”Mobile Computer Vision-Based Applications for Food Recognition and Calorific Estimation: A Systematic Review.” Healthcare, 2023. 1

[4] Xiong, Yiming. (2023). Food Image Recognition based on ResNet. Applied and Computational Engineering. 8. 623-629. 10.54254/2755-2721/8/20230284.