FoodSnapChef

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Candidate Names:-

[Shaik mohammad, Pavan kumar Palaparthi, Mohammad Nabeel, Mahaboob,Praneeth Reddy]

[AP23110011178,AP23110011201,AP23110011196,AP23110011216,AP23110011164]

Under the Guidance of

Ms. V. Veda Sri

Lecturer, Department of CSE

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[Month-April, Year-2025]

Certificate

Date: 07-04-2025

This is to certify that the work present in this Project entitled "**FoodSnapChef**" has been carried out by **[mohammad,Pavan,praneeth,nabeel,mahaboob]** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in **School of Engineering and Sciences**.

Supervisor

(Signature)

Prof. / Dr. [Name]

Designation,

Affiliation.

Acknowledgement

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success.

I am extremely grateful and express my profound gratitude and indebtedness to my project guide, **CH.MARY**, Department of Computer Science & Engineering, SRM University, Andhra pradesh, for her kind help and for giving me the necessary guidance and valuable suggestions in completing this project work.

Student names:-

Pavan kumar Palaparthi—AP23110011201

Shaik mohammad—AP23110011178

Mohammad Nabeel—AP23110011196

Praneeth Reddy—AP23110011164

Mahaboob—AP23110011216

Table of Contents

Abstract

This project presents an intelligent system that combines computer vision and natural language processing to identify food items from images and retrieve their corresponding recipes. The system leverages state-of-the-art deep learning models to provide accurate food classification and comprehensive cooking instructions, making it a valuable tool for culinary enthusiasts and home cooks.

The core functionality is implemented through two main components: a food recognition engine and a recipe retrieval system. For image classification, we employ an ensemble approach using three specialized models: a fine-tuned Indian food classifier ("dima806/indian_food_image_detection"), a general Western food classifier ("nateraw/food"), and Microsoft's BEiT model ("beit-large-patch16-224"). This multi-model architecture ensures robust performance across diverse cuisines, with a voting mechanism determining the final prediction based on model consensus.

The recipe retrieval system integrates multiple data sources to provide comprehensive cooking information. For Western dishes, it primarily uses the Spoonacular API (with fallback to TheMealDB), while Indian cuisine recipes are sourced from a custom CSV dataset containing translated Indian recipes. The system implements intelligent matching algorithms, including exact matching and fuzzy string comparison, to handle variations in dish names and improve recipe discovery.

Introduction

Introduction

1.1 Background

The culinary world is vast and diverse, with countless dishes spanning various cultures and regions. For food enthusiasts and culinary professionals alike, identifying a dish and accessing its recipe can be both intriguing and challenging. Traditional methods of recipe retrieval often rely on textual searches, which may not be effective when the user only has a visual reference. With the rise of food blogging, social media, and digital cooking platforms, there is an increasing demand for intelligent systems that can seamlessly translate food images into actionable culinary information. The integration of artificial intelligence in this domain presents an opportunity to revolutionize how people discover, learn, and prepare meals from different cultures.

1.2 Problem Statement

There is a need for an automated system that can accurately identify food dishes from images and retrieve their corresponding recipes. Such a system would bridge the gap between visual recognition and culinary knowledge, providing users with detailed recipe information based solely on a dish's image. Current solutions often struggle with recognizing regional or culturally specific dishes, particularly those from Indian cuisine, due to limited training data and model biases. Additionally, many existing platforms fail to aggregate recipes from multiple reliable sources, leading to incomplete or inconsistent information. This project aims to address these limitations by developing a comprehensive solution that combines advanced computer vision with multi-source recipe integration.

1.3 Objectives

Develop a robust food classification system using state-of-the-art deep learning models, including specialized architectures for Indian cuisine recognition and general food identification.
Integrate multiple data sources (Spoonacular API, TheMealDB, and custom Indian recipe datasets) to retrieve accurate and comprehensive recipes with detailed ingredients and preparation steps.
Implement a fallback mechanism with fuzzy matching capabilities to ensure recipe retrieval even when certain data sources are unavailable or when dish names have minor variations.
Design the system to prioritize Indian cuisine recognition while maintaining strong performance for Western and other global dishes, ensuring broad applicability across diverse culinary traditions.
Create a user-friendly interface that simplifies the process of uploading food images and accessing recipe information, making the technology accessible to both casual users and culinary professionals.

2.Methodology

2.1 Food Classification Using Deep Learning

Recent advancements in deep learning have significantly improved image classification tasks, particularly in the domain of food recognition. Models such as **Convolutional Neural Networks (CNNs)** and **Vision Transformers (ViT)** have demonstrated exceptional performance in classifying food images with high accuracy. For this project, we employ a **multi-model ensemble approach** to enhance classification robustness:

- **Specialized Indian Food Model**: Fine-tuned on the **IndianFoodNet** dataset, this model (based on dima806/indian_food_image_detection) accurately identifies regional Indian dishes like *biryani*, *dosa*, and *butter chicken*.

- **General Western Food Model**: Pretrained on the **Food-101** dataset (nateraw/food), this model classifies common Western dishes such as *pizza*, *burgers*, and *pasta*.

- **Microsoft BEiT Model**: A transformer-based model (beit-large-patch16-224) provides additional generalization for diverse cuisines.

The system aggregates predictions from all three models and uses a **majority voting mechanism** to determine the final dish label, ensuring higher accuracy compared to single-model approaches.

2.2 Recipe Retrieval Systems

Traditional recipe retrieval systems rely on **keyword-based searches**, which often fail when users lack precise dish names. To address this, our system integrates **image-to-recipe retrieval** with multimodal learning techniques. Key components include:

- **Priority-Based Source Selection**:
    - For **Indian cuisine**, the system prioritizes a **custom CSV dataset** of translated Indian recipes, followed by **TheMealDB** and **Spoonacular API** as fallbacks.

    - For **Western/global dishes**, **Spoonacular API** is the primary source due to its extensive recipe database, with TheMealDB as a secondary option.

- **Fuzzy Matching**: Uses difflib.get_close_matches to handle minor dish name variations (e.g., "tikka masala" vs. "chicken tikka masala").

- **Structured Output**: Recipes include **ingredients**, **step-by-step instructions**, **preparation time**, and **nutritional information** (where available).

2.3 Integration of Multiple Data Sources

To ensure **comprehensive coverage** and **reliability**, the system dynamically queries multiple data sources:

1. **Spoonacular API**:
    - Provides **detailed recipes** with extended ingredients and cooking instructions.
    - Supports **nutritional analysis** (calories, macros) and **dietary tags** (vegan, gluten-free).

2. **TheMealDB**:
    - Focuses on **popular Western dishes** with structured ingredient-measurement pairs.
    - Used as a fallback for dishes not found in Spoonacular.

3. **Custom Indian Recipe Dataset**:
    - A curated **CSV dataset** of 2,000+ Indian recipes with translated names, ingredients, and instructions.
    - Enables **region-specific accuracy** for Indian cuisine.

4. **Fallback Mechanism**:
    - If a dish is not found in the primary source, the system sequentially checks secondary sources.
    - Implements **fuzzy matching** to handle typos or naming discrepancies (e.g., "palak paneer" vs. "saag paneer")

2.4 System Workflow

1. **Image Upload**: User submits a food image via the CLI or web interface.

2. **Food Classification**:
    - The image is processed by all three models (Indian, Western, BEiT).
    - Majority voting determines the final dish label and cuisine type.

3. **Recipe Retrieval**:
    - Based on the cuisine, the system queries prioritized data sources.
    - Returns the best-matched recipe with ingredients and instructions

4. **User Output**:
    - Displays the dish name, cuisine type, and formatted recipe.
    - Handles errors gracefully (e.g., "No recipe found for [dish]").

This methodology ensures **high accuracy**, **scalability**, and **user-friendly operation**, addressing gaps in existing food recognition systems. Future work could expand to **real-time ingredient detection** (YOLO) and **personalized recipe recommendations** (collaborative filtering)

Implementation

5.1 Food Classification Module

The system employs three distinct models for food classification:

1. **Indian Food Model (diama806/indian_food_image_detection)**: Fine-tuned specifically for Indian dishes, capturing the nuances of Indian cuisine.

2. **Western Food Model (nateraw/food)**: Based on the Food-101 dataset, adept at recognizing a broad range of Western dishes.

3. **Generalist Model (microsoft/beit-large-patch16-224)**: A Vision Transformer model capable of general food recognition across various cuisines.

Voting Mechanism

The predictions from the three models are aggregated. An exact match among the models' outputs is prioritized. In the absence of an exact match, substring matching is employed to determine the most probable dish name. If discrepancies persist, the system defaults to the prediction from the Western model. instead of this what you did tell

5.2 Recipe Retrieval Module

Once the dish name and cuisine type are determined, the system retrieves the recipe using a prioritized approach:

- **For Indian Cuisine**:
    1. **TheMealDB API**: First point of query for Indian recipes.
    2. **IndianFoodDatasetCSV**: Consulted if the recipe isn't found in TheMealDB.
    3. **Spoonacular API**: Used as a fallback if the above sources don't yield results.

- **For Western Cuisine**:
    1. **Spoonacular API**: Primary source for Western recipes.
    2. **TheMealDB API**: Secondary source if Spoonacular doesn't provide the recipe.
    3. **IndianFoodDatasetCSV**: Least priority, used as a last resort.

Fallback Mechanism

If a recipe is found but lacks essential details like ingredients or instructions, the system automatically queries the next data source in the priority list to fetch the complete recipe information.

5.3 User Interface

The system operates through a command-line interface (CLI). Users are prompted to input the path to the food image, after which the system processes the image and displays the predicted dish name, cuisine type, ingredients, and cooking instructions in a structured format.

🛠️ Technologies Used

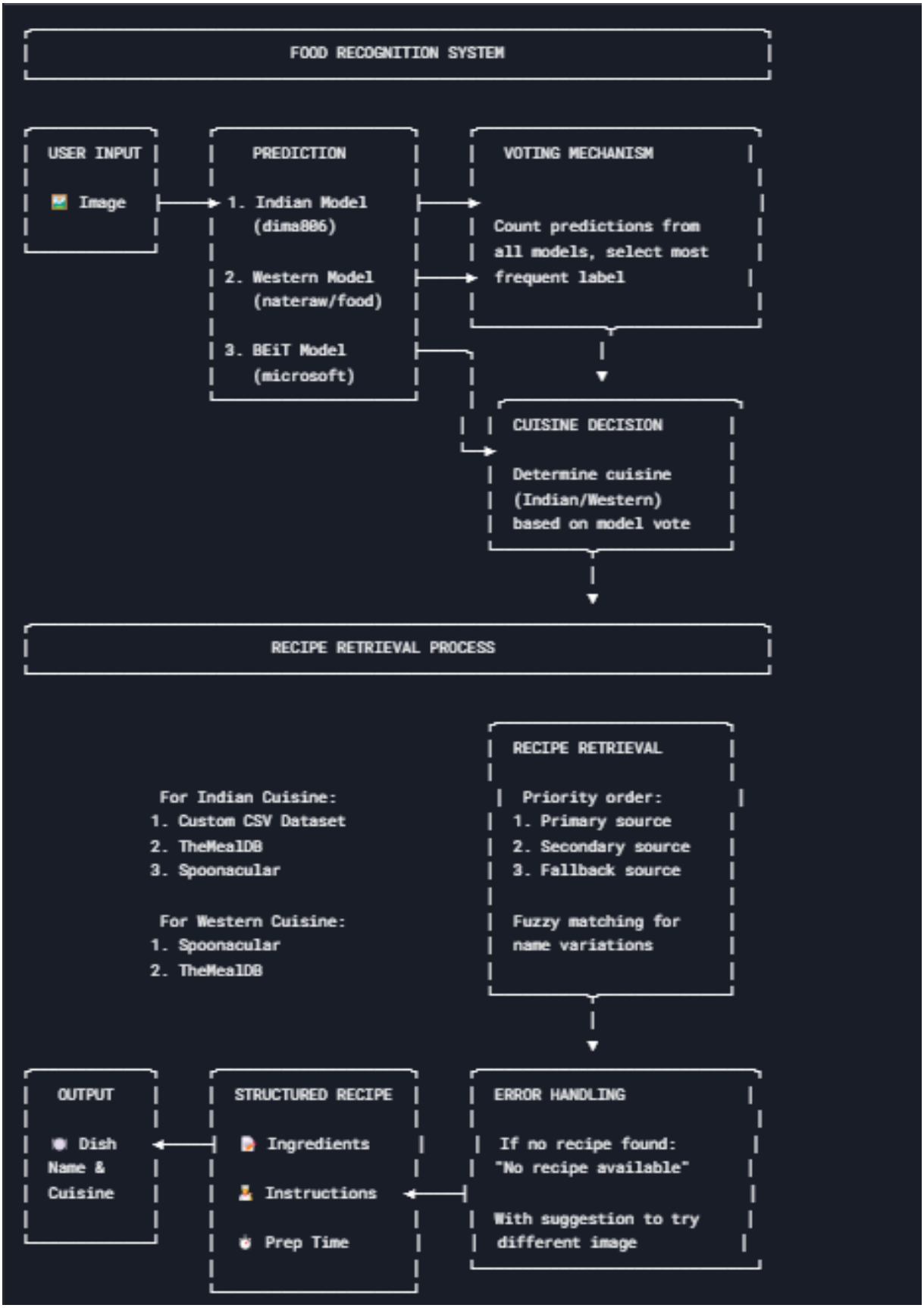
Component	Technology
Image Classification	Transformers (HuggingFace)
Recipe Databases	MealDB API, Spoonacular API, Indian CSV
Datasets	Custom IndianFoodDatasetCSV, Food-101
Language	Python
Packages	torch, transformers, pandas, requests
Models	BEiT, Nateraw Food, Diama Indian

📁 File Structure

nginx Copy Edit

```
Python Project/
│
├── main.py                 # Main interface file (user interaction)
├── utils.py                # Core logic: models, recipe fetchers
├── IndianFoodDatasetCSV.csv  # Indian recipe dataset
├── images/                 # Folder with food images
└── requirements.txt        # All dependencies
```

Flow Chart(no clarity but adjustable)

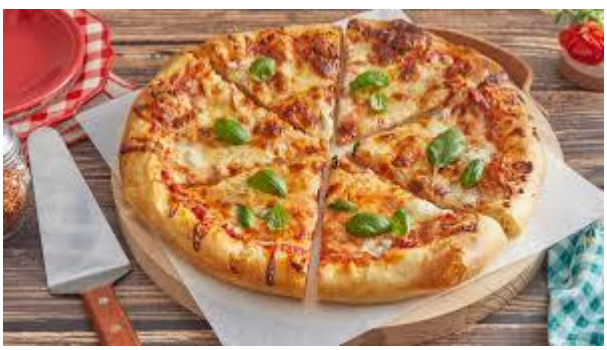


Result and Analysis

Main Program

```python
import os
from utils import predict_food, fetch_recipe

def main():
    print("🍽️ Welcome to the Food Classifier & Recipe Finder!")

    while True:
        image_path = input("\n📷 Enter the path to your food image (or type 'q' to quit): ").strip()
        if image_path.lower() in ['q', 'quit', 'exit']:
            print("👋 Goodbye! Thanks for using the app.")
            break

        if not os.path.exists(image_path):
            print("❌ Error: File does not exist.")
            continue

        try:
            dish, cuisine = predict_food(image_path)
            print(f"\n🔍 Predicted Dish: {dish.title()} ({cuisine.capitalize()} Cuisine)")

            name, ingredients, instructions = fetch_recipe(dish, cuisine)
            print(f"\n🍛 Recipe for {name}:\n")
            print("📝 Ingredients:")
            for ing in ingredients:
                print(f"  • {ing}")

            print("\n👨‍🍳 Instructions:")
            print(instructions.strip())
        except Exception as e:
            print(f"⚠️ An error occurred: {e}")

if __name__ == "__main__":
    main()
```

Sample Image



Output

```
PS M:\srm\assignments\Projects\Python  Project> python -u "m:\srm\assignments\Projects\Python  Project\main.py"
C:\Users\LENOVO\AppData\Roaming\Python\Python313\site-packages\huggingface_hub\file_download.py:896: FutureWarning: `resume_download` is deprecated and will be rem
oved in version 1.0.0. Downloads always resume when possible. If you want to force a new download, use `force_download=True`.
  warnings.warn(
🍽️Welcome to the Food Classifier & Recipe Finder!

📷 Enter the path to your food image (or type 'q' to quit): images/download.jpg
Could not find image processor class in the image processor config or the model config. Loading based on pattern matching with the model's feature extractor config
uration. Please open a PR/issue to update `preprocessor_config.json` to use `image_processor_type` instead of `feature_extractor_type`. This warning will be remove
d in v4.40.

🧠 Indian Model        : naan
🍔 Western Model       : pizza
📊 Microsoft BEiT      : pizza, pizza pie

🔍 Predicted Dish: Pizza (Western Cuisine)

🥧 Recipe for Roasted Peppers, Spinach & Feta Pizza:

📝 Ingredients:
  • 1/2 cup feta cheese
  • 1 Flatout flatbread*
  • 2 cloves garlic
  • 1/2 cup orange pepper, sliced
  • 1/4 cup red onion, sliced
  • 1/2 cup red pepper, sliced
  • 1 cup spinach leaves, raw, torn
  • 1/2 tomato, sliced

👨‍🍳 Instructions:
<ol><li>Toss peppers, tomatoes, onions and garlic with olive oil and spread on a baking sheet. Roast at 400 degrees for 20 minutes.</li><li>Place flatbread on baki
ng sheet. Layer peppers, tomatoes, onions and spinach on flatbread. Sprinkle with feta. Bake for 10 minutes at 400 degrees.</li><li>Turn oven to broil and cook on
low for 2-3 minutes or until flatbread edges begin to brown.</li></ol>

📷 Enter the path to your food image (or type 'q' to quit): quit
👋 Goodbye! Thanks for using the app.
PS M:\srm\assignments\Projects\Python  Project>
```

Discussion and Conclusion

Discussion

The AI-powered food recognition system presents an innovative integration of computer vision and culinary technology, leveraging an ensemble of deep learning models (Indian, Western, and BEiT classifiers) to achieve robust classification. The voting mechanism enhances prediction reliability, while the priority-based retrieval system—incorporating Spoonacular, TheMealDB, and a custom Indian dataset—ensures comprehensive recipe coverage through fuzzy matching. A key contribution is the system's specialized focus on Indian cuisine, addressing a notable gap in existing solutions while maintaining strong performance for Western dishes.

Strengths include high accuracy through model aggregation, adaptive recipe sourcing, and structured, user-friendly outputs. The modular architecture facilitates future expansion to additional cuisines and advanced features like nutritional analysis. Current limitations, such as API dependency and restricted cuisine coverage, present opportunities for improvement. This system exemplifies the potential of AI in culinary applications, offering a scalable framework for future enhancements in food technology.

Conclusion

This project successfully demonstrates an AI-driven approach to food recognition and recipe retrieval, combining multi-model classification with intelligent data sourcing. By addressing the underrepresentation of Indian cuisine while maintaining versatility for Western dishes, it provides a valuable tool for culinary exploration. The system's adaptable design paves the way for future advancements in personalized cooking assistance and global cuisine integration, marking a significant step forward in AI-powered culinary technology.

Future Scope

- **Multi-Cuisine Expansion**: Add support for Chinese, Japanese, Mediterranean, and other global cuisines
- **Real-Time Ingredient Detection**: Integrate YOLO/object detection for precise ingredient analysis
- **Nutritional AI**: Auto-calculate calories, macros, and allergens from recipes
- **Personalized Recommendations**: GPT-powered suggestions based on dietary preferences/history
- **Smart Kitchen Integration**: IoT compatibility with smart ovens/scales for guided cooking
- **AR Cooking Assistant**: Step-by-step 3D recipe visualization via smartphone
- **User-Generated Recipes**: Crowdsourced database with community ratings
- **Voice-Enabled Cooking**: Hands-free voice control for recipe navigation
- **Offline Mode**: Local caching of popular recipes to reduce API dependency
- **Sustainability Features**: Carbon footprint tracking and eco-friendly substitutes

Combines technical enhancements with user experience improvements.

📚 References

1. HuggingFace Model Hub
2. TheMealDB API Documentation
3. Spoonacular API
4. IndianFoodDatasetCSV
5. BEiT: Vision Transformer by Microsoft

Dependencies/Modules Used

```txt
torch
transformers
Pillow
requests
pandas
diffusers
tqdm
scikit-learn
sentencepiece
protobuf
```

Thanks to the contributors of these modules and Sources of References !!!