App to help blind people shopping in grocery stores

CS341 DESIGN PROJECT REPORT SUBMITTED

Submitted by

1 – Abhijit P.J.
30 – Joseph Davis
49 – Praful Kumar
53 – Sarthak Anil

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT ENGINEERING COLLEGE

THRISSUR - 680009

November 2020

Abstract

A blind person may find it difficult to recognise items or products in a grocery store. They have to rely on their natural instincts to identify products. This is a tedious endeavour; the aim of this study/design is to develop an application to help them. The complete system design of the application is achieved through the implementation of several concepts and features. The paper uses convolutional neural network for object detection. The training set is used to train the model while the validation set is used to test the accuracy of the model during the training process. Testing the accuracy of the model between the training process makes the model more efficient and accurate. The structure of this convolutional neural network is designed with an input layer followed by mid layers which are then followed by output layers. The number of neurons in the output layer depends on the number of types of object which the model can classify. Researchers around the world has come up with various CNNs. In this project, we have collected the various papers compared and made a design combining the advantages of the previous researchers and removing the disadvantages.

Introduction

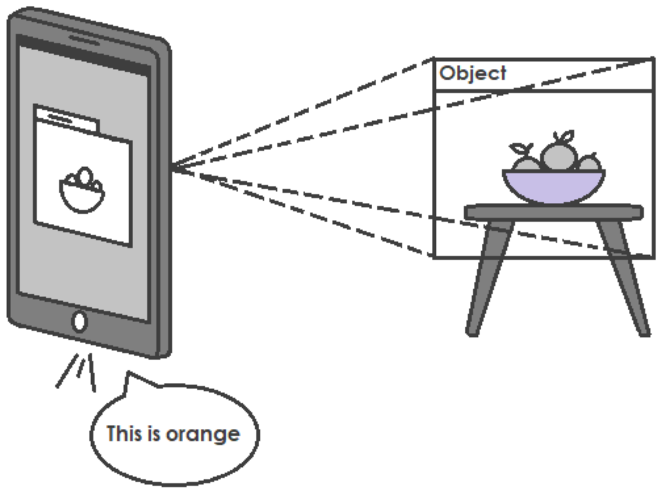
Blind people usually find it difficult to shop in grocery stores since they sometimes can't figure out items in the store. There are instances when they fail to recognize the items placed on the shelves of the store. We as a group came up with a solution to this problem.

Our solution is to develop a mobile application that helps blind ones to recognize the products in front of them using the app. After running it on any mobile device, the app utilizes the camera to scan the object placed in front of it. On completion of scanning, it recognizes the item and speaks out the result through the device speakers. The user, who is blind, will be able to hear it out loudly or can utilize headphones.

The application has a camera screen that lets the user tap on it. Once he/she taps, further processes are triggered and thereafter he/she will hear the result. It doesn't have any complex user interface. A simple tap on the screen is enough. This is to ensure the smooth operation of the app by the blind person since any sort of buttons might make the app more complicated for them to use.

In cases of the app failing to recognize any object, it simply prompts the user to consult a nearby person or the shopkeeper for help. It also collects the images which couldn't be recognized. These images are later utilized to improve the app's object recognition activity. Also, the application is good enough to recognize the items placed under poor lighting conditions.

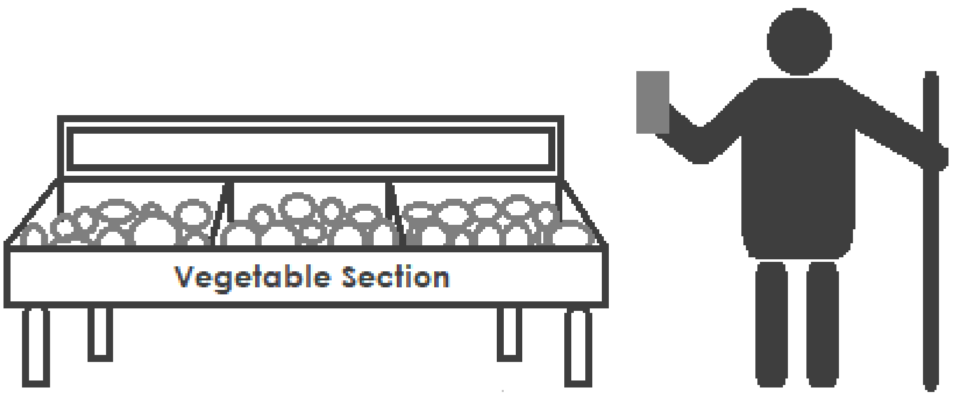
For example, a blind person visits a grocery store to buy tomatoes. Let us say he is standing in front of the vegetable section. He/she has to recognize tomatoes. We know that tomatoes are spherical. There are several other vegetables with are similar in shape. Hence it becomes more challenging for the blind person to recognize tomatoes from among other vegetables. This is where our app comes in handy.



He/she just needs to use the app and then point out the rear-camera of his/her mobile device towards the object. Then he/she have to tap the screen for the scanning and recognition process to begin. Once this is complete, the app will utilize the mobile speakers to speak out the message, "this is tomato". Otherwise, in cases of failure, the app will give a message, "Unable to recognize. Please consult nearby person for help".

There are chances of new items or products being placed in the stores. In that case, the app might not recognize the item in the first instance. It will rather store those images under the category of unrecognized images in the local devices. This data is collected and later it will be used to improve the model which the app utilizes for recognizing objects. The improved model will be pushed as a monthly update. There might be several users and therefore all those sets of data will increase the efficiency of the model and boost the scalability of the application.

The above picture shows the illustration of the example mentioned above. This gives a clear idea of the scenario where a blind person will utilize the application to recognize the object in a grocery store.



Literature Review

[1] **Max-Pooling Convolutional Neural Networks for Vision-based Hand Gesture Recognition** Focuses on the development of real-time hand gesture-based HRI interface for mobile robots. The author uses big and deep neural network (NN) combining convolution and max-pooling (MPCNN) for supervised feature learning and classification of hand gestures given by humans to mobile robots using coloured gloves. In this method the hand contour is retrieved by colour segmentation, then smoothened by morphological image processing which eliminates noisy edges. Their big and deep MPCNN classifies 6 gesture classes with 96% accuracy. Their implementation uses a vocabulary of 6 hand gestures it was stated by the author that the vocabulary can be extended to 11 classes using two hands, i.e. from finger count 0 to 10. The obtained results from their research show that vision-based gesture recognition system can be effectively employed for HRI, even for small and relatively not powerful robots, such as domestic mobile robots (e.g., Roomba and Scooba), and swarm robotic systems.

[2] **FAST IMAGE SCANNING WITH DEEP MAX-POOLING CONVOLUTIONAL NEURAL NETWORKS** points out the fact that even though Deep Neural Networks now excel at image classification, detection and segmentation, their high computational complexity can bring even the most powerful hardware to its knees. The author shows how dynamic programming can speed up the process by orders of magnitude, even when max-pooling layers are present. Their plain MATLAB implementation of the image-based approach handled the complications due to max-pooling layers interleaved with convolutional layers, avoiding unnecessary computations and greatly sped up forward-propagating deep neural networks on sliding windows. In conclusion their results show that the image-based implementation yields a dramatic speedup over patch-based approaches. In particular, MATLAB-image yields a 32-fold speedup when compared to the highly-optimized GPU-patch implementation, despite the former being implemented in a slower environment and without attention to low-level optimizations. Effectively a simple MATLAB implementation yields a 32-fold speedup over a highly optimized patch-based GPU implementation.

[3] **Assessment of Object Detection Using Deep Convolutional Neural Networks** aims to highlight the state-of-the-art approaches based on the deep convolutional neural networks especially designed for object detection from images. Deep convolutional neural networks have provided promising results for object detection by alleviating the need for human expertise for manually handcrafting the features for extraction. It allows the model to learn automatically by letting the neural network to be trained on large-scale image data using powerful and robust GPUs in a parallel way, thus, reducing training time. Several factors are responsible for proliferation for DCNNs viz. (i) Availability of large training datasets and fully annotated datasets (ii) Robust GPU to train large-scale neural network models in a parallel way (iii) State-of-the-art training strategies and regularization methods. Object detection is one of the crucial challenges in computer vision and it is efficiently handled by DCNN, Restricted Boltzmann Machine (RBM), autoencoders, and sparse coding representation. This paper aims to highlight state-of-the-art approaches for object detection based on the DCNNs.

[4] **Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks** present a hybrid DNN (HDNN), by dividing the maps of the last convolutional layer and the maxpooling layer of DNN into multiple blocks of variable receptive field sizes or max-pooling field sizes, to enable the HDNN to extract variable-scale features. The deep convolutional neural network (DNN), as a feature learning architecture, has yielded superior performance in many object recognitions tasks. The DNN uses the convolution layers and the max-pooling layers. The hidden layers and the output layer combine the extracted features for classification. This paper proposes HDNNs by dividing all maps of the highest convolutional and max-pooling layer of DNN into multiple blocks of variable receptive field sizes or max-pooling field sizes. The paper proves that HDNN is capable of extracting multiscale features. Experiments on vehicle database of the City of San Francisco given in the paper, show that HDNN outperforms DNN.

[5] **Classification of Fruits Using Computer Vision and a Multiclass Support Vector Machine** states the fact that automatic classification of fruits via computer vision is a complicated task due to the various properties of numerous types of fruits. To overcome this, the author proposes a novel classification method based on a multi-class kernel support vector machine (kSVM) with the desirable goal of accurate and fast classification of fruits. Firstly, fruit images were acquired by a digital camera, and then the background of each image was removed by a split-and-merge algorithm; Second, the colour histogram, texture and shape features of each fruit image were extracted to compose a feature space; Third, principal component analysis (PCA) was used to reduce the dimensions of feature space; Finally, three kinds of multi-class SVMs were constructed, i.e., Winner-Takes-All SVM, Max-Wins-Voting SVM, and Directed Acyclic Graph SVM. Meanwhile, three kinds of kernels were chosen, i.e., linear kernel, Homogeneous Polynomial kernel, and Gaussian Radial Basis kernel; finally, the SVMs were trained using 5-fold stratified cross validation with the reduced feature vectors as input. The experimental results demonstrated that the Max-Wins-Voting SVM with Gaussian Radial Basis kernel achieves the best classification accuracy of 88.2%. As for the computation time, the Directed Acyclic Graph SVMs performs swiftest.

[6] **Multi-class fruit detection based on image region selection and improved object proposals** proposes a novel approach for multi-class fruit detection using effective image region selection and improved object proposals. Five complementary features, namely local binary patterns (LBPs), histograms of oriented gradient (HOGs), LBP based on magnitude of Gabor feature (GaborLBP), global colour histograms, and global shape features, are utilized to improve the detection accuracy. An optimal combination of regions (i.e., features) is selected using an image region selection method based on feature similarity and cross-validation accuracy. To combine the strength of the five complementary features, a weighted score-level feature fusion approach based on the average confidence coefficient is used. During detection, an object proposal method, "EdgeBoxes," is improved by calibrating scores considering the image region similarity to generate windows that are likely to contain fruits and to speed up detection. The experimental results showed that the image region selection method can select an effective and optimal combination of regions, which exhibits better

recognition accuracy than the method without image region selection. This proposed method demonstrated a low miss rate (0.0377) at 0.0682 false positives per image (FPPI) and outperforms some baselines: multi-class fruit detection using the traditional sliding window mechanism, the well-known deformable parts model (DPM) method, convolutional neural networks features (CNN) with support vector machine (SVM) for classification (CNN + SVM), cascade detection framework and faster RCNN in terms of the miss rate vs. FPPI and precision vs. recall curves. The proposed multi-class fruit detection in this paper can detect multi-class and multiple fruits in a variety of sizes, backgrounds, angles, locations, and image conditions.

[7] **Vision-based rock-type classification of limestone using multi-class support vector machine** states the fact that rock-type classification is a challenging and difficult job due to the heterogeneous properties of rocks. In this paper, the author proposes an image-based rock-type analysis and classification method. Colour, morphology, and textural features were extracted from the captured image and a total of 189 features were recorded. The multi-class support vector machine (SVM) algorithm was then applied for rock-type classification. The hyper-parameters and the number of input features of the SVM model were selected by genetic algorithm. The results from the study revealed that the SVM model performed best when 40 features were selected out of the 189 extracted features. It also demonstrated that the overall accuracy of the proposed technique for rock type classification is 96.2 %. A comparative study at the end shows that the proposed SVM model performed better than a competing neural network model in this case study.

[8] **Classification and Grading Rice Using Multi-Class SVM** proposes machine algorithm to grade (Premium, Grade A, Grade B and Grade C) the rice kernels using Multi-Class SVM. Maximum Variance method was applied to extract the rice kernels from background, then, after the chalk has been extracted from rice. The percentage of Head rice, broken rice and Brewers in rice samples were determined using ten geometric features. Multi-Class SVM classified the rice kernel by examining the Shape, Chalkiness and Percentage of Broken (Head Rice, Broken and Brewers) kernels. The SVM classify accurately more than 86%. Based on the results, it was concluded that the system was enough to use for classifying and grading the different varieties of rice grains based on their interior and exterior quality

[9] In the paper **Real-time kiwifruit detection in orchard using deep learning on Android ™ smartphones for yield estimation** single shot multibox detector (SSD) with two lightweight backbones MobileNetV2 and InceptionV3 were employed to develop an Android APP named KiwiDetector for field kiwifruit detection. An 8-bit quantization method was used to compress model size and improve detection speed by quantizing weight tensor and activation function data of convolutional neural networks from 32-bit floating point to 8-bit integer. Detection test was performed on 100 selected kiwifruit field images with resolution of 3,968 × 2,976 pixels using the four models on a HUAWEI P20 smartphone. Results showed that MobileNetV2, quantized MobileNetV2, InceptionV3, and quantized InceptionV3 obtained true detected rate (TDR) of 90.8%, 89.7%, 87.6%, and 72.8%, respectively. The TDR of MobileNetV2 and quantized MobileNetV2 was generally consistent and higher than InceptionV3 and quantized InceptionV3. For processing an image on the smartphone,

MobileNetV2, quantized MobileNetV2, InceptionV3, and quantized InceptionV3 took about 163 ms, 103 ms, 1085 ms, and 685 ms with model sizes of 17.5 MB, 4.5 MB, 96.1 MB, and 24.1 MB, respectively. Quantized MobileNetV2 reached a significant TDR with the fastest detection speed and the smallest model size.

[10] **ActiVis: Mobile object detection and active guidance for people with visual impairments** project aims to deliver a mobile system that is able to guide a person with visual impairments towards a target object or area in an unknown indoor environment. For the development of this system object detection, mobile computing, action generation and human-computer interfacing to interpret the user's surroundings and present effective guidance directions is used. Their approach to direction generation uses a Partially Observable Markov Decision Process (POMDP) to track the system's state and output the optimal location to be investigated. This system includes an object detector and an audio-based guidance interface to provide a complete active search pipeline.

[11] **AI benchmark: Running deep neural networks on android smartphones** presents a study of the current state of deep learning in the Android ecosystem and describe available frameworks, programming models and the limitations of running AI on smartphones. This give an overview of the hardware acceleration resources available on four main mobile chipset platforms: Qualcomm, HiSilicon, MediaTek and Samsung. This paper also presents the real-world performance results of different mobile SoCs collected with AI Benchmark1 that are covering all main existing hardware configurations.

[12] The paper **FACE AND FACIAL EXPRESSIONS RECOGNITION FOR BLIND PEOPLE** how various algorithms are used to detect face, the user wears camera glasses, and system speaks the saved person's name when his face comes in view of the camera. The various algorithm used are Viola Jones for face detection, PCA (principal component analysis) in which they recognize an unknown test image by comparing it with the known training images stored in the database as well as give information regarding the person recognized. These techniques work well under robust conditions like complex background, different face positions.

[13] **MobileNets for flower classification using TensorFlow** state the fact that Google's inception-v3 model takes more time and space for classification with high accuracy. In this paper, they have shown experimental performance of MobileNets model on TensorFlow platform to retrain the flower category datasets, which can greatly minimize the time and space for flower classification compromising the accuracy slightly.

[14] **ImageNet Large Scale Visual Recognition Challenge** is a benchmark in object category classification and detection on hundreds of object categories and millions of images. The challenge has been run annually from 2010 to present, attracting participation from more than fifty institutions. This paper describes the creation of this benchmark dataset and the advances in object recognition that have been possible as a result. They discuss the challenges of collecting large-scale ground truth annotation, highlight key breakthroughs in categorical object recognition, provide a detailed analysis of the current state of the field of large-scale

image classification and object detection, and compare the state-of-the-art computer vision accuracy with human accuracy.

[15] The paper "**Hyperspectral fruit and vegetable classification using convolutional neural networks**" tells about classification for fruits and vegetables. the paper uses ImageNet- a pretrained neural network for the purpose. with slight modifications. An additional data compression layer has been added to be able to classify the hyperspectral images with the RGB pre-trained network. To isolate the benefit of increased spectral resolution for the classification, the same analysis was also performed with pseudo-RGB images calculated from the hyperspectral images. The results show that the hyperspectral image data increases the average classification accuracy from 88.15% to 92.23%. The approach can easily be extended to other applications.

[16] **Mobile Object Detection using TensorFlow Lite and Transfer Learning** evaluate the viability of using deep learning models for object detection in real-time video feeds on mobile devices in terms of object detection performance and inference delay as either an end-to-end system or feature extractor for existing algorithms. The results show a significant increase in object detection performance in comparison to existing algorithms with the use of transfer learning on neural networks adapted for mobile use.

[17] **TensorFlow: A Vegetable Classification System and Its Performance** focus on vegetable recognition using Deep Neural Network (DNN) as a method to visualize the knowledge in the vegetable production. This paper presents the performance of our vegetable classification using TensorFlow framework with 10 kinds of vegetables. This also propose a VegeCare tool using AI, which manages the growth of vegetable for farmers. The evaluation results show that the learning accuracy was more than 70%. The important point to note is that the performance of vegetable classification results of leafy vegetables was degraded.

[18] **Classification of vegetables using TensorFlow** is purposing the glimpse of the recognition of a particular vegetable. This is being implemented on the TensorFlow platform, which is making use of OpenCV as the main library database. Firstly, the given frame is converted into an image and differentiated into cubical parts from which the features are extracted, so to converged it into the data set. Having these values, the certain frame is categorized into one of the sets of images provided, at the conclusion side percentage-wise isolation of objects is done, and here the vegetables are being identified and corresponding action should be executed. Highlighting the uniqueness of usage of this idea, would result into involution of more prominent ways of segregation of vegetables in a food production industry as per the requirement, in the territory wherein the only agriculture holds the backbone of economy. The use of image processing techniques is of outstanding implication for the analysis of agricultural operations. Vegetable and vegetable classification is one of the major applications that can be utilized in the supermarket to automatically detect the kind of the vegetable or vegetable purchased by the customer and to generate the costs for it. A simple android app has been developed to carry out this task in this research.

System Design

A blind person may find it difficult to recognise items or products in a grocery store. They have to rely on their natural instincts to identify products. This is a tedious endeavour,the aim of this study/desing is to develop an application to help them.

The complete system design of the application is achieved through the implementation of several concepts and features discussed below.

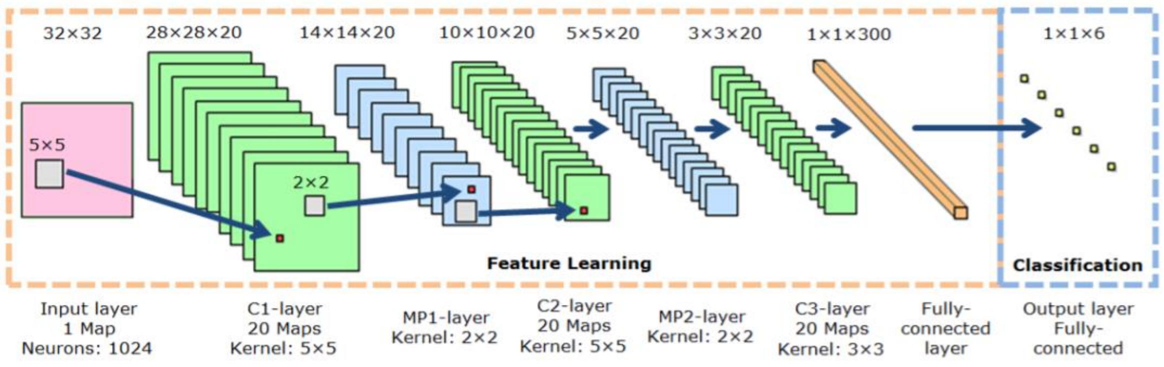
The app takes help from a model to recognize the images scanned through the camera. This model is designed using TensorFlow. Tensorflow is a core open-source library that helps to develop and train Deep-Learning models. The entire library is imported using the command "import tensorflow".

The model mentioned above is made using a convolutional neural network. It is made up of several layers of neurons each with an activation function assigned to it. This model has to undergo a training process before it can be utilized in the mobile device.

At the initial stages of the training process, several sample images of the objects are collected. These images are sorted and grouped into several categories based on the content of the image. For example, all the images containing mangoes are grouped. Similarly, images of other fruits, vegetables, and products are grouped. Once the grouping is complete, the images in each group are further divided into a training set and validation set in the ratio 9:1 respectively.

The training set is used to train the model while the validation set is used to test the accuracy of the model during the training process. Testing the accuracy of the model between the training process makes the model more efficient and accurate.

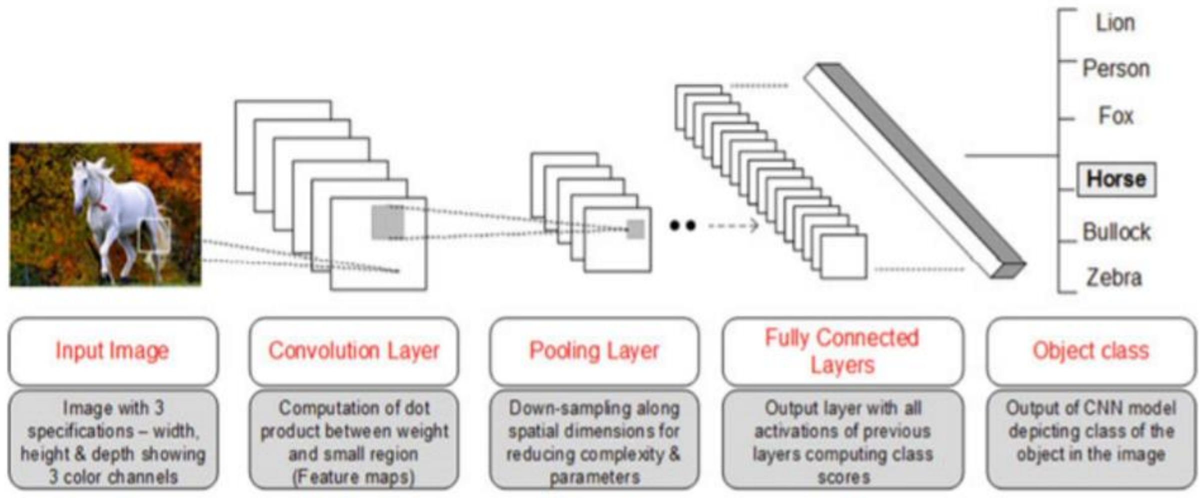
The structure of this convolutional neural network is designed with an input layer followed by mid layers which are then followed by output layers. The number of neurons in the output layer depends on the number of types of object which the model can classify. Also note, that the number of classifiable objects can be changed at the time of need. Each image is applied with sets of filters before it is broken down in the model training process. These filters help to highlight specific characteristics of each image. Also, a compression mechanism is applied to each image. This is called the Max-Pooling method. Then the entire neural structure is compiled.



[1]

There might be issues of overfitting, where the model becomes too good in identifying a particular object. This will make it less efficient when a new image is brought to it. To tackle this problem, each image is rotated, tilted, skewed, and mirrored. This generates several new images that are then utilized for model training.

[3]



Lion
Person
Fox
Horse
Bullock
Zebra
Input Image
Image with 3 specifications – width, height & depth showing 3 color channels
Convolution Layer
Computation of dot product between weight and small region (Feature maps)
Pooling Layer
Down-sampling along spatial dimensions for reducing complexity & parameters
Fully Connected Layers
Output layer with all activations of previous layers computing class scores
Object class
Output of CNN model depicting class of the object in the image

The model training is initiated using the “model.fit()” function is TensorFlow. Once the training is complete, the model is saved using the “model.save()” function. The saved model is in the “Keras” or “H5” file format. This model cannot be used directly in mobile devices because it requires high computation to run this kind of TensorFlow model. Execution of these models will increase latency and make the app look sloppy with frequent crashes. This is where TensorFlow-Lite comes in handy.

The model which was designed in TensorFlow is converted to TensorFlow-Lite using the convertor function. Thereafter, the Keras-model is changed to TFLite-model and the new file extension is “tflite”. These TFLite models are optimized for mobile devices with very acute computational power. Hence the app does not show latency and therefore it responds quickly. This completes the training process and the model is ready to be utilized.

The most common devices found are android and ios. Let's take the android as an example. The framework of the android application is designed in Android-Studio. There are several functional folders inside the studio each with a specific task. The resources for the app are placed under the ‘assets’ folder. This folder stores pictures, clips, templates, etc. that are later used in the app. The model we designed is placed in this folder.

The dependencies for TensorFlow-Lite are defined in the app code. This ensures error-free execution of the app on mobile devices. The app code consists of model loading commands and arguments. The complete code is executed and the ‘apk’ executable file is generated in the end.

The user points the camera towards any object and then he/she taps the screen. During this time a picture of the object is captured, and then it is passed into the model. The model predicts the name of the object and then returns a list of possible names along with their confidence scores. The confidence score is a probability value between 0 and 1. Here, we have defined a threshold value. Thereafter the names having values higher than the threshold are stored in the list while the rest are removed. Then the name having max value is taken out and passed into the 'text-to-speech' function.

If the model struggles to recognize an object, it will return names with lower confidence scores (probability value) than the threshold value. Here, we prompt a message, "Unable to recognize. Please consult nearby person for help" and then store this image in the local device database.

When the mobile has proper network connectivity, the images stored in the local device database is uploaded to the cloud server. When these arrive in the server they are sorted out using visual recognition algorithms. Then they are used to retrain the model which was used in those devices. The new model is tested for accuracy and then it is published and uploaded back to the cloud server. Thereafter it is pushed as a monthly update to all devices. This increases the scalability of the application because there might be several devices from which data are collected to retrain the model. This also improves the efficiency of the model.

Whenever a new product arrives at the store, the app might fail to recognize it. Therefore, it may store that scanned image in the local device database and later it will be uploaded to the cloud server. We then access those images to retrain the model. After the completion, a new model is created. This new model is distributed across all devices through a monthly update.

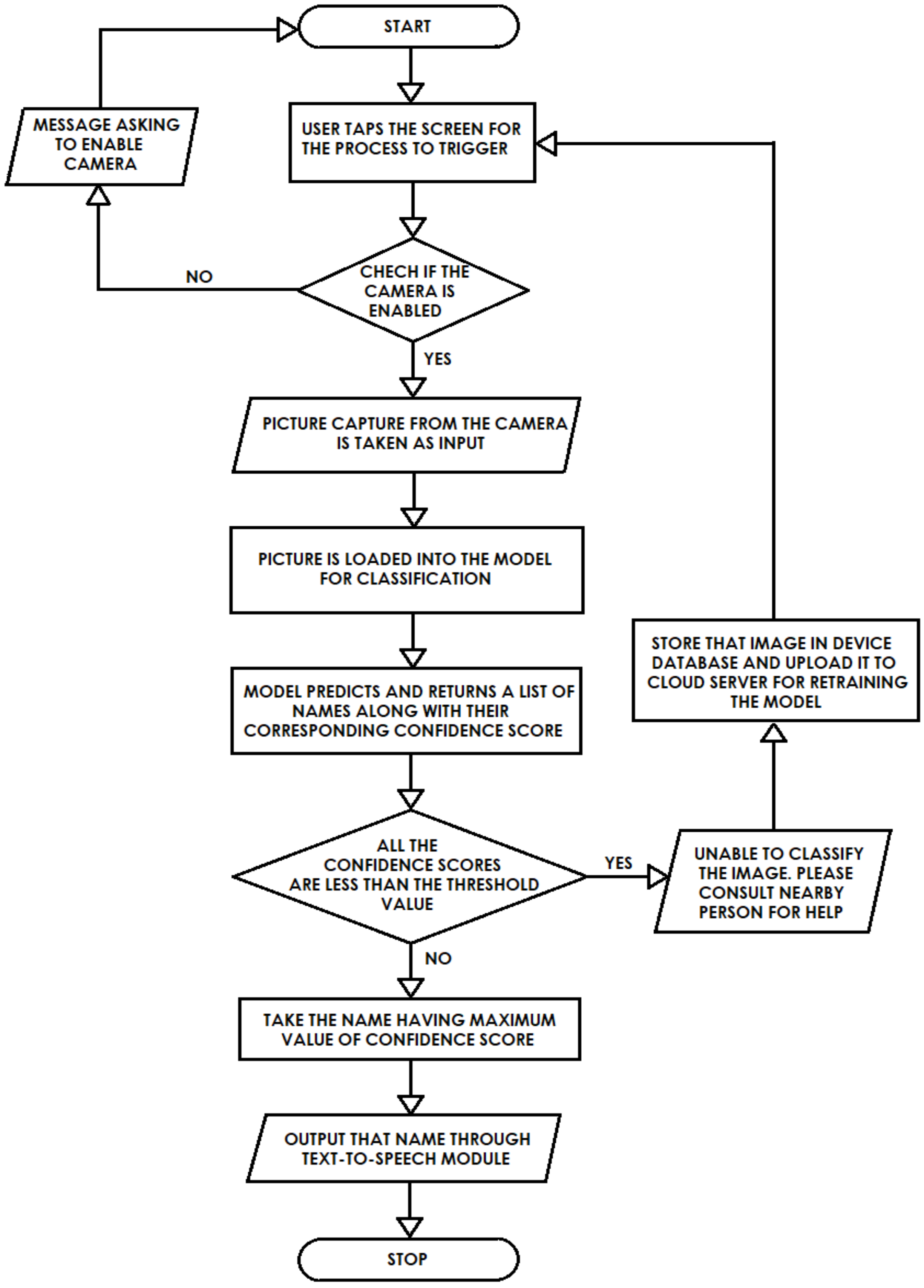
This was one of the challenges we faced during design development. But using the local database we overcame this problem. Another issue was when similar objects are placed side by side. At this time the model will predict a list of names along with their confidence scores respectively.

A concept called transfer learning is used for retraining the model. Whenever the set of new images arrive from the cloud, the current model is broken down into layers and then the last layer is selected. From this layer onwards, the new layers are created with a greater number of neurons. Then these newly developed layers are compiled into the model. It saves time by avoiding the training of the entire model from the starting layer. This is called transfer learning. Once the retraining is complete, the new model is generated and distributed.

The 'text-to-speech' module is provided in android studio. This module helps to convert a given text into an audio message. It accepts the input as its parameter and then does the conversion. The audio can be heard through device speakers. The user can hear it directly or use his/her headphones.

There are similar projects that are similar to some extent. For example, there is an app named 'KNFB' which reads printed text for the blind. Just the app and point it towards the document. Once the scanning is complete it reads the text for the user. A disadvantage of this app is that it not good enough to read handwritten texts. Therefore it can't be used for reading handwritten documents.

The detailed flowchart explaining the entire system is given below.


START
USER TAPS THE SCREEN FOR THE PROCESS TO TRIGGER
MESSAGE ASKING TO ENABLE CAMERA
CHECH IF THE CAMERA IS ENABLED
NO
YES
PICTURE CAPTURE FROM THE CAMERA IS TAKEN AS INPUT
PICTURE IS LOADED INTO THE MODEL FOR CLASSIFICATION
MODEL PREDICTS AND RETURNS A LIST OF NAMES ALONG WITH THEIR CORRESPONDING CONFIDENCE SCORE
ALL THE CONFIDENCE SCORES ARE LESS THAN THE THRESHOLD VALUE
YES
UNABLE TO CLASSIFY THE IMAGE. PLEASE CONSULT NEARBY PERSON FOR HELP
STORE THAT IMAGE IN DEVICE DATABASE AND UPLOAD IT TO CLOUD SERVER FOR RETRAINING THE MODEL
NO
TAKE THE NAME HAVING MAXIMUM VALUE OF CONFIDENCE SCORE
OUTPUT THAT NAME THROUGH TEXT-TO-SPEECH MODULE
STOP

Conclusion and future scope

This paper tries to find a way to help blind people in our community by making them more self-reliant while shopping at grocery stores. They usually need someone to help them pick out things in a bigger store or hypermarket. This is not an issue since there are dedicated people over there to help them out. But situations are different in small local stores. There might be a single attendee in the store, making it a difficult situation. Keeping this in mind we set out to design an application that can help them. The selected model training method is similarly used by many others and is proven to be reliable in our case too.

A model is trained from scratch using TensorFlow and is converted into TFLite models since TFLite models require less computation power. We have faced some changeless during our design. The main one being the task to incorporate all the fruits and vegetables and train a perfect ideal model. This requires a large raw data and computational power. To overcome this we implement a cache feature in the individual user device. Whenever the application fails to detect an item it is pushed into the cache and monthly this cache is flushed into the server. Afterward, all the images that were flushed into the server are categorized and segregated into clusters like we did for initial model training and use transfer learning to retrain the model, this solves both the problems as data is collected in chunks and computational power required is less as the data chunks are low in size compared to large data sets.

Overall, the design proves to be effective and the initial tests are done through a pre-trained model confirm this.

Regarding future scope, improvements can be made on model training and retraining. In this study, we used a convolutional neural network for our model training. Future studies can be made to check the performance and efficiency of the current model concerning other models like support vector machine (SVM). The detection of the image heavily depends on the camera module on the device on which the application is used, improvements can be made such that the application is to warn the user if there are any imperfections like a foggy, dirty, scratched lens. Also, improvements can be made for faster response and providing a more interactive experience for the user other than the robotic voice. More categories of grocery items can be added in future model training.

References

[1] J. Nagi *et al.*, “Max-pooling convolutional neural networks for vision-based hand gesture recognition,” in *2011 IEEE International Conference on Signal and Image Processing Applications, ICSIPA 2011*, 2011, pp. 342–347, doi: 10.1109/ICSIPA.2011.6144164.

[2] A. Giusti, D. C. Cireşan, J. Masci, L. M. Gambardella, and J. Schmidhuber, “Fast image scanning with deep max-pooling convolutional neural networks,” in *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings*, 2013, pp. 4034–4038, doi: 10.1109/ICIP.2013.6738831.

[3] A. R. Pathak, M. Pandey, S. Rautaray, and K. Pawar, "Assessment of object detection using deep convolutional neural networks," in *Advances in Intelligent Systems and Computing*, 2018, vol. 673, pp. 457–466, doi: 10.1007/978-981-10-7245-1_45.

[4] X. Chen, S. Xiang, C. L. Liu, and C. H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 10, pp. 1797–1801, 2014, doi: 10.1109/LGRS.2014.2309695.

[5] Y. Zhang and L. Wu, "Classification of Fruits Using Computer Vision and a Multiclass Support Vector Machine," *Sensors*, vol. 12, no. 9, pp. 12489–12505, Sep. 2012, doi: 10.3390/s120912489.

[6] H. Kuang, C. Liu, L. L. H. Chan, and H. Yan, "Multi-class fruit detection based on image region selection and improved object proposals," *Neurocomputing*, vol. 283, pp. 241–255, Mar. 2018, doi: 10.1016/j.neucom.2017.12.057.

[7] S. Chatterjee, "Vision-based rock-type classification of limestone using multi-class support vector machine," *Appl. Intell.*, vol. 39, no. 1, pp. 14–27, Jul. 2013, doi: 10.1007/s10489-012-0391-7.

[8] H. Kaur and B. Singh, "Classification and Grading Rice Using Multi-Class SVM," *Int. J. Sci. Res. Publ.*, vol. 3, no. 4, 2013, Accessed: Nov. 21, 2020. [Online]. Available: www.ijsrp.org.

[9] Z. Zhou, Z. Song, L. Fu, F. Gao, R. Li, and Y. Cui, "Real-time kiwifruit detection in orchard using deep learning on Android ^{TM} smartphones for yield estimation," *Comput. Electron. Agric.*, vol. 179, p. 105856, 2020, doi: 10.1016/j.compag.2020.105856.

[10] J. C. Lock, A. G. Tramontano, S. Ghidoni, and N. Bellotto, "ActiVis: Mobile object detection and active guidance for people with visual impairments," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Sep. 2019, vol. 11752 LNCS, pp. 649–660, doi: 10.1007/978-3-030-30645-8_59.

[11] A. Ignatov *et al.*, "AI benchmark: Running deep neural networks on android smartphones," *arXiv*. pp. 0–0, 2018, Accessed: Nov. 21, 2020. [Online]. Available: http://ai-benchmark.com.

[12] B. Vishwakarma, P. Dange, A. Chavan, A. Chavan, and H. Galiyal, "FACE AND FACIAL EXPRESSIONS RECOGNITION FOR BLIND PEOPLE," *Int. Res. J. Eng. Technol.*, 2017, Accessed: Nov. 21, 2020. [Online]. Available: www.irjet.net.

[13] N. R. Gavai, Y. A. Jakhade, S. A. Tribhuvan, and R. Bhattad, "MobileNets for flower classification using TensorFlow," in *2017 International Conference on Big Data, IoT and Data Science (BID)*, Dec. 2017, vol. 2018-Janua, pp. 154–158, doi: 10.1109/BID.2017.8336590.

[14] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015, doi: 10.1007/s11263-015-0816-y.

[15] J. Steinbrener, K. Posch, and R. Leitner, "Hyperspectral fruit and vegetable

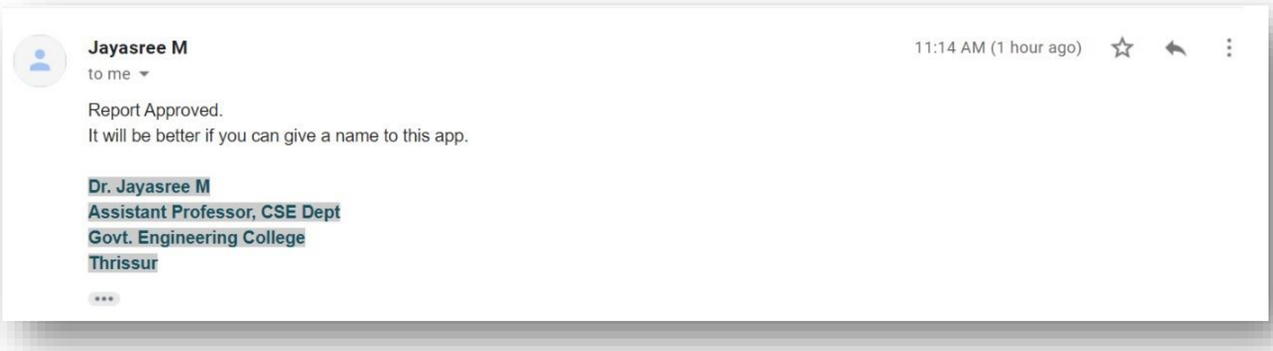
classification using convolutional neural networks," *Comput. Electron. Agric.*, vol. 162, pp. 364–372, Jul. 2019, doi: 10.1016/j.compag.2019.04.019.

[16] O. Alsing, "Mobile Object Detection using TensorFlow Lite and Transfer Learning," 2018. Accessed: Nov. 22, 2020. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-233775.

[17] N. Ruedeeniraman, M. Ikeda, and L. Barolli, "TensorFlow: A Vegetable Classification System and Its Performance Evaluation," in *Advances in Intelligent Systems and Computing*, Jul. 2020, vol. 994, pp. 132–141, doi: 10.1007/978-3-030-22263-5_13.

[18] O. Patil, "Classification of Vegetables using TensorFlow," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 6, no. 4, pp. 2926–2934, Apr. 2018, doi: 10.22214/ijraset.2018.4488.

Approval of mentor


Jayasree M
to me
11:14 AM (1 hour ago)

Report Approved.
It will be better if you can give a name to this app.

Dr. Jayasree M
Assistant Professor, CSE Dept
Govt. Engineering College
Thrissur

Approval of Coordinator