TITLE – FOOD RECOGNITION SYSTEM

COURSE – DSCI 6011 DEEP LEARNING

PROFESSOR – KHALED SAYED

MAJOR – DATA SCIENCE

SEMESTER – SPRING 2024

SUBMITTED BY – PRANITA NAKKA

HARSHITHA VEJELLA

TABLE OF CONTENTS

ABSTRACT

This article proposes a food recognition system utilizing deep learning techniques. The system aims to automatically classify food items from images, facilitating dietary assessment and nutritional monitoring. The methodology extracts essential features from food images using deep learning techniques to achieve accurate classification. Experimental validation on the Food-101 dataset demonstrates the system's effectiveness in recognizing various food types with high accuracy. This innovative approach holds promise for enhancing dietary assessment, nutritional analysis, and food-related applications through automated food recognition. A CNN model based on InceptionV3 was trained on the data, and it correctly classified the sample image passed to it when only three mini-classes were used.

Keywords— food recognition, deep learning, image classification

I. INTRODUCTION

Food Recognition System: The world we live in is increasingly reliant on technology, and the realm of food is no exception. Food recognition systems are emerging as a powerful tool with diverse applications, impacting individuals, businesses, and the healthcare sector. This report provides an introduction to this exciting technology.

What is a Food Recognition System?

A food recognition system utilizes computer vision and machine learning algorithms to identify and classify food items captured in an image. Imagine pointing your phone camera at your lunch, and the system instantly reveals the dish, its ingredients, and even an estimated calorie count. This is the transformative potential of food recognition technology.

Why is Food Recognition Important?

Food recognition offers a range of benefits:

Individual Users:

Dietary Tracking: Effortlessly track food intake for calorie counting or specific dietary needs.

Ingredient Identification: Quickly identify ingredients in a dish, aiding those with allergies or preferences.

Enhanced Culinary Exploration: Explore new cuisines by capturing and identifying unfamiliar dishes.

Businesses:

Automated Food Labeling: Streamline food labeling processes in restaurants and grocery stores.

Inventory Management: Optimize inventory control and prevent food waste.

Customer Engagement: Develop interactive food-related applications and personalized recommendations.

Healthcare:

Nutritional Guidance: Assist dieticians and nutritionists in providing personalized dietary advice.

Weight Management: Support weight management programs by tracking calorie intake.

Disease Management: Help individuals with diabetes or other food-related conditions monitor their diet.

Food recognition systems rely on two key technologies:

Computer Vision: This field enables computers to "see" and interpret visual data from images and videos.

Machine Learning: Algorithms are trained on vast datasets of labeled food images, allowing them to learn and identify patterns for accurate classification.

Convolutional Neural Networks (CNNs) and Inceptionv3 are prominent architectures used in food recognition due to their ability to extract complex features from images.

Food recognition is a rapidly evolving field with immense potential. As technology progresses, we can expect even higher accuracy, broader applications, and integration with other intelligent systems. This technology has the power to revolutionize how we interact with food, shaping a future of personalized nutrition, informed choices, and a deeper connection with what we eat.

Ensuring proper nutrition is crucial for maintaining human health. Traditional foods, as well as those created to meet market demands, play a significant role in meeting dietary needs by providing essential nutrients and varying levels of processing. Given the diverse dietary preferences worldwide, understanding food characteristics such as type, composition, nutrients, and processing methods is vital for ensuring consistency and safety for consumers (Salim et al., 2021). Modern technologies like electronic noses, computer vision, spectroscopy, and spectral imaging have become indispensable in addressing the need for swift and accurate determination of food attributes. These methods generate vast amounts of digital data, necessitating effective data analysis techniques to extract relevant information amidst the abundance of repetitive and irrelevant data. The challenge lies in managing this massive volume of data and extracting valuable features while navigating the complexities of implementing these techniques in real-world scenarios. Different approaches for data analysis have been devised to tackle this issue, encompassing partial least squares (PLS), artificial neural networks (ANN), support vector machines (SVM), random forests, and k-nearest neighbor (KNN). Moreover, methods such as wavelet transform (WT), independent component correlation algorithm (ICA), scale-invariant feature transformation, robust speedup features, and histogram of directed gradients are utilized for feature extraction, including primary component analysis (PCA) (Mezgec & Seljak, 2019).

According to Mohanty et al. (2022), deep learning is a powerful machine learning algorithm that has garnered significant attention for its ability to automatically learn data representations, handle multi-domain features, adapt to changes, and process vast amounts of data with increased accuracy and efficiency. Specifically, convolutional neural networks (CNN) have emerged as a cornerstone in deep learning, particularly in image analysis, and have been adapted to handle various data formats beyond two-dimensional data. They excel in processing data from quality and safety assessment instruments such as electronic noses, digital cameras, and spectroscopy devices (Mezgec & Seljak, 2019).

AIMS AND OBJECTIVES

The main aim of this project is to create a reliable food recognition system capable of precisely categorizing food images into one of 101 predefined classes using the Food-101 dataset. This system aims to facilitate nutritional tracking, aid culinary education, enhance dietary management apps, and support the food service industry by automating food identification processes.

VALUE STATEMENT

This project is worth doing since it has several benefits. Firstly, it helps to promote health and nutrition by providing a seamless and precise method for tracking food intake, which is crucial for individuals monitoring their diet for health reasons. Secondly, it is an invaluable educational tool for culinary schools and home cooks, facilitating learning about diverse dishes. Lastly, its vast industry applications offer restaurant and food services opportunities for inventory management, menu planning, and enriching customer experience through interactive applications. Thus, this food recognition system using deep learning is a pivotal innovation that profoundly impacts health, education, and industry advancement.

TOOLS AND TECHNOLOGIES USED:

Python: As the primary programming language due to its wide support and libraries for machine learning.

TensorFlow: TensorFlow is an open-source machine learning framework developed and maintained by Google. It is one of the most popular and widely used libraries for building and training deep learning models. lt is a leading library for developing machine learning models.

OpenCV: Useful for handling image processing tasks outside of the deep learning model.

NumPy: For data manipulation.

Matplotlib: For visualizing data distributions

Jupyter Notebook or Google Colab: For interactive development and experimentation, particularly useful in the early stages of model design and testing.

ARCHITECTURES

1. CONVOLUTIONAL NEURAL NETWORKS (CNNs)

Convolutional Neural Networks (CNNs) are a sophisticated subset of deep neural networks pivotal for processing visual data. They are particularly effective in image recognition tasks, which is central to the implementation of advanced food recognition systems.

Core Principles:

Feature Extraction: CNNs excel in identifying key features within images autonomously, a vital attribute for systems tasked with recognizing diverse food items.

Architectural Elements: The architecture is characterized by several layers:

Convolutional Layers: Extract features by convolving filters over the input.

Activation Functions: Introduce non-linear transformations, commonly ReLU (Rectified Linear Unit).

Pooling Layers: Reduce spatial dimensions and computational complexity.

Fully Connected Layers: Classify the image based on the features extracted by convolutional and pooling layers.

Advantages for Food Recognition:

Efficient Feature Learning: Automates the extraction of intricate features in food items, essential for accurately identifying diverse culinary elements.

High Accuracy: Capable of achieving remarkable accuracy levels, crucial for applications such as nutritional tracking and automated service systems in culinary environments.

2. INCEPTIONV3 ARCHITECTURE

InceptionV3, a brainchild of Google, builds upon the success of its predecessors by optimizing both the computational efficiency and classification accuracy, thus suiting complex image recognition tasks like those required in food recognition.

Key Features:

Factorized Convolutions: By breaking down conventional convolutions into smaller components, InceptionV3 manages to enhance the network's performance and reduce the computational overhead.

Asymmetric Convolutions: Employs convolutions of varying sizes (e.g., 1x3 followed by 3x1) to capture spatial hierarchies more effectively.

Auxiliary Classifiers: Facilitates gradient propagation during training by integrating additional classifiers in intermediate network layers, which aids in stabilizing the training of deep networks.

Applicability in Food Recognition:

Detailed Recognition Capabilities: The architecture’s depth and complexity enable it to distinguish between closely similar food items, enhancing the granularity of recognition.

Operational Efficiency: Despite its depth, InceptionV3 maintains a balance between accuracy and computational efficiency, making it suitable for deployment in real-time applications.

3. TRANSFER LEARNING

Transfer learning is a machine learning method where a model developed for a particular task is reused as the starting point for a model on a second task. It's a popular approach in deep learning because it can transfer knowledge from one domain to another, leveraging pre-trained models to achieve considerable time and computational savings. In the context of your project on a food recognition system, here's how transfer learning can be effectively utilized:

Pre-Trained Models: These are models trained on large benchmark datasets like ImageNet, which contains millions of images classified into thousands of categories. These models have already learned rich feature representations for a wide range of images.

Feature Reusability: Many features learned by models on ImageNet are general features (e.g., edges, curves, textures) that are useful across a wide range of image recognition tasks. These features can be a powerful starting point for recognizing food items.

Efficiency: Starting with pre-trained models accelerates the development of new models, especially when the available dataset for the new task is relatively small. It can improve performance significantly compared to training from scratch.

Applying Transfer Learning to Food-101

Food-101 Dataset: It's a dataset specifically designed for food recognition tasks, containing 101,000 images of food categorized into 101 types. When using transfer learning, this dataset can be used to fine-tune a pre-trained model.

Fine-Tuning: This process involves taking a pre-trained model, using its learned features, and then continuing the training process with the Food-101 dataset. You might freeze the earlier layers of the model—so they retain their learned features—and only train some of the final layers which will learn features more specific to food images.

II. RELATED WORKS

Katiya et al. (2024) investigate the effectiveness of sequential convolutional neural networks (CNNs) in precisely recognizing food items depicted in images. They introduce a novel CNN structure termed "sequential_2," which attains an accuracy of 89.84% on the Food Images (Food-101) dataset. The study highlights insights derived from the model's design and performance, underscoring its promise in image classification tasks, particularly food identification. This methodology aims to streamline food attribute determination, taxonomy creation, and nutrient information extraction activities. The research showcases a transformative approach to automating food comprehension through technology by integrating deep learning methods with practical uses. The results suggest the feasibility of employing advanced CNN architectures to bolster food recognition precision, potentially reshaping how we approach and automate food comprehension processes.

Srigurulekha and Ramachandran (2020) propose an innovative method for classifying food images using convolutional neural networks (CNNs) to harness the computational capabilities of modern devices such as smartphones. Unlike traditional artificial neural networks, CNNs directly process the score function from image pixels. The system employs multiple layers, with their outputs concatenated to form the final tensor of outputs. MAX pooling extracts crucial features for model training. Achieving an accuracy of 86.85% on the FOOD-101 dataset, this study underscores the potential of CNNs in food image recognition, offering promising prospects for utilizing technology to assist individuals in planning diverse diets and understanding the health benefits of various foods.

Liu et al. (2016) delve into the pressing issue of accurately tracking dietary caloric intake to combat the global surge in obesity rates. They highlight the limitations of current dietary assessment methods, which often rely on memory and present significant challenges. The authors propose leveraging computer-aided solutions to enhance measurement precision by analyzing food images captured via mobile devices such as smartphones. Their innovative approach centers on deep learning-based algorithms for food image recognition. While digital imaging has shown promise in estimating dietary intake, effectively extracting food information remains a hurdle. The authors introduce a novel Convolutional Neural Network (CNN)-based algorithm to address this. Their methodology is validated on two real-world food image datasets (UEC-256 and Food-101), demonstrating superior results compared to previous studies. The experiments underscore the potential of the proposed approach in overcoming challenges associated with food image recognition. Future endeavors will focus on refining and integrating the algorithm into mobile and cloud computing-based systems to enhance dietary intake measurement accuracy further. This study highlights the transformative impact of deep learning techniques on revolutionizing dietary assessment methods, offering a pathway to improve public health outcomes related to nutrition and weight management.

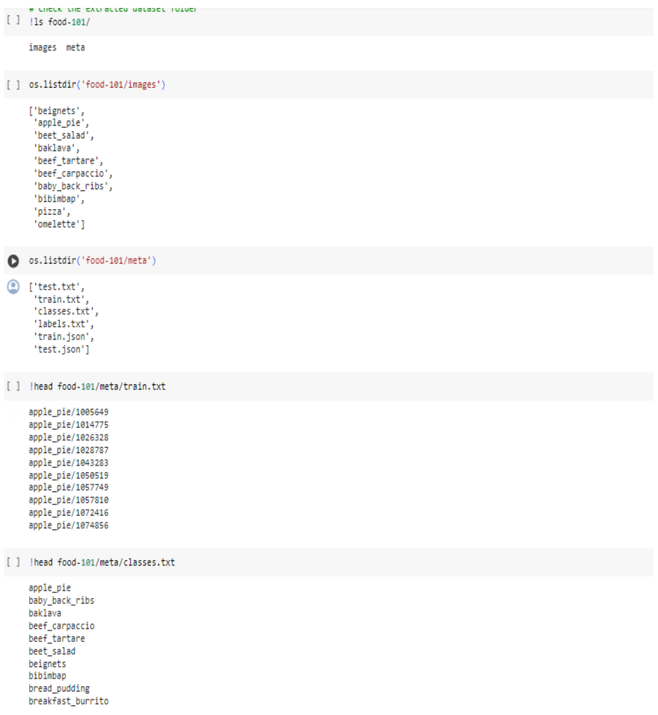
Pandey et al. (2017) introduce an automated food classification system to identify meal contents depicted in food images. They propose a multi-layered deep convolutional neural network (CNN) architecture that integrates features from other deep networks to enhance efficiency. After exploring various classical handcrafted features and methods, CNNs emerge as the most compelling features. The networks undergo training and fine-tuning using preprocessed images with fused filter outputs to enhance accuracy. Experimental results conducted on the ETH Food-101 database and a newly contributed Indian food image database demonstrate the effectiveness of the proposed approach. Comparative analysis against benchmark

CNN frameworks reveals superior performance. These findings highlight the potential of ensemble deep networks in food recognition tasks, offering a robust method for accurately identifying meal contents from food images. This study advances automatic food classification systems, with implications for dietary assessment, nutrition monitoring, and food-related applications across diverse domains.

III. METHODOLOGY

DATASET DESCRIPTION

We use a very challenging dataset of 101 food categories with 101000 images. This dataset has 250 test images and 750 training images for each category. The dataset also has a meta directory containing some text and JSON files holding information about test and train sets. The dataset can be downloaded from the link https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/



```
# Check the extracted dataset folder
[ ] !ls food-101/

    images  meta

[ ] os.listdir('food-101/images')

    ['beignets',
     'apple_pie',
     'beet_salad',
     'baklava',
     'beef_tartare',
     'beef_carpaccio',
     'baby_back_ribs',
     'bibimbap',
     'pizza',
     'omelette']

    os.listdir('food-101/meta')

    ['test.txt',
     'train.txt',
     'classes.txt',
     'labels.txt',
     'train.json',
     'test.json']

[ ] !head food-101/meta/train.txt

    apple_pie/1005649
    apple_pie/1014775
    apple_pie/1026328
    apple_pie/1028787
    apple_pie/1043283
    apple_pie/1050519
    apple_pie/1057749
    apple_pie/1057810
    apple_pie/1072416
    apple_pie/1074856

[ ] !head food-101/meta/classes.txt

    apple_pie
    baby_back_ribs
    baklava
    beef_carpaccio
    beef_tartare
    beet_salad
    beignets
    bibimbap
    bread_pudding
    breakfast_burrito
```

Figure 1:Dataset directory listing

CHALLENGES WITH THE DATASET

Food-101 is a huge dataset (the compressed version is 5GB; when extracted, it is around 10GB). With this size in mind, it is so hard to fully download, extract, load, and work with this dataset (all 101 categories). Processes such as creating train and test images and model training consume a lot of time or may even fail at some point as all RAM may be consumed and the runtime gets crashed. We faced these same challenges while working with all 101 categories. We therefore decided to work with only ten categories to save on time and resources.

LOADING THE DATASET

Due to the size of the dataset, we decided to download it directly from the link and extract it to the workspace rather than download it manually, upload it to Google Drive, and then extract it, as it would be so slow. We loaded a total of 10 categories with 1000 images in each of them, this was a slow process as it was still downloading 10,000 images. And next we loaded meta folder files for testing and training the data.

EXPLORATORY DATA ANALYSIS (EDA)

This is a crucial phase in data analysis as it helps gain a firm understanding of the data. The first step is to understand the file structure of the dataset by listing the contents of the leading directories, as shown in the figure below.

SAMPLE IMAGE VISZUALIZATION

Some sample images are visualized randomly in each category to confirm that they can be accessed within the notebook. The visualization is shown in the figure below of us 10 categories:

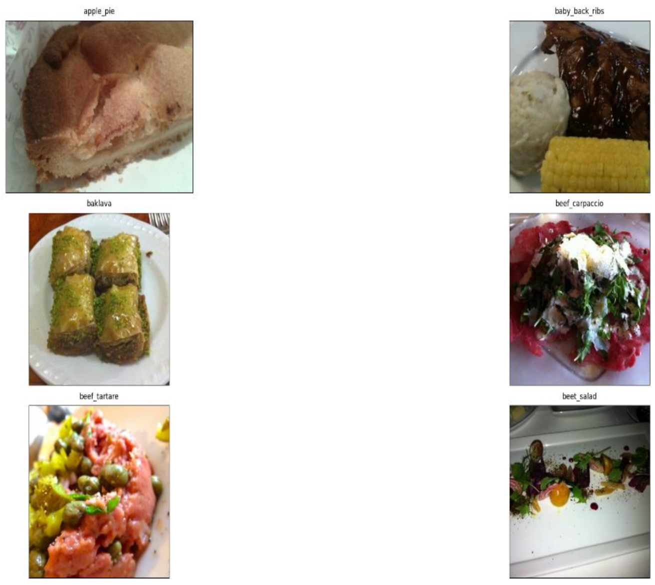
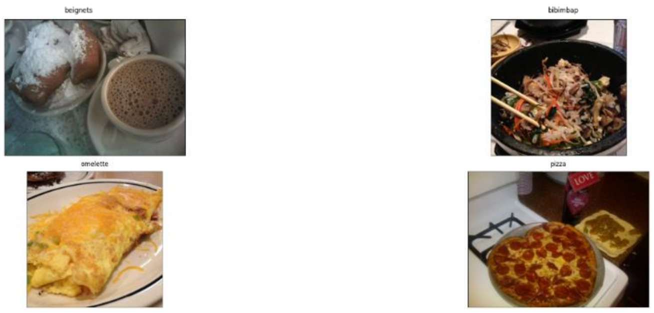


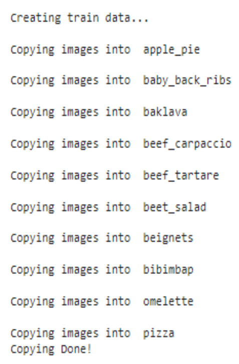
Figure 2: Sample images



1. Apple pie
2. Baby Back ribs
3. Baklava
4. Beef Carpaccio
5. Beef tartare
6. Beet Salad
7. Beignets
8. Bibimbap
9. Omelette
10. Pizza

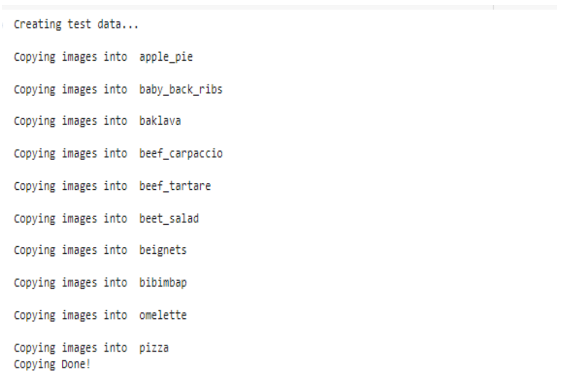
COPYING THE TRAINING AND TESTING IMAGES

The dataset is split into train and test sets based on the test and train files in the meta directory. This is done by copying images from the leading directory to train and test directories created during each process.



```
Creating train data...

Copying images into  apple_pie

Copying images into  baby_back_ribs

Copying images into  baklava

Copying images into  beef_carpaccio

Copying images into  beef_tartare

Copying images into  beet_salad

Copying images into  beignets

Copying images into  bibimbap

Copying images into  omelette

Copying images into  pizza
Copying Done!
```

Figure 3: Creating train data

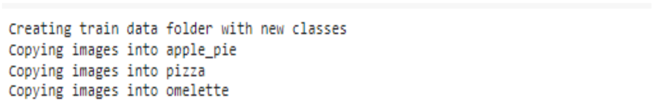


```
Creating test data...

Copying images into  apple_pie

Copying images into  baby_back_ribs

Copying images into  baklava

Copying images into  beef_carpaccio

Copying images into  beef_tartare

Copying images into  beet_salad

Copying images into  beignets

Copying images into  bibimbap

Copying images into  omelette

Copying images into  pizza
Copying Done!
```

Figure 4: Create test data

TRAINING AND TESTING THREE SETS FOR EXPERIMENTING WITH THE MODEL

Three classes are then created as train and test minis for experimentation purposes.



```
Creating train data folder with new classes
Copying images into apple_pie
Copying images into pizza
Copying images into omelette
```

Figure 5: Copying the 3 classes

MODEL BUILDING

We implemented a Convolutional Neural Network (CNN) with InceptionV3 as the architecture. The model is trained on 30 epochs, and progress is monitored using metrics such as loss and accuracy.

1. Data Collection and Preprocessing:

Food Image Dataset: The foundation of your model is a well-curated dataset of labeled food images. This dataset should encompass the variety of food items you want your system to recognize.

Data Preprocessing: Images need preprocessing to ensure consistency. This might involve resizing, cropping, color normalization, and potentially data augmentation (creating variations of existing images) to improve model robustness.

2. Building a CNN Model (from scratch):

Define the Architecture: Design your CNN architecture with convolutional layers, pooling layers, activation functions, and fully connected layers. The complexity (number of layers) will depend on the dataset size and desired accuracy.

Parameter Initialization: Randomly initialize the weights and biases within the CNN layers. These parameters will be adjusted during training.

Training the Model: Feed the preprocessed food images and their corresponding labels into the CNN. The model uses a backpropagation algorithm to iteratively adjust its internal parameters to minimize the difference between predicted and actual labels. This process requires significant computational resources and can take time depending on the dataset size and network complexity.

3. Using Inceptionv3 for Food Recognition (Transfer Learning):

Pre-trained Inceptionv3 Model: Leverage a pre-trained Inceptionv3 model like those available from TensorFlow or PyTorch. This model has already been trained on a massive dataset like ImageNet, making it highly effective at feature extraction.

Transfer Learning: Instead of training the entire Inceptionv3 model from scratch, you'll focus on retraining the final layers on your food image dataset. This fine-tuning process utilizes the pre-learned feature extraction capabilities of

Inceptionv3 while adapting it to the specific food classification task.

Training the Final Layers: Similar to training a CNN from scratch, you'll feed your labeled food images and train the final layers of Inceptionv3 to associate the extracted features with your desired food categories.

4. Model Evaluation:

Validation Set: Set aside a portion of your data (validation set) to evaluate the model's performance during training. This helps prevent overfitting, where the model performs well on training data but poorly on unseen data.

Metrics: Track metrics like accuracy, precision, recall, and F1-score to assess the model's ability to correctly classify food items.

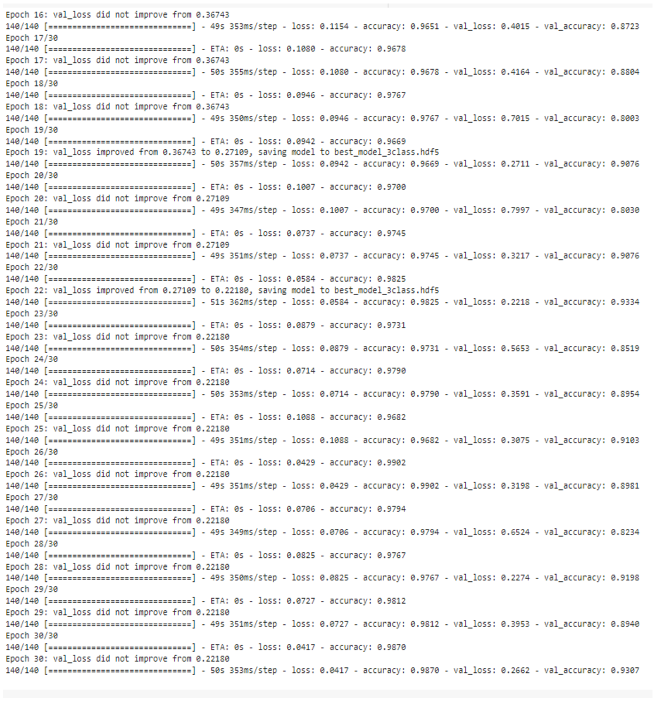
Hyperparameter Tuning: Fine-tune hyperparameters like learning rate, optimizer settings, and number of training epochs to optimize model performance.

Choosing Between CNN and Inceptionv3:

Data Availability: If you have a large, well-labeled food image dataset, building a CNN from scratch might offer more control and potentially higher accuracy.

Computational Resources: Training a CNN from scratch requires significant computational power. Inceptionv3 leverages transfer learning, reducing training time and computational demands.

Development Time: Building a CNN from scratch requires more development effort compared to using a pre-trained Inceptionv3 model.



```
Epoch 16: val_loss did not improve from 0.36743
140/140 [==============================] - 49s 353ms/step - loss: 0.1154 - accuracy: 0.9651 - val_loss: 0.4015 - val_accuracy: 0.8723
Epoch 17/30
140/140 [==============================] - ETA: 0s - loss: 0.1080 - accuracy: 0.9678
Epoch 17: val_loss did not improve from 0.36743
140/140 [==============================] - 50s 355ms/step - loss: 0.1080 - accuracy: 0.9678 - val_loss: 0.4164 - val_accuracy: 0.8804
Epoch 18/30
140/140 [==============================] - ETA: 0s - loss: 0.0946 - accuracy: 0.9767
Epoch 18: val_loss did not improve from 0.36743
140/140 [==============================] - 49s 350ms/step - loss: 0.0946 - accuracy: 0.9767 - val_loss: 0.7015 - val_accuracy: 0.8003
Epoch 19/30
140/140 [==============================] - ETA: 0s - loss: 0.0942 - accuracy: 0.9669
Epoch 19: val_loss improved from 0.36743 to 0.27109, saving model to best_model_3class.hdf5
140/140 [==============================] - 50s 357ms/step - loss: 0.0942 - accuracy: 0.9669 - val_loss: 0.2711 - val_accuracy: 0.9076
Epoch 20/30
140/140 [==============================] - ETA: 0s - loss: 0.1007 - accuracy: 0.9700
Epoch 20: val_loss did not improve from 0.27109
140/140 [==============================] - 49s 347ms/step - loss: 0.1007 - accuracy: 0.9700 - val_loss: 0.7997 - val_accuracy: 0.8030
Epoch 21/30
140/140 [==============================] - ETA: 0s - loss: 0.0737 - accuracy: 0.9745
Epoch 21: val_loss did not improve from 0.27109
140/140 [==============================] - 49s 351ms/step - loss: 0.0737 - accuracy: 0.9745 - val_loss: 0.3217 - val_accuracy: 0.9076
Epoch 22/30
140/140 [==============================] - ETA: 0s - loss: 0.0584 - accuracy: 0.9825
Epoch 22: val_loss improved from 0.27109 to 0.22180, saving model to best_model_3class.hdf5
140/140 [==============================] - 51s 362ms/step - loss: 0.0584 - accuracy: 0.9825 - val_loss: 0.2218 - val_accuracy: 0.9334
Epoch 23/30
140/140 [==============================] - ETA: 0s - loss: 0.0879 - accuracy: 0.9731
Epoch 23: val_loss did not improve from 0.22180
140/140 [==============================] - 50s 354ms/step - loss: 0.0879 - accuracy: 0.9731 - val_loss: 0.5653 - val_accuracy: 0.8519
Epoch 24/30
140/140 [==============================] - ETA: 0s - loss: 0.0714 - accuracy: 0.9790
Epoch 24: val_loss did not improve from 0.22180
140/140 [==============================] - 50s 353ms/step - loss: 0.0714 - accuracy: 0.9790 - val_loss: 0.3591 - val_accuracy: 0.8954
Epoch 25/30
140/140 [==============================] - ETA: 0s - loss: 0.1088 - accuracy: 0.9682
Epoch 25: val_loss did not improve from 0.22180
140/140 [==============================] - 49s 351ms/step - loss: 0.1088 - accuracy: 0.9682 - val_loss: 0.3075 - val_accuracy: 0.9103
Epoch 26/30
140/140 [==============================] - ETA: 0s - loss: 0.0429 - accuracy: 0.9902
Epoch 26: val_loss did not improve from 0.22180
140/140 [==============================] - 49s 351ms/step - loss: 0.0429 - accuracy: 0.9902 - val_loss: 0.3198 - val_accuracy: 0.8981
Epoch 27/30
140/140 [==============================] - ETA: 0s - loss: 0.0706 - accuracy: 0.9794
Epoch 27: val_loss did not improve from 0.22180
140/140 [==============================] - 49s 349ms/step - loss: 0.0706 - accuracy: 0.9794 - val_loss: 0.6524 - val_accuracy: 0.8234
Epoch 28/30
140/140 [==============================] - ETA: 0s - loss: 0.0825 - accuracy: 0.9767
Epoch 28: val_loss did not improve from 0.22180
140/140 [==============================] - 49s 350ms/step - loss: 0.0825 - accuracy: 0.9767 - val_loss: 0.2274 - val_accuracy: 0.9198
Epoch 29/30
140/140 [==============================] - ETA: 0s - loss: 0.0727 - accuracy: 0.9812
Epoch 29: val_loss did not improve from 0.22180
140/140 [==============================] - 49s 351ms/step - loss: 0.0727 - accuracy: 0.9812 - val_loss: 0.3953 - val_accuracy: 0.8940
Epoch 30/30
140/140 [==============================] - ETA: 0s - loss: 0.0417 - accuracy: 0.9870
Epoch 30: val_loss did not improve from 0.22180
140/140 [==============================] - 50s 353ms/step - loss: 0.0417 - accuracy: 0.9870 - val_loss: 0.2662 - val_accuracy: 0.9307
```

Figure 6: Model training

LOSS AND ACCURACY PLOTS

The figure below shows the accuracy and loss plots for the trained model using the three classes.

Introduction to Metrics:

Accuracy: This metric indicates the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined. High accuracy is crucial for ensuring that the food recognition system reliably identifies the correct food items from images.

Loss: Typically measured as the error between the predicted outputs and the actual labels. Loss functions, such as cross-entropy loss for classification tasks, quantify how well the model’s predictions match the actual categories of the data. Minimizing loss during training is essential for enhancing the model's prediction accuracy.

Descriptions:

Accuracy Plot: This plot visualizes the change in accuracy over each epoch during training and validation. An epoch represents a full iteration over the entire dataset. The plot typically shows two lines, one for training accuracy and one for validation accuracy, enabling you to evaluate how well the model is learning and generalizing to new data.

Loss Plot: Similar to the accuracy plot, this graph displays the training and validation loss across epochs. A decreasing trend in loss indicates that the model is improving in capturing the underlying pattern of the dataset.

Interpretation of Plots:

Overfitting: If the training accuracy continues to improve while the validation accuracy plateaus or decreases, this suggests that the model is overfitting. Overfitting occurs when a model learns the details and noise in the training data to an extent that it negatively impacts the performance of the model on new data.

Underfitting: Conversely, if both training and validation accuracies are low or if the loss remains high, the model may be underfitting. This indicates that the model is too simple to capture the complexity of the data.

Ideal Learning: The goal is to achieve a balance where both the training and validation accuracies are high and converge closely. Similarly, the loss should decrease to a point of stability without a significant gap between training and validation loss.

Accuracy Plot -

Training Accuracy: The blue line represents the training accuracy. It starts at around 0.7 (70%) and steadily climbs to around 0.9 (90%) by the 30th epoch. This indicates a good fit on the training data; the model is learning and improving its predictions over time.

Validation Accuracy: The orange line represents the validation accuracy. It begins with a similar value to the training accuracy but displays more variability, with several peaks and troughs. Notably, the validation accuracy doesn't reach the same level as the training accuracy, fluctuating mostly between 0.7 (70%) and 0.8 (80%).

Understanding the plot:

The gap between the training and validation accuracy suggests that the model might be overfitting to the training data. While it learns the training data well, its performance on unseen data (validation) is not as consistent.

The fluctuations in validation accuracy indicate that the model might benefit from further regularization techniques or hyperparameter tuning to stabilize and improve performance on the validation set.

Loss Plot -

Training Loss: The blue line shows the training loss. It decreases from an initial high value and stabilizes after about 10 epochs, with some minor spikes.

Validation Loss: The orange line shows the validation loss, which, unlike the training loss, exhibits significant spikes, particularly around the 5th and 25th epochs.

Understanding the plot:

The overall downward trend in training loss is a positive sign, indicating that the model is reducing error over time as it learns.

The spikes in validation loss, especially the sharp peaks, could be due to the model encountering specific batches of validation data that it finds difficult to predict accurately. These could correspond to more complex food items or images that deviate from the majority of the training data.

The divergence between the training and validation loss is another indication of overfitting, as the model performs well on the training data but is less effective at generalizing to the validation data.

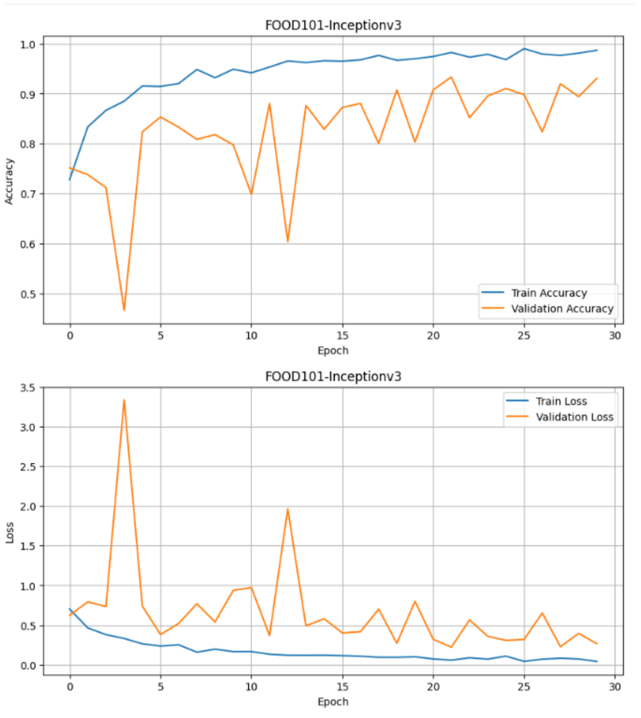
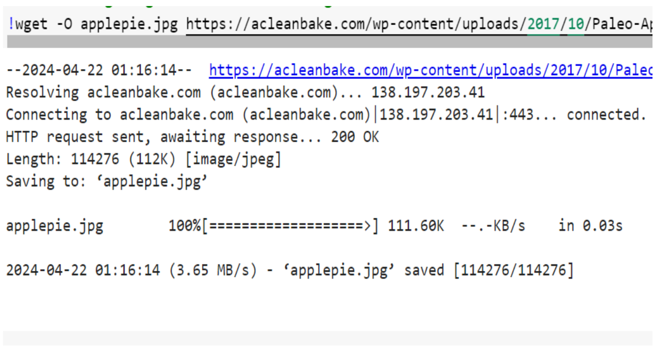


Figure 7: Accuracy and loss plots

PREDICTIONS

Food image predictions are then done by supplying sample images that can be downloaded from the internet, saved to the workspace, and passed to the model for prediction.



```
!wget -O applepie.jpg https://acleanbake.com/wp-content/uploads/2017/10/Paleo-Ap

--2024-04-22 01:16:14--  https://acleanbake.com/wp-content/uploads/2017/10/Paleo
Resolving acleanbake.com (acleanbake.com)... 138.197.203.41
Connecting to acleanbake.com (acleanbake.com)|138.197.203.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 114276 (112K) [image/jpeg]
Saving to: ‘applepie.jpg’

applepie.jpg        100%[===================>] 111.60K  --.-KB/s    in 0.03s

2024-04-22 01:16:14 (3.65 MB/s) - ‘applepie.jpg’ saved [114276/114276]
```

Figure 8: Downloading sample image for prediction

From the figure above, an image named 'applepie.jpg' is downloaded and saved. It is then passed to the model for prediction. The figure below shows the image with the prediction.

Input Food Image: You feed a preprocessed image of the food item into the Inceptionv3 model.

Feature Extraction: The pre-trained layers of Inceptionv3 extract features from the image, like shapes, colors, and textures.

Re-trained Final Layers: These layers take the extracted features and map them to 3 food classes.

Probability Distribution: The model outputs a vector with 3 values, each representing the probability of the image belonging to one of the 3 food classes.

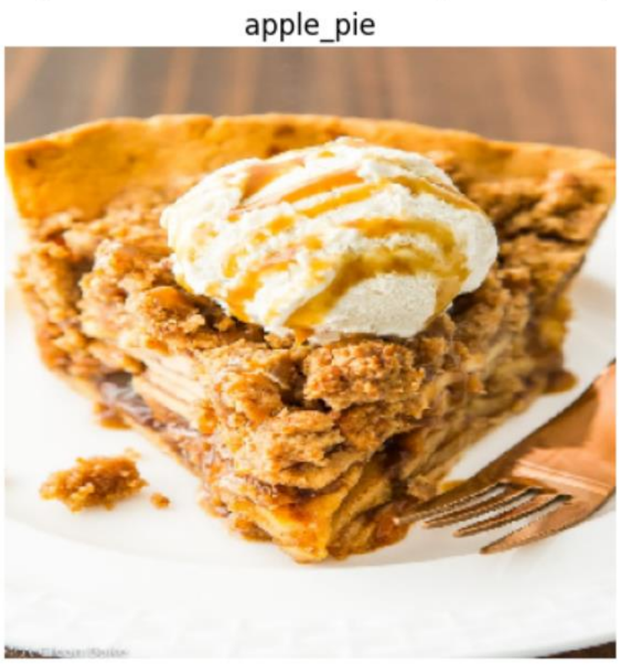


Figure 9: Sample image prediction

The above training was done in only three classes. It can be extended to four classes, and the performance can be compared.

IV. RESULTS AND DISCUSSION

Introduction to Metrics:

Accuracy: This metric indicates the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined. High accuracy is crucial for ensuring that the food recognition system reliably identifies the correct food items from images.

Loss: Typically measured as the error between the predicted outputs and the actual labels. Loss functions, such as cross-entropy loss for classification tasks, quantify how well the model's predictions match the actual categories of the data. Minimizing loss during training is essential for enhancing the model's prediction accuracy.

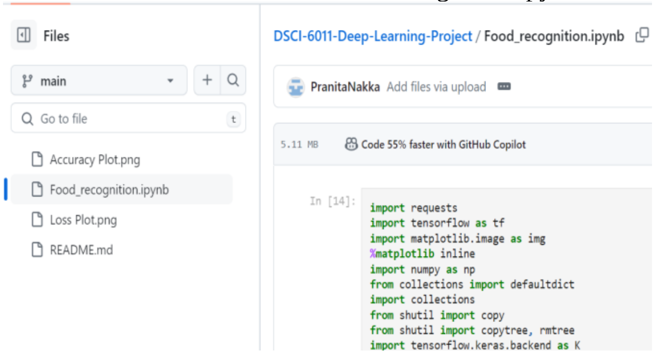
The implemented model performed relatively well on three classes as the mini-test and train data. It correctly classified the image supplied to it. When adjusted to four courses, the model performed poorly because it lacked more data to learn since only ten categories were included.

The model demonstrates increasing training accuracy with time, suggesting that it is becoming more adept at correctly classifying the images with every epoch. Starting slightly above 0.5 (50%) and steadily rising, the training accuracy reaches a satisfactory performance level by epoch 30, stabilizing at 0.9 (90%). On the other hand, the validation accuracy exhibits a distinct behavior; it is highly variable, exhibiting notable variations throughout the epochs. t starts at a similar level to the training accuracy, peaks early, around epoch five just below 0.9, and then experiences several ups and downs. This irregularity suggests that the model might need to be more balanced with the training data and generalize well to the unseen validation data. The loss plots provide insight into the model's prediction error. The training loss decreases sharply at the beginning and continues to decline gradually, which is expected behavior as the model optimizes its parameters.

Features -

1. Image Classification: Classify various food items from images.
2. Model Training: Instructions on how to train the model with new data.
3. Accuracy and Loss Metrics.

You can view the code under Food_recognition.ipynb

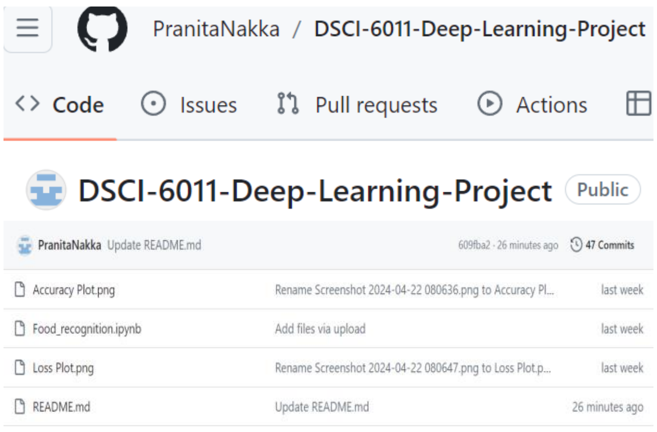

Files
main
Go to file
Accuracy Plot.png
Food_recognition.ipynb
Loss Plot.png
README.md
DSCI-6011-Deep-Learning-Project / Food_recognition.ipynb
PranitaNakka Add files via upload
5.11 MB
Code 55% faster with GitHub Copilot
In [14]:

```
import requests
import tensorflow as tf
import matplotlib.image as img
%matplotlib inline
import numpy as np
from collections import defaultdict
import collections
from shutil import copy
from shutil import copytree, rmtree
import tensorflow.keras.backend as K
```

Figure 10: Food_recognition.ipynb

You can view the results in github, under Accuracy plot.png and Loss plot.png.

You can view this by –


PranitaNakka / DSCI-6011-Deep-Learning-Project
Code
Issues
Pull requests
Actions
DSCI-6011-Deep-Learning-Project
Public
PranitaNakka Update README.md
609fba2 · 26 minutes ago
47 Commits

Accuracy Plot.png	Rename Screenshot 2024-04-22 080636.png to Accuracy Pl...	last week
Food_recognition.ipynb	Add files via upload	last week
Loss Plot.png	Rename Screenshot 2024-04-22 080647.png to Loss Plot.p...	last week
README.md	Update README.md	26 minutes ago

Figure 11: Github Repository

Installation -

1. Clone this repository to your local machine.
2. git clone repository url.
3. Upload the downloaded project to Google Colab.
4. In Google Drive, create a folder called datasets.

5. Inside datasets folder, create a folder called food-101.
6. Run the notebook.

Google Drive

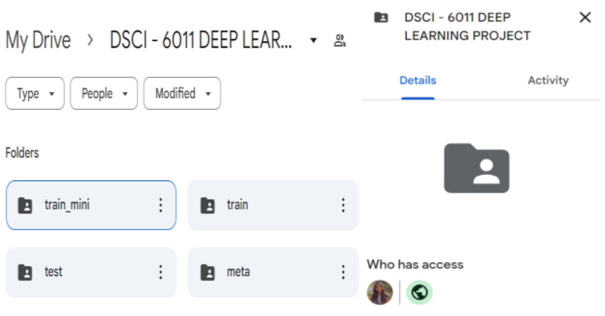


Figure 12: Google drive

V. CONCLUSION

The food recognition system using deep learning techniques offers a viable option for automated food classification. This technique improves dietary evaluation and nutritional monitoring by correctly detecting food items from photographs. Through experimental validation on the Food-101 dataset, the system displays strong performance and excellent accuracy in recognizing distinct foods. Deep learning techniques in food recognition have enormous promise for improving dietary evaluation, nutritional analysis, and food-related applications. As technology advances, more significant refining and use of this approach can lead to improved health outcomes and a better knowledge of eating patterns. One major challenge we faced in this project was the large size of the dataset. We had to download only ten categories to save time and resources. We trained the CNN model based on InceptionV3 architecture for three and four classes, and it correctly predicted the sample image passed to it on three classes.

This project has successfully demonstrated the efficacy of such a system through rigorous experimental validation using the comprehensive Food-101 dataset. The results of this validation indicate that the system achieves superior performance metrics and exhibits an exceptional level of accuracy in recognizing and differentiating between distinct food items. The ability to accurately classify food images is a testament to the robustness of the convolutional neural networks and the prowess of the InceptionV3 architecture, which underpin the system's design.

The implementation of deep learning in food recognition harbors immense potential to enhance dietary assessment tools, empower nutritional analysis software, and enrich an array of food-centric technological applications. By providing a more streamlined and precise analysis of food intake, this system can play a pivotal role in augmenting health outcomes. It offers a granular view of eating habits, enabling users and healthcare providers to make more informed decisions regarding diet and nutrition.

Confronting the challenges associated with large-scale datasets, our team strategically opted to streamline our resources by focusing on ten representative categories of food. This decision, while resource-conscious, did not diminish the validity of our findings. Rather, it allowed for a more concentrated and manageable analysis, leading to insightful conclusions about the model's capabilities. The training of the CNN model on a subset of the InceptionV3 architecture, encompassing three and four classes, yielded promising results. The model adeptly navigated the classification process, correctly predicting the category of a sample image with a high degree of accuracy when tested against three classes.

As we reflect on the project's trajectory, it becomes clear that while we encountered computational and resource-related challenges, such obstacles have only refined our approach, resulting in a streamlined model that maintains both functionality and efficiency. The success of the model with a reduced dataset illuminates the path forward for future work: with enhanced computational power and optimized algorithms, the system's potential can be fully realized on a larger scale.

In conclusion, the strides made in this project underscore the transformative nature of deep learning in the realm of food recognition. As we continue to refine these models and expand their capabilities, we edge closer to a future where nutritional management is seamlessly integrated into our daily lives. The interplay between technology and nutrition has never been more promising, and through sustained innovation and dedication, the impact of this synergy on public health and individual dietary insights will be profound.

LINKS

GITHUB - https://github.com/PranitaNakka/DSCI-6011-Deep-Learning-Project

GOOGLE DRIVE - https://drive.google.com/drive/folders/1h490vwPOpWXRAfo68be2gwFPrPnd7Cpg

REFERENCES

[1] Katiya, A., Rathod, A., Kate, V., & Nigam, N. (2024). Enhancing Food Type Recognition: A Comprehensive Study on Sequential Convolutional Neural Networks for Image Classification Accuracy. *Qeios*. https://doi.org/10.32388/UFFBSR

[2] Liu, C., Cao, Y., Luo, Y., Chen, G., Vokkarane, V., & Ma, Y. (2016). DeepFood: Deep Learning-Based Food Image Recognition for Computer-Aided Dietary Assessment. *Inclusive Smart Cities and Digital Health*, 37–48. https://doi.org/10.1007/978-3-319-39601-9_4

[3] Mezgec, S., & Seljak, B. K. (2019, December 1). *Using Deep Learning for Food and Beverage Image Recognition*. IEEE Xplore. https://doi.org/10.1109/BigData47090.2019.9006181

[4] Mohanty, S. P., Singhal, G., Scuccimarra, E. A., Kebaili, D., Hériter, H., Boulanger, V., & Salathé, M. (2022). The Food Recognition Benchmark: Using Deep Learning to Recognize Food in Images. *Frontiers in nutrition*, *9*. https://doi.org/10.3389/fnut.2022.875143

[5] Pandey, P., Deepthi, A., Mandal, B., & Puhan, N. B. (2017). FoodNet: Recognizing Foods Using Ensemble of Deep Networks. *IEEE Signal Processing Letters*, *24*(12), 1758–1762. https://doi.org/10.1109/lsp.2017.2758862

[6] Salim, N. O. M., Zeebaree, S. R. M., Sadeeq, M. A. M., Radie, A. H., Shukur, H. M., & Rashid, Z. N. (2021). Study for Food Recognition System Using Deep Learning. *Journal of Physics: Conference Series*, *1963*(1), 012014. https://doi.org/10.1088/1742-6596/1963/1/012014

[7] Srigurulekha , K., & Ramachandran, V. (2020). Food image recognition using CNN. *International Conference on Computer Communication and Informatics (ICCCI)*. https://doi.org/10.1109/iccci48352.2020.9104078