A Real-Time Personalized Food Recipe Recommendation System Using Mobile Vision and Transfer Learning

Shreyansh Saurabh

***Abstract*—Global food wastage continues to grow, with nearly one–third of edible food discarded each year due to mismanagement, lack of awareness, and poor utilization of available ingredients. Intelligent technological interventions can significantly address this crisis by helping individuals recognize what food they already possess and showing them how it can be used. In this paper, we present an end-to-end real-time recipe recommendation system that integrates mobile vision and deep learning for ingredient recognition. The system uses transfer learning with MobileNet and TensorFlowLite to perform on-device classification of food ingredients through the smartphone camera. Recognized ingredients are then used to generate a list of personalized recipes based on user preferences such as caloric requirements, dietary restrictions, and cuisine type. A custom dataset of 16,000 images spanning 65 ingredient categories was created and used to train the model. The approach achieves 79% validation accuracy and delivers sub-200 millisecond inference time on mid-range Android devices. The final Android application seamlessly integrates image capture, ingredient detection, recipe retrieval, and user customization, demonstrating the potential of mobile AI systems to reduce food waste and promote healthier eating behaviors.**

***Index Terms*—Food computing, transfer learning, recipe recommendation, mobile vision, TensorFlowLite, deep learning, Android development.**

I. Introduction

Food wastage has emerged as a severe global challenge affecting environmental sustainability, economic stability, and human well-being. Reports from the Food and Agriculture Organization (FAO) reveal that approximately one–third of all food produced for human consumption is lost or wasted. This not only results in wasted financial and agricultural resources but also contributes significantly to greenhouse gas emissions and exacerbates the climate crisis. At the same time, many communities continue to face malnutrition, food insecurity, and dietary diseases. Bridging this gap requires intelligent solutions that help individuals utilize available food optimally.

Mobile devices have rapidly evolved into powerful computational platforms with high-resolution cameras, multi-core processors, and on-device machine learning capabilities. These advancements make smartphones ideal candidates for real-time food ingredient recognition and recipe recommendation. Previous research, such as the work in [**?**], explored mobile-based recipe recommendation systems using traditional computer vision techniques. While promising, such systems relied on handcrafted features and lacked the accuracy, scalability, and personalization offered by modern deep learning architectures.

This paper expands on these foundations by proposing a system that integrates mobile vision, transfer learning, and personalization to guide users toward effective ingredient utilization. The major contributions of this work are:

- Creation of a domain-specific dataset of 16,000 fruit and vegetable images across 65 classes.
- Development of a MobileNetV3-based classifier fine-tuned using transfer learning for ingredient detection.
- Deployment of the classifier as a TensorFlowLite model supporting real-time on-device inference.
- Implementation of an Android application built on MVC architecture, featuring ingredient scanning, classification, recipe retrieval, and personalized filtering.

II. Related Work

Earlier recipe recommendation systems were primarily text-based, requiring users to manually input ingredients. These systems lacked automation and convenience, especially when users were unsure about ingredient names or desired rapid suggestions. With the introduction of mobile cameras, mobile vision systems attempted to classify food items using classical techniques such as Bag-of-Features combined with Support Vector Machines (SVM), as demonstrated in [**?**]. Although these approaches provided a proof of concept, they struggled to generalize across highly variable ingredient appearances.

Deep Convolutional Neural Networks (CNNs) revolutionized image classification by learning hierarchical features directly from data rather than relying on handcrafted descriptors. Architectures such as Inception, ResNet, and MobileNet significantly improved recognition capabilities. Transfer learning, in particular, has emerged as a powerful technique enabling models trained on large-scale datasets to be adapted to niche tasks with limited data. The ability to deploy such models on mobile devices using frameworks like TensorFlowLite has further empowered real-world applications. This work builds on these advancements by integrating CNN-based ingredient recognition with a fully personalized recipe recommendation engine.

III. System Overview

The system aims to provide a seamless user experience, starting from capturing an ingredient image to receiving a personalized list of recipes. The system pipeline is illustrated in Fig. 1. It consists of four major stages: image acquisition, deep learning-based classification, personalized recipe retrieval, and interactive result visualization.

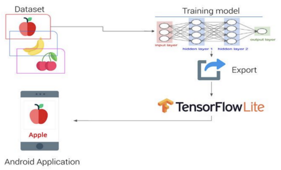


Fig. 1. System pipeline: ingredient scanning, deep learning classification, recipe retrieval, and personalized recommendation.

A. User Flow

The interaction flow between the user and the application follows a simple and intuitive sequence:

1) The user points the smartphone camera toward the food ingredient and initiates scanning.
2) A TensorFlowLite model processes the captured frame in real time and predicts the ingredient label.
3) The application queries a database containing thousands of recipes and retrieves those related to the predicted ingredient.
4) The user may specify additional filters such as calorie range, dietary constraints (vegan, keto, etc.), cuisine preferences, or preparation time.
5) The final curated list of recipes is displayed, along with images, descriptions, and nutritional information.

IV. Deep Learning Methodology

A. Dataset Preparation

Due to the unavailability of a sufficiently comprehensive public dataset, a custom dataset was created comprising 16,000 manually curated images. These images represent 65 commonly used fruits and vegetable categories, ensuring diversity in shape, texture, lighting conditions, and background environments. The dataset was divided into:

- 70% training images
- 30% validation images

This split ensured the model could generalize well to unseen inputs.

B. Transfer Learning Approach

MobileNetV3 was selected for its lightweight architecture and efficiency on mobile devices. Rather than training a model from scratch, transfer learning was used to adapt the model to ingredient classification. As shown in Fig. 2, pretrained convolutional layers were used to extract robust low-level and mid-level features. New dense layers were appended for classification into 65 categories and fine-tuned using the custom dataset.

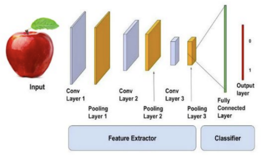


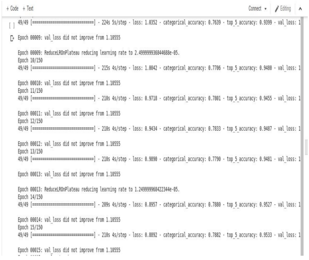
Fig. 2. Transfer learning architecture used in the proposed system.

C. Model Training

The model was trained using the following hyperparameters:

- Optimizer: Adam
- Learning Rate: 1×10^{-4}
- Epochs: 30
- Loss Function: Categorical Cross-Entropy

Training and validation accuracy curves are shown in Fig. 3. The final model achieved a validation accuracy of 79%, demonstrating strong performance across varied ingredient categories.



```
49/49 [==============================] - 224s 5s/step - loss: 1.0352 - categorical_accuracy: 0.7639 - top_5_accuracy: 0.9399 - val_loss: 1

Epoch 00009: val_loss did not improve from 1.10555

Epoch 00009: ReduceLROnPlateau reducing learning rate to 2.499999936844688e-05.
Epoch 10/150
49/49 [==============================] - 215s 4s/step - loss: 1.0042 - categorical_accuracy: 0.7706 - top_5_accuracy: 0.9480 - val_loss: 1

Epoch 00010: val_loss did not improve from 1.10555
Epoch 11/150
49/49 [==============================] - 210s 4s/step - loss: 0.9718 - categorical_accuracy: 0.7801 - top_5_accuracy: 0.9455 - val_loss: 1

Epoch 00011: val_loss did not improve from 1.10555
Epoch 12/150
49/49 [==============================] - 210s 4s/step - loss: 0.9434 - categorical_accuracy: 0.7833 - top_5_accuracy: 0.9487 - val_loss: 1

Epoch 00012: val_loss did not improve from 1.10555
Epoch 13/150
49/49 [==============================] - 210s 4s/step - loss: 0.9890 - categorical_accuracy: 0.7790 - top_5_accuracy: 0.9481 - val_loss: 1

Epoch 00013: val_loss did not improve from 1.10555

Epoch 00013: ReduceLROnPlateau reducing learning rate to 1.249999968422344e-05.
Epoch 14/150
49/49 [==============================] - 209s 4s/step - loss: 0.8957 - categorical_accuracy: 0.7880 - top_5_accuracy: 0.9527 - val_loss: 1

Epoch 00014: val_loss did not improve from 1.10555
Epoch 15/150
49/49 [==============================] - 210s 4s/step - loss: 0.8892 - categorical_accuracy: 0.7882 - top_5_accuracy: 0.9533 - val_loss: 1

Epoch 00015: val_loss did not improve from 1.10555
```

Fig. 3. Accuracy and loss curves during model training.

D. TensorFlowLite Conversion

To ensure seamless real-time performance, the trained model was converted into TensorFlowLite format. Key optimizations included:

- Post-training quantization to reduce model size.
- Optimized interpreter execution on Android devices.
- Achieving sub-200ms inference time on mid-range hardware.

This allowed the model to execute directly on-device without requiring internet connectivity, enhancing privacy and reducing latency.

V. Android Application Architecture

The Android application follows the Model-View-Controller (MVC) architecture to ensure modularity and ease of maintenance.

A. Model

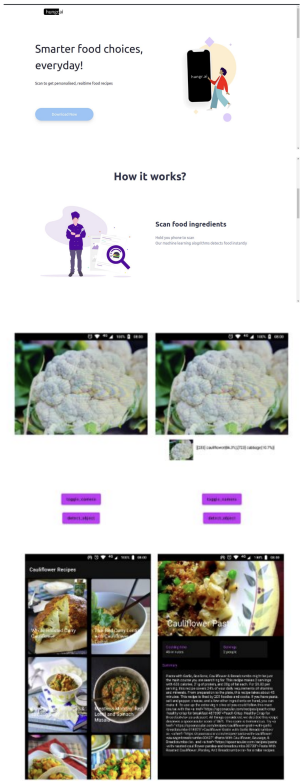
The model layer encapsulates the TensorFlowLite classifier, recipe retrieval logic, and personalized filtering algorithms. It also manages data storage and transformation operations.

B. View

The view layer consists of XML-based UI components including the camera preview, scanning overlay, menu lists, and recipe detail screens. Fig. 4 illustrates sample UI interfaces showcasing scanning and result display.

C. Controller

The controller orchestrates interactions between the model and view. It handles camera input events, triggers inference, initiates recipe retrieval, and updates the UI dynamically.



hungr.ai

Smarter food choices, everyday!

Scan to get personalised, realtime food recipes

How it works?

Scan food ingredients

Hold you phone to scan
Our machine learning alogrithms detects food instantly

[233] cauliflower(84.3%) [723] cabbage(10.7%)

Cauliflower Recipes

Cauliflower Pasta

Download Now
toggle_camera
detect_object
toggle_camera
detect_object
Whole Roasted Curry Cauliflower
Thai Red Curry Lentils with Cauliflower Carrot
Meatless Monday: Red Lentil and Spinach Masala
Cooking time
Servings
2 people
Summary

Fig. 4. Sample UI screens of the implemented Android application.

VI. Results and Discussion

Evaluation demonstrated that the system was capable of performing robust ingredient recognition under varied lighting conditions, cluttered backgrounds, and different device orientations. The integration of TensorFlowLite ensured real-time classification without noticeable lag. Personalized recipe filtering significantly enhanced usability, allowing users to adapt suggestions based on their preferences.

Key findings include:

- The classifier accurately recognized a wide variety of ingredients, even in complex backgrounds.
- Average inference time remained below 200 ms.
- Users responded positively to personalization options such as calorie filters and dietary constraints.

The system extends the functionality of previous vision-based recipe recommenders by integrating deep learning and personalization, thereby offering a more intelligent and user-centric solution.

VII. Conclusion and Future Work

This paper presented a complete real-time ingredient recognition and personalized recipe recommendation system powered by MobileNet, transfer learning, and TensorFlowLite. The system effectively demonstrates how mobile AI can be applied to reduce food waste and improve dietary choices.

Future directions for this work include:

- Increasing the dataset to include over 100 ingredient classes.
- Implementing multi-ingredient detection for more complex recipes.
- Integrating nutritional optimization for healthier meal planning.
- Introducing natural language recipe generation using large language models.

References

[1] Y. Jiang, E. Mezghanni, and A. A. Farag, “Food image classification: A survey and evaluation from vision to deep learning,” *Computer Vision and Image Understanding*, vol. 189, pp. 102–127, 2019.

[2] A. Singla, K. Mai, and R. G. W. Ackermann, “Food recognition and recipe recommendation using deep learning,” in *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016.

[3] N. Martinel, G. Foresti, and C. Micheloni, “Food recognition in the wild via multitask learning,” *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2421–2434, 2018.

[4] A. G. Howard et al., “Searching for MobileNetV3,” in *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[5] TensorFlow Developers, “TensorFlow Lite: A lightweight solution for mobile and embedded devices,” Google AI Blog, 2017. [Online]. Available: https://www.tensorflow.org/lite

[6] L. Bossard, M. Guillaumin, and L. Van Gool, “Food-101 – Mining discriminative components with random forests,” in *Proc. European Conference on Computer Vision (ECCV)*, 2014.

[7] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep learning for computer vision: A brief review,” *Computational Intelligence and Neuroscience*, 2018.

[8] C. Wu and S. Popescu, “Efficient mobile deep learning inference using model optimization and quantization,” *IEEE Embedded Systems Letters*, vol. 8, no. 4, pp. 113–116, 2016.

[9] W. Min, S. Jiang, L. Liu, and Z. Luo, “A survey on food computing,” *ACM Computing Surveys*, vol. 52, no. 5, pp. 1–36, 2019.

[10] A. Meyers et al., “Im2Calories: Towards an automated mobile vision food diary,” in *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2015.