A Report
ON

Calorific Counter App for Indian Food Items by Image

By

Name of the Student:	**ID.No.**
Mahak Kothari	2018A7PS0232G
Praveen Sridhar	2018A7PS0166G

AT

TamilNadu Health System Reforms-IT, Chennai

A Practice School I station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

(June 2020)

A Report On

Calorific Counter App for Indian Food Items by Image

By

Name of the Student:	ID.No.:	Disciplines
Mahak Kothari	2018A7PS0232G	B.E. Computer Science Engineering
Praveen Sridhar	2018A7PS0166G	B.E. Computer Science Engineering

Prepared in partial fulfillment of the Practice School-I Course Nos.
BITS C221/ BITS C231/ BITS C241
AT
TamilNadu Health System Reforms-IT, Chennai
A Practice School I station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

(June 2020)

Acknowledgment

We would like to express our sincere thanks to Tamil Nadu Health System Reforms, Chennai, for providing us the opportunity and resources to carry out the project work. We would specifically thank Dr. Rajashekar (Tamil Nadu Health System Reforms) for his guidance, co-operation, and invaluable suggestions. We are also grateful to Ms. Vijayalakshmi Praveen (Tamil Nadu Health System Reforms) for helping us with the onboarding process and providing us necessary feedback and support. We are grateful to other members of Tamil Nadu Health System Reforms, Chennai, for providing necessary comments and encouragement.

We are thankful to Dr. Nitin Chaturvedi, (PS-1 faculty for Tamil Nadu Health System Reforms, Chennai) for guiding and coordinating the Practice School program with Tamil Nadu Health System Reforms, Chennai. Our sincere thanks to the Practice School Division (PSD), BITS Pilani for implementing the Practice School I program.

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE- PILANI

Practice School Division

Station: TamilNadu Health System Reform - IT, Chennai

Centre: Chennai, India

Duration: 6 weeks **Date of start:** May 18, 2020

Date of submission: June 23, 2020

Title of Project: Calorific Counter App for Indian Food Items by Image

ID NO.	**Name**	**DISCIPLINE**
2018A7PS0232G	Mahak Kothari	B.E. Computer Science Engineering
2018A7PS0166G	Praveen Sridhar	B.E. Computer Science Engineering

Name of Mentor: Dr. Rajashekar

Name of PS Faculty: Dr. Nitin Chaturvedi

Key Words: Computer Vision, Image classification, Convolutional Neural Network, Transfer Learning, Ensemble Learning, Quantization, Pruning, Nutrients, Android app development, Java, Android Studio.

Project Areas: Android Development, Computer Vision.

Abstract: The process of image classification is quite an interesting field and when it is applied for health application it becomes much more interesting and useful for people in their daily lives. In this report, an approach to classify images of food items using Convolutional Neural Networks is presented. Since traditional Neural Networks require huge data, image augmentation techniques were used to increase the data, and transfer learning was used to improve accuracy. Upon successful completion of training, the model was then imported using the Tensorflow lite API in android studio.

Signature of Student	**Signature of PS Faculty**
Date: 23/06/2020	**Date:**

Table of Contents

Contents **Page No.**

i.Cover page 1

ii. Title page 2

iii. Acknowledgment 3

iv. Abstract Sheet 4

1) Introduction 7

2) Main Text

a) Proposed Methodology 8

b) Dataset 9

c) Image Preprocessing 10

d) Convolutional Neural Network 11

e) Ensemble Learning 13

f) Pruning 14

g) Quantization 15

h) Developing the app in android studio 16

i) Nutritional content 18

3) Conclusion 19

4) References 20

5) Glossary 21

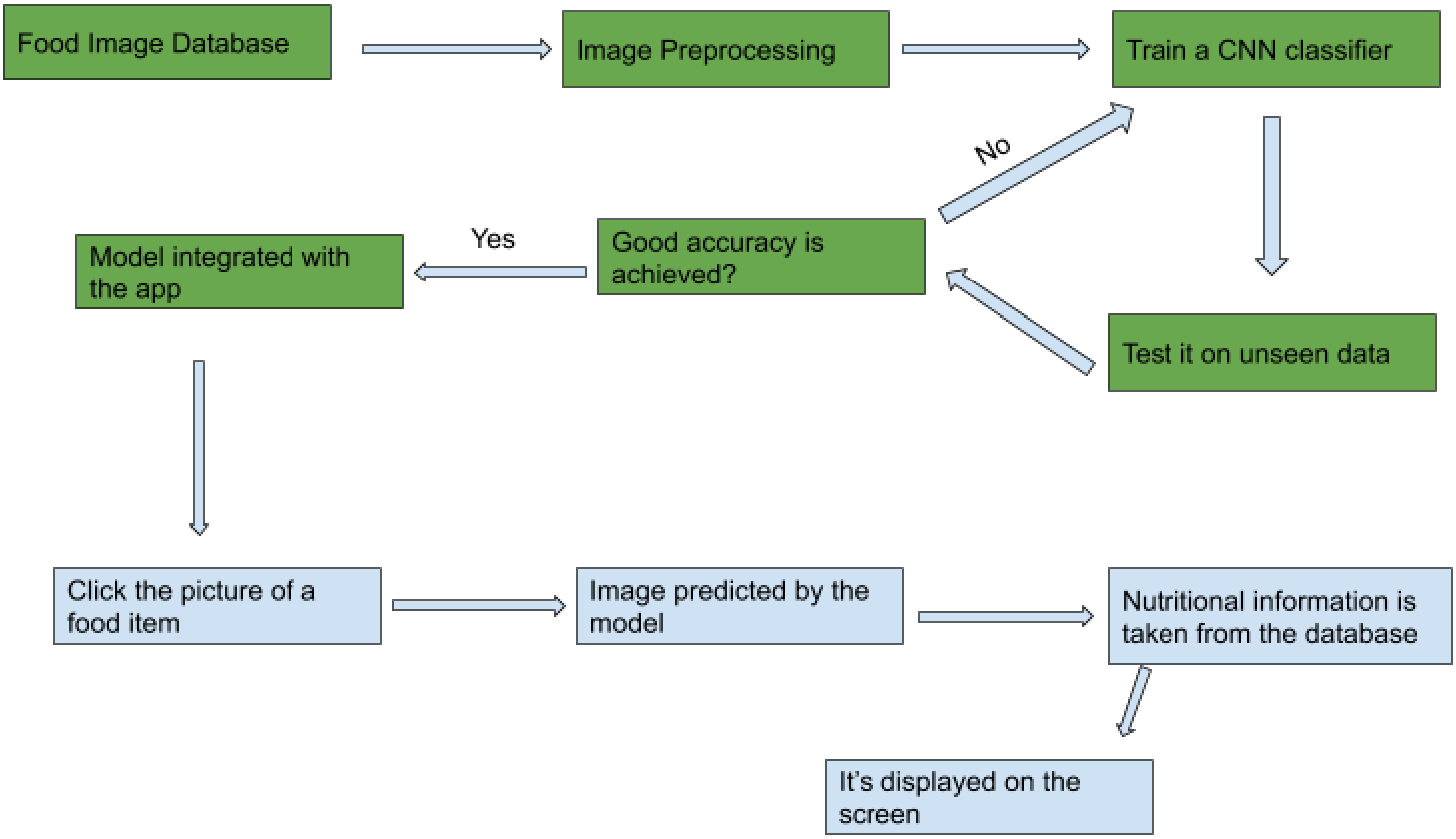
Introduction

In the current age, people are more conscious about their food and diet to avoid either upcoming or existing diseases. Traditional food journals require manual storing of information about the food we consumed in a dairy which is a very tedious and non-efficient way. In this era of smart technology, we require an efficient and technical solution to this problem.
A majority of the people are overeating and not being active enough. Given how stressed and busy people are today, it's impossible to forget to keep track of the food that they eat. This only increases the importance of proper classification of food.

In this report, a method to classify the images of food items for diet monitoring applications is discussed. Classification is done using Convolutional Neural Networks(CNNs). To increase the number of images data augmentation techniques are used and to improve the accuracy transfer learning and ensemble learning is applied. Quantization was used to decrease the model size. Since CNNs are capable of extracting features from an image automatically, it's used for food classification.

Due to the increased use of smartphones, we have developed a mobile application that clicks the picture of a food item then it is passed through a CNN which classifies it and gives the nutritional information about that. Previously some work was done by the researchers on this topic but most of the work doesn't take into account the Indian food items. Since Dataset was not available for Indian food items so web scraping was used to make the dataset. Dataset is available here.

Proposed Methodology


Food Image Database
Image Preprocessing
Train a CNN classifier
No
Model integrated with the app
Yes
Good accuracy is achieved?
Test it on unseen data
Click the picture of a food item
Image predicted by the model
Nutritional information is taken from the database
It's displayed on the screen

- Boxes in Green are explained in the below sections
- Blue Boxes shows how the app works.

Dataset

The dataset consists of 8 Food items:- Daal, Rice, Chapati, Idli, Kachori, Paneer Tikka, Apple, and Banana.

We have approximately 200 images per item.

The dataset was created by keeping in mind which food items are consumed most in our country.

Dataset was scraped from the web using this tutorial and then some bad images which were not useful for the classification were manually deleted.

Image Preprocessing

Various Image augmentation techniques we applied to increase the dataset as training a CNN requires a huge amount of data.

Some of the Techniques are:-

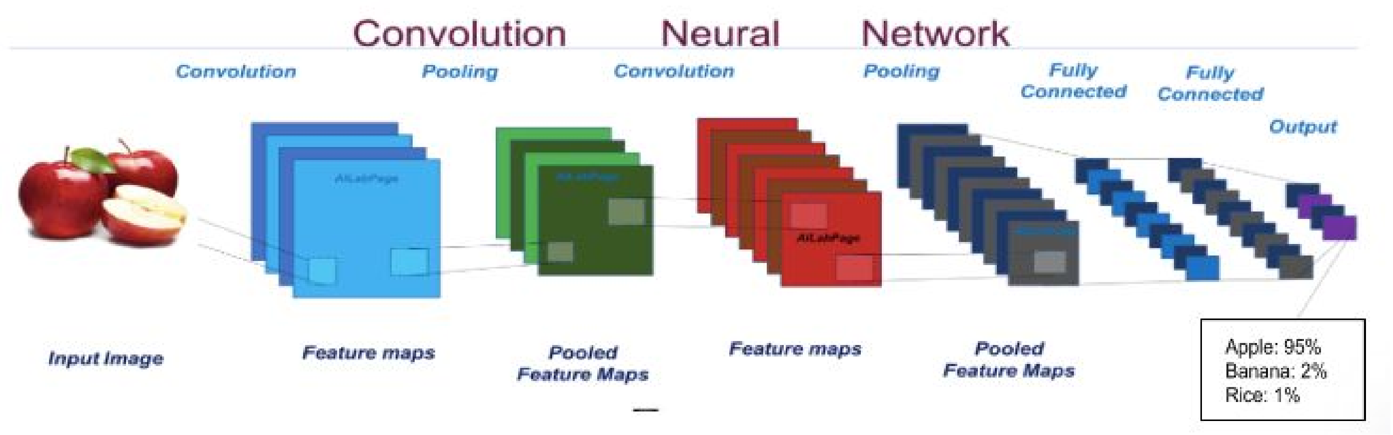
1) Rotation:- Randomly rotated image between (0 - 180degree)
2) GrayScale:- Randomly made some of the images as black and white.
3) Horizontal Flip:- Randomly flipped some of the images horizontally.
4) Vertical Flip:- Randomly flipped some of the images vertically.
5) Random Crop: Randomly cropped some part of the images and then trained the model on them.
6) Color jitter:- Changed brightness, hue, and contrast of the images.

As Apple remains an Apple after applying this transformation we have increased the dataset size by a good amount. This also prevents overfitting of the data.

However, it was observed that applying vertical flip decreased the model’s accuracy. Without the vertical flip, the model’s accuracy increased from 85% to around 95%, so it wasn’t included in the final model.

Convolutional Neural Network (CNN)

A convolutional neural network is a class of deep neural networks, most commonly applied to visual data (images).
This network primarily consists of the Convolutional layers, Pooling, Fully Connected layers, and some activation functions.



Convolutional Layer:- Each convolutional layer contains a series of filters known as convolutional kernels. The filter is a matrix of integers that are used on a subset of the input pixel values, the same size as the kernel. Each pixel is multiplied by the corresponding value in the kernel, then the result is summed up for a single value for simplicity representing a grid cell, like a pixel, in the output channel/feature map.

Pooling Layer:- Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network. The pooling layer operates on each feature map independently.

Fully Connected Layer:- Fully Connected Layer is simply, feed-forward neural networks. Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.

Activation Functions:- The purpose of an activation function is to add some kind of non-linear property to the function, which is a neural network. Without the activation functions, the neural network could perform only linear mappings from inputs x to the outputs y.
There are various types of activation functions like sigmoid, ReLU, softmax, etc. each of them having a unique property.

Output:- The output from the last fully connected layer is passed through the softmax activation function which converts it into probability.

Loss:- There are various types of loss functions to calculate how our predictions differ from the actual value. It is then used to update the weights of the convolutional layer through a process of backpropagation. The Loss function which I have used is the cross-entropy loss.

This way keep on updating the weights till we achieve good accuracy.

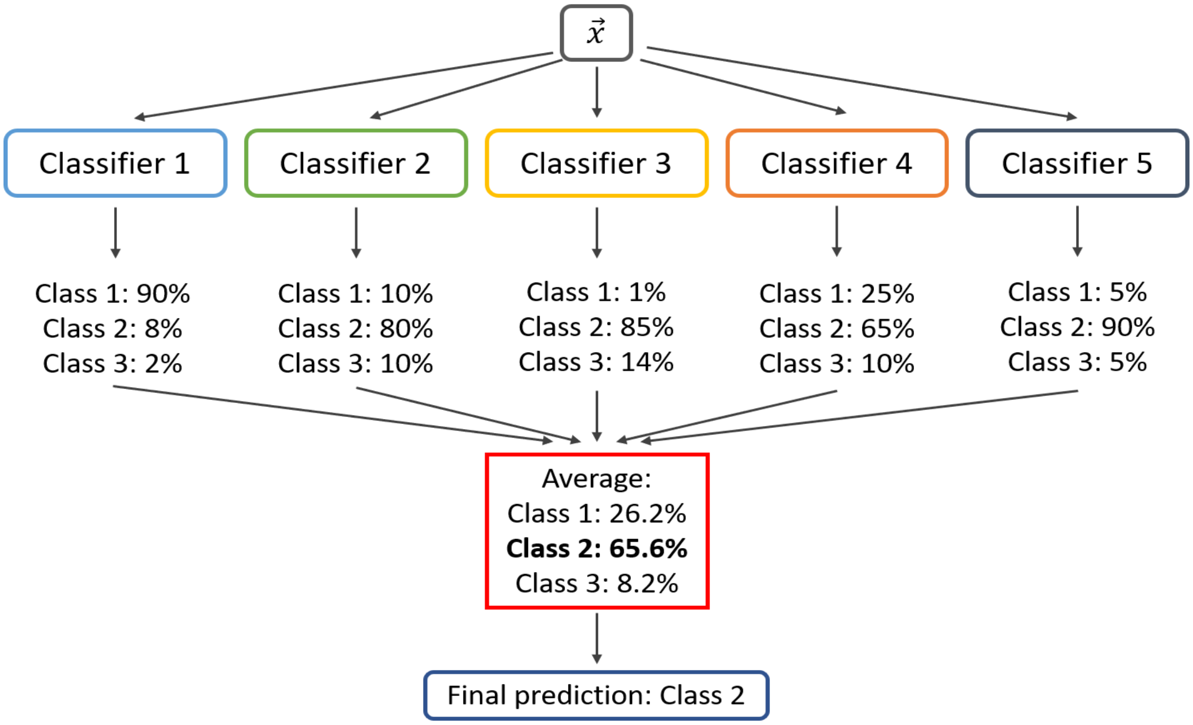
Transfer Learning:- Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.
It is used to speed up training and improve the performance of a deep learning model and it is also helpful if we have a small dataset.

The pretrained which I have used is EfficientNetB7 which is a State Of The Art model.

The code for CNN can be found here.

Ensemble Learning

Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. We take the probability from each model and then average them out to find the best suitable answer.



For Ensemble learning, models used are Resnext101 by FaceBook, EfficientNet b7 by AutoML MNAS, and ResNet101 by Microsoft.

With the use of ensemble learning, accuracy improved but model size increased a lot more (around 600 MB), and therefore this technique was not suitable for this application as this would increase the app size which is not commercially viable.

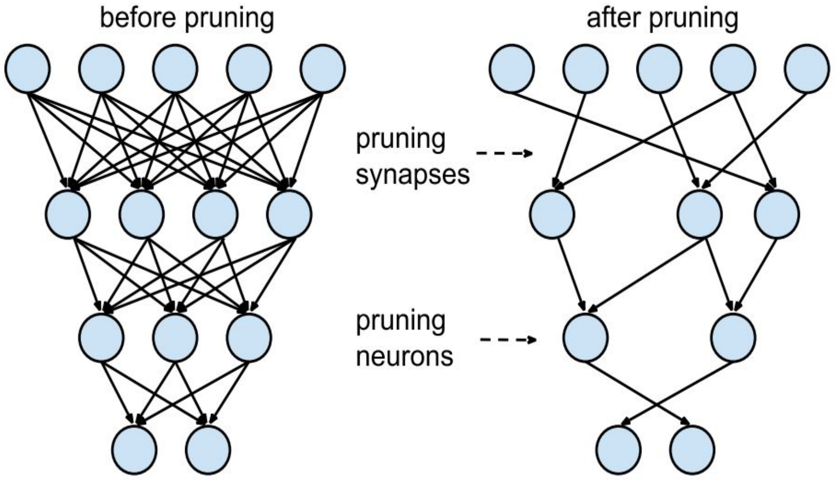
Therefore techniques like quantization and pruning were used to reduce the model size and maintain accuracy.

Pruning

Pruning is a way to reduce the size of the neural network through compression. After the network is pre-trained, it is then fine-tuned to determine the importance of connections. This is done through the ranking of the neurons from the network. The basic principles of pruning include removing unimportant weighted information using second derivative data. This results in better generalization results, improved speed of processing the results, and reduced size as well

Pruning is usually done in an iterative fashion, to avoid the pruning of necessary neurons. This also ensures that an important part of the network is not lost, as neural networks are a black box. The first step is to determine which neurons are important and which aren't.

Pruning can come in various forms, with the application of the method depending on the kind of output that is required from the developer.



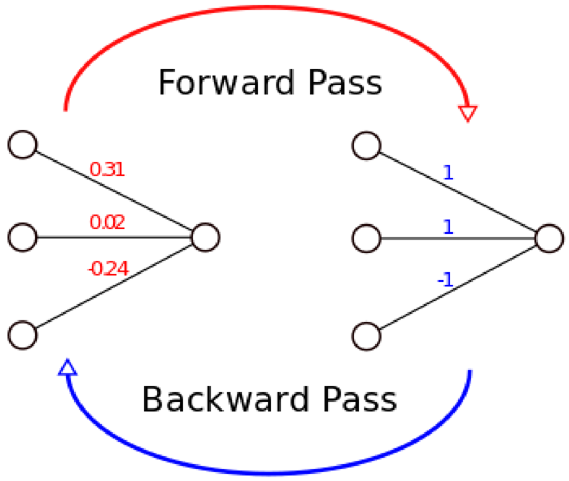
Types of Pruning:-

a) Weight pruning or Synapse pruning
b) Neurons pruning

In our case pruning doesn't affect the model size by a huge factor maybe because the efficient net is already an optimized model. Therefore it was not used in our model.

Quantization

Quantization for deep learning is the process of approximating a neural network that uses floating-point numbers by a neural network of low bit-width numbers.



During training, there are effectively two networks: float-precision and binary-precision.

The binary-precision is updated in the forward pass using the float-precision, and the float-precision is updated in the backward pass using the binary-precision.

Binary parameters do not have a derivative. Instead, the binary parameters are treated as if they were float-valued for the purpose of gradient calculation.

The binary- precision network is saved for inference, therefore, the deep learning model size becomes significantly smaller. In general, the size of the model and the number of computations during training is not as important - this only happens once.

32-bit float is converted into an 8-bit integer. Therefore the size of the model decreased by approximately ¼ (From 173 MB to 49 MB). The accuracy of the model is not affected much. First, the model was trained using TensorFlow(Keras) and later model was quantized with tflite API.

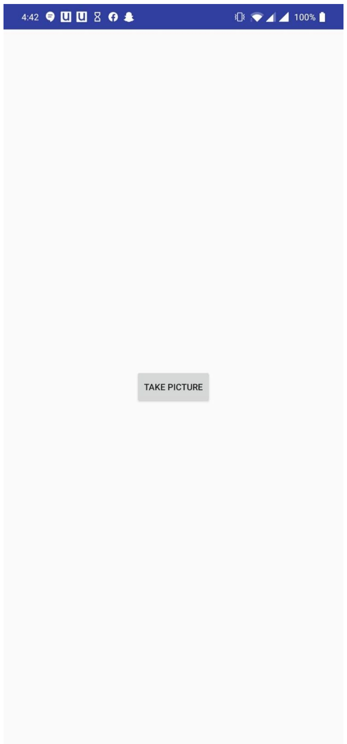
The code for the Quantization can be found here.

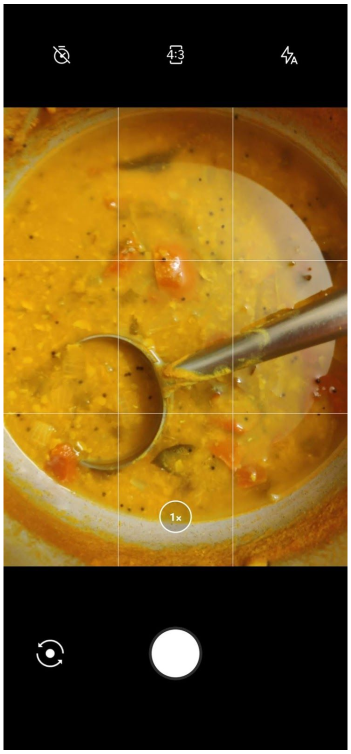
The Quantized model can be found here.

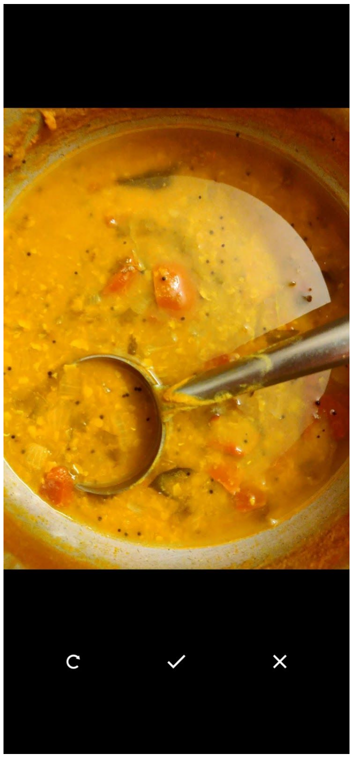
Developing the app in Android studio:

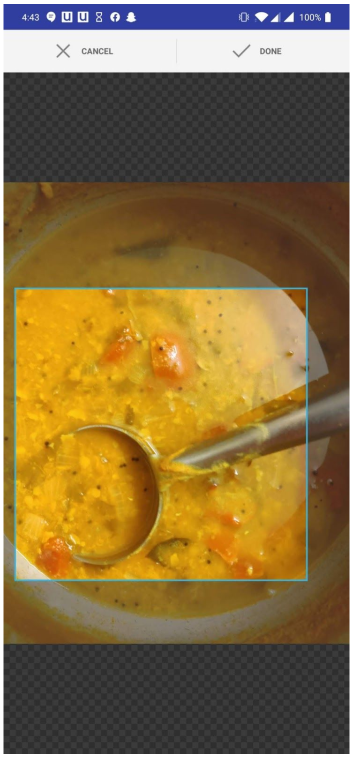
The trained TensorFlow lite model was imported to an android studio. For this purpose, the dependency for the TensorFlow lite library was included in the build.gradle file. The app requests permission to use the camera and the device storage because the image taken is stored on the phone and sent to the next activity. When the app is launched, the first activity has a take picture button that launches the device camera. Once the user has clicked the image, they should crop the image to make sure that the image is a square image. After the cropping is complete and confirmed, the classifying activity is started. In this activity, the user has 2 buttons. One, where he/she can classify the image and the second one to go back to the previous activity (to take another picture). The user can also see the image they have taken in this activity and an ImageView was used for this purpose. When the user clicks on the classify button, the TensorFlow lite model runs using the onboard hardware. Once the TensorFlow model completes running, all the returned data is written to a TextView. Currently, the details of all the food items are stored using a HashMap with the name of the food item being the key, and the carbs, fats, proteins, and energy values being the value associated with the key. Once all this data is written to the TextView and is visible on the screen, the user can then click the back button and be taken back to the first activity where they can click a picture (app images on the next page). The final app can be found here.

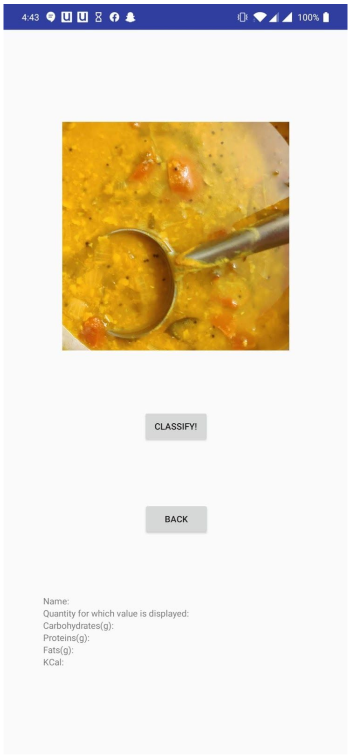
App



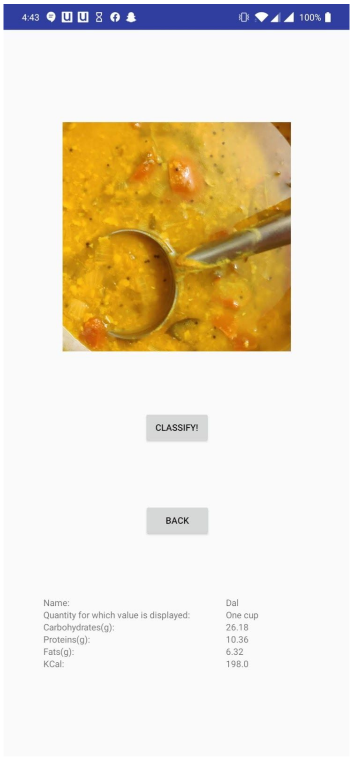








Name:
Quantity for which value is displayed:
Carbohydrates(g):
Proteins(g):
Fats(g):
KCal:



Name:	Dal
Quantity for which value is displayed:	One cup
Carbohydrates(g):	26.18
Proteins(g):	10.36
Fats(g):	6.32
KCal:	198.0

Nutritional Content

Food item	Proteins (g)	Carbohydrate (g)	Fat (g)	Calories (KCal)
Dal (1 cup)	10.36	26.18	6.32	198
Rice (1 cup)	4.2	44.08	0.44	204
Chapati (one)	2.34	13	0.62	68
Idli (one)	1.91	7.89	0.19	40
Kachori (one)	2.46	8.92	2.5	68
Paneer Tikka (per piece)	3.94	3.1	1.46	40
Apple (one medium-sized)	0.36	19.06	0.23	72
Banana (one medium-sized)	1.29	26.95	0.39	105

Nutritional Information is taken from here.

Conclusion

In this report, we have addressed the effectiveness of CNNs for food image recognition and detection. First, a food image dataset was constructed from images present on google. Second, CNN was applied to the recognition of 8 Indian food items and its performance was evaluated. Due to lack of data, image augmentation was found useful and transfer learning helped a lot to improve the accuracy. Finally applied quantization to achieved an accuracy of 95.6% on the testing set with the model of size 49 MB. This model was then imported using a Tensorflow lite library into the android studio. An app was built which would allow the user to take a picture on their phone and pass the image as an argument to the model. The model then returns an array that tells us the food item that is seen in the image.

There was a problem in integrating the quantized model with the app so, for now, we have added the unquantized model. There is a wide variety of food items that are available, so for that, we have thought to release regular updates of the app and with each update, we will increase the number of food items.

References

1) Attokaren, David Joseph & Fernandes, Ian & Sriram, A. & Srinivasa Murthy, Y.V. & Koolagudi, Shashidhar. (2017). Food classification from images using convolutional neural networks. 2801-2806. 10.1109/TENCON.2017.8228338.

2) Kagaya, Hokuto & Aizawa, Kiyoharu & Ogawa, Makoto. (2014). Food Detection and Recognition Using Convolutional Neural Network. 10.13140/2.1.3082.1120.

3) https://cs231n.github.io/

4) http://www.scholarpedia.org/article/Ensemble_learning#:~:text=Ensemble%20learning%20is%20the%20process,%2C%20function%20approximation%2C%20etc.)

5) https://medium.com/@joel_34050/quantization-in-deep-learning-478417eab72b#:~:text=Quantization%20for%20deep%20learning%20is,cost%20of%20using%20neural%20networks.

6) https://towardsdatascience.com/pruning-deep-neural-network-56cae1ec5505

7) https://www.tensorflow.org/lite/guide/inference#load_and_run_a_model_in_python

8) https://www.youtube.com/watch?v=JnhW5tQ_7Vo&t=161s

Glossary

1) **Activation Function**:- to allow Neural Networks to learn complex decision boundaries, we apply a nonlinear activation function to some of its layers.
2) **Backpropagation:-** Backpropagation is an algorithm to efficiently calculate the gradients in a Neural Network and update the weights.
3) **Categorical Cross-Entropy Loss:-** It is a popular loss function for categorization problems and measures the similarity between two probability distributions. It is given by `L = -sum(y * log(y_prediction))` where `y` is the probability distribution of true labels (typically a one-hot vector) and y_prediction is the probability distribution of the predicted labels, often coming from a softmax.
4) **Fine-Tuning:-** Fine-Tuning refers to the technique of initializing a network with parameters from another task and then updating these parameters based on the task at hand.
5) **Softmax:-** The softmax function is typically used to convert a vector of raw scores into class probabilities at the output layer of a Neural Network used for classification.
6) **Ensemble Learning:-** a process that combines the predictions from multiple neural network models to reduce the variance of predictions and improve accuracy.
7) **Quantization:-** process of approximating a neural network that uses floating-point numbers by a neural network of low bit-width numbers.
8) **Pruning:-** It's a model optimization technique that involves eliminating unnecessary values in the weight tensor.