A MINI PROJECT REPORT ON

FOOD RECOGNITION AND INGREDIENT DECODER

SUBMITTED IN PARTIAL FULFILLMENT FOR THE

DEGREE OF BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE AND TECHNOLOGY

BY

22 - MS. VIJAYA GOVEKAR

13 - MS. KAVERI CHAVAN

04 - MS. POONAM BADHE

UNDER THE GUIDANCE OF

MS. PRAJAKTA GOTARNE

USHA MITTAL INSTITUTE OF TECHNOLOGY
S. N. D. T. WOMEN'S UNIVERSITY
MUMBAI – 400049
2021 – 2022.

CERTIFICATE

This is to certify that **Ms. Poonam Badhe, Ms. Kaveri Chavan and Ms. Vijaya Govekar** has successfully completed the Mini Project on **Food Recognition** and Ingredient Decoder, the partial fulfillment for the bachelor's degree in **Computer Science and Technology** during the year 2021-2022 as prescribed by SNDT Women's University.

GUIDE

Prof. Prajakta Gotarne

HEAD OF THE DEPARTMENT

Prof. Kumud Wasnik

PRINCIPAL

Dr. Shikha Nema

EXAMINAR 1

EXAMINAR 2

INDEX

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

ABSTRACT

Indian cuisine consists of a variety of regional dishes, due to which food image recognition is a necessity. Transfer Learning is widely used in multi-class image classification whereby a machine exploits the knowledge gained from a previous task to improve generalization about another. In this paper, we propose a custom-built CNN model and a transfer learning-based model for the purpose of food recognition and classification.

CHAPTER 2

PROBLEM STATEMENT

People are becoming more aware of the impact of food consumption on health. They are becoming more conscious of the importance of leading a healthy lifestyle. Food identification is crucial in this process. The food domain can be divided into three categories: the ingredients it contains, the number of ingredients, and the quantity of ingredients. Detection of food ingredients can assist people in adhering to a healthy diet and awareness of what they are consuming.

CHAPTER 3

LITERATURE SURVEY

Sr no.	Year	Title	Purpose/ Context	Advantage	Limitations
1	Aug 2021 International Research Journal of Engineering and Technology (IRJET)	Indian Food Image Recognition with MobileNetV2 Authors- Priya N , Preetam Kumari2, Poorvika N, Sanjana R, Dr. Hema Jagadish	A custom-built CNN model and a transfer learning based MobileNetV2 model for the purpose of food recognition and classification is proposed. A calorie estimation algorithm based on image features and nutritional information is proposed.	The dataset consists of 12 classes of Indian food images, with 100 images per class.	After experimentation, it was found that the MobileNetV2 model outperformed the custom CNN model with an accuracy of 79.45%.

Sr no.	Year	Title	Purpose/ Context	Advantage	Limitations
2	Dec 2021 The Maharaja Sayajirao University of Baroda	Overview of Deep Learning in Food Image Classification for Dietary Assessment System Author- Bhoomi Shah and Hetal Bhavsar	This paper defines the role of deep learning techniques based on a convolutional neural network for food object recognition. Studied the challenges in food recognition.	Different networks available and its comparison, various deep learning frameworks, the segmentation and classification methods, the performance achieved in terms of Top-1 and Top-5 accuracy, and the comparison of all the CNN architecture with its advantages and disadvantages were studied.	There are a variety of food domains that have not been touched yet as not much work has been done on recognition of liquid food items, and also, there is very less work done till now for European food and Indian cuisine.

CHAPTER 4

INTRODUCTION

Food is a necessity in everyone's life. Obesity is increasing at an alarming rate, endangering the lives of many people. Controlling calorie intake is essential for preventing obesity and a variety of other diseases. In addition, analyzing food images and estimating calorie intake can assist people in adhering to a healthy food diet. Due to the availability of smartphones, health monitoring systems are available to people, which can be accessed all the time. It can also help regular people keep up with their daily diet. Food recognition and calorie estimation have grown in popularity in today's world as people become more health-conscious about their diets. This can be accomplished in two steps. The first step is to successfully recognize a food image, and then the calorie can be estimated in the second step.

Indian cuisine consists of possibly hundreds of varieties of regional and traditional foods, due to which food image recognition becomes an essential task. The rising popularity of food blogging and image tagging pose requirements for a food recognition system customized to recognize regional foods. Due to the wide variety of Indian foods, the nutritional content varies largely. Not being conscious of our food intake can lead to diseases such as diabetes and obesity. A study reported that more than 650 million adults worldwide and 150 million adults in India are obese. A few techniques that exist for multi-class image classification are SVM, KNN, and Artificial Neural Networks. The Transfer Learning technique has shown promising results in the field of image classification. Transfer learning is a deep learning technique where a model is trained to learn and store the knowledge from one problem and use the same model to other similar problems. Convolution Neural Network is a deep learning technique that has gained popularity in image recognition tasks due to its high accuracy and robustness.

Characteristics of food include its type, composition, nutrients, amount of ingredients, etc. The main objective of the project is to detect various ingredients from the image of a prepared dish. Using machine and deep learning algorithms we can identify the characteristics of food, but we mainly focus on the ingredients it contains. Data analysis can be done as it contains enormous data volume which is of repetitive and irrelevant material. The people will be able to identify the food products in real-time, which does not require contact or seeing the food.

CHAPTER 5

EXISTING SYSTEM

In this section, we present some of the most common food calorie measuring methods that have been developed in the last few years. The objective here is to describe the main advan- tages and drawbacks of these methods, in order to demonstrate the novelty and contribution of our proposed system.

There are some methods existing for dietary assessments which involve self-reporting and manually recorded instruments. Most of these systems ask the user to input the details of the ingredients presented in the food. But it has some issues with the evaluation of calorie con-sumption. Calorie consumption by a participant is prone to be biased, i.e. underestimating and under reporting of food intake to increase the accuracy and to reduce the bias current methods are enhanced by mobile cloud computing system, which makes use of devices such as smartphones to capture dietary and calorie information. Next to this step, dietary and calorie information are analyzed by employing the computing capacity of the cloud for an assessment. However, users still have to enter the information manually. For food classifi- cation, previous work was concentrated on basic machine learning algorithms like Random Forest and SVM with hand-tailored features. These methods are generally based on relative or spatial relationships of features. But these methods come with computational cost at a large scale. Random forest came up with an accuracy of 90% whereas SVM came up with an accuracy of 92.2%. The system often fails to detect various food portions in mixed food; it also fails to segment them properly.

Though plenty of research and development efforts have been made in the field of visual based dietary and calorie information analysis, the efficient extraction of information from food images remains a challenging issue. In some methods for classification, K-NN (k nearest neighbor) is used which takes much time to train the images and classify. If data is not assumed properly, data loss may occur.A common problem of most of the models designed in this area is the identification of meal. Most of the models developed are found to have some demerits. Some of them lack precision, others require a lot of user intervention.

CHAPTER 6

PROPOSED SYSTEM

CNN:

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

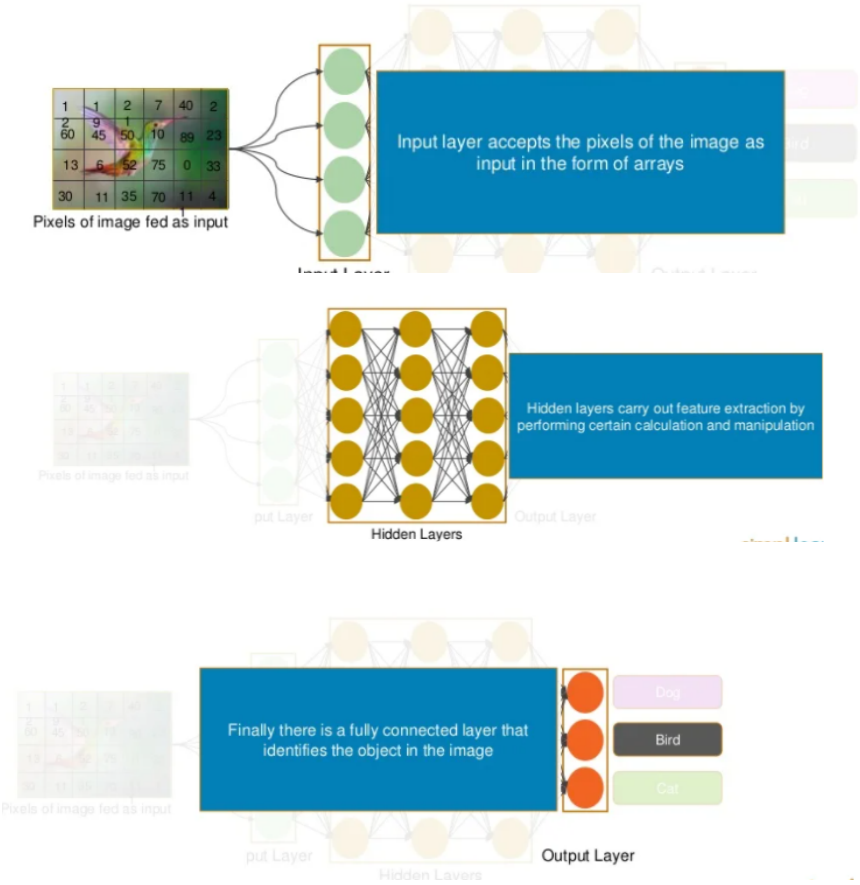


Fig 6.1 Working of Image Recognition Using CNN

CNN ARCHITECTURE:

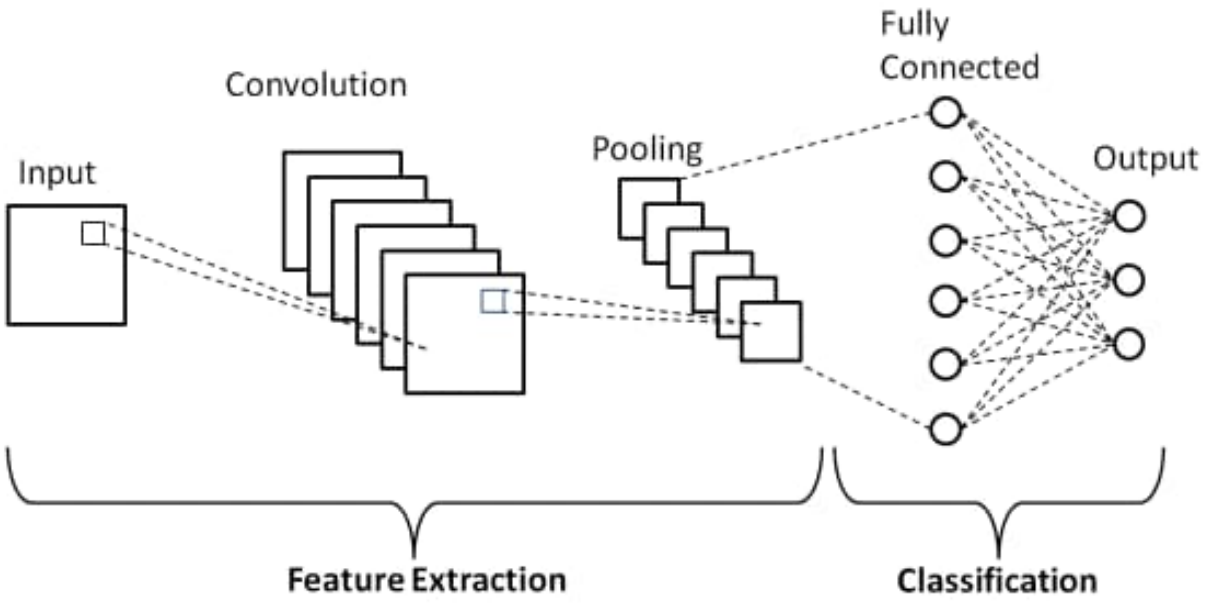


Fig 6.2 Basic Architecture

There are two main parts to a CNN architecture
A convolution tool that separates and identifies the various features of the image for analysis in a process called Feature Extraction A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.

VGG 16

VGG stands for Visual Geometry Group; it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. The “deep” refers to the number of layers with VGG-16 consisting of 16 convolutional layers. The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNet also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures.

This model achieves *92.7% top-5* test accuracy on the ImageNet dataset which contains *14* million images belonging to 1000 classes.

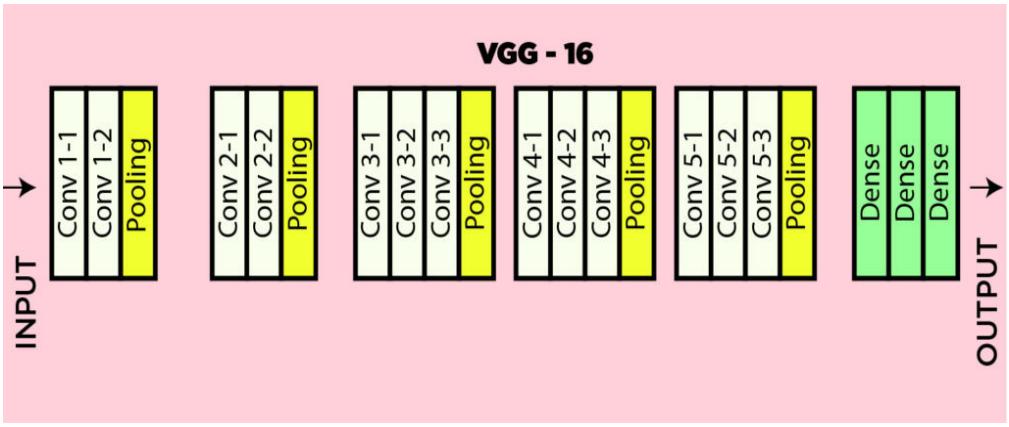


Fig 6.3 VGG16 Architecture

CHAPTER 7

SYSTEM ARCHITECTURE

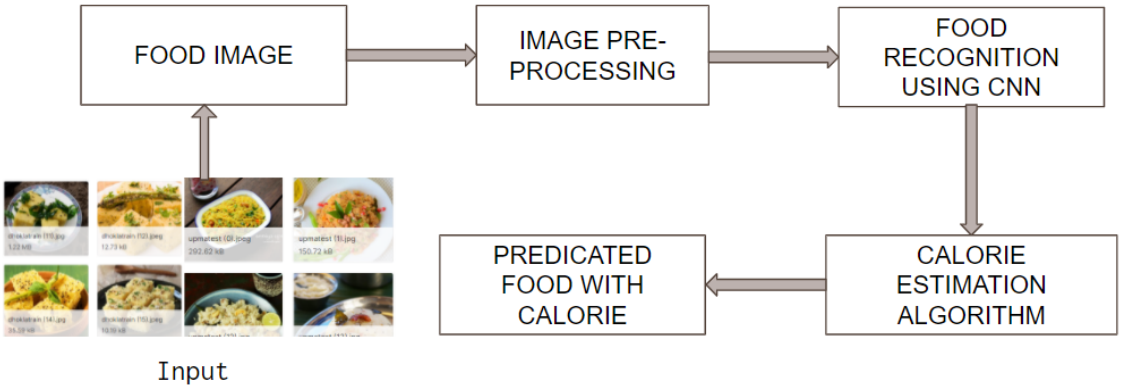


Fig 7.1 Proposed Architecture

Due to the nature of food items, the problem of food image detection and recognition is difficult. Foods are typically deformable objects, making defining their structure difficult. Furthermore, some food types can have a high intra-class (similar foods look very different) and low inter-class (different foods look very similar) variance, making the process of specifying the food type even more challenging. We will be approaching deep learning to solve this problem. Many problems in computer vision necessitate the definition of complex features, which can be difficult and time-consuming to define manually. Deep learning alleviates this by allowing computational models composed of multiple processing layers to learn these features and represent the input data with them automatically. Deep learning models have significantly improved the best results in a variety of research fields, including computer vision.

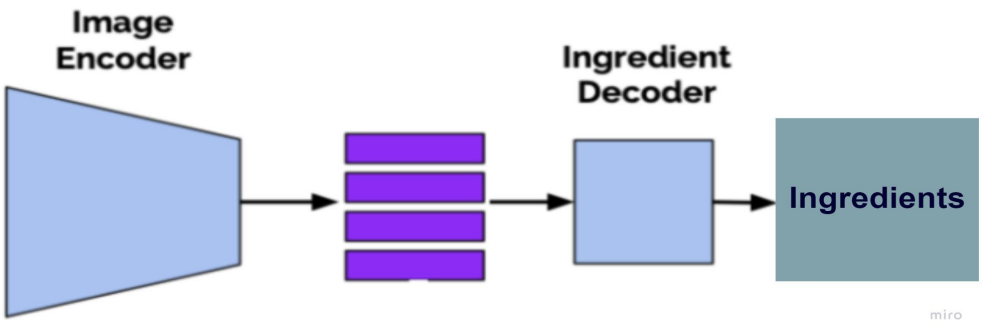


Fig 7.2 Proposed Decoder

CHAPTER 8

HARDWARE AND SOFTWARE REQUIREMENT

Software Requirements:

Google Colab, Python 3, Tensorflow, VGG 16, Keras

Hardware Requirements:

Graphical User Interface (GPU), Windows 10

CHAPTER 9

IMPLEMENTATION

SETUP

```
[ ]  import tensorflow as tf
     from tensorflow import keras

     from keras.layers import Input, Lambda, Dense, Flatten
     from keras.models import Model
     from keras.applications.vgg16 import VGG16
     from keras.applications.vgg16 import preprocess_input
     from keras.preprocessing import image
     from keras.preprocessing.image import ImageDataGenerator
     from keras.models import Sequential
     import numpy as np
     from glob import glob
     import matplotlib.pyplot as plt
```

Fig 9.1 Importing Libraries

Load the data: Food dataset

Raw data download

```
[ ]  # re-size all the images to this
     IMAGE_SIZE = [224, 224]


[ ]  train = '/content/drive/MyDrive/food20dataset/train_set.csv'
     test= '/content/drive/MyDrive/food20dataset/test_set.csv'
```

Fig 9.2 Import Dataset

Now we have a FoodDataset folder which contains two subfolders, train and test. Each subfolder contains image files for each category.

Generate a Dataset

```
[ ]  # re-size all the images to this
     IMAGE_SIZE = [224, 224]

[ ]  train = '/content/drive/MyDrive/food20dataset/train_set.csv'
     test= '/content/drive/MyDrive/food20dataset/test_set.csv'

[ ]  # add preprocessing layer to the front of VGG
     vgg = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

     Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
     58892288/58889256 [==============================] - 0s 0us/step
     58900480/58889256 [==============================] - 0s 0us/step

[ ]  # don't train existing weights
     for layer in vgg.layers:
       layer.trainable = False

[ ]   # useful for getting number of classes
     folders = glob('/content/drive/MyDrive/food20dataset/train_set/*')

[ ]  # our layers - you can add more if you want
     x = Flatten()(vgg.output)

[ ]  # x = Dense(1000, activation='relu')(x)
     prediction = Dense(len(folders), activation='softmax')(x)
```

Fig 9.3 Training Dataset

Visualize the data

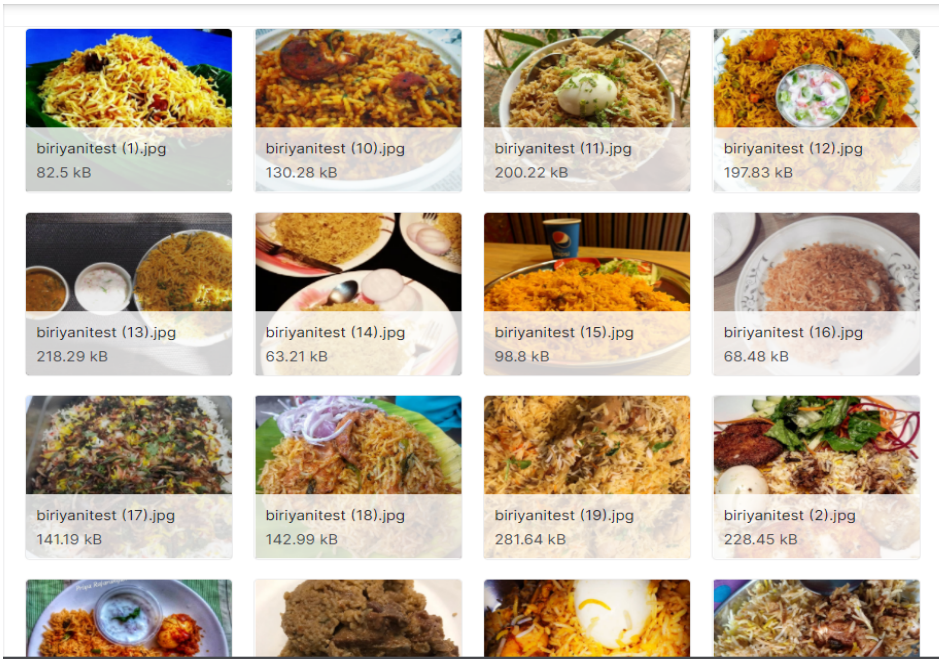


Fig 9.4 Dataset

Using image data augmentation

When you don't have a large image dataset, it's a good practice to artificially introduce sample diversity by applying random yet realistic transformations to the training images, such as random horizontal flipping or small random rotations. This helps expose the model to different aspects of the training data while slowing down overfitting.

augumentation

```
[ ]  from keras.preprocessing.image import ImageDataGenerator

     train_datagen = ImageDataGenerator(rescale = 1./255,
                                        shear_range = 0.2,
                                        zoom_range = 0.2,
                                        horizontal_flip = True)

     test_datagen = ImageDataGenerator(rescale = 1./255)

     training_set = train_datagen.flow_from_directory('/content/drive/MyDrive/food20dataset/train_set',
                                                      target_size = (224, 224),
                                                      batch_size = 32,
                                                      class_mode = 'categorical')

     test_set = test_datagen.flow_from_directory('/content/drive/MyDrive/food20dataset/test_set',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 480 images belonging to 6 classes.
Found 120 images belonging to 6 classes.
```

Fig 9.5 Augmentation of Dataset

Standardizing the data

Our images are already in a standard size (180x180), as they are being yielded as contiguous float32 batches by our dataset. However, their RGB channel values are in the [0, 255] range. This is not ideal for a neural network; in general you should seek to make your input values small. Here, we will standardize values to be in the [0, 1] by using a Rescaling layer at the start of our model.

Fig 9.6 Train the model

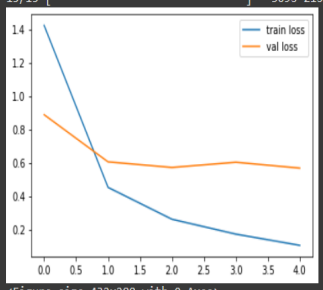
```
# create a model object
model = Model(inputs=vgg.input, outputs=prediction)
# view the structure of the model
model.summary()
```

```
Model: "model_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0

block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792

block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928

block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0

block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856

block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584

block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0

block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168

block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080

block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080

block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0

block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160

block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808

block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808

block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0

block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808

block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808

block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808

block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0

flatten_1 (Flatten)          (None, 25088)             0

dense_1 (Dense)              (None, 6)                 150534

=================================================================
Total params: 14,865,222
Trainable params: 150,534
Non-trainable params: 14,714,688
_________________________________________________________________
```

Fig 9.7 Run inference on new data

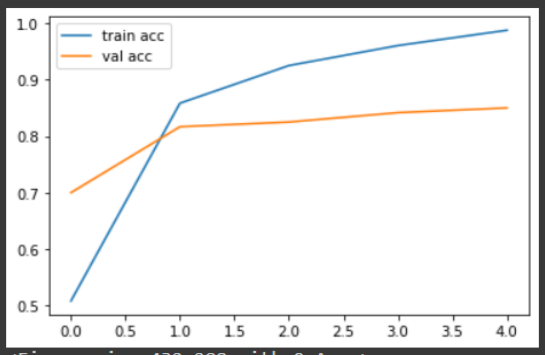
```
[ ]  r = model.fit_generator(
       training_set,
       validation_data=test_set,
       epochs=5,
       steps_per_epoch=len(training_set),
       validation_steps=len(test_set)
     )
     # loss
     plt.plot(r.history['loss'], label='train loss')
     plt.plot(r.history['val_loss'], label='val loss')
     plt.legend()
     plt.show()
     plt.savefig('LossVal_loss')
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which

Epoch 1/5
15/15 [==============================] - 388s 25s/step - loss: 1.4227 - accuracy: 0.5083 - val_loss: 0.8875 - val_accuracy: 0.7000
Epoch 2/5
15/15 [==============================] - 285s 19s/step - loss: 0.4517 - accuracy: 0.8583 - val_loss: 0.6048 - val_accuracy: 0.8167
Epoch 3/5
15/15 [==============================] - 283s 19s/step - loss: 0.2607 - accuracy: 0.9250 - val_loss: 0.5714 - val_accuracy: 0.8250
Epoch 4/5
15/15 [==============================] - 283s 19s/step - loss: 0.1715 - accuracy: 0.9604 - val_loss: 0.6026 - val_accuracy: 0.8417
Epoch 5/5
15/15 [==============================] - 309s 21s/step - loss: 0.1042 - accuracy: 0.9875 - val_loss: 0.5674 - val_accuracy: 0.8500
```




```
<Figure size 432x288 with 0 Axes>
```

```
[ ]  # accuracies
     plt.plot(r.history['accuracy'], label='train acc')
     plt.plot(r.history['val_accuracy'], label='val acc')
     plt.legend()
     plt.show()
     plt.savefig('AccVal_acc')
```




```
<Figure size 432x288 with 0 Axes>
```

predictions

```
[ ]  img = keras.preprocessing.image.load_img(
         "/content/drive/MyDrive/food20dataset/test_set/dosa/dosatest (18).jpg", target_size=IMAGE_SIZE
     )
     img_array = keras.preprocessing.image.img_to_array(img)
     img_array = tf.expand_dims(img_array, 0)  # Create batch axis

     predictions = model.predict(img_array)
     score = predictions[0]
     print(score)
```

```
[0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00 0.0000000e+00
 5.3348755e-25]
```

```
[ ]  class_names=['Biriyani','Chapati','Dosa','Gulab Jamun','Halwa','Idly']

     output_class=class_names[np.argmax(score)]
     print("The predicted class is: ", output_class)
```

```
The predicted class is:  Dosa
```

CHAPTER 10

FUTURE SCOPE

This project of ingredient decoder will find out the present ingredients in the image given as input by the user. This project is not only useful for normal people but also for visually impaired people to know what they are consuming. This project of ingredient decoder will be based on Indian cuisine and if possible, we might add daily sugar, carbohydrates, and fat detection models to this project as in India more than 50% of the population is suffering from obesity. This might help them calculate their intake.

CHAPTER 11

CONCLUSION

We have studied the ResNet and VGG 16 model and implemented both in our project. We conclude that VGG 16 gave accuracy of 96.76% on the training dataset and 81% on the validation dataset. Whereas, ResNet gave much less accuracy when compared with the VGG 16 model. We have also implemented an ingredient decoder further.

CHAPTER 12

REFERENCES

Indian Food Image Recognition with MobileNetV2
Priya N V1, Preetam Kumari2, Poorvika N3, Sanjana R4, Dr. Hema Jagadish5

Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).

Kagaya, Hokuto, Kiyoharu Aizawa, and Makoto Ogawa. "Food detection and recognition using convolutional neural network." In *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 1085-1088. 2014.

Kaggle,food20 dataset(indian food) | Kaggle

Shah, Bhoomi, and Hetal Bhavsar. "Overview of Deep Learning in Food Image Classification for Dietary Assessment System." In *Intelligent Systems, Technologies and Applications*, pp. 265-285. Springer, Singapore, 2021.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.

O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." *arXiv preprint arXiv:1511.08458* (2015).

https://github.com/krishnaik06/Transfer-Learning