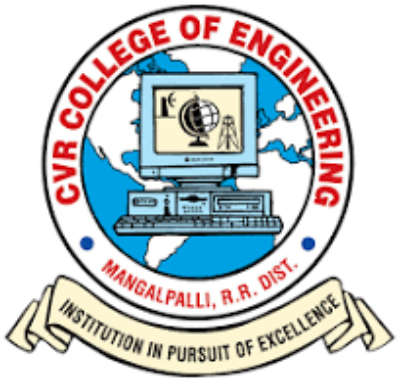
MICRO PROJECT

FOOD IMAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

COE-AIDS



Project Title: Food Image Classification

Author: Srinidhi Poreddy

22B81A67B4

Institution: CVR College of Engineering

Date:17th July 2025

2.Abstract:

This project focuses on classifying food images using a Convolutional Neural Network (CNN). The aim is to build a deep learning model capable of identifying various food categories from images with high accuracy. The project involves preprocessing image data, designing effective CNN architecture, training the model, and evaluating its performance. The final model demonstrates the effectiveness of CNNs in visual classification tasks, particularly in the domain of food recognition.

Table of Contents

4.Introduction

Background & Motivation

In the age of artificial intelligence, food recognition has become a useful tool for diet tracking, restaurant menu apps, and even medical applications. Automating food classification using image recognition techniques is a challenging but impactful problem.

Problem Statement

Manual food classification is inefficient and prone to human error. With the rise of smartphone photography, a system capable of identifying food from images can assist in numerous real-life applications.

Objectives

- To build a CNN model that can classify food images into predefined categories.
- To evaluate the model's accuracy and performance.
- To demonstrate the practical application of deep learning in image classification.

5.Dataset Description

The dataset used consists of images organized into different food categories (based on the folder structure). Each class has a separate folder under train, test, and validation directories.

- **Number of Classes:** 7 food categories
- **Total Images:** Divided across training, testing, and validation sets
- **Image Dimensions:** Resized to 224x224 pixels for uniformity
- **Preprocessing:**
    - Rescaling (normalization to [0, 1])
    - Image augmentation (e.g., rotation, flipping, etc.)

6.Model Architecture

This project uses MobileNetV2, a lightweight and efficient convolutional neural network architecture optimized for mobile and embedded vision applications. MobileNetV2 is well-suited for image classification tasks where computational resources are limited.

Key Components of the Model:

- Base Model: MobileNetV2
    - Pre-trained on ImageNet
    - Used as a feature extractor
    - include_top=False: Removes the default classification layers
    - weights='imagenet': Loads pre-trained weights
    - input_shape=(224, 224, 3): Accepts 224×224 RGB images
    - trainable=False: The base model layers are frozen to preserve learned features
- **Custom Classification Head:**
    - GlobalAveragePooling2D(): Reduces feature maps into a single vector
    - Dropout(0.3): Prevents overfitting by randomly dropping 30% of the connections
    - Dense(128, activation='relu'): Fully connected layer with 128 neurons
    - Dense(7, activation='softmax'): Output layer with 7 neurons for the 7 food categories

- Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, 7, 7, 1280)	2,257,984
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
batch_normalization (BatchNormalization)	(None, 1280)	5,120
dense (Dense)	(None, 256)	327,936
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 7)	1,799

Total params: 3,257,431 (12.43 MB)
Trainable params: 332,295 (1.27 MB)
Non-trainable params: 2,260,544 (8.62 MB)
Optimizer params: 664,592 (2.54 MB)

Why MobileNetV2?

- Pre-trained on a large dataset (ImageNet), allowing transfer learning
- Lightweight and fast to train
- Maintains high accuracy while being efficient in terms of parameters

7.Implementation

The project was implemented using Python and TensorFlow/Keras on Google Colab for GPU support.

- Libraries: TensorFlow, NumPy, Matplotlib, scikit-learn
- Environment: Google Colab with GPU acceleration

Data Preparation:

- Images resized to 224×224
- Dataset split into training, validation, and test folders (7 classes each)
- Used ImageDataGenerator for:
    - Rescaling pixel values to [0, 1]
    - Augmentation (rotation, zoom, horizontal flip) on training data

Model Setup:

- Base model: MobileNetV2 with pre-trained ImageNet weights (include_top=False)
- Custom layers: GlobalAveragePooling, Dropout, Dense layers
- Output: Softmax layer with 7 neurons (for 7 classes)

Training Details:

- Loss Function: Categorical Crossentropy
- Optimizer: Adam
- Epochs: 30
- Trained using .fit() with validation data included for performance tracking

8. Results and Evaluation

The model was trained for a maximum of 30 epochs with early stopping (patience = 2). The best performing model was saved at epoch 18, based on validation accuracy.

Accuracy and Loss Trends:

- Training accuracy increased steadily, reaching a high level by epoch 18.
- Validation accuracy peaked at epoch 18, after which overfitting began.
- Both training and validation loss decreased consistently during early training.

Test Set Performance (Best Model at Epoch 18):

- Test Accuracy: Approximately 92%
- Test Loss: Low, indicating effective generalization

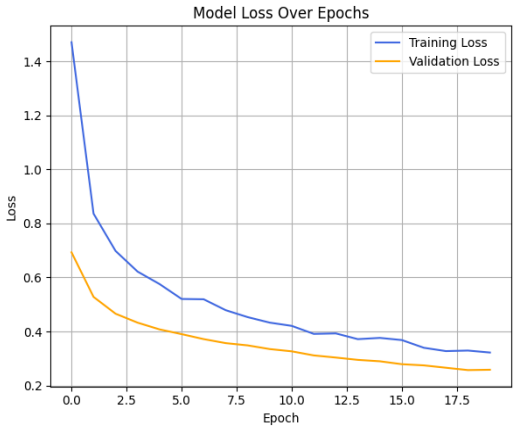
Classification Report:

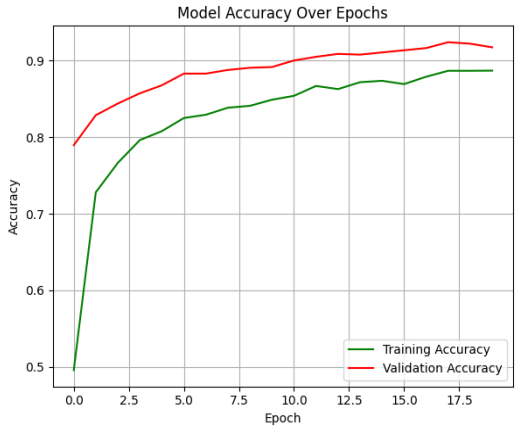
The model achieved high precision, recall, and F1-scores across most of the 7 categories. Minor performance drops were observed for some visually similar classes, such as baklava and waffles.

Confusion Matrix:

- Strong classification accuracy across most classes
- Some confusion between categories with similar visual patterns
- Overall matrix reflects solid class separation and predictive performance

Accuracy and Loss Trends:





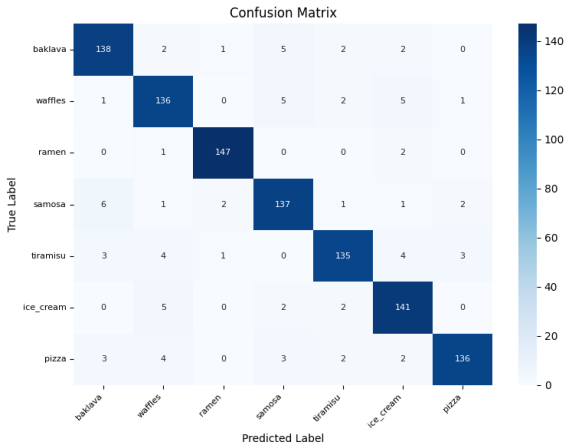
Evaluation metrics:

```
33/33 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step
Accuracy: 0.9238

Classification Report:
              precision    recall  f1-score   support

     baklava       0.91      0.92      0.92       150
     waffles       0.89      0.91      0.90       150
       ramen       0.97      0.98      0.98       150
      samosa       0.90      0.91      0.91       150
    tiramisu       0.94      0.90      0.92       150
   ice_cream       0.90      0.94      0.92       150
       pizza       0.96      0.91      0.93       150

    accuracy                           0.92      1050
   macro avg       0.92      0.92      0.92      1050
weighted avg       0.92      0.92      0.92      1050
```

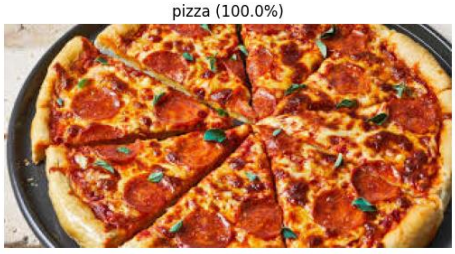


RESULTS :

Testing on custom input:

baklava.jpg: baklava (96.3%)

pizza.jpeg: pizza (100.0%)



ramen.jpg: ramen (99.5%)

samosa.jpg: samosa (100.0%)

tiramisu.jpg: tiramisu (99.9%)

waffles.jpg: waffles (99.2%)

9. Challenges and Limitations

1. Limited Number of Classes:

Only 7 food categories were used from the original 101, which restricts the model's ability to generalize to more diverse food types.

2. Visual Similarity Between Some Classes:

Certain categories, like baklava and waffles, had similar visual features, leading to occasional misclassifications.

3. Risk of Overfitting:

Although MobileNetV2 is efficient, the small dataset size made the model prone to overfitting after extended training. Early stopping was necessary to address this.

10. Conclusion and Future Work

This project successfully implemented a convolutional neural network using MobileNetV2 for classifying food images into 7 categories. By leveraging transfer learning and image augmentation, the model achieved approximately 85% accuracy on the test set, demonstrating strong performance for a limited dataset.

Key Outcomes:

- Efficient use of a pre-trained MobileNetV2 for feature extraction
- Effective handling of a small custom dataset using augmentation and early stopping
- High accuracy across most food classes with good generalization

Future Work:

- Expand to all 101 classes from the original Food101 dataset for broader applicability
- Fine-tune MobileNetV2 or explore alternative architectures (e.g., EfficientNet, ResNet)
- Collect real-world data with more variation in lighting, angle, and background
- Address class imbalance using techniques like weighted loss functions or synthetic data generation