Multimedia Tools and Applications (2021) 80:33011–33032
https://doi.org/10.1007/s11042-021-11329-6

Smartphone-based food recognition system using multiple deep CNN models

Abdulnaser Fakhrou[1] · Jayakanth Kunhoth[2] · Somaya Al Maadeed[2]

Received: 21 September 2020 / Revised: 26 January 2021 / Accepted: 23 July 2021 /
Published online: 12 August 2021

Abstract
People with blindness or low vision utilize mobile assistive tools for various applications such as object recognition, text recognition, etc. Most of the available applications are focused on recognizing generic objects. And they have not addressed the recognition of food dishes and fruit varieties. In this paper, we propose a smartphone-based system for recognizing the food dishes as well as fruits for children with visual impairments. The Smartphone application utilizes a trained deep CNN model for recognizing the food item from the real-time images. Furthermore, we develop a new deep convolutional neural network (CNN) model for food recognition using the fusion of two CNN architectures. The new deep CNN model is developed using the ensemble learning approach. The deep CNN food recognition model is trained on a customized food recognition dataset.The customized food recognition dataset consists of 29 varieties of food dishes and fruits. Moreover, we analyze the performance of multiple state of art deep CNN models for food recognition using the transfer learning approach. The ensemble model performed better than state of art CNN models and achieved a food recognition accuracy of 95.55 % in the customized food dataset. In addition to that, the proposed deep CNN model is evaluated in two publicly available food datasets to display its efficacy for food recognition tasks.

Keywords Food classification · Deep learning · Ensemble learning · Assistive system · Visual impairment

✉ Jayakanth Kunhoth
j.kunhoth@qu.edu.qa

Abdulnaser Fakhrou
afakhrou@qu.edu.qa

Somaya Al Maadeed
s_alali@qu.edu.qa

[1] Department of Psychological Sciences, College of Education, Qatar University, Doha, Qatar

[2] Department of Computer Science and Engineering, Qatar University, Doha, Qatar

1 Introduction

The data provided by WHO [1] describes that globally there exist about 285 million visually impaired individuals. Among 285 million individuals, 19 million individuals fall under the 0-14 age category. The main causes of visual impairments in children are nutrition deficiency, premature birth, birth defects, infections, etc. Vision impairments can negatively affect the social development and cognitive development of a child. People with visual impairments utilize various assistive systems in their day to day life for multiple tasks such as object recognition, navigation, text recognition, etc [15].

The advancement of technology in recent years made the development of assistive systems simpler. Current technological advancement allows developers to embed assistive technology in smart handheld devices. In the last decade, various assistive object recognition systems have been proposed for people with visual impairments [6, 17, 26, 32]. Most of the systems utilize computer vision technology to recognize objects. Computer vision technology adopts various image processing and machine learning algorithms to recognize the objects from imageries. Computer vision-based systems proposed for object recognition tasks can be classified into two types; tag-based systems and non-tag-based systems. Tag-based systems [21, 27] utilize visual markers or cues attached to the objects for recognizing them. A mobile device embedded with a smart scanner or camera is assigned to scan or capture the tags for visual marker identification. Non- tag-based systems [7, 9, 16] do not utilize any visual marker or barcodes. Instead, they process the raw imageries and apply various image feature detection algorithms and machine learning algorithms to recognize the objects. These systems can recognize various objects such as packed foods, currencies, smartphones, furniture, etc. Among existing object recognition systems, only a few works are focused on food recognition for people with visual impairments.

Food recognition or classification is an important task in visual object recognition. Since many of the food dishes are similar in appearance, size, and color, vision-based food recognition is more challenging. Most of the existing food recognition systems were developed for food calorie estimation as well as dietary assessment. Unlike existing food recognition works, this work is focused on the development of a precise smartphone application to recognize food dishes as well as fruits for children with visual impairments. Children with minimum knowledge of smartphone operation can utilize this application to identify the food dishes and fruits from natural scenes. This food recognition application with trained deep CNN model can be utilized to improve the experience and confidence of children during dining activity. Moreover, the application can be extended for educating the children with visual impairments.The major contributions of the proposed work are,

- Development of a food recognition application for visually impaired children.
- Application of transfer learning methodology for food recognition using multiple pre-trained deep CNN models trained in ImageNet [10] dataset.
- Development of an ensemble model that integrates the two CNN models for improved food recognition performance.
- Development of a new food dataset that contains images of both food dishes and fruits for food recognition challenge.
- Evaluation of proposed ensemble model in a sufficiently large two publicly available food recognition datasets.

The overall architecture of the proposed system and smartphone application for food recognition is given in Fig. 1. The proposed system is comprised of two parts; an Android application for real-time food recognition and a server-side for building the deep CNN model for food recognition. The Android application is configured to capture the images continuously using the embedded camera and run inference on each image for recognizing the food in real-time. The application will read out the name of the recognized food via the speaker.

On the server side, a deep neural network is trained for recognizing the food dishes and fruits. The deep CNN model for food recognition is developed by transfer learning methodology. Existing deep CNN models pre-trained on the ImageNet dataset are used for transfer learning. The trained model is optimized and quantized to work in the Android application. If the original trained deep CNN model (unoptimized and nonquantized model) is directly deployed in the edge devices like smartphones, the smartphone will either run out of memory, or prediction on the image will be very slow. Because of that, the trained raw deep CNN model was optimized and quantized before loading to the Android application. Model optimization will reduce the latency as well as inference cost. Moreover, an optimized deep CNN model can be deployed in smartphones that are having processing, memory, and storage constraints. Once the model is trained, it is converted to the TensorFlow Lite (TFLite) format (optimized and quantized model). The TensorFlow lite framework provides a collection of tools for deploying the TensorFlow machine learning models in edge devices like smartphones. During TFLite conversion, the model is optimized and reduced its size by about 90%. The optimized TFlite model is loaded in the Android application for recognizing the food images.

Once the Android application is loaded with the food recognition deep CNN model, then the application does not require any aid from an external server or pc. It can process the images and run inference on the images in real-time without any internet connection. The visual sensor embedded in the mobile device or the smartphone camera is assigned for capturing the scenes in front of the user. The captured frames are subjected to preprocessing since the raw image format does not match with the input of the deep CNN model. The image preprocessing function will extract each frame and preprocess the frame

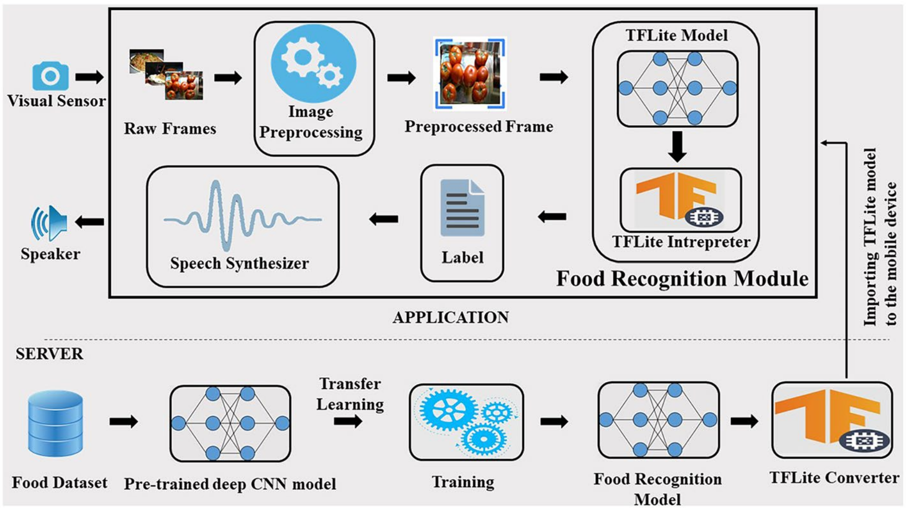

Fig. 1 Overall System Architecture

according to the input format of the deep CNN model. Later, the optimized deep CNN model runs inference on the preprocessed image frame. The output of the deep CNN model is the probability values for each class. The resulted probability values are interpreted into the class labels using the interpreter. Moreover, the labels with a confidence value of more than 90 % are only considered for speech synthesizing. The speech synthesizer is used to generate voice feedback for the users. Once the speech synthesizer receives the recognized label which is having a confidence value of more than 90 %, it generates the speech using Android text to speech SDK. The recognized label name is notified to the user via the speaker of the smartphone.

The remaining part of the paper is arranged as follows. In Sect. 2 we discuss the related works briefly. Detailed description about the custom food recognition dataset, public datasets used for the evaluation of the proposed method and the proposed methodology are given in Sect. 3. The evaluation experiments, results and discussion of results are provided in Sect. 4. And paper is concluded in Sect. 5.

2 Related work

Significant numbers of food recognition systems have been proposed in the last decade. Most of the proposed systems were focused on recognizing the food dishes or cuisines. Among them, a few of the systems were extended to food volume estimation and calorie estimation for dietary assessment. In this section, we provide a brief discussion on the food recognition systems proposed in the last decade.

Existing food classification systems utilized two different types of approaches; traditional approaches and deep learning approaches for recognizing the food dishes. The traditional approach follows a feature extraction method to extract the visual image features from the images of the food dish and represent them as a feature vector. Various image feature descriptor algorithms such as scale-invariant feature transform (SIFT), histogram of oriented gradients (HOG), local binary pattern (LBP), etc. were utilized in the literature for the feature extraction process. Moreover, morphological features such as color, shape, and size were also included for enhanced food recognition. The extracted feature vectors are utilized to train the classification model using various machine learning supervised algorithms such as support vector machine (SVM), K-Nearest Neighbors (K-NN), etc.

Kong et al. [20] proposed a mobile phone-based dietary assessment system named DietCam. The DietCam utilized SIFT feature descriptor algorithm and nearest neighbor algorithm to recognize the food dishes. Matsuda et al. [24] proposed a multiple-food recognition system that uses various image feature descriptors such as SIFT, CSIFT, HOG, Gabor texture, and color feature. SVM algorithm was employed for the classification task. Kawano and Yanai [18] proposed a real-time smartphone-based food recognition system for classifying 256 food dishes. The food recognition module in the proposed system used RootHOG (a variant of the HOG) feature vector and one vs rest linear classifier to classify the images into respective categories. A linear SVM classifier was trained with SIFT feature vectors for food recognition in [5]. The extracted SIFT features from images were converted to the bag of features or bag of visual words using the hierarchical K means clustering approach. Reference [29] proposed an SVM classifier based food recognition system. The proposed system integrated various features including texture, color, size, and shape for building the feature vector of images. Farinella et al. [11] proposed a food retrieval system that utilizes an SVM classifier for the food recognition task. The SVM classifier was trained with the combination of various handcrafted features. Pairwise rotation invariant co-occurrence LBP, SIFT, and Bag of texton were utilized to create the image feature vector.

In recent years, deep learning displayed its advancement over traditional machine learning algorithms for various classification tasks. Unlike traditional machine learning approaches, the deep learning methods don't require any explicit feature extraction algorithm. Moreover, deep learning methods showed exceptional improvements over traditional methods for various computer vision problems such as image classification, object recognition, face recognition, etc. Since 2014, deep neural networks are employed for food classification problems. Kagaya et al. [14] compared the performance of the CNN and traditional methods for the food classification task. The proposed method was evaluated in a custom dataset that contains 10 different types of food dishes. Obtained results show that the CNN-based approach outperformed the traditional approach (SIFT-Bag of Words) and displayed significant improvement in classification accuracy. Bossard et al. [8] introduced a large challenging dataset for the food recognition task. The proposed dataset, Food 101 contains 101000 images falling under 101 different classes. They analyzed the performance of the AlexNet CNN model and Random Forest algorithm for the food classification task. AlexNet CNN model performed better than the Random Forest algorithm in Food 101 dataset. Yanai and Kawano [35] implemented the transfer learning methodology for food classification using a pre-trained AlexNet model. ALexNet model was initially trained on the ImageNet dataset. The pre-trained AlexNet model was fine tunned for classifying the food images. The transfer learning approach using the GoogleNet model pre-trained on the ImageNet dataset was implemented in [25]. The pre-trained GoogleNet model was retrained on the Food-101 dataset. Liu et al. [22] proposed a smartphone-based food recognition system for dietary assessment. The proposed system consists of a smartphone for capturing the images and a cloud server for processing the image. A trained deep CNN model was employed in the server for the food classification task. The deep CNN model for food classification was developed by finetuning a pre-trained GoogleNet model on the image dataset.

Pandey et al. [28] introduced EnsembleNet by combining 3 different CNN models for classifying the food images. Initially, three CNN models (pre-trained GoogleNet, pre-trained AlexNet, and pre-trained ResNet) were separately finetuned on the food dataset. The ouput of the three models were fused for improved food classification performance. Yu et al. [36] introduced deep layer aggregation in deep CNN models. The deep layer aggregation based approach improved the food classification accuracy compared to finetuned pre-trained models. Foresti et al. [23] proposed a wide slice residual network (WISeR) for food recognition. The proposed deep CNN architecture WISeR consists of two deep CNN branches; a residual network and a slice network. WISeR achieved a state of the art classification performance in multiple public food recognition datasets. PAR-Net [30] introduced for food recognition consists of a fully connected layer and three subnetworks; a primary network, an auxiliary network, and a regional network. Each subnetwork of PAR-Net is based on a popular deep CNN architecture ResNet. Each subnetwork is a pre-trained ResNet model. All layers of the pre-trained ResNet model in each subnetwork were retrained in the Food dataset. The performance of PAR-Net is analyzed by arranging different types of ResNet(50, 34, or 101) in different combinations. PAR-Net built with ResNet 101 subnetworks delivered maximum classification accuracy.

Aguilar and Radeva [4] proposed a food recognition system by integrating local and flat classifiers. The evaluation of the proposed method was carried out in the MAFood-121 . It contains food from 11 different cuisines classified into 121 classes. The workflow of the proposed method is training separate classifiers for dishes in each cuisine, a flat classifier for all the dishes in the dataset irrespective of cuisines, and a classifier for recognizing the cuisines. Results from the cuisine classifier, flat classifier, and local classifier classifiers are analyzed for recognizing the food dish. Multi-task learning CNN model based on regularized uncertainty for food analysis was proposed in [3]. The proposed model can predict the dish name, Cuisine name, and category name (vegetable, meat or bread, etc.) for a given food image. ResNet 50 architecture has been utilized for building the base of the regularized uncertainty based multi-task learning (RUMTL)model. The proposed RUMTL model achieved better accuracy compared to the single-task model for dish recognition in the MAFood-121 dataset. Kayikei et al. [19] proposed a smartphone-based mobile application for recognizing Turkish food dishes. A pre-trained Inception V3 model finetuned on the Food 24 dataset was utilized for recognizing the food dishes. Moreover, they analyzed the performance of various pre-trained CNN models such as InceptionV3, Inception-ResNetV2, ResNet50, and Xception for the food classification task. Inception V3 outperformed all other pre-trained CNN models in Food 24 dataset. Table 1 displays a summary of various food recognition and classification systems proposed in the literature.

Considerable numbers of food image datasets for food classification challenges are proposed in the literature. Some of the popular and commonly used datasets are Food-101 [8], UEC Food-100 [24], and UEC Food-256 [18]. Most of the existing datasets contain food dishes from specific cuisine or a specific region only. But MAFood–121 [3] is a generic dataset and it contains food dishes from 11 different cuisines categorized into 121 classes. Another limitation of existing food datasets is that the majority of them contain food dishes only but not varieties of fruits or vegetables.

3 Materials and methods

3.1 Dataset

The evaluation of the proposed approach is carried out in three different food recognition datasets. The main aim of the proposed work is to develop a mobile application for recognizing both food dishes and fruits. Therefore a food dataset containing images of both food dishes and varieties of fruits is required to train the deep CNN model. In this context, a new food dataset [2] (of both food dishes and fruit varieties is customized for food recognition challenge. The sample images of the customized dataset are given in Fig. 2. The dataset is constructed by collecting food dish images from a popular food recognition dataset Food-101 [8] and varieties of fruit's images from a fruit recognition dataset [13]. Images of 15 types of food dishes (global cuisines) like cupcakes, ice cream, fried rice, french fries, donuts are extracted from Food-101. The images of 15 food classes are merged with images of 14 types of fruits such as banana, apple, mango, etc. available in the fruit dataset [13]. The proposed food dataset contains 31127 images classified into 29 different classes. Each class in the dataset contains 1000-1200 images approximately. All the images are converted into 224×224 pixels for our experiments. Among 31127 images, 21474 images are used to train the deep CNN models, 2751 images are assigned for validation and the rest 6902 images are utilized to evaluate the performance of the trained models.

Moreover, two publicly available food recognition datasets are used to assess the performance of the proposed method. A large public dataset and a comparatively small public dataset are selected for the evaluation of the proposed method. MAFood-121 [3] is a multi-attribute public food dataset available for food recognition problems. It

Table 1 Comparative Analysis of Food Recognition Methods Proposed in the Literature

Reference	Approach	Methodology	Dataset	Accuracy
Kong et al. [20] 2012	Traditional	SIFT and Visual words (K means Clustering). Classification: NN algorithm.	Custom dataset (5 classes)	92.00 %
Matsuda et al. [24] 2012	Traditional	SIFT, CSIFT, HOG, Gabor Texture, and color. Classification: MKL SVM	Custom dataset (100 classes)	21.00 %
Kawano et al. [18] 2014	Traditional	RootHOG, Fisher vector, and Color. Classifier: One vs rest	UEC-Food256	50.01 %
Anthimopoulos et al. [5] 2014	Traditional	SIFT, Bag of features. Classifier: L-SVM	Custom dataset (11 classes)	78.00 %
Pouladz et al. [29] 2015	Traditional	Texture, Color, Size, and shape. Classifier :SVM	Custom dataset (30 classes)	94.50 %
Farinella et al. [11] 2016	Traditional	PRICoLBP, SIFT, and Bag of textons. Classifier: SVM	UNICT-FD1200	75.70%
Kagaya et al. [14] 2014	Deep learning	CNN	Custom dataset (10 classes)	73.70 %
Bossard et al. [8] 2014	Deep learning	CNN (AlexNet)	Food-101	56.40%
Yanai et al. [35] 2015	Deep learning	CNN Transfer Learning (AlexNEt)	Food-101	70.41 %
Meyers et al. [25] 2016	Deep learning	CNN Transfer Learning (GoogleNet)	Food-101	79.00 %
Liu et al. [22] 2017	Deep learning	CNN Transfer Learning (GoogleNet)	Food-101	77.40%
Pandey et al. [28] 2017	Deep learning	EnsembleNet (GoogleNet + AlexNet + ResNet)	Food-101	72.12 %
Yu et al. 2018 [36]	Deep learning	DLA	Food-101	90.00%
Martinel et al. [23] 2018	Deep learning	WISeR	Food-101	90.27%
Qiu et al. [30] 2019	Deep learning	PAR-Net	Food-101	90.40 %
Tan et al. [34] 2019	Deep learning	CNN Transfer Learning (EfficientNet)	Food-101	93.00%
Yanai et al. [35]	Deep learning	CNN Transfer Learning (AlexNet)	UEC-FOOD 100	78.8%
Liu et al. [22] 2017	Deep learning	CNN Transfer Learning (GoogleNet)	UEC-FOOD 100	76.30%
Martinel et al. [23] 2018	Deep learning	WISeR	UEC-FOOD 100	89.60%
Yanai et al. [35] 2015	Deep learning	CNN Transfer Learning (AlexNet)	UEC-FOOD 256	67.60 %
Liu et al. [22] 2017	Deep learning	CNN Transfer Learning (GoogleNet)	UEC-FOOD 256	54.70%
Martinel et al. [23] 2018	Deep learning	WISeR	UEC-FOOD 256	83.20%
Aguilar et al. [4] 2019	Deep learning	CNN Transfer Learning (ResNet), Integration of local and flat classifier.	MAFood-121	81.62 %
Aguilar et al. [3] 2019	Deep learning	RUMTL	MAFood-121	83.82 %
Kayikei et al. [19] 2019	Deep learning	CNN Transfer Learning (Inception V3)	Food-24	88.00 %

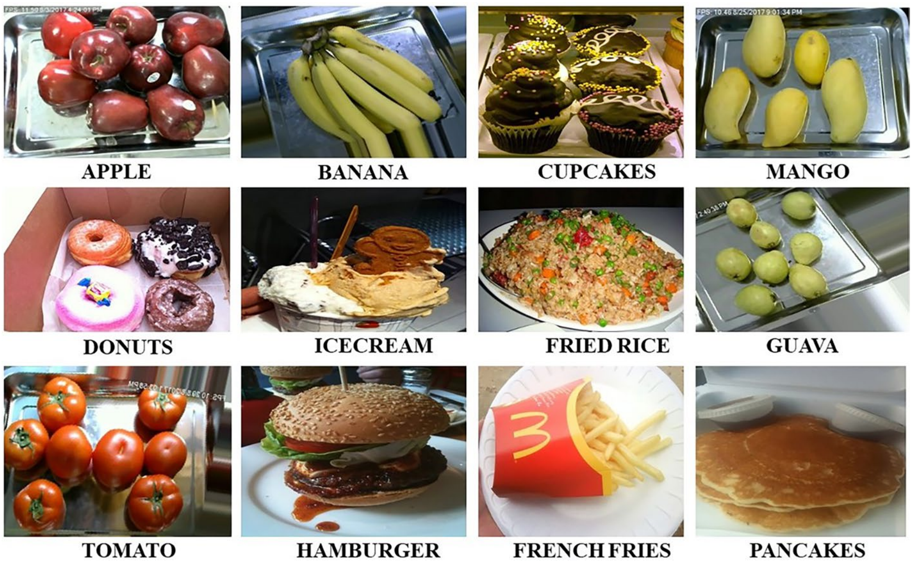

Fig. 2 Sample images from custom food dataset

contains 21175 images classified into 121 food classes. Moreover, 121 food classes contain food dishes from 11 different cuisines around the world. Each image has multiple labels; dish name, cuisine name, and food category name. The second public food dataset used for evaluation is Food24 [19]. Food24 contains about 11000 food images categorized into 24 food classes. All of the food classes in Food24 belongs to Turkish cuisine. The sample images from MAFood-121 and Food24 dataset is given in Fig. 3 respectively.

3.2 Methodology

3.2.1 Deep convolutional network

Convolutional Neural Network is a category of neural networks that contain neurons with biases and weights in each layer. The basic setting of a deep CNN model is shown in Fig. 4. The major elements of a deep CNN model are convolutional layers, batch Normalization

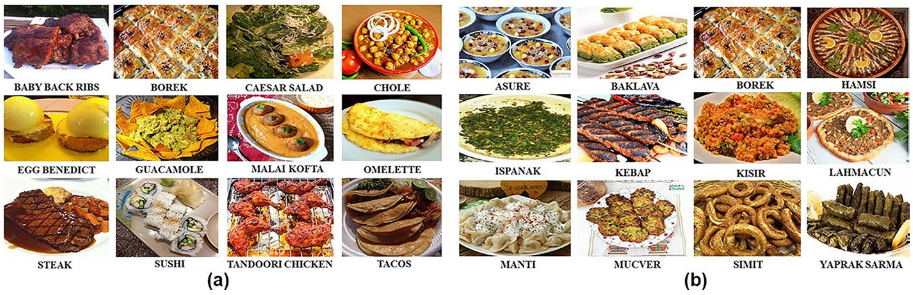

Fig. 3 Sample images from Public food dataset: (**a**) MAFood-121 dataset [3] (**b**) Food-24 dataset [19]

layers, pooling layers, and fully connected layers/dense layers. Batch normalization is absent in some of the popular deep CNN architecture such as AlexNet, VGG. Because the batch normalization technique has not existed before VGG or AlexNet.

Consider an input image i . For each local patch of the image i, the convolutional layer computes the neuron's output when i is passed through the convolutional layer. In a convolution operation, a filter or kernel slides over each pixel of the image to generate the feature map. The feature map is the resultant dot product of neurons' coefficient and input image vector. The convolution operation is applied either to the input image or feature map generated from the previous layer of the architecture.

Consider F_i^{c-1} is the feature map or output from the previous convolution or other previous layers and F_i^c is the feature map generated in the current convolutional layer. Then, $F_i^c = f(x)$, where

$$x = \sum_{i \in N_K} F_i^{c-1} * w_k^c + b_k^c \tag{1}$$

N_K : The number of kernels, b_k^c : bias value, w_k^c : current layer's weight matrix and *f(x)* is an activation function. Activation function is applied on the output of the convolutional operation *'x'* to compute the feature map F_i^c.

The activation function or transfer function is applied to the output of convolutional operation for transferring or mapping it to a particular interval such as [0, 1] or [-1, 1]. The popular activation functions used in deep CNN models are linear function (similar to a straight line equation), Sigmoid function ($f(x) = \frac{1}{1+e^{-x}}$), Tanh function ($f(x) = tanh(x)$) and RELU function.

RELU is the most widely used activation function and it is defined as,

$$f(x) = \begin{cases} 0, & if x < 0 \\ x, & if x \geq 0 \end{cases} \tag{2}$$

The batch Normalization layer is included to speed up the learning process. Batch normalization is responsible for normalizing the output from the previous activation layer . Furthermore, they improve the stability of the neural network. Pooling layers are applied to

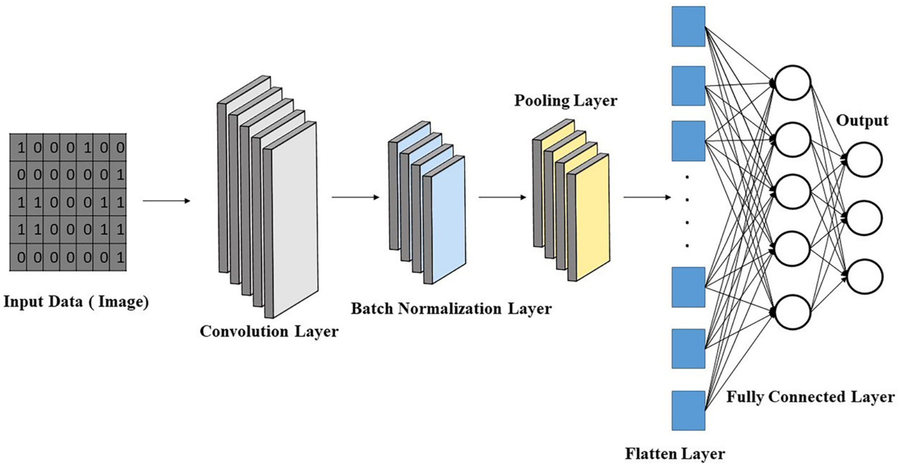

Fig. 4 Basic setting of a deep CNN architecture

the feature maps for reducing their spatial size. It allows the model to learn the important features of the image. Moreover, this dimensionality reduction technique helps the deep CNN model to achieve translation-invariance.

Multidimensional feature maps resulted from convolutional layers are flattened into a 1-dimensional array by using the flatten layer. The flattened features or resulted 1-dimensional array is fed to the fully connected layer/dense layer. The fully connected layer, a simple feed-forward neural network aggregate the 1-dimensional feature array to build the prediction model. Instead of the flatten layer, the global average pooling layer can also be used to reduce the spatial dimension of the multidimensional feature maps.

The output layer computes the probability of an input image for being in each class. The class with the highest probability is selected as the prediction of the model. The softmax function is widely used in the output layer to compute the probability. Softmax function normalizes the output of the dense layer.

For an input image i , the softmax function in the output layer estimates the probability that image i belongs to a class k_c by the following equation.

$$p(y = k_c|i;P) = \frac{e^{P_{k_c}^T i}}{\sum_{k_j=1}^{n} e^{P_{k_j}^T x}} \tag{3}$$

where n is the number of classes and P is the parameter of the model.

3.2.2 Transfer learning

Transfer learning is a prevalent research methodology in the field of machine learning. Here, the knowledge learned while interpreting a particular research problem is being reutilized for interpreting other similar research problems. In deep learning, transfer learning methodology utilizes the knowledge learned by a pre-trained model to train a new deep CNN model in a new dataset. Usually, deep CNN models require a large amount of data for training from scratch. Training deep CNN models with an inadequate amount of data can lead to a condition where the generalization ability of the model gets worsen and the model gets over-fitted to the training data. On the other hand, the transfer learning technique makes the development of deep CNN models possible even if a huge amount of data is not available for training the deep CNN architecture. One of the requirements for the transfer learning method is the availability of a pre-trained CNN model which is trained in a sufficiently large dataset. Transfer learning can be achieved in two ways; transfer learning via the feature extraction approach and transfer learning via the finetuning approach.

In transfer learning via the feature extraction approach, a pre-trained deep CNN model is used as a feature extractor. The images from a new dataset are passed through the pre-trained model in batches to retrieve feature vectors of the new image dataset. The extracted features are utilized to develop a new prediction model by training machine learning classifiers. Transfer learning via fine tunning approach utilizes a pre-trained model as a starting point to train on the new data set. The same pre-trained model is retrained on a new dataset after making some modifications to it. The modifications made in the pre-trained model are replacing the final classification layer with a new classification layer according to the number of classes in the new dataset, adding flatten layers, replacing fully connected layers, etc. In this work, transfer learning via finetuning is implemented. Figure 5 represents the general setting of the finetuning approach employed in this work.

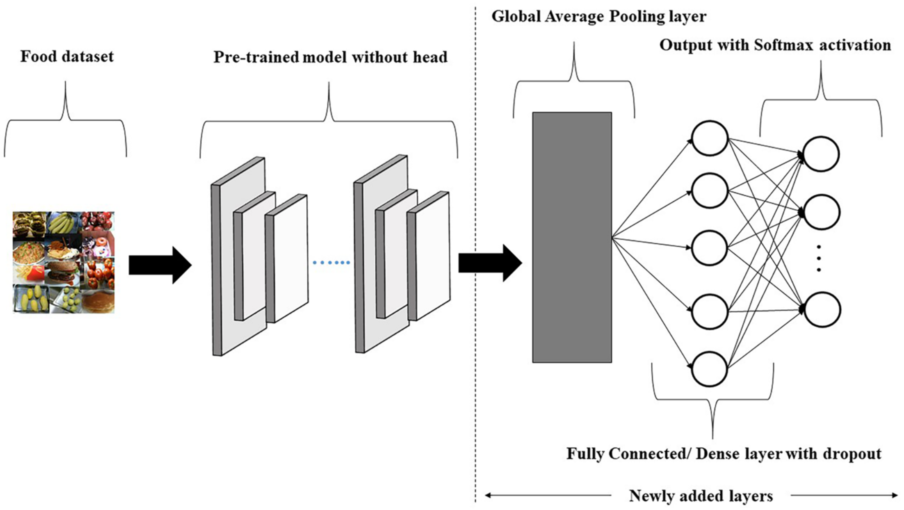
Food dataset
Pre-trained model without head
Global Average Pooling layer
Output with Softmax activation
Fully Connected/ Dense layer with dropout
Newly added layers

Fig. 5 Transfer learning via finetunning the pre-trained deep CNN

Three different pre-trained CNN models are used in this work to develop 5 different deep CNN food recognition models. VGG -16, InceptionV3, and DenseNet201 pre-trained in the ImageNet dataset are used for the transfer learning task. The pre-trained model is implemented without the head, which means the original fully connected layers in the pre-trained model are removed. Then, three new layers are added on top of the headless pre-trained model for fine-tuning. Generally, on top of the pre-trained model without the head, a flatten layer is added for converting multidimensional features into 1 dimension. In this approach instead of a flatten layer, global average pooling is added. The global average pooling has some advantages over the flatten layer. The flatten layer simply converts a tensor of any shape to a 1-dimensional tensor. While the global average pooling layer implements average pooling on the spatial dimension until each becomes one. As a result of the average pooling operation, the global average pooling layer produces the vector with better representation. After the global average pooling layer, a fully connected / dense layer with 512 neurons is added. Also, a dropout is applied to the fully connected layer to prevent overfitting. Finally, the output layer with the softmax activation function is added for the classification of the food images. Five different types of deep CNN models are developed using three pre-trained deep CNN models. The description of the developed five deep CNN models is given in Table 2.

The above-given abbreviation is used in the following sections to mention the five different deep CNN models.

Table 2 Description of five developed CNN models

Sl.NO	Model Description	Abbreviation
1	Finetuning the last convolutional block of pre-trained VGG [31] model.	TL_VGG
2	Finetuning the last two inception blocks of the pre-trained InceptionV3 [33] model.	TL_IV
3	Finetuning the last dense block of pre-trained DenseNet201 [12] model.	TL_DN
4	Finetuning all layers of pre-trained InceptionV3 model.	TL_ IVF
5	Finetuning all layers of the pre-trained DenseNet201 model.	TL_DNF

3.2.3 Ensemble learning

Ensemble learning is a popular and widely used approach in machine learning where multiple models are integrated for enhanced performance. The resultant model from the ensemble learning technique is known as the ensemble model. The ensemble model contains multiple machine learning models with the same or different algorithms trained on a set of the same or similar data. The multiple models included in the ensemble models are known as base models. Based on the problem and the algorithm of the base models, various approaches are used in the literature to combine the predictions of them to build the ensemble model. Commonly used two approaches are hard voting and soft voting. The hard voting a.k.a majority voting approach considers the class label with the most votes among all base models/classifiers as the ensemble decision. Let C1, C2, and C3 the three base classifiers in an ensemble model, and A and B are the two class labels. For an image *i*, the predicted label by each classifier is, C1=A, C2=B, and C3=A. Out of 3 classifiers, 2 classifiers predicted the class label as A, which means A received majority votes, so class A is the decision of the ensemble model.

The soft voting a.k.a the average voting approach will compute the average of probabilities obtained in each base classifier of the ensemble model for each class. The class with the highest average probability is considered as the decision of the ensemble model.

Algorithm 1: Ensemble learning using soft voting approach

```
Input: An Image $x$
initialization;
C=$[C_0, C_1, C_2, C_3, ......C_{N-1}]$ and K=$[K_0, K_1, K_2, K_3, ......K_{L-1}]$,
  where C is the list of classifiers and K is the list of class labels.
for each $C_i$ in $C$ do
    create an array $A_{C_i}$ for storing the probability values.
    for each $K_j$ in $K$ do
        compute probability $p(K_j|x)_{C_i}$ that the image $x$ belongs to
          class $K_j$ using classifier $C_i$:
        $p(K_j|x)_{C_i} = \frac{e^{P^T_{K_j} x}}{\sum_{K_l=0}^{L-1} e^{P^T_{K_l} x}}$ , where P is the parameter of the
          classifier $C_i$
        Append $p(K_j|x)_{C_i}$ to the array $A_{C_i}$
    end
end
create an array $A_C$ for storing the average probability values.
for each $K_j$ in $K$ do
    compute the average of probalities that $x$ belongs to $K_j$ in each
      calssifier;
    $p(K_j|x)= \frac{\sum_{C_i=0}^{N-1} p(K_j|x)_{C_i}}{N}$ , where $p(K_j|x)_{C_i} = A_{C_i}(j)$
    Append $p(K_j|x)$ to the array $A_C$
end
$y$=argmax($A_C$), where $A_C = [p(K_0|x), p(K_1|x), .....p(K_{L-1}|x)]$
Result: Class Label $y$
```

Let C1, C2, and C3 are the three base classifiers in an ensemble model, and A and B are the two class labels. For an image *i*, classifier C1 predicts the probability that *i* belongs to class A and B are 0.6 and 0.4 respectively. Similarly, classifier C2 predicts the probability that *i* belongs to class A and B are 0.3 and 0.7 respectively. And classifier C3 predicts the probability that *i* belongs to class A and B are 0.35 and 0.65 respectively. From the provided probability by each base classifiers, the average probability for each class is A= 0.42 and B = 0.58. The probability that the image *i* belongs to class B is higher, so B is the decision of the ensemble model.

Figure 6 illustrates the general working of the TL_Ensemble model.

The proposed ensemble model (TL_Ensemble) consists of two deep CNN models. A DenseNet201 model with all layers finetuned on the food dataset (TL_DNL) and An InceptionV3 model with all layers finetuned on the food dataset. The soft voting or average voting approach is implemented in the TL_Ensemble model to ensemble the base deep CNN models. The two deep CNN models included in the TL_Ensemble model are configured to run inference on the query image separately. In Fig. 6, P_{11}, P_{12}, … , P_{1N} represents the probability that the query image belongs to class 1, 2, … N respectively in the first classifier. Similarly P_{21}, P_{22}, … , P_{2N} indicates the probability output of the second classifier.

In the custom food dataset there exist 29 classes. For a query image *i*, the DenseNet201 model extracts the CNN features. And Computes the probability of the extracted features for belonging to each of the 29 classes. The softmax function added in the final layer of the DenseNet201 model is utilized for the probability calculation. Once the probability is calculated the DenseNet201 model generates an array with 29 cells. Each cell contains the numerical value of probability that the query image belongs to the respective class. I.e., the first cell of the resulted array contains the numerical value of probability that the query image belongs to the first class, the second cell contains the numerical value of probability that the query image belongs to the second class, and so on. Similarly, the InceptionV3 model extracts the features from the query image and computes the probability. The InceptionV3 model also generates an array with 29 cells where each cell contains the numerical

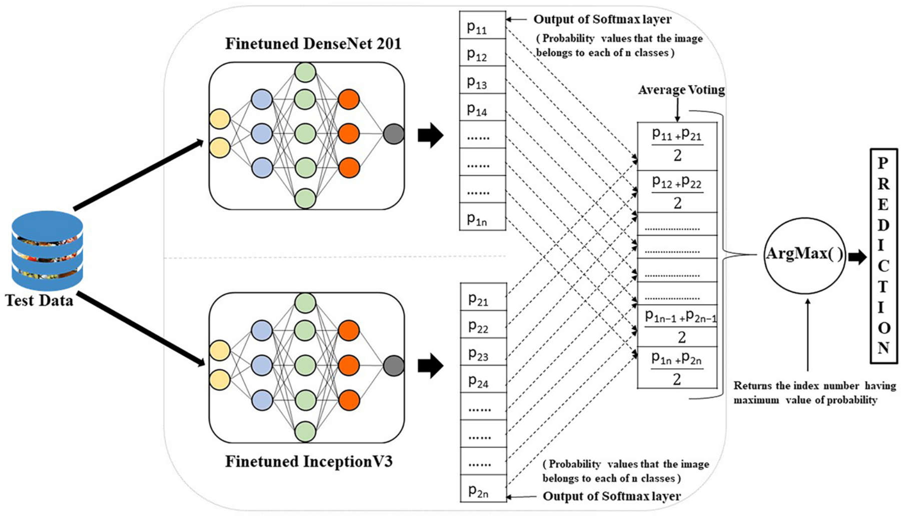

Fig. 6 Proposed ensemble model using average voting approach

value of probability. Once the inference on the query image is completed by DenseNet201 and InceptionV3 models, the average voting algorithm is employed to ensemble the results of two independent models. The average voting algorithm generates an array of 29 cells by averaging the respective cells of arrays generated by the DenseNet201 model and the InceptionV3 model. The array generated by the average voting algorithm provides the final average probability values for the query image. The array index with the maximum value is the final prediction of the TL_Ensemble model. Suppose in the final generated array, index 5 contains the maximum value. That means the query image *i* belongs to class 5.

4 Evaluation and results

The proposed deep CNN methods were evaluated in three different food recognition datasets. Training deep CNN models is computationally expensive and it requires a powerful workstation. In this work, a laptop with Intel Core i7 -6700 CPU @ 2.60GHZ processor, 24 GB RAM, and 6 GB Nvidia GTX-1060 GPU was used for training the deep CNN models. The training and implementation of the deep CNN models were carried out in python language. The deep CNN model was built using the ‘tf.Keras’ API of the TensorFlow library (version 2.0).

Model evaluation is a key task where various performance metrics of the model are analyzed to interpret its effectiveness for the respective problem. In the case of a multi-label classification problem, overall classification accuracy (Accuracy) is the most commonly used performance metric. Accuracy is defined as the ratio of the correct prediction to total observations in the test data. Other than accuracy, we considered the other three performance metrics such as precision, recall, and F1 score to analyze the performance of the classification model.

The accuracy of the classifier is calculated using the following Eq. (4), where True Positives (TP) are the number of positive observations that are correctly classified, True Negatives (TN) are the number of negative observations that are classified as negative, False Positives (FP) are the number of negative observation that are falsely classified as positive and False Negatives (FN) are the number of positive observations that are falsely classified as negative

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

Precision defines, how many of the observations are correct among the positively classified observations. The precision metric is calculated using the following Eq. (5), where TP is the number of true positives and FP is the number of false positives generated in the classifier.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall metric is equivalent to the accuracy of positive observations. It means how many positive observations are classified correctly. The recall is calculated using the following Eq. (6) where, TP is the number of true positives, and FP is the number of false negatives obtained in the classifier. Recall measures the sensitivity of the model.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

F1 is defined as the harmonic mean of recall and precision. The F1 score of the classifier is computed using the following Eq. (7).

$$F1score = 2 \times \frac{(Recall \times Precision)}{(Recall + Precision)} \tag{7}$$

4.1 Evaluation in custom food dataset (dataset 1)

The custom food dataset contains images of food dishes and fruits classified into 29 different classes. All the images are of dimension 224×224 pixels. The train, validation, and test splits are pre-arranged where the train set, validation set, and test set contains 21474, 2751, and 6902 images respectively.

The hyperparameters and input parameter used for training (fine-tuning) all the deep CNN models in the custom food data set are given in Table 3.

Data augmentation was applied to the training data using the image data generator tool available in the TensorFlow library. Data augmentation is applied to increase the diversity of the data and combat the overfitting problem. The data augmentation techniques applied to the training data are described in Table 4.

VGG16 model, Inceptionv3 model, and DenseNet201 model pre-trained on ImageNet dataset are used for finetuning. Five different CNN models as described in Table 2 are developed in this work. Moreover, an ensemble of the two-deep CNN models is also developed for improved food recognition. Before fine-tuning the upper layers of each CNN model, the newly added fully connected layer of each model is trained on the food dataset. In each CNN model, the fully connected layer is trained for 3 epochs by freezing all other upper layers. After 3 epochs the earlier layers of the model are finetuned on the food dataset. In the case of the VGG16 model, the last convolutional block is finetuned for 20 epochs. Similarly, the last two inception blocks of the inceptionV3 model and the last dense block of the DenseNet201 model are finetuned on the food dataset for 20 epochs separately. In the next two experiments, all the layers of InceptionV3 and DenseNet201 are trained on the food dataset for 20 epochs. Five different CNN training sessions are conducted for developing five deep CNN models. At the beginning of each training session, the initial learning rate was set to 1e-4. During training, the learning rate is reduced by a factor of 0.5 when the validation metric hit the plateau. The ensemble model was built by integrating the DenseNet201 deep CNN model with all layers finetuned and InceptionV3 deep CNN model with all layers finetuned.

Table 3 Paremeters for experiments

Parameters	value
Input size	224×224
Optimization algorithm	Adam
Initial learning rate	1e-4
Batch Size	16

Table 4 Applied data augmentation techniques

Technique	Value
Rotation	40
Width shift range	0.2
Height shift range	0.2
Shear Range	0.2
Zoom Range	0.2
Horizontal Flip	True

The accuracy, precision, recall, and F1 score obtained for each food recognition model trained on the custom food dataset are given in Fig. 7.

In the custom food dataset, the deep CNN model developed by finetuning the last convolutional block of pre-trained VGG16 (TL_VGG) achieved a classification accuracy of 89.66%. Among all 6 methods evaluated in the custom food dataset, the least value of classification accuracy was recorded in the TL_VGG model. It is expected that TL_VGG can't perform well as the other five deep CNN models. Except for VGG, other deep CNN models are having complex architecture. Moreover, they have already recorded better classification performance in the ImageNet dataset compared to VGG. Still, we used the VGG deep CNN model in our experiments because of its simple architecture and to analyze how a simple CNN model perform in food classification problem. Even though the accuracy recorded by TL_VGG is the least among all developed methods, still it is an acceptable value for food classification.

The deep CNN model developed by finetuning the last two inception blocks of the InceptionV3 model (TL_IV) performed better than the TL_VGG model and obtained a classification accuracy of 92.55 %. Compared to the TL_VGG model, the TL_IV model achieved a significant increment of about 3 % in classification accuracy. TL_IVF

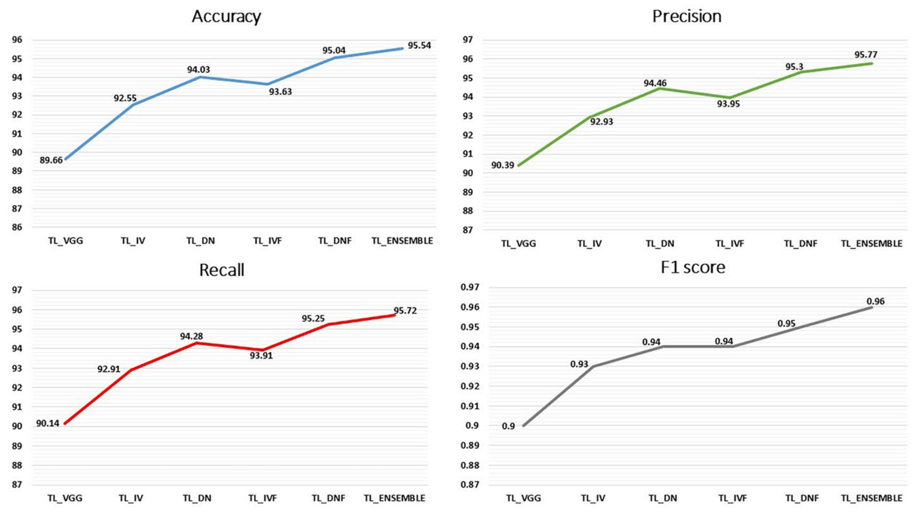

Fig. 7 Accuracy, Precision, Recall and F1 score obtained for developed deep CNN models in custom food dataset

is developed by finetuning all the layers of the InceptionV3 on the custom food dataset. The TL_IVF model surpassed the TL_IV model in terms of classification performance and achieved a classification accuracy of 93.63 %. Finetuning all layers of InceptionV3 improved the classification accuracy by more than 1 %.

The DenseNet deep CNN model (TL_DN) built by finetuning the last dense block of pre-trained DenseNet 201 displayed a better classification performance than InceptionV3 based deep CNN models (TL_IV and TL_IVF). The classification accuracy of the TL_DN model is 94.03 %, which is about 1.4 % more than the TL_IV model and 0.4 % more than the TL_IVF model. Even though DenseNet based models have 3 million fewer parameters than Inception based models, they delivered a better classification performance for food classification in custom food recognition dataset. The deep CNN model developed by finetuning all the layers of the DenseNet201 deep CNN model (TL_DNF) further improved the food classification accuracy. The TL_DNF model obtained a classification accuracy of 95.05 % in the custom food recognition dataset.

The proposed ensemble model (TL_Ensemble) developed by integrating multiple deep CNN models achieved the best accuracy among all other deep CNN models in the custom food recognition dataset. The TL_Ensemble model consists of TL_IVF and TL_DNF models. The predictions from TL_IVF and TL_DNF models are integrated by the average voting approach and the resultant TL_Ensemble model yielded a classification accuracy of 95.54 %. The recorded classification accuracy value of the TL_Ensemble model demonstrates its effectiveness and superiority over all other deep CNN models for food classification problems.

The accuracy metric is fair enough for analyzing the performance of classifiers in a multi-label classification problem unless the dataset is unsymmetric. The custom food dataset used for evaluation is neither symmetric but not highly unsymmetric. Each class contains 1000 to1200 images. Because of that the average precision, recall, and F1 score of each deep CNN classifier were computed. The TL_Ensemble model achieved the maximum value of precision and recall among all the deep CNN models. All of the deep CNN models achieved an F1 score equal to or more than 0.9. It displays the efficacy of all the developed deep CNN models for food classification. The TL_Ensemble model yielded the highest F1 score (0.96) which is quite near to the possible maximum value 1.

Table 5 presents the confusion matrix of the TL_Ensemble model (the one which outperformed all other models in the custom food dataset). The confusion matrix provides a more insightful analysis of the classification model. Each row and column of the confusion matrix corresponds to true class and predicted class respectively. The diagonal cells correspond to correctly classified observations and the off-diagonal cells correspond to incorrectly classified observations. In the confusion matrix, the numerical values 0 to 28 represent each class were, (0) Apple, (1)Banana, (2)Carambola, (3)Guava, (4)Kiwi, (5)Mango, (6)Orange, (7)Peach, (8)Pear, (9) Persimmon, (10)Pitaya, (11)Plum, (12) Pomegranate, (13) Tomatoes, (14) Cupcakes, (15) Donuts, (16)Falafel, (17) French fries, (18) Fried rice, (19) Hamburger, (20)Hot dog, (21)Hummus, (22) Ice cream, (23)Muskmelon, (24) Pancakes, (25)Pizza, (26) Samosa, (27)Tiramisu, and (28) Waffles.

4.2 Real time evaluation of ensemble model in smartphone devices

The response time of the system is one of the main metrics to be considered while developing an assistive system for people with visual impairments. Especially in the case of systems that use the deep CNN models for prediction, inference time, or prediction time of the

Table 5 Confusion matrix of TL Ensemble model evaluated in custom food dataset

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
0	**238**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	**222**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
2	0	0	**208**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	**240**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**234**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	**266**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	**232**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	**206**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	**218**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	**209**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	**221**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	**228**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	**239**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	**224**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**223**	4	2	0	0	0	1	1	7	0	0	0	1	4	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	**203**	3	0	1	2	1	0	0	0	0	0	2	2	1
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	**221**	2	2	3	7	8	1	0	0	1	3	3	2
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**221**	0	3	3	0	1	0	0	0	2	0	2
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	**239**	0	0	3	1	0	0	1	1	0	1
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	5	2	**205**	14	1	1	0	0	0	0	1	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	6	**240**	0	3	0	0	0	2	0	1
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	6	3	4	**226**	5	0	8	1	2	3	1
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	3	0	1	2	3	**236**	0	3	0	0	5	6
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**236**	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	1	1	1	**234**	0	1	6	4
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	2	3	4	0	0	2	**198**	0	1	2
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	3	0	1	1	1	1	2	0	1	2	**263**	1	1
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	1	0	2	7	0	1	0	1	**245**	1
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	0	0	0	3	1	0	2	0	6	2	0	2	**217**

deep CNN models in realtime are analyzed. Because most of the deep CNN models require a high computational and high memory environment to work well without any delay.

This section provides prediction time analyses of the TL_Ensemble model (Quantized and optimized model) in smartphone devices. The prediction time of a deep CNN model may vary according to the hardware conditions of the smartphone devices. Owing to that fact, in our experiment we used multiple smartphone devices having different hardware configurations to assess the prediction time of the TL_Ensemble model. Before loading the TL_Ensemble model to the smartphone, it is quantized and optimized using TFLite API. The proposed system is made to run on four smartphone devices for two minutes separately and the average prediction time of the TL_Ensemble model is recorded. Figure 8 displays the average prediction time of the quantized and optimized TL_Ensemble model in four

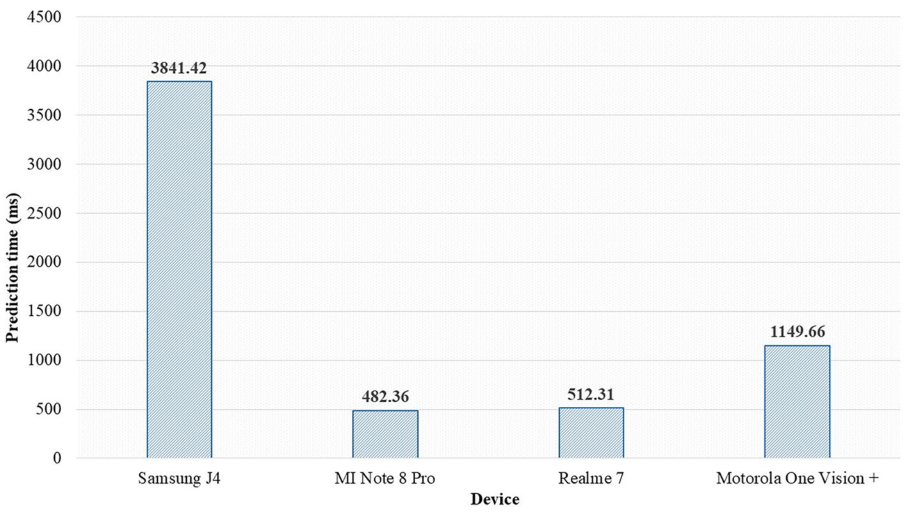

Fig. 8 Average prediction time of TL_Ensemble model in Smartphones

different smartphone devices. Samsung J4 (2 GB RAM), Redmi MI note8 Pro (6 GB RAM), Realme 7 (6 GB R^{am}), and Motorola One Vision Plus (4 GB RAM) are the four smartphone devices used for experiments.

Highest average prediction time (highest delay) is recorded in a smartphone device with 2 GB memory. Compared to the current market standard of smartphones, devices with 2 GB RAM are outdated. It is clear from the figure that the model performed well in smartphones having good hardware configurations. TL_Ensemble model achieved an average prediction time of 482.36 milliseconds in a device with 6 GB RAM. That means the system can recognize the food in less than half a second in real situations.

4.3 Evaluation of ensemble model in public food dataset

To prove the effectiveness and consistency of our ensemble deep CNN model for food recognition problems, the ensemble model (TL_Ensemble) is evaluated in two public food datasets. The first dataset contains large varieties of food dishes falling under different cuisines all over the world and the second data set contains varieties of food dishes commonly found in the middle east region. Overall food dish classification accuracy is considered as the metric for analyzing the performance of the TL_Ensemble model in this evaluation experiment.

The first public dataset used for evaluation is MAFood-121 [3], which contains images of 121 different varieties of food dishes. The dataset has a preassigned train, validation, and test splits. The same preassigned split is followed for training and evaluating the TL_ Ensemble deep CNN model. The dataset contains 21175 images, where 72.5 % of them are used for training the classifier, 12.5 % of total images are used for validation during training and the rest 15 % is used for testing the performance of the classifier. All layers of InceptionV3 (Pre-trained in ImageNet dataset) and all layers of DenseNet201 (Pre-trained in ImageNet dataset) are finetuned on the MAFood-121 dataset for 16 epochs and 15 epochs respectively. Before finetuning all the layers of both deep CNN models, the newly added fully connected layer of both deep CNN models is finetuned on the dataset for 3 epochs. To reduce overfitting, data augmentation techniques mentioned in table 3 are applied to the training data. The same input parameter and hyperparameter mentioned in table 4 are used for training the deep CNN models on the MAFood-121 dataset. The learning rate of the training is configured to reduce by a factor of 0.5 when the validation metric hit the plateau. After training the multiple deep CNN classifiers, the TL_Ensemble model is built by combining their output using the average voting approach. The classification accuracy of the TL_Ensemble model in the MAFood-121 dataset is given in Table 6 along with the accuracy of state of art methods evaluated in the same dataset.

.

Table 6 Performance comparison of TL_Enseble model in MAFood-121 dataset

Methods	Overall Accuracy
Single Task classifier [3]	82.50 %
RUMTL [3]	83.82 %
Flat classifier [4]	81.37 %
Local + Flat classifier [4]	81.62 %
TL_Ensemble	**84.95** %

Table 7 Performance comparison of TL_Enseble model in Food24 dataset

Methods	**Overall Accuracy**
Xception [19]	83 %
InceptionResNetV2 [19]	85 %
InceptionV3 [19]	88 %
TL_Ensemble	**89.40** %

It is clear from table 5 that the proposed ensemble model TL_Ensemble outperformed all the state of art methods in the MAFood-121 dataset for the food dish classification task. The single task classifier [3] is based on ResNet50 architecture and achieved 82.50% dish classification accuracy in the MAFood-121 dataset. The integration of local (classifier for each cuisine) and flat classifier (single dish classifier for 121 classes) proposed in [4] achieved a classification accuracy of 81.62 %.RUMTL [3] is a multi-task learning model developed for food dishes classification as well as cuisine and category classification. The food dish classification accuracy obtained by RUMTL in the MAFood-121 dataset is 83.82%, which was the maximum accuracy recorded earlier for dish classification tasks in the MAFood-121 dataset. Our TL_Ensemble model achieved a food dish classification accuracy of 84.95 %, which is 1 % more than the second-best method RUMTL. The obtained results indicate the effectiveness of our ensemble model for the classification of a large variety of food items.

The second public dataset used for the evaluation of the TL_Ensemble model is Food24 [19] which contains 24 different food dishes from Turkish cuisine. Food24 consists of about 11000 images, where 8600 images are preassigned for training the classifier and the rest of the images are for evaluating the performance of the classifier. The same train and test split are used for the training and testing of the TL_Ensemble model. 10 % of the training data has been utilized for validation tasks during training. The newly added fully connected layer of both InceptionV3 and DenseNet201 model is finetuned for 3 epochs followed by finetuning all the layers of InceptionV3 and DenseNet201 for 6 and 7 epochs respectively. The same input parameter and hyperparameters mentioned in table 3 are used for training the deep CNN models. The learning rate of training was reduced by a factor of 0.5 after each couple of epochs. The performance of the TL_Ensemble model in the Food24 dataset is given in Table 7 along with the performance of state of art methods evaluated in the same dataset.

In the Food24 dataset, the proposed TL_Ensemble model achieved a classification accuracy of 89.4 %. The TL_Ensemble model achieved better classification accuracy than all CNN models developed in [19] for food classification in the Turkish food dataset, Food24.

5 Conclusion

In this work, we develop a smartphone-based food recognition application for children with visual impairments. The proposed application utilizes a trained deep CNN model to recognize the food items. Moreover, we propose an ensemble model that integrates multiple deep CNN models using the soft voting approach for improved food recognition. The obtained food classification performance of the ensemble model in multiple food datasets shows its effectiveness for food classification problems. Furthermore, the ensemble model is employed in different smartphone devices for real-time prediction analysis. Devices with good hardware configuration can run the model without delay and provide real-time predictions. We contribute a food recognition dataset that contains both food dishes and fruit varieties for food classification problem. In the future, we will extend our smartphone application to support multiple platforms. Moreover, we will focus on the development of a deep CNN model for detecting and recognizing multiple food items concurrently.

Acknowledgements This publication was made possible by Qatar University collaborative grant number QUCG-CED-20/21-2 from the Qatar University. The findings achieved herein are solely the responsibility of the author.

Funding Open access funding provided by the Qatar National Library.

References

1. (2011) Vision impairment and blindness. https://www.who.int/newsroom/fact-sheets/detail/blindness-and-visualimpairment. Accessed 22 Jan 2021
2. (2020) Custom food dataset. https://www.dropbox.com/sh/irxb953mt9od181/AAD8QLXzuBZpdMdsKyFEWFwFa?dl=0. Accessed 22 Jan 2021
3. Aguilar E, Bolaños M, Radeva P (2019) Regularized uncertainty-based multi-task learning model for food analysis. J Vis Commun Image Represent 60:360–370
4. Aguilar E, Radeva P (2019) Food recognition by integrating local and flat classifiers. In: Morales A, Fierrez J, Sánchez JS, Ribeiro B (eds) Pattern Recognition and Image Analysis. pp. Springer International Publishing, Cham, pp 65–74
5. Anthimopoulos MM, Gianola L, Scarnato L, Diem P, Mougiakakou SG (2014) A food recognition system for diabetic patients based on an optimized bag-of-features model. IEEE J Biomed Health Inform 18(4):1261–1271
6. Bagwan SMR, Sankpal L (2015) Visualpal: A mobile app for object recognition for the visually impaired. In 2015 International Conference on Computer, Communication and Control (IC4). Indore, India, pp 1–6
7. Bashiri FS, LaRose E, Badger JC, D'Souza RM, Yu Z, Peissig P (2018) Object detection to assist visually impaired people: A deep neural network adventure. In Advances in Visual Computing. Springer International Publishing, Cham, pp 500–510
8. Bossard L, Guillaumin M, Van Gool L (2014) Food-101–mining discriminative components with random forests. In European conference on computer vision. Springer, Zurich, Switzerland, pp 446–461
9. Chincha R, Tian Y (2011) Finding objects for blind people based on surf features. In 2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW). Georgia, USA, pp 526–527
10. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition. Florida, USA pp 248–255
11. Farinella GM, Allegra D, Moltisanti M, Stanco F, Battiato S (2016) Retrieval and classification of food images. Comput Biol Med 77:23–39
12. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Hawaii, USA pp 4700–4708
13. Hussain I, He Q, Chen Z, Xie W (2018) Fruit recognition dataset (version v 1.0)[dataset]. zenodo

14. Kagaya H, Aizawa K, Ogawa M (2014) Food detection and recognition using convolutional neural network. In Proceedings of the 22nd ACM International Conference on Multimedia. Florida, USA pp 1085–1088
15. Karkar A, Al-Maadeed S (2018) Mobile assistive technologies for visual impaired users: A survey. In 2018 International Conference on Computer and Applications (ICCA). Beirut, Lebanon, pp 427–433
16. Karkar A, Kunhoth J, Al-Maadeed S (2020) A scene-to-speech mobile based application: Multiple trained models approach. In 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT). Doha, Qatar, pp 490–497
17. Karkar A, Puthren M, Al-Maadeed S (2018) A bilingual scene-to-speech mobile based application. In 2018 International Conference on Computer and Applications (ICCA). Beirut, Lebanon, pp 1–240
18. Kawano Y, Yanai K (2014) Foodcam-256: a large-scale real-time mobile food recognitionsystem employing high-dimensional features and compression of classifier weights. In Proceedings of the 22nd ACM international conference on Multimedia. Florida, USA, pp 761–762
19. Kayıkçı Ş, Başol Y, Dörter E (2019) Classification of turkish cuisine with deep learning on mobile platform. In 2019 4th International Conference on Computer Science and Engineering (UBMK). Samsun, Turkey, pp 1–5
20. Kong F, Tan J (2012) Dietcam: Automatic dietary assessment with mobile camera phones. Pervasive Mob Comput 8(1):147–163
21. Lanigan PE, Paulos AM, Williams AW, Rossi D, Narasimhan P (2006) Trinetra: Assistive technologies for grocery shopping for the blind. In ISWC. Georgia, USA, pp 147–148
22. Liu C, Cao Y, Luo Y, Chen G, Vokkarane V, Yunsheng M, Chen S, Hou P (2017) A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure. IEEE Trans Serv Comput 11(2):249–261
23. Martinel N, Foresti GL, Micheloni C (2018) Wide-slice residual networks for food recognition. In 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). Nevada, USA, pp 567–576
24. Matsuda Y, Hoashi H, Yanai K (2012) Recognition of multiple-food images by detecting candidate regions. In 2012 IEEE International Conference on Multimedia and Expo. Melbourne, Australia, pp 25–30
25. Meyers A, Johnston N, Rathod V, Korattikara A, Gorban A, Silberman N, Guadarrama S, Papandreou G, Huang J, Murphy KP (2015) Im2calories: towards an automated mobile vision food diary. In Proceedings of the IEEE International Conference on Computer Vision. Santiago, Chile, pp 1233–1241
26. Nakamura D, Takizawa H, Aoyagi M, Ezaki N, Mizuno S (2017) Smartphone-based escalator recognition for the visually impaired. Sensors 17(5):1057
27. Nicholson J, Kulyukin V, Coster D (2009) Shoptalk: independent blind shopping through verbal route directions and barcode scans. The Open Rehabilitation Journal 2(1):11–23
28. Pandey P, Deepthi A, Mandal B, Puhan NB (2017) Foodnet: Recognizing foods using ensemble of deep networks. IEEE Signal Process Lett 24(12):1758–1762
29. Pouladzadeh P, Shirmohammadi S, Bakirov A, Bulut A, Yassine A (2015) Cloud-based svm for food categorization. Multimed Tools Appl 74(14):5243–5260
30. Qiu J, Lo FPW, Sun Y, Wang S, Lo B (2019) Mining discriminative food regions for accurate food recognition. In BMVC. Cardiff, UK, pp 158–168
31. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition
32. Singh S, Choudhury S, Vishal K, Jawahar C (2014) Currency recognition on mobile phones. In 2014 22nd International Conference on Pattern Recognition. Stockholm, Sweden, pp 2661–2666
33. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, USA, pp 1–9
34. Tan M, Le Q (2019) EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning. PMLR, Long Beach, USA, pp 6105–6114
35. Yanai K, Kawano Y (2015) Food image recognition using deep convolutional network with pre-training and fine-tuning. In 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). Turin, Italy, pp 1–6
36. Yu F, Wang D, Shelhamer E, Darrell T (2018) Deep layer aggregation. In Proceedings of the IEEE conference on computer vision and pattern recognition. Salt Lake City, USA, pp 2403–2412