StreetFoodBD: A comprehensive image dataset to classify Bangladeshi street foods

MD Shamsur Rahman Sami[a], Tasnia Shahrin[a], Jannatin Tajri[a], Joty Saha[a], Mohammad Asif[a]

[a]Department of Computer Science and Engineering, University of Dhaka, Nilkhet Rd, Dhaka 1000, Bangladesh

Corresponding author. E-mail address: shamsurrahman07052001@gmail.com

Article info

Dataset link: StreetFoodBD: Bangladeshi Street Food Dataset (Original data)

Keywords:

image classification

deep learning

CNN

VGG19

Random Forest

Bangladeshi street food

computer vision

dataset

Abstract

Bangladesh is home to a diverse and vibrant street food culture that is loved locally but often unfamiliar to international visitors and rural populations. To help bridge this gap, we introduce StreetFoodBD, a curated image dataset of Bangladeshi street foods for the purpose of food name classification using deep learning. We collected 688 photos ourselves and also scraped many more images from the internet. The project was carried out in three phases, and we trained three different models: a 3-layer CNN, a CNN combined with VGG19, and a Random Forest classifier, to classify these foods based on their images. This dataset is intended to help foreigners and villagers unfamiliar with local food names by providing instant recognition from photos.

Specifications Table

Subject	Machine Learning, Deep Learning
Specific subject area	Food image classification of Bangladeshi street foods using deep learning techniques
Type of data	Images (JPG)
Data collection	Collected via mobile photography and web scraping; manually labeled
Data source location	Dhaka, Bangladesh
Data accessibility	Publicly available on GitHub: StreetFoodBD Dataset
Related research article	None

Value of the Data

- This dataset offers one of the first structured collections of Bangladeshi street food images, a category significantly underrepresented in existing public datasets like Food-101 [2] or UECFOOD100 [3]. It addresses a regional data gap by focusing on traditional and culturally specific items not commonly found in global datasets.

- Researchers working on food recognition, low-resource dataset generalization, or transfer learning can reuse these data to benchmark models or develop culturally aware applications, especially in underrepresented regions. It also provides a valuable use case for domain adaptation, where pre-trained models (e.g., on Food-101) can be fine-tuned using StreetFoodBD.

- The dataset can be useful for developing assistive tools for tourists and locals unfamiliar with Bangladeshi street food. Similar efforts such as the FoodAI project in Singapore [4] have shown success in helping users identify meals using image recognition, suggesting a clear reuse potential in real-world mobile applications.

- Given that a portion of the images (688) were manually collected, it adds value by providing authentic and diverse visual conditions, useful for testing model robustness under real-world lighting, background, and framing variations—issues often absent in curated datasets.

- The dataset can support work in data augmentation, class balancing techniques, and model evaluation in scenarios with both imbalanced and balanced data. Since the three phases reflect varied dataset conditions (imbalanced, expanded, and balanced), they serve as ready-made setups for experimental design in ML research.

Background

Street food is an integral part of Bangladeshi culture, offering a wide variety of unique, flavorful dishes that are visually diverse and regionally distinct. Despite its cultural and culinary significance, there has been limited effort to digitally catalog these foods for use in machine learning applications. Most existing food image datasets, such as Food-101 [2] and UECFood256 [5], focus on international cuisines and overlook region-specific foods from South Asia. This lack of representation poses challenges for developing accurate food recognition systems tailored to the Bangladeshi context.

To address this gap, we developed **StreetFoodBD**, a curated dataset specifically aimed at classifying Bangladeshi street foods using deep learning techniques. Our motivation stems from the need to support tourists, mobile applications, and digital health tools that rely on automated food recognition. The dataset also enables researchers to explore food classification in underrepresented cultural domains, an area where diversity in training data has been shown to improve model robustness and fairness [6, 7]. StreetFoodBD not only contributes to cultural preservation but also serves as a valuable benchmark for developing and evaluating models trained on non-Western food data.

Data Description

The **StreetFoodBD** dataset was created to support the classification of Bangladeshi street food items through image-based machine learning. It includes images of 10 distinct food classes: *Fuchka*, *Jhalmuri*, *Kalavuna*, *Khichuri*, *Mishti*, *Pitha*, *Puri*, *Roshmalai*, *Shingara*, and *Sugarcane Juice*.

Data collection occurred in three phases:

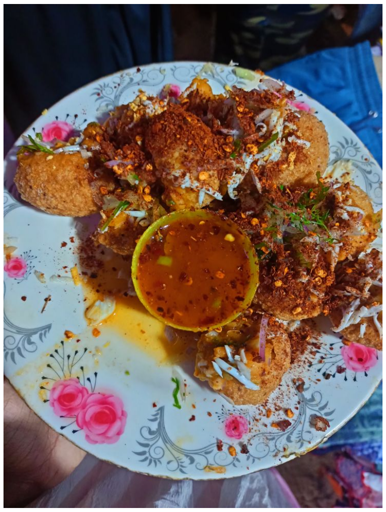
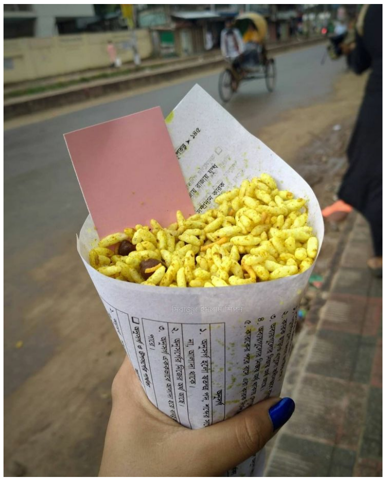
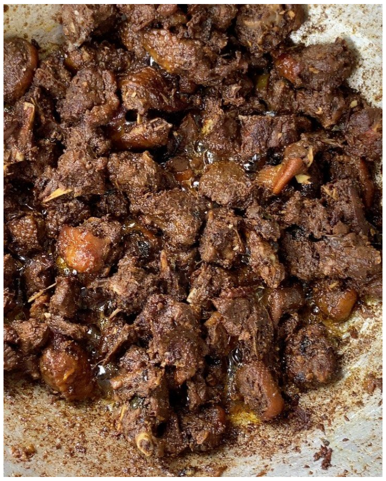
- **Phase 1**: 1,107 images across 13 food classes, with imbalanced class distributions.
- **Phase 2**: Expanded to 1,695 images across the same 13 classes, still imbalanced.
- **Phase 3**: Focused on 10 selected classes, deliberately balanced to include 1,475 images total.

Of the total images, **688 were manually captured**—photographed directly at street vendors using mobile phones in Dhaka, Bangladesh. The rest were **scraped from public online sources** such as blogs, recipe sites, and social media. All images are in **JPG format**, organized into subfolders by food class under the main `StreetFoodBD` directory.

Images were resized and preprocessed to a fixed input size of **224 × 224 pixels** for compatibility with the CNN and VGG19 models used in training. The original images are also preserved for reproducibility.

Each class folder clearly labels the food item and provides a diversity of real-world visual conditions—lighting, angle, background—that is essential for training models that generalize well. Studies show that incorporating regional variability in food datasets significantly improves the robustness and accuracy of recognition systems in real-world applications [8].

Sample images and characteristics of 10 street food varieties used in this dataset.

Food Name	**Description**	**Image**
Fuchka	Hollow crispy balls filled with spicy water, mashed potato, and chickpeas.	
Jhalmuri	Spiced puffed rice mixed with peanuts, onions, chilies, and mustard oil.	
Kalavuna	Traditional dry-fried beef curry cooked with local spices.	

Food Name	Description	Image
Pitha	Sweet or savory rice cake, often filled with jaggery or coconut.	
Khichuri	Spiced rice and lentil dish often served with egg or meat.	

Food Name	Description	Image
Mishti	Soft Bengali sweets made from chhena and soaked in sugar syrup. Includes kalojam, sandesh, and chomchom.	
Puri	Deep-fried flatbread served with curry or sweets.	
Roshmalai	Soft cheese balls soaked in sweetened, cardamom-flavored milk.	

Food Name	Description	Image
Shingara	Crispy triangle pastry filled with spiced potato or meat.	
Sugarcane Juice	Freshly pressed juice from raw sugarcane stalks.	

StreetFoodBD dataset information in brief.

Parameter	**Description**
Dataset Name	StreetFoodBD
File Format	JPG
Image Size	Resized to 224 × 224 pixels
Number of Classes	10 (final balanced dataset)
Total Images	1,475 (final balanced dataset)
Images per Class	±152 (Evenly distributed in Phase 3)
Collection Methods	Manual photography (688 images); web scraping
Image Preprocessing	Resized and normalized for model input; originals preserved
Directory Structure	`StreetFoodBD/ → [Class Name]`
Applicable Domains	Image classification, computer vision, food recognition

This dataset provides a valuable resource for training and evaluating machine learning models on Bangladeshi street foods, a domain with limited existing public datasets. Its diversity and real-world image quality make it suitable for research in food image classification, computer vision, and cultural food recognition.

Experimental Design, Materials and Methods

1. Dataset Collection and Preparation

The dataset used in this study was collected in three distinct phases, combining images obtained via mobile photography and web scraping. A flowchart of the overall pipeline for data collection and preparation is provided below.

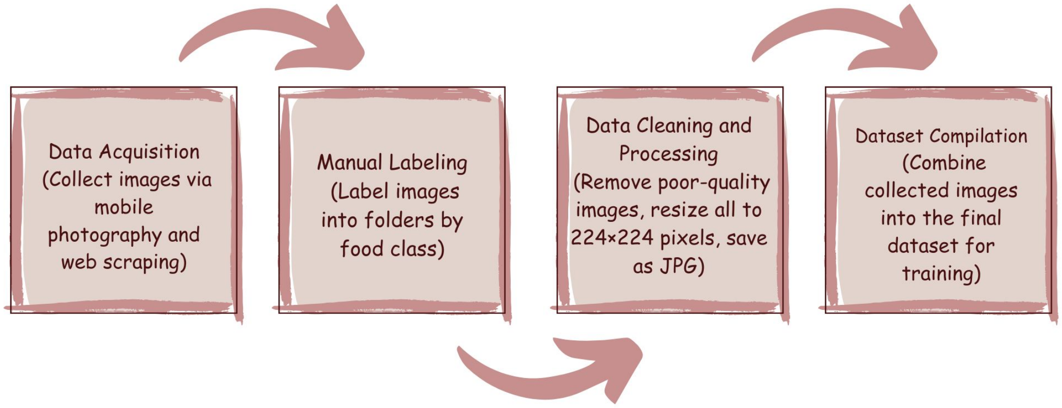

Data Acquisition (Collect images via mobile photography and web scraping)
Manual Labeling (Label images into folders by food class)
Data Cleaning and Processing (Remove poor-quality images, resize all to 224×224 pixels, save as JPG)
Dataset Compilation (Combine collected images into the final dataset for training)

Figure 1: *Steps of data collection and preparation*

Each food class was stored in separate folders named after the food item. The final dataset used in the third phase consisted of 1475 images across 10 balanced classes (Fuchka, Jhalmuri, Kalavuna, Khichuri, Mishti, Pitha, Puri, Roshmalai, Shingara, Sugarcane Juice). All images were resized to 224×224 pixels and stored in JPG format.

2. Capturing Images of Various Street Foods

A large portion of the dataset (688 images) was captured using mobile phone cameras in varying lighting conditions and angles. This ensured diversity and robustness in training. Real-world conditions such as different backgrounds, occlusions, and food presentation styles were preserved to reflect practical deployment scenarios. Similar to findings by Khan et al., real-world variability enhances model robustness [8].

3. Types of Models Used

Three main models were used to analyze the classification task across phases:

- A baseline CNN with three convolutional layers. - A transfer learning model combining CNN with pretrained VGG19 layers. - A Random Forest classifier applied on feature-extracted representations.

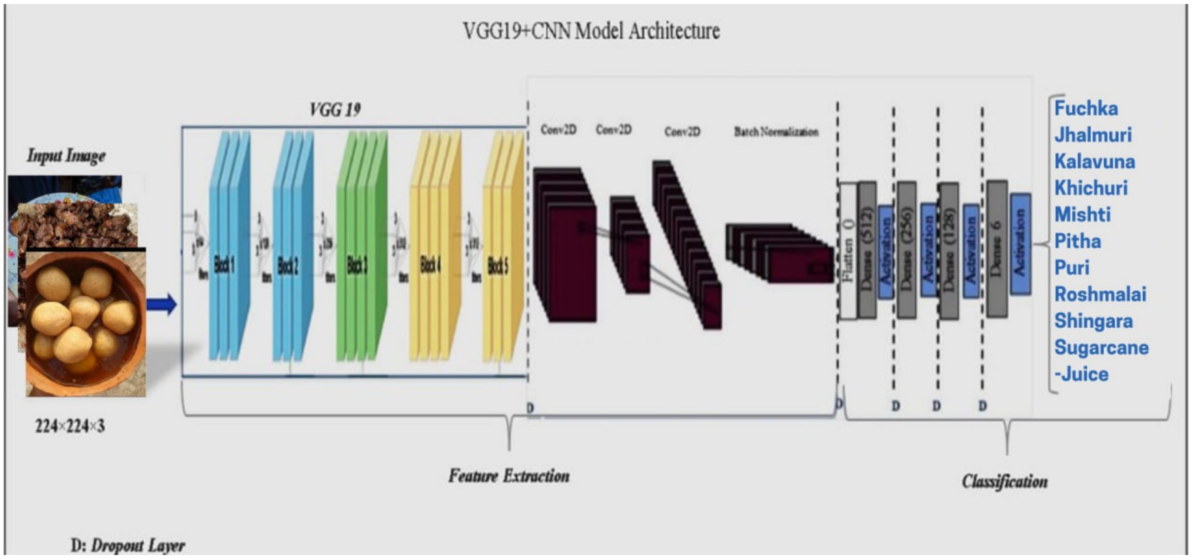


Figure 2: *CNN + VGG19 Model Architecture used in the classification task.*

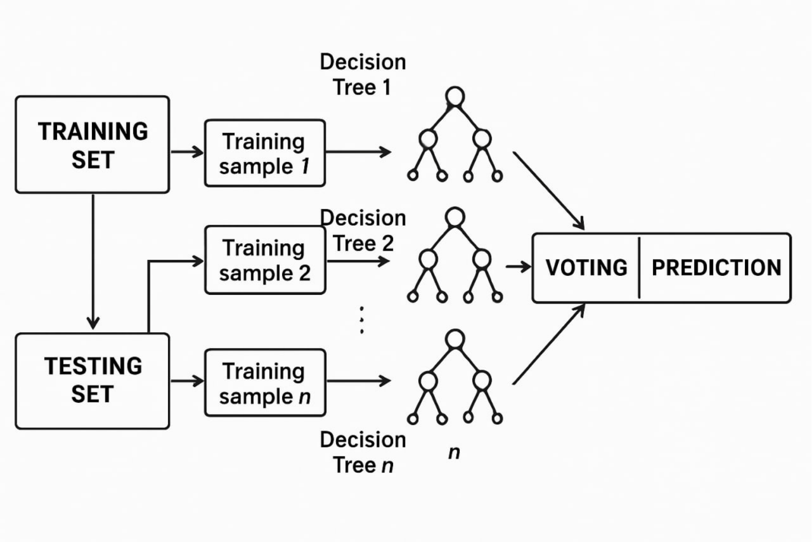
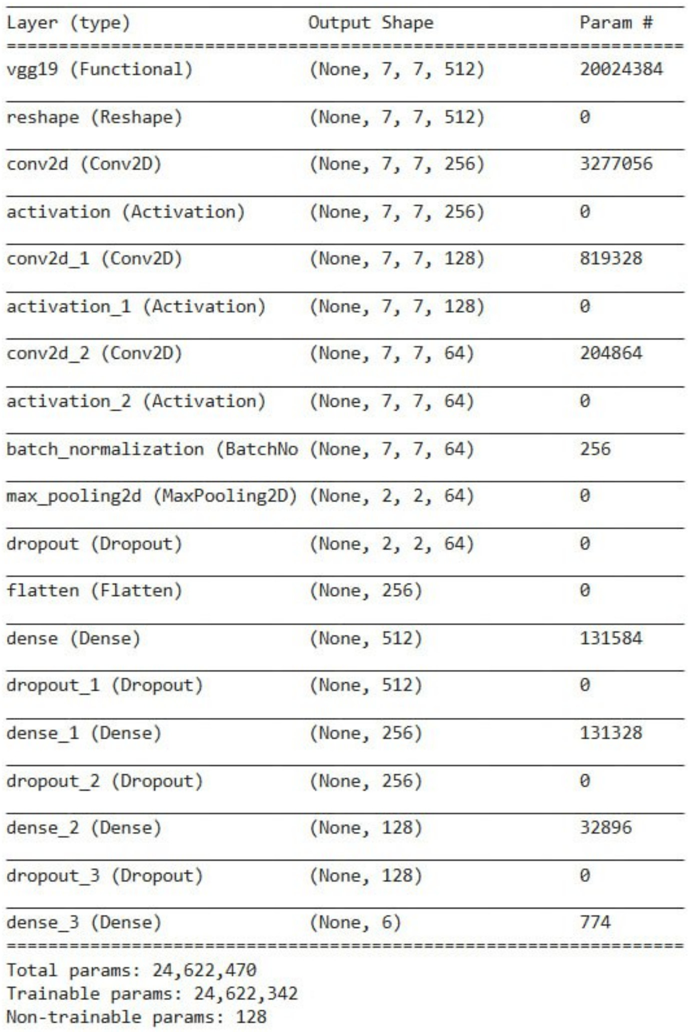


Figure 3: *Random Forest classification pipeline using image feature extraction.*

Number of Parameters



```
Layer (type)                 Output Shape              Param #
=================================================================
vgg19 (Functional)           (None, 7, 7, 512)         20024384
_________________________________________________________________
reshape (Reshape)            (None, 7, 7, 512)         0
_________________________________________________________________
conv2d (Conv2D)              (None, 7, 7, 256)         3277056
_________________________________________________________________
activation (Activation)      (None, 7, 7, 256)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 7, 7, 128)         819328
_________________________________________________________________
activation_1 (Activation)    (None, 7, 7, 128)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 7, 7, 64)          204864
_________________________________________________________________
activation_2 (Activation)    (None, 7, 7, 64)          0
_________________________________________________________________
batch_normalization (BatchNo (None, 7, 7, 64)          256
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 2, 2, 64)          0
_________________________________________________________________
dropout (Dropout)            (None, 2, 2, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 512)               131584
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               131328
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 774
=================================================================
Total params: 24,622,470
Trainable params: 24,622,342
Non-trainable params: 128
```

Figure 4: *Number of parameters used in CNN+VGG19 model.*

4. Phase 1: Initial Model Evaluation

Phase 1 used 1107 images across 10 imbalanced classes.
-CNN Model:

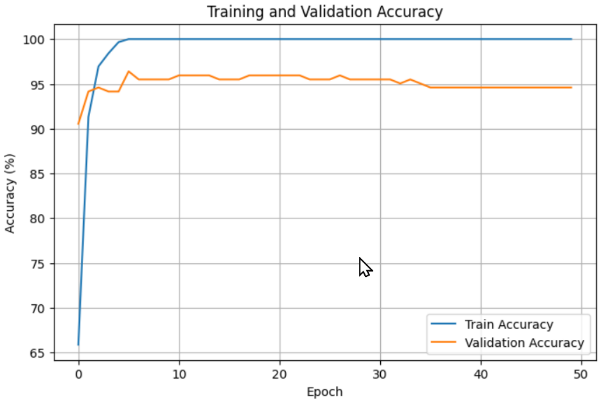


Figure 5: *Training and validation accuracy over epochs (Phase 1 CNN)*

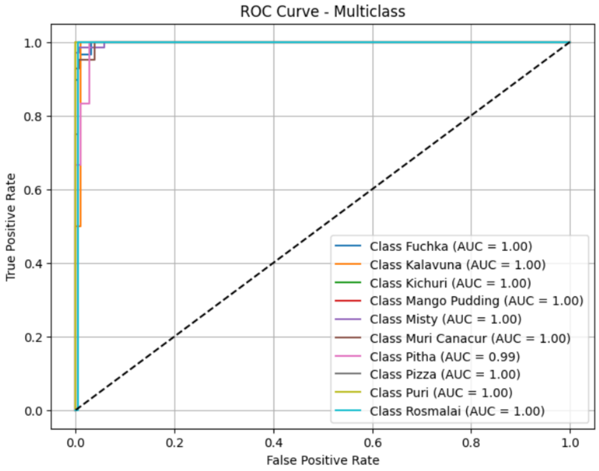


Figure 6: *ROC Curve (Phase 1 CNN, multiclass)*

- Accuracy: 94.59%
- Precision: 94.06%
- Recall: 94.59%
- ROC AUC: 0.9989

Dataset:

- Total images: 1107
- Classes: 10 (Fuchka, Kalavuna, Kichuri, Mango Pudding, Misty, Muri Canacur, Pitha, Pizza, Puri, Rosmalai)

Data Distribution:

Class	Train Images	Val Images	Total Images	% Train	% Val
Fuchka	100	29	129	11.30%	13.06%
Kalavuna	19	2	21	2.15%	0.90%
Kichuri	140	31	171	15.81%	13.96%
Mango Pudding	12	2	14	1.35%	0.90%
Misty	293	68	361	33.11%	30.63%
Muri Canacur	150	42	192	16.95%	18.92%
Pitha	19	6	25	2.15%	2.70%
Pizza	31	4	35	3.50%	1.80%
Puri	107	37	144	12.10%	16.67%
Rosmalai	14	1	15	1.58%	0.45%

Table 2: Data distribution across training and validation splits for Phase 1 CNN model.

Training Details:

- Device: CPU
- Training images: 885
- Validation images: 222
- Number of epochs: 50
- Data loading time: 28.48 seconds
- Total training time: 9180.68 seconds (approximately 2 hours 33 minutes)

Training and Validation Accuracy (Selected Epochs):

Epoch	Train Loss	Train Accuracy	Validation Accuracy	Epoch Time (sec)
1	1.1686	65.88%	90.54%	178.83
2	0.3141	91.30%	94.14%	176.27
3	0.1008	96.95%	94.59%	176.18
6	0.0014	100.00%	96.40%	177.47
10	0.0001	100.00%	95.50%	179.20
20	0.0000	100.00%	95.95%	179.86
30	0.0000	100.00%	95.50%	188.36
50	0.0000	100.00%	94.59%	189.22

Table 3: Training loss, accuracy, validation accuracy, and epoch time for selected epochs.

Notes:

- Training loss rapidly decreased to near zero by epoch 6.
- Training accuracy reached 100% from epoch 6 onwards.
- Validation accuracy peaked around epochs 6–20 (95–96%) then stabilized around 94.5–95.5%.
- Each epoch duration ranged approximately between 178 and 189 seconds.

-VGG19+CNN Model:

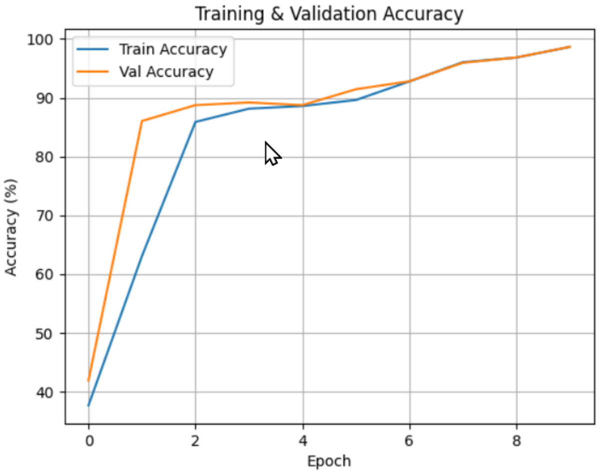


Figure 7: *Training and validation accuracy over epochs (Phase 1 VGG19+CNN)*

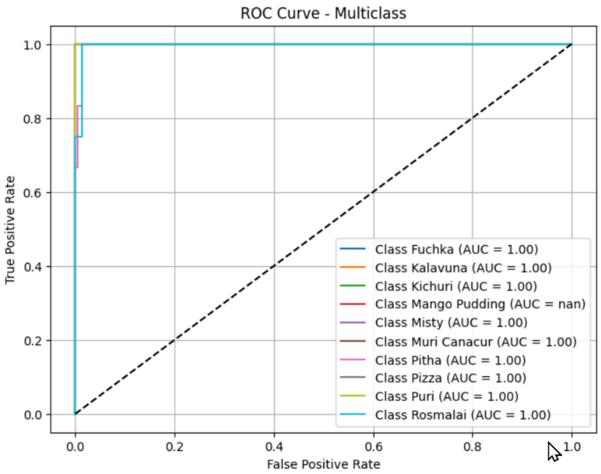


Figure 8: *ROC Curve (Phase 1 VGG19+CNN, multiclass)*

- Accuracy: 98.65%
- Precision: 98.72%
- Recall: 98.65%
- ROC AUC: 0.9999

Model Configuration:

- Model: VGG19 pretrained on ImageNet with custom CNN classifier
- Total Epochs: 10
- Device used: CPU

Dataset Overview:

- Total images: 1107
- Number of classes: 10
- Training set: 885 images
- Validation set: 222 images

Class-wise Data Distribution:

Class	Train Images	Val Images	Total Images	% Train	% Val
Fuchka	111	18	129	12.54%	8.11%
Kalavuna	16	5	21	1.81%	2.25%
Kichuri	134	37	171	15.14%	16.67%
Mango Pudding	14	0	14	1.58%	0.00%
Misty	287	74	361	32.43%	33.33%
Muri Canacur	149	43	192	16.84%	19.37%
Pitha	19	6	25	2.15%	2.70%
Pizza	27	8	35	3.05%	3.60%
Puri	117	27	144	13.22%	12.16%
Rosmalai	11	4	15	1.24%	1.80%
Total	**885**	**222**	**1107**	**100%**	**100%**

Table 4: Class-wise image distribution in Phase 1 dataset (CNN + VGG19)

Epoch-wise Training Results:

Table 5: Training Summary: Phase 1 – CNN + VGG19 Model

Epoch	Train Accuracy	Validation Accuracy	Epoch Time (sec)
5	88.59%	88.74%	763.84
6	89.60%	91.44%	767.00
7	92.77%	92.79%	764.12
8	96.05%	95.95%	767.20
9	96.84%	96.85%	788.48
10	98.64%	98.65%	763.84

Total Training Time: Approximately 130 minutes (2 hours 10 minutes)

5. Phase 2: Dataset Expansion

This phase increased the dataset to 1695 images across the same 13 classes, still imbalanced.

-CNN Model:

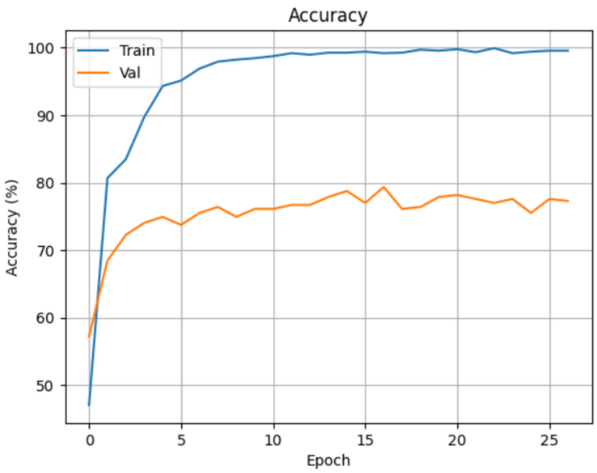


Figure 9: *Training and validation accuracy over epochs (Phase 2 CNN)*

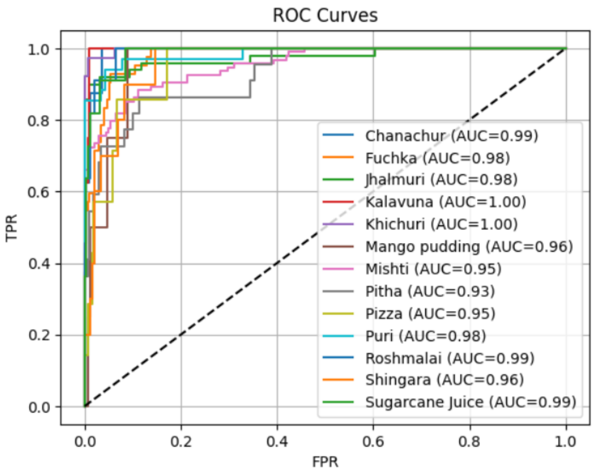


Figure 10: *ROC Curve (Phase 2 CNN, multiclass)*

- Accuracy: 79.35%
- Precision: 80.81%

- Recall: 79.35%
- ROC AUC: 0.9710

Dataset:

- **Total images:** 1695
- **Classes:** 13 (Chanachur, Fuchka, Jhalmuri, Kalavuna, Khichuri, Mango pudding, Mishti, Pitha, Pizza, Puri, Roshmalai, Shingara, Sugarcane Juice)

Data Distribution:

- **Training images:** 1357
- **Validation images:** 338

Training Details:

- **Device:** CPU
- **Number of epochs:** 27 (early stopping)
- **Average epoch time:** approximately 5 minutes 30 seconds

Class wise data distribution:

Class	**Train Images**	**Val Images**	**Total Images**	**% Train**	**% Val**
Chanachur	37	8	45	2.73%	2.37%
Fuchka	165	42	207	12.16%	12.43%
Jhalmuri	172	49	221	12.68%	14.50%
Kalavuna	27	8	35	1.99%	2.37%
Khichuri	164	39	203	12.09%	11.54%
Mango pudding	14	4	18	0.82%	1.18%
Mishti	357	94	451	26.31%	27.81%
Pitha	70	22	92	5.16%	6.51%
Pizza	52	7	59	3.83%	2.07%
Puri	153	34	187	11.28%	10.06%
Roshmalai	48	11	59	3.54%	3.25%
Shingara	60	10	70	4.42%	2.96%
Sugarcane Juice	37	11	48	2.73%	3.25%

Table 6: **Data distribution across training and validation splits for Phase 2 CNN model.**

Epoch wise training results:

Epoch	Train Loss	Train Accuracy	Validation Accuracy	Epoch Time (sec)
1	1.8197	47.05%	57.23%	270
5	0.3154	94.32%	74.93%	334
10	0.1153	98.45%	76.11%	334
15	0.0510	99.26%	78.76%	342
20	0.0350	99.56%	77.88%	334
25	0.0329	99.41%	75.52%	343
27	0.0257	99.56%	77.29%	331

Table 7: **Training loss, accuracy, validation accuracy, and epoch time for selected epochs of Phase 2 CNN model.**

Notes:

- Training loss steadily decreased from 1.82 to near 0.03 by epoch 27.
- Training accuracy rapidly improved to over 99% after epoch 10 and plateaued.
- Validation accuracy peaked at 78.76% at epoch 15, then fluctuated around 75–78%.
- Early stopping triggered at epoch 27 due to no further validation improvement.
- Average epoch duration was about 5 minutes 30 seconds.

-VGG19+CNN Model:

Due to hardware limitations, the model did not complete full training. The highest recorded accuracy was 75.81% at epoch 28.

Dataset:

- Total images: 1695
- Classes: 13 (Chanachur, Fuchka, Jhalmuri, Kalavuna, Khichuri, Mango Pudding, Mishti, Pitha, Pizza, Puri, Roshmalai, Shingara, Sugarcane Juice)

Data Distribution:

- Training images: 1338
- Validation images: 357

Class-wise data distribution:

Class	Train Images	Val Images	% Train	% Val
Chanachur	36	9	2.69%	2.52%
Fuchka	160	47	11.96%	13.17%
Jhalmuri	181	40	13.52%	11.20%
Kalavuna	29	6	2.17%	1.68%
Khichuri	169	34	12.62%	9.52%
Mango Pudding	16	2	1.20%	0.56%
Mishti	350	101	26.15%	28.29%
Pitha	82	10	6.12%	2.80%
Pizza	48	11	3.59%	3.08%
Puri	152	35	11.36%	9.80%
Roshmalai	44	15	3.29%	4.20%
Shingara	50	20	3.74%	5.60%
Sugarcane Juice	39	9	2.91%	2.52%

Training Details:

- Device: CPU
- Number of epochs: 30
- Average epoch time: approximately 20 minutes

Epoch wise training results:

Epoch	Train Loss	Train Accuracy	Validation Accuracy
1	2.5669	4.42%	3.54%
5	2.1172	32.89%	40.41%
10	1.5122	62.54%	69.62%
15	1.1574	69.47%	71.68%
20	0.9754	69.99%	72.86%
25	0.8221	74.85%	75.22%
28	0.7677	76.33%	75.81%

Notes:

- Training loss decreased steadily from 2.57 at epoch 1 to 0.77 at epoch 28.

- Training accuracy improved significantly from 4.42% to 76.33%.
- Validation accuracy showed substantial improvement, reaching about 75.81% by epoch 28.
- Each epoch lasted approximately 20 minutes.
- Training was ongoing at epoch 29 when this output was recorded.

6. Phase 3: Balanced Dataset Evaluation

In this phase, the dataset was reduced to 1475 images across 10 balanced food classes. Due to resource constraints, neither model completed full training, but we report their best observed accuracies.

-CNN Model:

- Accuracy (at best epoch): 64.07%

Dataset:

- Total images: 1475
- Classes: 10 (Fuchka, Jhalmuri, Kalavuna, Khichuri, Mishti, Pitha, Puri, Roshmalai, Shingara, Sugarcane Juice)

Data Distribution:

- Training images: 1180
- Validation images: 295

Class-wise data distribution:

Class	Train Images	Val Images	% Train	% Val
Fuchka	117	35	9.92%	11.86%
Jhalmuri	120	32	10.17%	10.85%
Kalavuna	123	29	10.42%	9.83%
Khichuri	126	26	10.68%	8.81%
Mishti	119	33	10.08%	11.19%
Pitha	127	25	10.76%	8.47%
Puri	121	31	10.25%	10.51%
Roshmalai	78	29	6.61%	9.83%
Shingara	124	28	10.51%	9.49%
Sugarcane Juice	125	27	10.59%	9.15%

Training Details:

- Device: CPU
- Number of epochs: 150
- Average epoch time: approximately 230 seconds

Epoch wise training results:

Epoch	Train Loss	Train Accuracy	Validation Accuracy
1	2.2350	29.32%	37.29%
5	1.2668	63.39%	55.59%
10	0.8632	73.05%	63.73%
50	–	–	–
100	–	–	–
121	0.0093	99.83%	63.73%
126	0.0126	99.75%	64.07%
130	0.0092	99.83%	63.05%
140	0.0190	99.58%	64.07%

Notes:

- Training loss decreased steadily from 2.24 at epoch 1 to near 0.01 by epoch 140.
- Training accuracy increased from 29.32% to above 99.5%.
- Validation accuracy improved from 37.29% to a maximum around 64.07%.
- The model shows signs of overfitting given the near-perfect training accuracy but validation accuracy plateauing.
- Each epoch took roughly 3.8 minutes on CPU.

-VGG19+CNN Model:

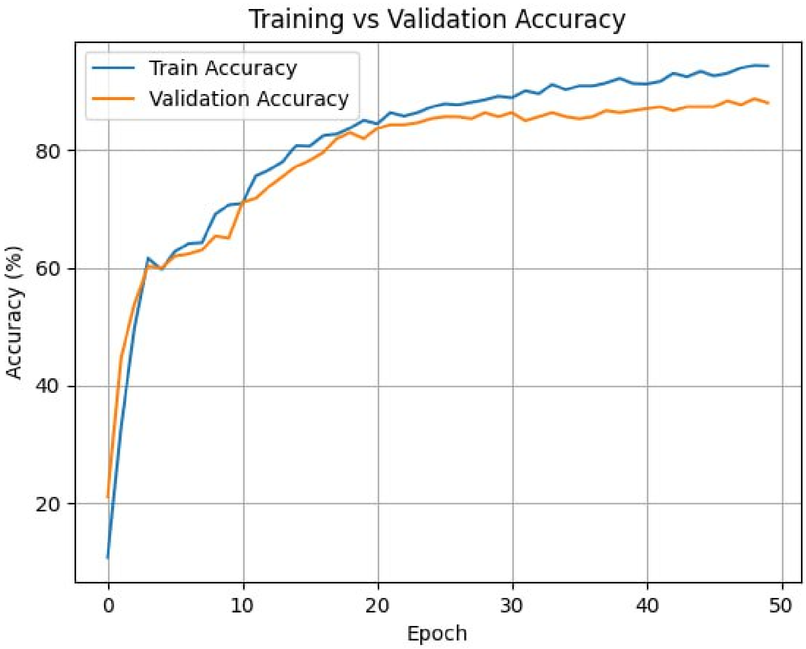


Figure 11: *Training and validation accuracy over epochs (Phase 3 VGG19+CNN)*

- Accuracy (at best epoch): 89.49%

Dataset:

- Total images: 1475
- Classes: 10 (Fuchka, Jhalmuri, Kalavuna, Khichuri, Mishti, Pitha, Puri, Roshmalai, Shingara, Sugarcane Juice)

Data Distribution:

- Training images: 1180
- Validation images: 295

Class-wise data distribution:

Class	Train Images	Val Images	% Train	% Val
Fuchka	121	31	10.25%	10.51%
Jhalmuri	122	30	10.34%	10.17%
Kalavuna	131	21	11.10%	7.12%
Khichuri	115	37	9.75%	12.54%
Mishti	125	27	10.59%	9.15%
Pitha	122	30	10.34%	10.17%
Puri	119	33	10.08%	11.19%
Roshmalai	86	21	7.29%	7.12%
Shingara	122	30	10.34%	10.17%
Sugarcane Juice	117	35	9.92%	11.86%

Training Details:

- Model: CNN + VGG
- Device: CPU
- Number of epochs: 50
- Average epoch time: approximately 31 seconds

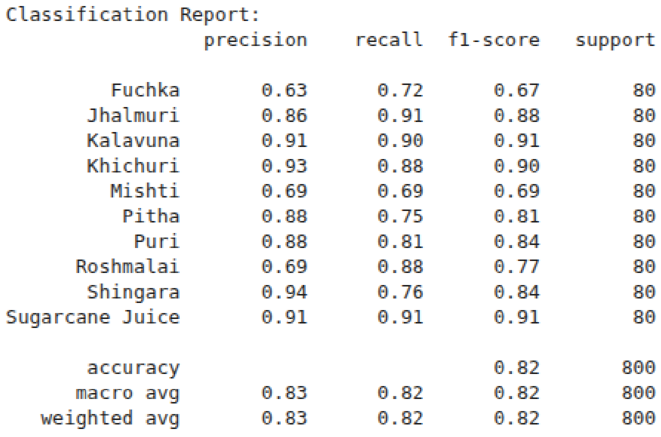
Epoch-wise training results:

Epoch	Train Accuracy	Validation Accuracy
1	13.73%	16.61%
2	30.68%	51.53%
3	56.86%	61.36%
5	56.95%	53.90%
10	68.98%	62.71%
15	81.27%	80.68%
20	85.51%	87.46%
25	88.98%	88.14%
27	90.42%	89.49%
28	90.00%	89.49%
29	90.93%	89.83%
30	90.76%	89.49%

Notes:

- Significant improvement observed from epoch 1 to 30 in both train and validation accuracy.
- Validation accuracy peaked at 89.83% by epoch 29, demonstrating effective learning and generalization.
- Model performance stabilized near convergence after epoch 27.
- No clear signs of overfitting; training and validation curves tracked closely.

As a baseline, a **Random Forest** model was trained on the Phase 3 dataset. Initially, each class contained approximately 152 images. To enhance model performance and balance the dataset, data augmentation techniques were applied, increasing the number of images to **400 per class**.



```
Classification Report:
                 precision    recall  f1-score   support

         Fuchka       0.63      0.72      0.67        80
       Jhalmuri       0.86      0.91      0.88        80
       Kalavuna       0.91      0.90      0.91        80
       Khichuri       0.93      0.88      0.90        80
         Mishti       0.69      0.69      0.69        80
          Pitha       0.88      0.75      0.81        80
           Puri       0.88      0.81      0.84        80
      Roshmalai       0.69      0.88      0.77        80
       Shingara       0.94      0.76      0.84        80
Sugarcane Juice       0.91      0.91      0.91        80

       accuracy                           0.82       800
      macro avg       0.83      0.82      0.82       800
   weighted avg       0.83      0.82      0.82       800
```

Figure 12: *Classification report for Random Forest.*

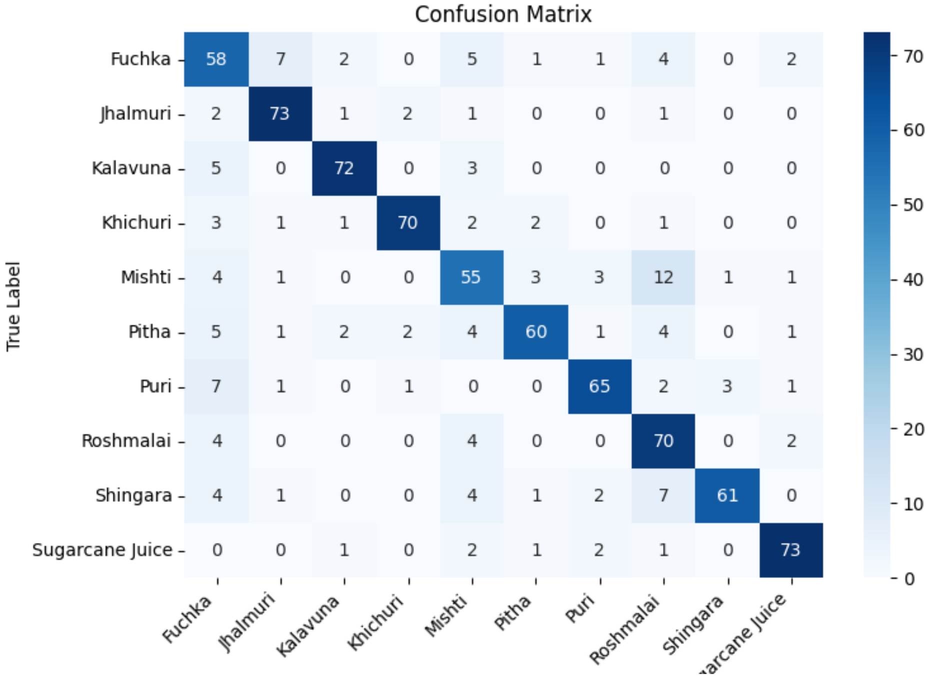


Figure 13: *Confusion matrix of Random Forest model.*

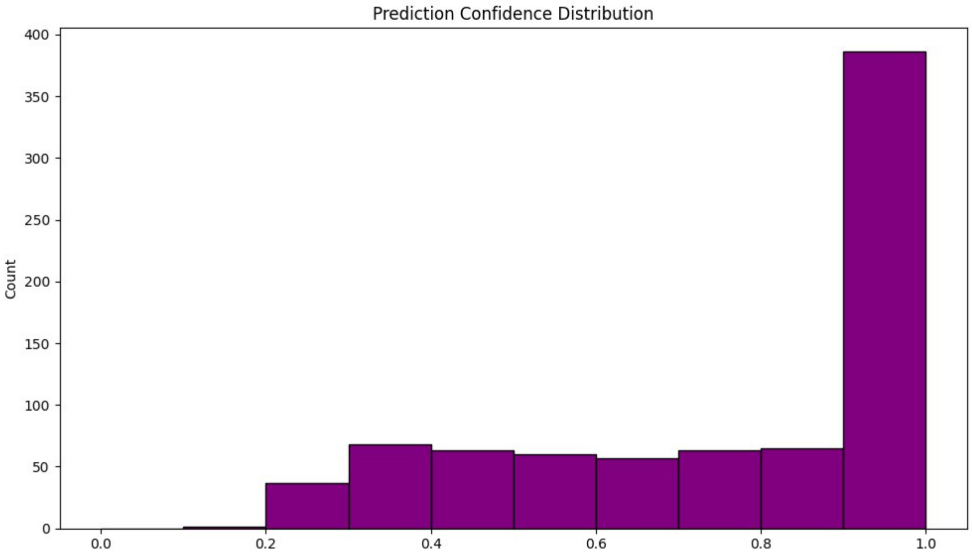


Figure 14: *Prediction confidence distribution.*

- Accuracy: 82.12%

7. Comparative Study of Models

In this study, we examined the performance of three different models—**Convolutional Neural Network (CNN)**, **CNN integrated with VGG19**, and **Random Forest (RF)**—across three experimental phases, each designed to test model robustness under evolving dataset conditions.

Accuracy and Learning Capability

Table 8: Performance summary of models across the three phases

Model	**Epochs**	**Training Time**	**Training Accuracy**	**Validation Accuracy**	**Final Accuracy**
			Phase 1		
CNN	50/50	2:42:00	100%	94.59%	94.59%
CNN+VGG19	10/10	02:00:00	98.64%	98.65%	98.65%
Random Forest	N/A	N/A	N/A	N/A	N/A
			Phase 2		
CNN	27/100	2:30:00	99.56%	77.29%	79.35%
CNN+VGG19	28/30	11:30:00	76.33%	75.81%	75.81%
Random Forest	N/A	N/A	N/A	N/A	N/A
			Phase 3		
CNN	140/150	09:30:00	99.85%	64.07%	64.07%
CNN+VGG19	29/50	12:30:00	90.93%	89.83%	89.49%
Random Forest	N/A	3.41s	98.23%	78.6%	82.12%

CNN showed strong performance on small, imbalanced datasets in Phase 1. However, its performance degraded in later phases, particularly in Phase 3, where the training was incomplete. CNNs are known to be lightweight but sensitive to data quality and quantity.

CNN+VGG19 consistently performed best, especially in Phase 1. Its ability to transfer learned features from large-scale datasets like ImageNet helped it generalize well to limited or unbalanced data. Despite incomplete training in Phases 2 and 3, its performance remained superior [9].

Random Forest, tested only on the balanced dataset in Phase 3, achieved an accuracy of 82.12%. While it does not capture spatial features like CNNs, it excels in interpretability, quick training, and robustness to small variations when features are discriminative enough [10].

Computational Requirements

CNN is efficient and easy to deploy in resource-constrained environments. CNN+VGG19 demands more computational power and memory but yields higher accuracy, especially with small datasets through transfer learning. Random Forest offers fast training and high explainability, making it suitable for deployment on low-power devices and for rapid iteration [11].

Table 9: Comparison of model computational requirements

Model	GPU Required	Memory Usage	Speed	Interpretability
CNN	Low	Low	Fast	Medium
CNN+VGG19	High	High	Slower	Low
Random Forest	No	Very Low	Very Fast	High

Suitability and Use Cases

Table 10: Best use-case scenario for each model

Model	Best Suited For
CNN	Real-time classification with limited computational resources
CNN+VGG19	High-accuracy tasks with access to powerful hardware
Random Forest	Interpretable, fast solutions with balanced and well-prepared data

In summary:

- CNN+VGG19 was the top-performing model overall and remains ideal for applications requiring high precision, assuming sufficient resources.
- Random Forest proved competitive on the final dataset, particularly valuable for its speed and interpretability.
- CNN served as a strong baseline but showed limitations when scaling or balancing data became crucial.

Challenges

The collection and preparation of our Bangladeshi street food dataset presented several real-world and technical challenges that impacted both data quality and model performance.

1. Financial and Logistical Barriers. Unlike synthetic or web-scraped datasets, our dataset required manual image capture by purchasing food from street vendors. To ensure data variety and authenticity, we captured multiple samples of each food item from different vendors in locations across Dhaka. This involved *significant financial costs*, travel time, and logistical coordination.

2. Requirement for Authentic Street Context. To maintain the dataset's integrity, we only captured images in *authentic street environments*—not indoor or home settings. Real-world elements such as food carts, plastic containers, or pavement backgrounds helped improve generalizability. However, this also introduced *cluttered and inconsistent* *backgrounds*, making feature extraction more challenging. Prior research indicates that context variability in street scenes can reduce model performance unless adequately handled with augmentation [12].

3. Lighting and Environmental Constraints. We captured images exclusively during *daylight hours* to avoid noise from artificial lighting. Despite this, shadows, cloud cover, and sunlight direction introduced *inconsistent illumination*, affecting model learning. Studies such as [13] highlight the impact of lighting on food classification performance.

4. Non-uniform Food Presentation. Presentation styles varied across vendors—even for the same dish—due to differences in garnish, packaging, and serving size. This high *intra-class variation* challenges deep models to generalize. Similar issues are noted in prior food recognition research [14].

5. Class Imbalance. Some food items like *Kalavuna* or *Roshmalai* had fewer images due to scarcity, leading to a class imbalance problem. This imbalance was especially significant in Phase 1 and Phase 2 and required mitigation strategies during model training.

Limitations

Despite careful methodology and phased experimentation, several limitations remain:

- **Small Dataset Size:** Our final dataset contained only 1475 images across 10 classes, which is relatively small for training deep models without overfitting.

- **Hardware Constraints:** Due to limited computational resources, training for CNN+VGG19 and even CNN was not fully completed in Phase 3, affecting model convergence and final metrics.

- **Geographic and Cultural Bias:** All images were collected from Dhaka city. As such, food presentation styles may not represent other regions of Bangladesh, introducing a geographic bias.

- **Manual Labeling:** Image labels were manually assigned. Although done carefully, there is a possibility of human error or subjectivity, especially in visually similar dishes.

- **Lack of Multimodal Data:** This study focused on image data only. Incorporating text descriptions, vendor metadata, or ingredients could have further improved model accuracy through multimodal learning approaches.

Ethics Statement

Images were taken or collected from public domains. No human faces or personal data were used. All photos were taken with vendor consent.

References

[1] G. M. M. Alshmrani, Q. Ni, R. Jiang, H. Pervaiz, and N. M. Elshennawy, “A deep learning architecture for multi-class lung diseases classification using chest X-ray (CXR) images,” *Alexandria Engineering Journal* https://www.sciencedirect.com/science/article/pii/S1110016822007104

[2] Bossard, L., Guillaumin, M., & Van Gool, L. (2014). Food-101 – Mining Discriminative Components with Random Forests. *European Conference on Computer Vision (ECCV)*, 2014. https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/

[3] Kawano, Y., & Yanai, K. (2014). Automatic expansion of a food image dataset leveraging existing categories with domain adaptation. *ECCV Workshops*, 2014. https://www.kaggle.com/datasets/rkuo2000/uecfood100

[4] FoodAI. (2020). Food Recognition with AI. Singapore Data Science Consortium. https://foodai.org/

[5] Kawano, Y., and K. Yanai. ”Automatic expansion of a food image dataset leveraging existing categories with domain adaptation.” In *European Conference on Computer Vision Workshops*, 2014. Available: https://link.springer.com/chapter/10.1007/978-3-319-16199-0_1

[6] Ciocca, G., Napoletano, P., & Schettini, R. (2017). Food recognition: A new dataset, experiments, and results. *IEEE Journal of Biomedical and Health Informatics*. Available: https://pubmed.ncbi.nlm.nih.gov/28114043/

[7] Shankar, S., Halpern, Y., Breck, E., Atwood, J., Wilson, J., & Sculley, D. (2017). No Classification without Representation: Assessing Geodiversity Issues in Open Data. arXiv preprint arXiv:1711.08536. Available: https://arxiv.org/abs/1711.08536

[8] S. Khan, S. Ali, M. Hayat, M. Bennamoun, F. Sohel, and R. Togneri, “A guide to deep learning-based food recognition: Datasets, methods, and applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 8414–8435, Dec. 2022. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC8700885/

[9] Deng, J., et al. “ImageNet: A Large-Scale Hierarchical Image Database.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. https://ieeexplore.ieee.org/document/5206848

[10] Wang, L. “Comparative Analysis of Random Forest and Deep Learning for Image Classification Tasks.” *Pattern Recognition Letters*https://www.sciencedirect.com/science/article/abs/pii/S0167865520302981

[11] Criminisi, A., Shotton, J., and Konukoglu, E. “Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning.” *Foundations and Trends in Computer Graphics and Vision*https://ieeexplore.ieee.org/document/8187032

[12] Hassannejad, H., Matrella, G., Ciampolini, P., De Munari, I., Mordonini, M., & Cagnoni, S. (2016). Food image recognition using very deep convolutional networks. https://dl.acm.org/doi/10.1145/2986035.2986042

[13] Meyers, A., Johnston, N., Rathod, V., Korattikara, A., Gorban, A., Silberman, N., Guadarrama, S., Papandreou, G., Huang, J., & Murphy, K. (2015). Im2Calories: Towards an Automated Mobile Vision Food Diary. https://ieeexplore.ieee.org/document/7410503

[14] Bolanos, M., Radeva, P., & Angulo, C. (2017). Food recognition for dietary assessment using deep learning. *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*. https://ieeexplore.ieee.org/abstract/document/7900117