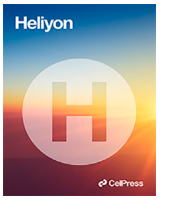
Heliyon 9 (2023) e16924

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon



Research article

Deep learning based object detection and surrounding environment description for visually impaired people

Raihan Bin Islam, Samiha Akhter, Faria Iqbal, Md. Saif Ur Rahman, Riasat Khan *

Electrical and Computer Engineering, North South University, Bangladesh

ARTICLE INFO

Keywords:
Assistive technologies
Machine learning
Mean average precision
Object detection
Random forest classifier
SSD MobileNet
Text-to-speech

ABSTRACT

Object detection, one of the most significant contributions of computer vision and machine learning, plays an immense role in identifying and locating objects in an image or a video. We recognize distinct objects and precisely get their information through object detection, such as their size, shape, and location. This paper developed a low-cost assistive system of obstacle detection and the surrounding environment depiction to help blind people using deep learning techniques. TensorFlow object detection API and SSDLite MobileNetV2 have been used to create the proposed object detection model. The pre-trained SSDLite MobileNetV2 model is trained on the COCO dataset, with almost 328,000 images of 90 different objects. The gradient particle swarm optimization (PSO) technique has been used in this work to optimize the final layers and their corresponding hyperparameters of the MobileNetV2 model. Next, we used the Google text-to-speech module, PyAudio, playsound, and speech recognition to generate the audio feedback of the detected objects. A Raspberry Pi camera captures real-time video where real-time object detection is done frame by frame with Raspberry Pi 4B microcontroller. The proposed device is integrated into a head cap, which will help visually impaired people to detect obstacles in their path, as it is more efficient than a traditional white cane. Apart from this detection model, we trained a secondary computer vision model and named it the "ambiance mode." In this mode, the last three convolutional layers of SSDLite MobileNetV2 are trained through transfer learning on a weather dataset. The dataset comprises around 500 images from four classes: cloudy, rainy, foggy, and sunrise. In this mode, the proposed system will narrate the surrounding scene elaborately, almost like a human describing a landscape or a beautiful sunset to a visually impaired person. The performance of the object detection and ambiance description modes are tested and evaluated in a desktop computer and Raspberry Pi embedded system. Detection accuracy and mean average precision, frame rate, confusion matrix, and ROC curve measure the model's accuracy on both setups. This low-cost proposed system is believed to help visually impaired people in their day-to-day life.

* Corresponding author.
E-mail addresses: raihan.islam@northsouth.edu (R.B. Islam), samiha.akhter@northsouth.edu (S. Akhter), faria.iqbal06@northsouth.edu (F. Iqbal), saif.rahman162@northsouth.edu (M. Saif Ur Rahman), riasat.khan@northsouth.edu (R. Khan).

https://doi.org/10.1016/j.heliyon.2023.e16924
Received 15 November 2022; Received in revised form 31 May 2023; Accepted 1 June 2023

Available online 7 June 2023

1. Introduction

Visually impaired or blind people survive their life in extreme hardness with significantly reduced vision. Some of them depend on various assistive technologies, such as braille displays, augmented reality-based intelligent glasses, and screen reader software [1]. According to the estimation of WHO, more than 2.25 billion people worldwide and 0.8 million population of Bangladesh are suffering from this disease [2]. This number is alarmingly estimated to be doubled by 2022 and tripled by the next 30 years. In a crowded country in Asia or Africa, it becomes quite difficult for visually impaired people to travel from one place to another by just using a traditional white cane. The white cane provides some essential navigational assistance to them by inspecting possible obstacles. But this device lacks the ability to help blind people to anticipate their surroundings with better precision [3]. This challenging scenario has inspired us to build an electronic device with object detection and audible feedback to make visually impaired people more independent. This project aims to better aid visually impaired people in assisting during navigation and visualizing the surrounding environment.

In recent years, active research has been done to implement intelligent systems for visually impaired persons' safe and autonomous movement. These assistive techniques are based on artificial intelligence techniques and advanced smartphone application devices. For instance, in [4], the authors introduced an intelligent eye Android mobile application that can detect light, color, various objects, and banknotes for visually impaired people. For light detection, the authors used the embedded light system of a smartphone. They utilized the OpenCV library to distinguish different colors, and they used a locally stored database in the mobile phone for object detection. They employed Android Studio to code in Java to develop their proposed mobile application, CraftAR toolbox for image recognition, and CNNdroid, an open-source library for running CNN models on Android devices. A built-in voice engine was used to read the detection results aloud so the user could hear them. A physical movement and kitchen activity detection system using machine learning approaches was proposed by Bhatlawande et al. [5] to assist blind individuals. Computer vision-based scale-invariant feature transform has been applied to extract attributes of the used dataset. Random forest model with optimized hyperparameters achieved the best performance with 0.808 accuracy. Mukhiddinov and his team [6] employed an improved version of the YOLOv4 deep learning technique for fruit and vegetable freshness identification. The system can classify five categories of fruits and vegetables, which has been developed employing a diverse dataset from various sources and lighting conditions. The upgraded YOLOv4 model achieved a 69.2% AP50 for the test samples. Finally, a smartphone application has been designed to instantaneously detect rotten fruits and vegetables.

Most assistive devices consist of different hardware components with embedded circuits and sensors. An assistive hardware tool was proposed in [7], where the authors built a simple portable device using the Raspberry Pi and TensorFlow object detection API to help blind people have an understanding of their surroundings. The proposed system consists of eyeglasses that work as a base for all the hardware components and can provide audio feedback to its user through a headphone in real time. In this work, Raspberry Pi 3B+ has been used as the major processing unit. A Raspberry Pi camera was used to stream the live feed, and TensorFlow object detection API was employed to detect objects. SSDLite MobileNetV2 was used as the detection model in this specific system, and eSpeak was integrated into the audible feedback network. Even though this system provides some promising features, it still lacks better quality, such as reading the objects aloud to its users. Nishajith et al. [8] implemented a smart wearable cap utilizing Raspberry Pi 3, NoIR camera, earphones, and power supply. The proposed device takes a picture of the object with the NoIR camera to detect the object with the help of the Google Brain Team using TensorFlow. It identified various objects from 90 different classes. Earphones were employed to describe the object using voice output with the help of a text-to-speech synthesizer named eSpeak. It takes the details of the object from the video captured and converts them into speech signals. The authors used the SSD MobileNetV1 COCO model and got an mAP score of 37, but the speed of their proposed model was slow. In [9], the authors designed a lightweight, low-cost, travel-friendly, and hands-free electronic navigation assistance prototype for blind people. Raspberry Pi Zero W was used as the utility device and linked it to the Google Cloud Vision API for remote image processing and audio output on a speaker or headphone. This entire arrangement was installed on a pair of spectacles and connected to a Raspberry Pi camera in the front, which was activated by the user pressing an initiating button. The camera will capture the real-time images and then process them remotely using the RES Google Vision API and the user's mobile phone's Wi-Fi. Finally, it will notify the voice output through audio linked via Bluetooth. By using this proposed device, the elderly and visually impaired people are able to distinguish between places, logos, lettering on the front, facial expressions, and so on. The authors utilized JSON and Python for reporting and presentation of this work. Some components of the proposed system are missing, such as the ability to deliver real-time aural feedback without the need for a button. In [10], authors designed an intelligent smart glass device to identify public signs in the outdoor environment. Initially, an HD camera attached to the front glass captures the video stream in real time. Next, this recording is passed to the Intel Edison lightweight module. Finally, these images are processed by the recognition technique of the OpenCV library, i.e., SIFT and SURF. The battery and Intel Edison module are placed on the two sides of the eyeglass. A vision assistance system for blind people has been proposed in [11] utilizing wearable sensors. The captured images will be sent to the cloud for classification. Finally, the CNN technique, ResNet, is used to process the images and identify various objects. Kumar et al. [12] implemented a smart sensor-based device to assist blind people's movements. This gadget employed pre-trained deep learning models for object detection and person tracking. Bouteraa [13] proposed a fuzzy logic-based smart gadget to aid with navigating situations. Raspberry Pi with a distance measuring device and IR sensor has been used. This device has been tested indoor and outdoor environments with different types of obstacles. Xie and his colleagues [14] employed the YOLO object detection model and RGBD depth sensing device for navigation support of blind people. The authors reported a 0.55 mAP coefficient and an excellent frame rate of 35 seconds. Mukhiddinov and Cho [15] developed an intelligent smart glass to detect real-time objects for blind people. The authors employed the open-source Low-Light dataset and a deep learning model established on transformer technique to detect various objects. A U-shaped ResNet architecture has been

employed for salient object recognition. Multiple sensing devices, e.g., GPS and ultrasonic sensors, have been used. The implemented object detection model accomplished 86.3% a.nd 81.7% precision and recall, respectively. Masud et al. [16] designed an intelligent walking stick for navigation barrier detection and classification employing Raspberry Pi 4B and various sensing devices. The authors used Viola-Jones and AdaBoost ensemble classifiers for object detection. Xia and others [17] developed a smart device for real-time object detection and traffic light classification to aid visual people navigation. The YOLOv3 Tiny model had been employed in the low-cost GD32F103VET6 MCU, incorporating a GPS, Wi-Fi and speaker module. The device accomplished 0.97 mAP for object detection and 95.52% accuracy for traffic light classification. Ashiq and his colleagues [18] implemented an object identification and tracking tool. The MobileNet framework was applied on a Raspberry Pi 3B embedded device. The applied model achieved 83.3% accuracy for the object detection task. Kumar and Jain [19] employed the YOLOv3 deep learning architecture for object detection and safe navigation of visually impaired persons. The authors reported 0.81 detection accuracy when the YOLOv3 model was deployed on a Raspberry Pi 3B integrated walking cane. Chang and team [20] devised an intelligent crosswalk recognition and smart navigation glass using an AI Edge MCI, camera and laser-ranging modules. The applied Inception SSD model detected the zebra crossing with approximately 92% accuracy and 75 ms mean detection time. In [21], the authors initiated a crosswalk detection and navigation system with a Raspberry Pi 4. The applied OpenCV-based model performed convincingly with a detection accuracy of 84.5%. Khalid and other researchers [22] introduced a traffic light detection and classification smart cap. The smart cap was utilized in a Raspberry Pi microcontroller. The OpenCV-based model showed 87.5% detection accuracy for traffic signal classification. Manjari et al. [23] developed an animal classification system employing the Jetson Nano edge device and ZED stereo camera. The authors collected 5,000 instances of six Asian animals. The SSD ResNet-50 approach obtained an mAP of 93.5% and 79.5% F1 score in classifying the detected animals. Bala et al. employed [24] the OpenCV model in the Raspberry Pi device for object detection and navigation assistance of blind populations. The designed cost-efficient device can detect various objects instantaneously.

After reviewing the related articles, we can conclude that significant efforts have been made to provide visually impaired individuals with object detection and navigation assistance. Various computer vision, deep learning techniques and advanced sensing devices have been employed in these works. Few of the articles developed a cost-efficient system using sophisticated deep learning techniques. Most of these works did not perform a comprehensive analysis of the implemented device demonstrating mean average precision (mAP), precision, recall, detection accuracy, AUC, user survey, cost of the device, etc. However, there is scope for improvement in developing an automatic system for surrounding environment descriptions with embedded systems.

This paper implements an automatic object detection and surrounding environment description system for visually impaired and older people. The major contribution of this work is as follows:

- An automatic object detection and ambiance narrative system is developed by applying the SSDLite MobileNetV2 technique and utilizing the MS COCO and an open-source weather dataset, respectively.
- The final layers of the MobileNetV2 model and their corresponding hyperparameters have been optimized using the particle swarm optimization (PSO) technique.
- The PSO-MobileNetV2 automatic deep learning-based navigation assistive systems are deployed in a cost-efficient hardware setup with Raspberry Pi 4 and Pi camera. Google text-to-speech module, PyAudio, playsound, and speech recognition are utilized to generate the audio output of the detected objects and surrounding atmosphere narration. Several pilot experiments have been conducted with the developed prototype to assess its functionality.
- The performance of the proposed system is evaluated in terms of mean average precision, frame rate, accuracy, user feedback ratings, etc. Finally, the implemented device's features and cost are compared to other existing works.

To the best of our knowledge, integrated PSO-MobileNet-based sophisticated deep learning techniques for object detection and surrounding environment description are implemented on an advanced embedded device, i.e., Raspberry Pi 4, for the first time in this work.

Section 2 discusses the proposed system's hardware and software design with related tables and figures. Section 3 depicts the results and operating performance of the implemented assistance tool with a brief discussion. Finally, section 4 concludes this research paper with some directions and scope for improvement in future work.

2. Proposed system

In this work, an intelligent approach has been implemented to help blind people navigate in the environment independently and get a sense of what is happening around them. This paper executes two modes: object detection and surrounding environment description. Voice assistance is used to notify its users about the object and the surrounding atmosphere.

2.1. Object detection

This paper uses the open-source MS COCO (Microsoft Common Objects in Context) dataset to construct the object detection model. The dataset contains 165,000 labeled images of 90 classes [25]. Interestingly, the MS COCO dataset has annotations for object detection, captioning, keypoint detection, stuff image segmentation, panoptic, and dense pose. In this work, the dataset has been used for object detection purposes only.

Fig. 1 shows some sample images from the MS COCO dataset. Next, we used the deep neural network-based Single Shot MultiBox Detector (SSD) [26] for the object detection model. This specific model is chosen because it can detect objects in real time and has

Fig. 1. Sample images from MS COCO Dataset.

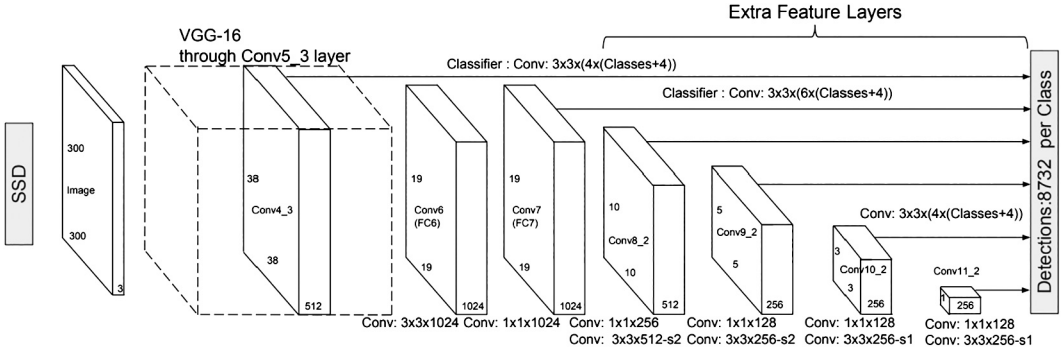


Fig. 2. SSD network architecture [26].

higher frames-per-second processing properties. SSD is an object detector that gives bounding boxes on an image or a real-time video and predicts a class. Single Shot MultiBox Detector is easily trained and can easily be integrated into a system to work as an object detector. It was trained in various datasets, including MS COCO, and provides a good mAP score [26]. The concepts of the Single Shot MultiBox Detector model are based on the following properties:

Feature Map Extraction: Initially, the featured maps are extracted by the VGG-16 network.

Single Shot: A single forward pass is done in the network to localize and classify the object.

MultiBox: The bounding boxes formed while detecting the image are mainly the MultiBox proposed by Szegedy [26].

Detector: Since the network works as an object detector, it focuses on detecting the objects and classifies them.

The basic architecture of the Single Shot MultiBox Detector has been depicted in Fig. 2. In this work, VGG-16 [27] has been used as the base architecture for the characteristic derivation of the SSD network. The fully connected layers are removed, and convolutional layers are added. Because of this activity, multiple features can be extracted efficiently. In this work, categorical cross-entropy is used to measure the loss of the SSD technique. It mainly measures the confidence of the network with respect to the objectness of the bounding box. The L2 norm function is used as the location loss in this work. It measures the difference in the distance between the predicted bounding box and the ground truth bounding box of the training set. The multibox loss is obtained from confidence and location losses as:

$$multibox\_loss = confidence\_loss + \alpha \times location\_loss \tag{1}$$

In (1), α denotes the balancing factor of location loss. There are different variations of SSD, and we used MobileNetV2 [28], also known as SSDLite, in this research for the SSD layer. SSDLite contains 1,917 boxes; it has various grids from 19×19 cells to 1×1 cells. Each cell contains three bounding boxes for the extensive grids, and for the others, it includes six boxes.

Fig. 3 demonstrates the architecture of the MobileNetV2 network. Since we are implementing software-based object detection in an embedded system, we are focusing on the model that works efficiently when implemented in the hardware device. Finally, we used TensorFlow object detection API [29] to run the proposed model.

After recognizing various objects, with the help of different speech recognition modules, Google text-to-speech (gTTS) [30], playsound, and PyAudio [31], we generated audio output for each class of the MS COCO dataset. The real-time video captures images frame by frame with approximately 15 seconds of delay, which is passed into the model. After the images are detected, the

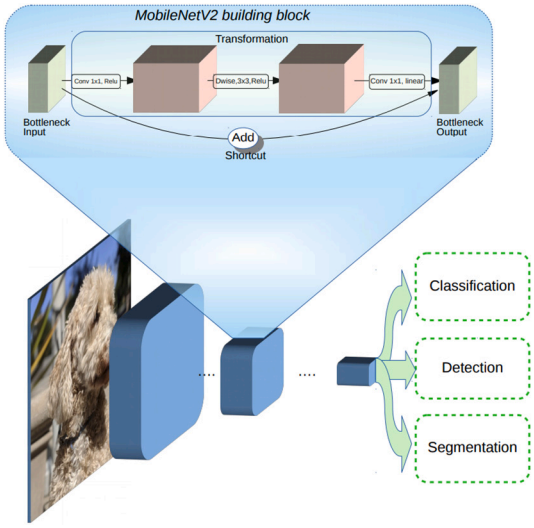


Fig. 3. MobileNetV2 architecture [28].

Fig. 4. Sample images from weather dataset of Kaggle.

specific audio file of the image class is considered, and the audio file is played accordingly. It is interesting to note that, the audio is played in a different thread simultaneously with the object detection model.

Frequent audio feedback creates discomfort for its user when the same object is detected repeatedly. To resolve this issue, we have adopted two steps in this work. First, we added a delay of 500 ms between each audio feedback, i.e., when two distinct objects are being detected, the system will take a moment (500 ms) before informing its user about the second object. The second strategy we have followed is creating a very simple algorithm to determine the previously detected object. If the detected objects in the next few frames are the same as the previous one, we skip creating the same audio feedback.

2.2. Ambiance mode

A significant contribution of this work is to implement the ambiance mode, a computer vision-based model trained on a different dataset than the object detection module we used in our project. As the name suggests, the ambiance mode aims to explain the feeling or mood associated with a particular place, person, or thing. This mode provides its user with a travel companion as well as a narrator. The sole purpose of the ambiance mode is to improve the quality of life of blind people by generating a scenic narrative of their surrounding environment.

The user of the proposed device needs to activate the mode through a voice command at the beginning. Once activated, it observes and detects the surrounding scenes as tags. These labels help the designed model to get an idea of what is happening in the surrounding environment and generate an audio speech based on that. Finally, the corresponding audio speech is narrated to its user through voice signals.

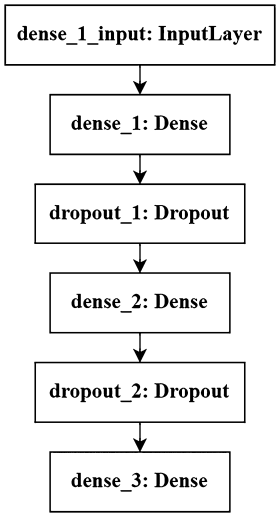


Fig. 5. Basic sequential model [32].

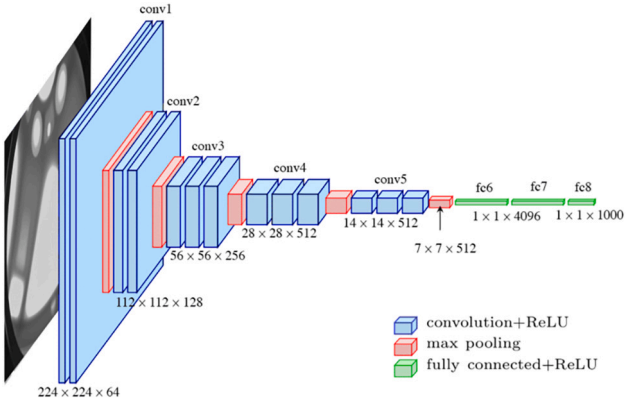


Fig. 6. VGG-16 architecture [27].

To implement the concept of the ambiance mode, this model has been trained on various scenes from five distinct weather classes. The classes are sunny, cloudy, foggy, sunrise, and rainy. For this training, a dataset of 500 images is utilized from each category, summing up to a total of 2,500 images. Fig. 4 shows some sample images from the obtained weather dataset of Kaggle. We have used 85 percent of the dataset for training and the remaining 15 percent for validation. It is worth mentioning that, the proposed ambiance mode system is initially implemented in a desktop PC, and then it is deployed in the embedded device. In the subsequent paragraphs, the implementation details are discussed individually.

For the personal computer, the open-source weather dataset was trained in the Keras Sequential model, which used VGG-16 and random forest [33] for surrounding environment depiction. The sequential model is made up of various linear layers. The layers are linearly connected so that the output of one layer leads to the next layer.

The architecture of the basic Keras sequential model is demonstrated in Fig. 5. VGG-16 is a convolution neural network approach that has been used for objection detection in ambiance mode. It has an arrangement of convolution and max pooling layers in the entire architecture. It has a convolution layer of 3×3 filter with a stride one and a max pool layer of a 2×2 filter with a stride 2. At the end of this architecture, it has two fully connected layers. The name VGG-16 refers to the number of layers with weights is 16 in this architecture.

Fig. 6 depicts the architecture of the VGG-16 model. We used VGG-16 as a feature extractor. Once the features are extracted, we pass the feature-extracted images to a random forest classifier. It is a classification algorithm that consists of decision trees. The output is established based on the predictions of the decision trees. The mean output of the trees is considered to make the final prediction. This process is efficient because it minimizes overfitting and increases precision at the same time. The working principles of the random forest classifier are depicted in Fig. 7.

For the ambiance mode implementation in the embedded system, we had to train the model in MobileNetV2, which gave us better results. The same VGG-16 architecture has not been used for Raspberry Pi because it caused errors while transferring the model into TFlite. MobileNetV2 worked perfectly fine in doing so; as a result, it has been chosen for the embedded implementation.

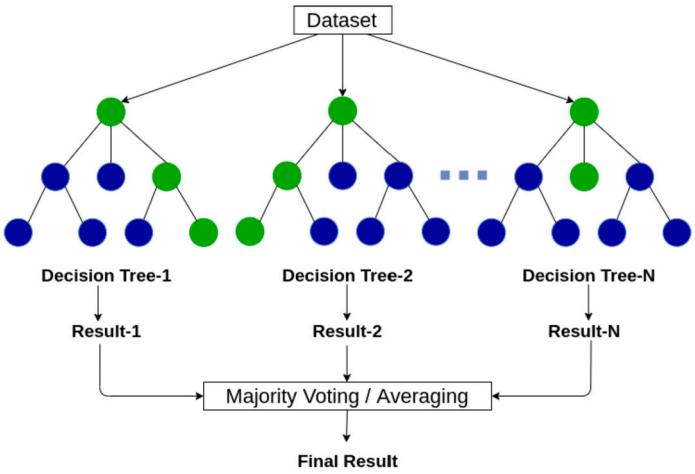


Fig. 7. Random forest architecture [34].

Table 1
Data augmentation techniques with parameters values.

Augmentation parameter	Value
rotation_range	90
brightness_range	[0.30, 0.60]
width_shift_range	0.50
height_shif_range	0.50
flipping	horizontal and vertical

As the obtained dataset is relatively small in size, the transfer learning technique is implemented instead of building our own model from scratch. In transfer learning, the weights of pre-trained deep learning models are used to facilitate the training process. The basic concept of transfer learning is to use the previously trained network architecture and its weights to speed up the training process as well as maintain good classification accuracy. Keras provides a wide range of pre-trained models that are trained on large-scale datasets such as "ImageNet." These pre-trained models already know a lot of features regarding the newly introduced dataset, which is much smaller in size than the actual dataset it was trained on. That's why transfer learning works better than a newly built model on a unique and smaller dataset.

As stated above, Keras provides many pre-trained models, but we had to choose the one that would fit perfectly in our embedded system, the Raspberry Pi 4 device. Finally, we selected the MobileNetV2 architecture as the base model for the ambiance mode. This specific lightweight model is chosen because it uses minimal resources and has a good mAP score and fast frame processing speed.

This work uses several traditional data preprocessing techniques to augment the data by rotating, shifting, increasing and decreasing the brightness, and flipping the images. Table 1 shows the data augmentation parameters with their specified values. Data augmentation is an excellent technique to increase the size of a dataset while exposing the essential regions and features of a specific image to the deep learning model. Once the data augmentation is done, it has been loaded to the base model MobileNetV2 with its pre-trained weights. The output layers of the base model are removed while loading the model. Then we add our own output layers on top of it. In our case, we added a dense layer of 4 classes with an activation function of softmax. We also set the last three layers of the pre-trained model as trainable. Thus, when the architecture training is initiated, the final three layers of the base model will also be trained along with the newly added layers.

Fig. 8 shows the overall summary of the proposed object detection and ambiance mode MobileNetV2 deep learning model with its trainable and non-trainable parameters. The pre-trained version of the lightweight MobileNetV2 network, trained on more than a million images from the ImageNet database, has been used in this work for object detection and ambiance mode description. This model is trained with Microsoft COCO and weather datasets for object detection and ambiance mode depiction, respectively. The last layers of the MobileNetV2 techniques have been modified in this work, as depicted in Fig. 8. These layers and their corresponding hyperparameters have been chosen employing the particle swarm optimization (PSO) technique. Gradient-based PSO approach has been used to search for these hyperparameters concerning the optimal solution in their corresponding search space. Particle Swarm Optimization (PSO) is a metaheuristic optimization algorithm inspired by the social behavior of bird flocking or fish schooling [35]. In PSO, a group of candidate solutions, i.e., particles, move through a search space to find the optimal solution [36]. Each particle has a position and a velocity that determines its movement in the search space. The particles communicate with each other and adjust their velocities based on their own best solution so far (i.e., the best position the particle has encountered) and the best solution found by any particle in the swarm. The PSO technique has been used because of its simplicity and fast convergence. It is a population-based algorithm that can efficiently explore the search space and avoid local optima.

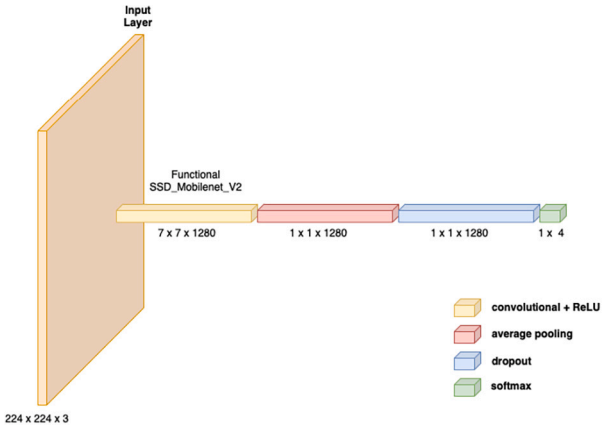


Fig. 8. Model training parameters for ambiance mode.

Table 2
MobileNetV2 model hyperparameters and corresponding search space.

Hyperparameters	Search space	Optimized value obtained from PSO algorithm
Pooling layer	[Max-pooling, Average pooling]	Average pooling
Learning rate	[0.0001, 0.0002, 0.0003, ..., 0.01]	0.0003
Dropout rate	[0.10, 0.20, 0.30]	0.20
Dense layer activation	[ReLU, Leaky ReLU, ELU]	ReLU

Table 2 illustrates the MobileNetV2 model hyperparameters, their search space and the corresponding optimized hyperparameters values obtained from the PSO algorithm. A dropout of 20 percent, obtained from the PSO technique, is applied so that it does not learn unnecessary features during the training process to avoid overfitting the model. We also used early stopping, a well-known technique in machine learning that helps prevent overfitting a model. It monitors the training session for each epoch and records the validation accuracy. If the validation accuracy does not improve for a given number of epochs, it simply stops the training. This technique provides the best possible accuracy out of a predefined number of epochs. To ensure minimal training loss, we used Adam optimizer with a learning rate of 0.0003. The significantly lower valued learning rate prevents the model from learning too many irrelevant features. A dropout of 20 percent and an Adam optimizer with a 0.0003 learning rate has also been selected.

After setting all the respective parameters, the proposed deep learning model has been trained on the Google Colab environment with an epoch of 30 and 32 steps per epoch. After fifteen epochs, we obtained an accuracy of 95 percent. Considering the small size of the utilized dataset, achieving an average accuracy of 95 percent with only fifteen epochs would not have been possible if we had built a model from scratch.

2.3. Hardware implementation of the proposed system

The ultimate goal of this work is to design a novel navigation-aiding device for object identification and description of the atmosphere of an environment. A list of the hardware components is utilized to implement this work, which is explained below:

- Raspberry Pi 4 Model B+ is utilized as the primary device since it is cost-efficient, portable, and performs efficiently compared to other embedded systems.
- Raspberry Pi camera v2 is used as a video capture device. It can capture videos up to 1080p30. It is connected to the Raspberry Pi's CSI port via a 15 cm ribbon wire.
- A 10000 mAh power bank is used as the primary power source. It has an input of 5 V/1A and an output of 5 V/2.1A.
- The final prototype of the device has been installed in a head cap. The Raspberry Pi is placed inside the cap, and the Pi camera is located outside.

The elementary hardware setup using the Raspberry Pi 4 embedded network and Pi camera have been demonstrated in Fig. 9. The entire design is lightweight, weighing approximately 140 g (0.31 lbs), including the fabric head cap, Raspberry Pi and Pi camera. The hardware devices are integrated into a head cap to implement the prototype of the proposed device. The inside and outside views of the actual system design are depicted in Fig. 10(a) and Fig. 10(b), respectively.

After turning on the device, the deep learning-based object detection model is loaded into the Raspberry Pi. Real-time video is captured constantly with the attached Raspberry Pi camera. The proposed model identifies various objects, and finally, the corresponding voice output is generated through earphones. The working consequences of object detection using the SSDLite MobileNetV2 model and audio feedback using Google text-to-speech have been depicted in Fig. 11.

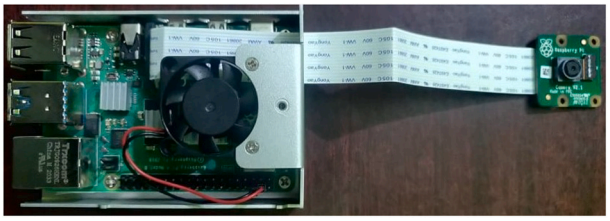


Fig. 9. Elementary hardware arrangement.

Fig. 10. System design of the prototype: (a) outside and (b) inside views.

Algorithm 1 Algorithm of the proposed object detection with SSDLite MobileNetV2 and audio feedback using gTTS.

Step 1: Initialize Raspberry Pi 4 and Load the Object Detection model.
Step 2: Optimize the hyperparameters of the SSDLite MobileNetV2 model with PSO.
Step 2.1:Initialize the swarm of solutions randomly in the search space according to Table 2.
Step 2.2: Evaluate the fitness of each particle based on a fitness function.
Step 2.3: Update the particle's velocity and position using its best and the swarm's best positions.
Step 3: Initialize the Pi camera and set it to stream the video frame.
Step 4: Set up the audio output device for feedback.
Step 5: Start the video stream from the Pi camera.
Step 6: For each frame:
Step 6.1: Pass the frame through the Object Detection model.
Step 6.1: Obtain the corresponding output, i.e., class of the detected object.
Step 6.2: Generate an audio feedback message and play it through the headphones indicating the detected class.
Step 7: Repeat Step 6 for each subsequent frame in the video stream.
Step 8: If the video stream stops or the program is interrupted, release any resources used by the program.

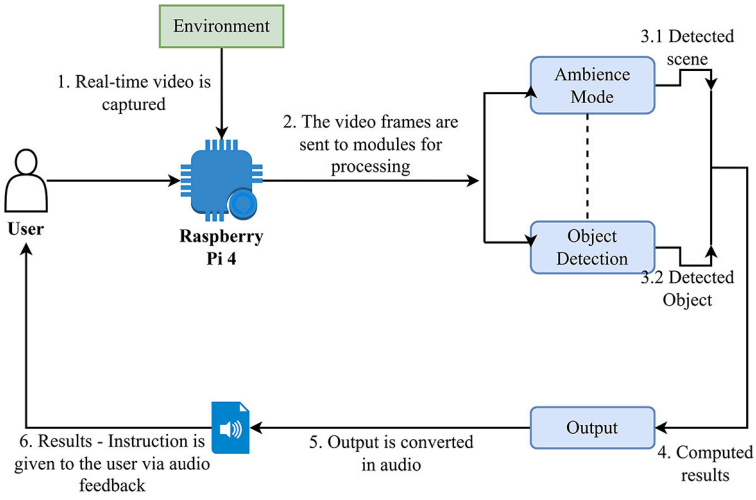


Fig. 11. Working sequences of object detection with SSDLite MobileNetV2 and audio feedback using gTTS.

Table 3
Average precision results for desktop PC.

Class	AP	Class	AP
person	0.80	chair	0.78
bed	0.76	potted plant	0.78
tv	0.80	cat	0.81
refrigerator	0.76	cell phone	0.84
car	0.75	couch	0.83

Table 4
Results of different models trained on the COCO dataset [37].

Models	Dataset	mAP	FPS
EfficientDet-D3	COCO	54.5	6.5
YOLOR-D6	COCO	55	30
YOLO V3	COCO	33.2	34
YOLO V4	COCO	43.25	83
YOLO V4-tiny	COCO	22	37.1
MobileNetV2 + SSDLite	COCO	23.4	88.6

3. Results and discussions

This section describes the software and hardware results of the proposed automatic voice-based object detection and surrounding environment description device for visually impaired people. At first, the deep learning-based object detection and ambiance mode performance are exhibited in a desktop personal computer. Finally, the results of the proposed hardware design are explained. The supplementary materials (videos) of the performed hardware experiments with the Raspberry Pi device can be found here.[1]

3.1. Results on desktop PC

In this research, the object detection model has been trained on AMD Ryzen 5 3600 6-Core Processor 3.60 GHz with 8 GB RAM and 1660 GPU. In this work, mean average precision (mAP) and frame per second (FPS), performance metrics for object detection tasks, have been evaluated. mAP is defined as:

$$mAP = \frac{1}{m}\sum_{j=1}^{j=m} AP_j \tag{2}$$

where AP_j denotes the average precision of class j and m is the total number of classes in the problem. FPS or frame rate determines the number of analyzed images and consequently determines the speed of the detection technique.

3.1.1. Object detection

In Table 3, the average precision of the selected object detection model is demonstrated on ten different classes for the desktop PC. The average precision (mAP) coefficients of the above categories contained in the COCO dataset are 0.791 for Desktop PC accomplished by the MobileNetV2 SSDLite model. This score defines how accurate a particular model is in detecting objects.

In Table 4, the mean average precision and frame per second (FPS) scores of different object detection models are demonstrated on the same Microsoft COCO dataset [37]. According to Table 3, EfficientDet-D3 has a higher mAP score with a very low frame rate. YOLO-D6 has the highest mAP score with an average FPS. In this work, we have incorporated the models into the embedded circuit, Raspberry Pi; we need a model with a significantly higher FPS and moderate mAP. Raspberry Pi 4 has a moderately lower processing power, which will significantly lower the FPS of the real-time live feed data. So using a model with low FPS will reduce the overall system's performance even more. MobileNetV2 with SSDLite has the highest FPS with a moderate mAP score. YOLOV4 also has a decent FPS with mAP scores higher than MobileNet, but it is not as lightweight as MobileNet. Because of this, finally, we chose to use the MobileNetV2 SSDLite CNN model for object detection as it offers a good trade-off between performance and accuracy.

Fig. 12 shows the receiver operating characteristic (ROC) curve of SSDLite MobileNetV2. For illustration purposes, the receiver operating characteristic curve of the SSDLite MobileNetV2 model is generated using the ground truth and predicted values of two classes only, i.e., cat and dog. The performance is believed to follow the same trend for the other object classes. This curve helps us understand how the model is performing at different classification thresholds. The area under the receiver operating characteristic curve is 0.91. The high AUC score indicates the high prediction accuracy of the proposed model.

Some of the detected objects by the proposed deep learning approach have been demonstrated in Fig. 13. The high confidence score of object detection exhibits the efficiency of the presented object detection model.

[1] https://github.com/RiasatKhanNSU/Object-Detection-and-Surrounding-Environment-Description-for-Visually-Impaired-People.

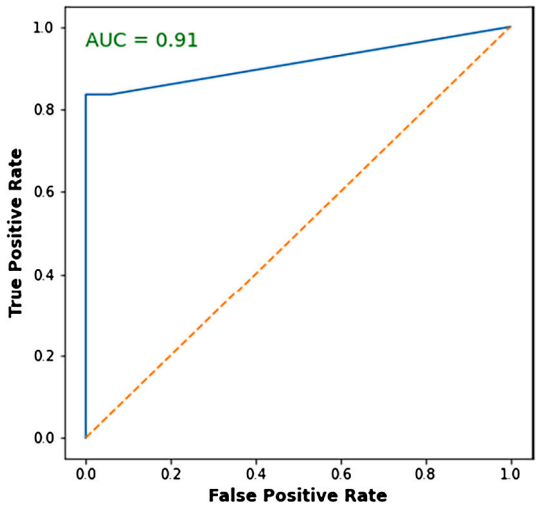


Fig. 12. ROC curve of the MobileNetV2 object detection model.

Fig. 13. Images of real-time detected objects.

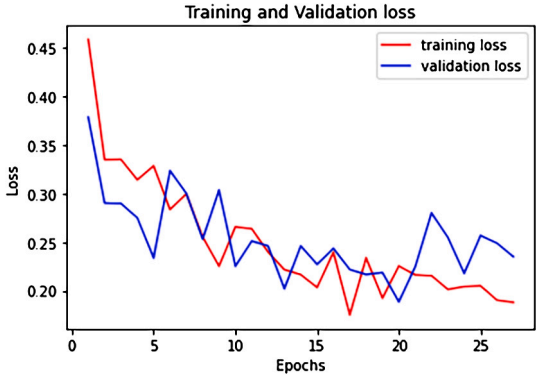


Fig. 14. Training and validation loss vs. epochs for ambiance mode.

After the system detects the objects of the MS COCO dataset, the audio feedback of the detected classes of various objects is generated for the blind people using gTTS, playsound, and PyAudio, and the speech recognition approach, as explained in Section 2.1. We have merged the audio feedback with the SSDLite MobileNetV2 model, where both processes work simultaneously.

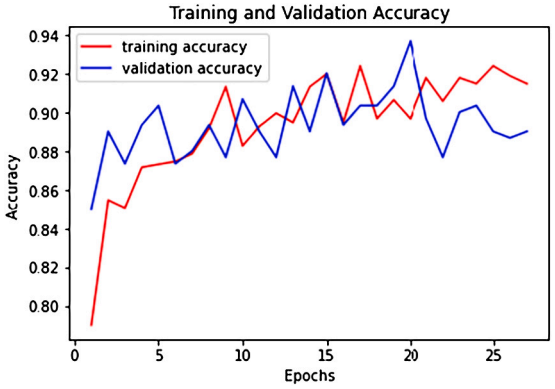


Fig. 15. Training and validation accuracies vs. epochs for ambiance mode.

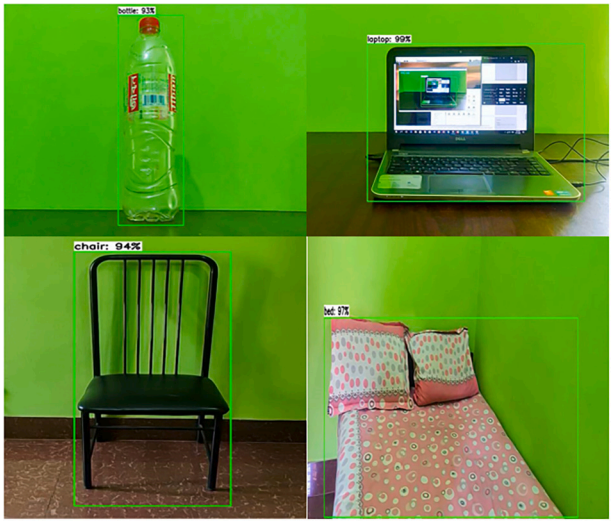


Fig. 16. Images of real-time detected objects with solid backgrounds by Raspberry Pi.

Fig. 17. Images of real-time detected objects with non-solid backgrounds and different lighting conditions by Raspberry Pi.

3.1.2. Ambiance mode

We used the open-source weather dataset from Kaggle to train the ambiance mode. The dataset has five classes of various conditions of weather, i.e., cloudy, foggy, rainy, shiny, and sunrise. Initially, we removed the noisy images and preprocessed the data to convert them into sRGB color profiles. Next, the dataset images are normalized, and then we read them using OpenCV. Finally, the VGG-16 architecture with a random forest classifier is employed. In Fig. 14, the training and validation losses indicate how well the model fits the training and testing new data. According to the figure, the training loss is initially high, but it reduces significantly with the increment of the epochs.

Fig. 15 illustrates the training and testing accuracies with the change of training epochs. The training accuracy increased significantly by approximately four times from the initial to final epochs. As the highest accuracy is obtained at 20 epochs, the training can be stopped at an earlier step. Lastly, the validation accuracy shows the performance of the proposed model on examples it has not observed.

Fig. 16 represents sample images of the detected objects with solid backgrounds by the Raspberry Pi device. The proposed deep learning technique can identify various objects with high accuracy.

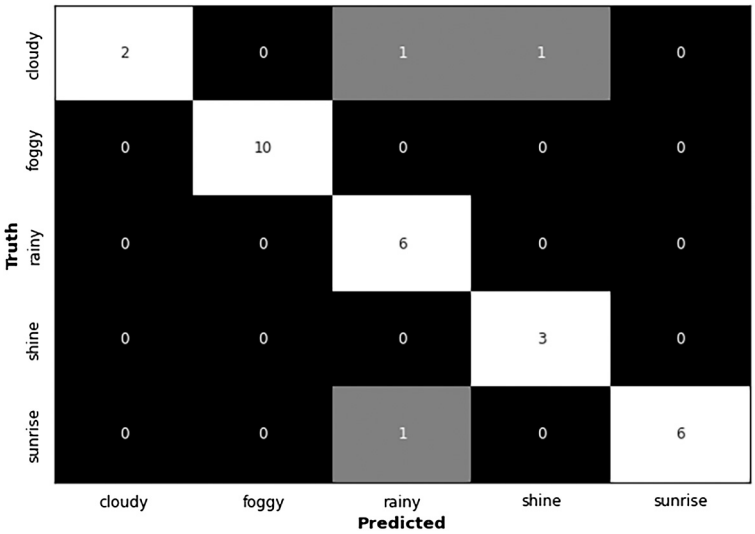


Fig. 18. Confusion matrix for ambiance mode.

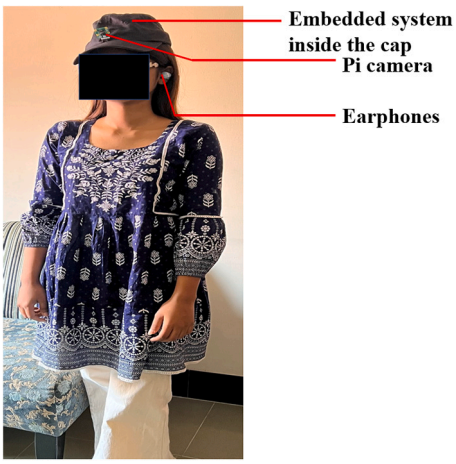


Fig. 19. User testing setup of the proposed device.

Various images of the detected objects with non-solid backgrounds, different lighting conditions and several objects in a single frame by the Raspberry Pi device have been illustrated in Fig. 17. The implemented MobileNetV2 SSDLite technique can detect these objects with decent accuracy.

Fig. 18 presents the confusion matrix of the proposed classifier for the five different weather classes of ambiance mode. It can be observed from the figure that, from a total of 30 instances, 27 and 3 cases are classified correctly and incorrectly, respectively. While predicting the environment, the implemented model gave three false positives, i.e., it detected two rainy weather as sunrise and cloudy and one shiny weather as cloudy.

3.2. Results on embedded system

Finally, the proposed obstacle detection and surrounding environment description system have been implemented in an embedded system to work in real time, illustrated in Fig. 19. The embedded system used in this research to implement the deep learning models is Raspberry Pi 4B with a Quad-core Cortex-A72 processor, 64-bit SoC @1.5 GHz and 4 GB RAM.

3.2.1. Object detection

This section presents the results for the proposed assistive device of object and obstacle detection implemented on Raspberry Pi.

In this work, various comprehensive real-time experiments (indoor and outdoor) have been performed to detect different objects with the proposed device. These objects have been placed approximately 0.05 m - 3 m from the Pi camera with different solid and non-solid backgrounds and lighting conditions. After careful inspection of the detected objects, the confusion matrix of some of the categories has been obtained, demonstrated in Fig. 20. According to this figure, the detection accuracy of the proposed embedded

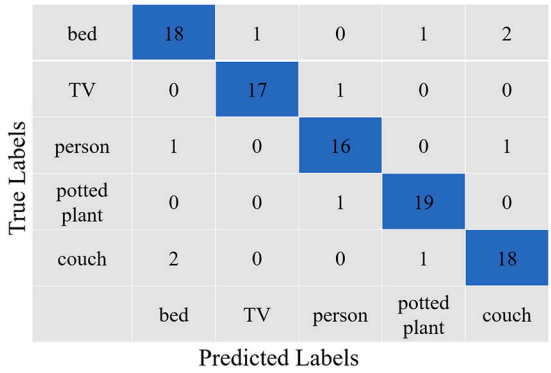


Fig. 20. Confusion matrix of the detected objects of the proposed embedded system.

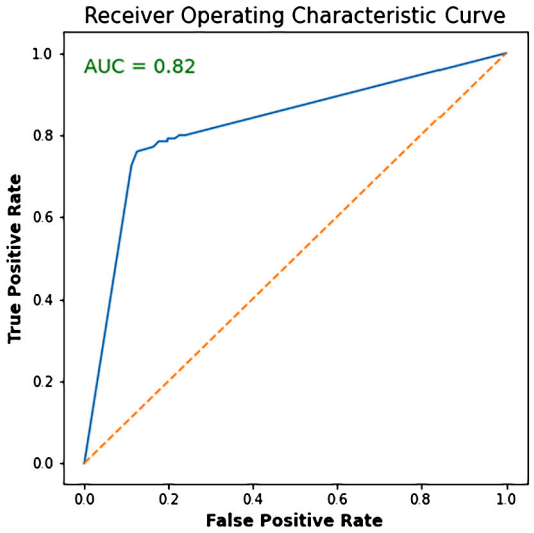


Fig. 21. ROC curve for object detection mode.

system Raspberry Pi-based hardware device is 88.89%. The average precision and recall for all the classes are 0.892 and 0.89, respectively. The average FPS of object detection of the Raspberry Pi is approximately 2.15. It should be mentioned that, if an object is classified correctly for three consecutive frames, it is considered a correct detection. Conversely, if an object is misclassified for three successive frames, it is regarded as an incorrect detection.

Next, the true and false positive rates FPR graph (ROC curve) of the MobileNetV2 model for object detection mode has been plotted in Fig. 21. According to this figure, the area under the ROC curve is 0.82.

In this work, we have used the back/primary camera of the Samsung Galaxy FE Android smartphone and connected it with the Raspberry Pi to investigate how the proposed system will work with higher-resolution images of object detection. The back camera of the smartphone has a relatively higher resolution of 12 MP, compared to 8 MP resolution of the Pi camera. The utilization of the higher resolution camera improves the average object detection confidence score from approximately 72% (with the Pi camera) to 84.5%. Conversely, the average FPS of object detection of the higher resolution smartphone camera reduces from approximately 2.15 (with the Pi camera) to 1.71.

3.2.2. Ambiance mode

For Raspberry Pi implementation of this mode, we used the class “sunrise” to detect both “sunset” and “sunrise” scenes and consequently, there are four categories of images. This process is done because pictures of both of these classes have almost similar features.

The training and validation losses with the change of epochs for the embedded device deployment of ambiance are illustrated in Fig. 22. Similar to the results of desktop PC, the losses decrease with the increment of epochs. As expected, the validation loss is slightly higher in the embedded system than desktop PC.

In Fig. 23, the training accuracy indicates the accuracy of our model in the embedded system on the examples it was constructed on, while the validation accuracy indicates the accuracy of our model on examples it has not seen. As mentioned earlier in Section 2.2, we have implemented a callback function provided by Keras, i.e., early stopping, while training the model of ambiance mode on the embedded system. It monitors the validation accuracy, halts the training process if the accuracy saturates, and saves the model

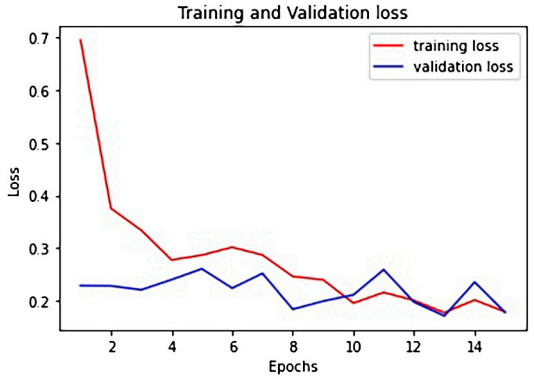


Fig. 22. Training and validation losses for ambiance mode on Raspberry Pi.

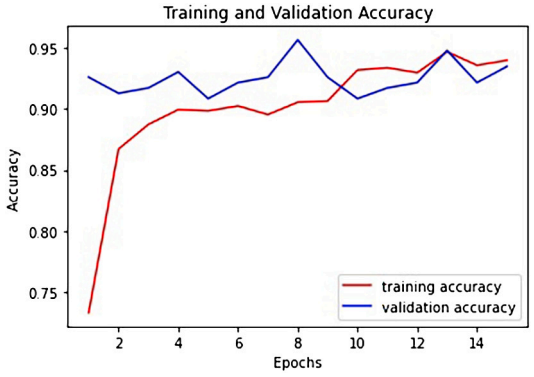


Fig. 23. Training and validation accuracy graphs for ambiance mode.

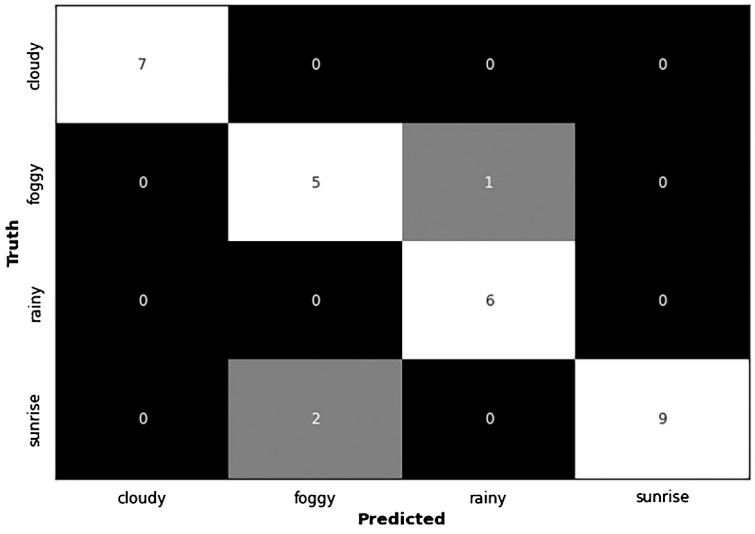


Fig. 24. Confusion matrix for ambiance mode in the embedded system.

to a point where it achieves the highest prediction accuracy. This technique prevents overfitting and increases the processing speed of the model. The number of epochs on the embedded system's model is less than the desktop computer's network for utilizing this function.

Fig. 24 shows the confusion matrix of the predicted class against the actual class for ambiance mode in the embedded system. According to this figure, in a total of 30 images, 27 instances are classified correctly. The cloudy and rainy weather conditions achieved the highest accuracy with no misclassification error.

Table 5
Narratives generated by the ambiance mode of the proposed system.

Class	Narrative description
Rainy	It is raining outside with a refreshing breeze. This weather seems like a boon for trees, birds, and animals.
Cloudy	The clouds have begun to gather in the sky. Beautiful bright sky is now starting to look really gloomy.
Sunrise	The sun is about to rise. The rays of the sun look warm and bracing to the eyes. The sun rays are adding quite a reddish hue to the sky.
Foggy	The weather today is quite hazy. It looks really pleasant, but the surroundings can hardly be observed because of all the mists around.

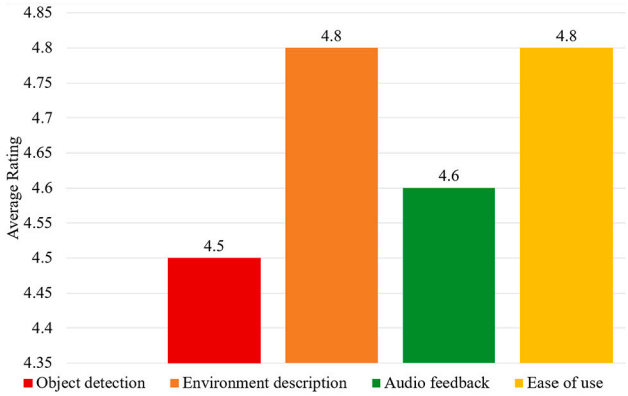


Fig. 25. Average evaluation rating of the proposed device from its users.

3.3. Audio feedback

In the object detection model, 90 different classes can be identified. We generated the audio feedback file for each class label using the Google text-to-speech library in a simple Python program. Next, these files are saved in MP3 format with a filename corresponding to each class label. Finally, all of the MP3 files are stored in a folder in the embedded system. Whenever the proposed deep learning model detects an object, the corresponding MP3 file is called by the program. The corresponding MP3 file is then played back to the user for audio feedback. In this way, the proposed system generates the audio outputs of the detected objects.

The ambiance mode also follows the same method stated above for audio feedback. Table 5 illustrates samples of the narratives of different types of weather conditions generated by ambiance mode.

In this study, several experiments have been conducted with the developed object detection and ambiance mode description device. Three volunteers (2 male and 1 female) participated in these investigations. An evaluation of the proposed device has been obtained from the volunteers concerning its various features on a scale of 1 to 5. As reported in Fig. 25, the average evaluation rating of the proposed device is 4.675. Specifically, the implemented device's users are convinced with its surrounding environment description and ease of use.

3.4. Comparison to other works

This section compares various features, applied deep learning models and the estimated cost of the proposed system implemented in this work to other similar AI-based assistive systems for visually impaired persons.

Table 6 summarizes the comparison of the proposed device with other similar works concerning their basic features, used embedded system, applied deep learning techniques, various performance metrics and approximated cost. Integration of complicated sensors and devices increases the overall cost of the system as in [10]. Several articles used sophisticated deep learning techniques to create an affordable solution. Most of these studies did not thoroughly examine the deployed device, displaying mean average precision (mAP), precision, recall, detection accuracy, AUC, user survey, device cost, etc. The implemented assistance system in this work utilizes deep learning approaches for obstacle detection and surrounding atmosphere description with a newer version of Raspberry Pi 4, decent performance and moderate overall cost.

3.5. Limitation and discussion

This study develops an object detection and ambiance mode depiction device for the navigational assistance of visually impaired people. The embedded system constructed on a Raspberry Pi 4B can detect various common objects with satisfactory accuracy of

Table 6
Comparison of the proposed device with other similar works.

Reference	Primary features	Embedded system	Applied model	Performance metrics	Estimated cost (USD)
Traditional White Cane	Search surroundings for navigation	N/A	N/A	N/A	25
[10]	Eyeglass to detect public sign with audio output	Intel Edison	OpenCV	N/A	240
[11]	Vision assistance with wearable sensors	N/A	Custom CNN	Precision = 0.766, Recall = 0.164	97
[38]	Text extraction and face recognition	Raspberry Pi 2	OpenCV	N/A	70
[7]	Obstacle avoidance and reading assistance	Raspberry Pi 3	OpenCV	N/A	68
[16]	Obstacle avoidance and object classification	Raspberry Pi 4B	Viola–Jones	N/A	N/A
[17]	Obstacle detection and traffic light classification	GD32F1 MCU	YOLOv3-tiny	mAP = 0.97	40.30
[18]	Object identification and tracking	Raspberry Pi 3B	MobileNet	Detection accuracy = 83.3%	N/A
[19]	Object detection and navigation	Raspberry Pi 3B	YOLOv3	Detection accuracy = 81%	N/A
[20]	Crosswalk recognition and navigation	AI Edge MCI module	Inception SSD	Detection accuracy = 92.2%	N/A
[21]	Crosswalk detection and navigation	Raspberry Pi 4	OpenCV	Detection accuracy = 84.5%	N/A
[22]	Traffic light detection and classification	Raspberry Pi	OpenCV	Detection accuracy = 87.5%	N/A
[23]	Animal detection and classification	Jetson Nano	SSD ResNet-50	mAP = 93.5%	N/A
[24]	Object detection and navigation	Raspberry Pi 4	OpenCV	N/A	N/A
This work	Obstacle detection and surrounding atmosphere description	Raspberry Pi 4B	SSDLite MobileNetV2	Detection accuracy = 88.89%, Precision = 0.892, Recall = 0.89	65

88.89% and 2.15 FPS. The open-source Microsoft COCO dataset has been used in this work to train the object detection model, which can identify 90 different familiar classes. While the COCO dataset contains a wide range of objects, it is not exhaustive and many objects are not included in the dataset, e.g., computer monitor image does not exist in the COCO dataset and is detected as a TV. The proposed assistive tool can detect humans as persons but cannot distinguish a specific person employing facial recognition techniques.

4. Conclusions

The most significant barrier for visually impaired people is imposing their security, dependability, and precise guidance, which introduces challenging problems in their navigation. In this paper, an automatic assistance habitual framework has been designed for detecting objects and encompassing environment description for blind people that helps them to move without the help of others. The suggested system is based on deep learning approaches, which are implemented by utilizing Raspberry Pi 4 and Pi camera. The pre-trained SSDLite MobileNetV2 model is trained on the COCO and an open-source weather dataset for object identification and ambiance description mode, respectively. The particle swarm optimization technique has been applied to choose the final layers and corresponding hyperparameters of the MobileNetV2 model. The proposed system is easy to operate, energy-efficient and cost-effective, which can be used for indoor and outdoor navigation. The proposed combined voice assistance feature offers image-to-text altering capabilities by speech and enables the visually impaired to understand what they are facing in front of them.

Future work could improve contextual information, ambiance, and flexibility in the design of the proposed assistive device. The size of the weather dataset used to develop the ambiance mode, which was relatively small, can be increased. This modification will increase the accuracy of the proposed ambiance mode model. The audio feedback on ambiance mode currently uses predefined texts that are then converted to speech by Google's text-to-speech engine. However, we can add complex natural language processing techniques to the system to make the interactions sound more lively. The storage, ram, and processor speed offered by the Raspberry Pi is sufficient for a small-scale project. In the future, the assistive system can be implemented on more powerful hardware like the Jetson Nano or a cloud server where we can utilize more resources to upgrade the model to describe complex scenes. In addition, the model can be trained on different situational and architectural datasets to describe more real-life situations or visiting places

to blind people, almost like a human companion. This work's implementation details will be available open-source so that other researchers can work on this project using various datasets to make it more adaptive to real-life events. Adding more sensors could also help the system extract more information about its surroundings, which in turn could help identify a specific scenario in greater detail. This addition will also improve the device's accessibility, making it useful to people with other disabilities as well. Creating a mobile app or incorporating our model into mobile devices could also increase its usability. Speech recognition and multimodal learning techniques can be added to the proposed detection system to improve its navigational functions. Reinforcement learning can be applied to enhance the performance of the proposed device.

CRediT authorship contribution statement

Raihan Bin Islam: Conceived and designed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper.

Samiha Akhter: Conceived and designed the experiments; Performed the experiments; Contributed reagents, materials, analysis tools or data; Wrote the paper.

Faria Iqbal: Md. Saif Ur Rahman: Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data.

Riasat Khan: Contributed reagents, materials, analysis tools or data; Wrote the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data associated with this study has been deposited at: https://cocodataset.org/#home.

References

[1] A. Bhowmick, S. Hazarika, An insight into assistive technology for the visually impaired and blind people: state-of-the-art and future trends, J. Multimodal User Interfaces 11 (2017) 1–24.
[2] P. Ackland, S. Resnikoff, R. Bourne, World blindness and visual impairment: despite many successes, the problem is growing, Community Eye Health 30 (2017) 71–73.
[3] I. Khan, S. Khusro, I. Ullah, Technology-assisted white cane: evaluation and future directions, PeerJ 6 (2018) 1–27.
[4] M. Awad, J.E. Haddad, E. Khneisser, T. Mahmoud, E. Yaacoub, M. Malli, Intelligent eye: a mobile application for assisting blind people, in: IEEE Middle East and North Africa Communications Conference, 2018, pp. 1–6.
[5] S. Bhatlawande, S. Shilaskar, A. Abhyankar, M. Ahire, A. Chadgal, J. Madake, Detection of exercise and cooking scene for assistance of visually impaired people, in: G. Ranganathan, R. Bestak, X. Fernando (Eds.), Pervasive Computing and Social Networking, Springer Nature Singapore, Singapore, 2023, pp. 493–508.
[6] M. Mukhiddinov, A. Muminov, J. Cho, Improved classification approach for fruits and vegetables freshness based on deep learning, Sensors 22 (2022).
[7] M.A. Khan, P. Paul, M. Rashid, M. Hossain, M.A.R. Ahad, An AI-based visual aid with integrated reading assistant for the completely blind, IEEE Trans. Human-Mach. Syst. 50 (2020) 507–517.
[8] A. Nishajith, J. Nivedha, S.S. Nair, J. Mohammed Shaffi, Smart cap - wearable visual guidance system for blind, in: International Conference on Inventive Research in Computing Applications, 2018, pp. 275–278.
[9] M. Cabanillas-Carbonell, A.A. Chávez, J.B. Barrientos, Glasses connected to Google vision that inform blind people about what is in front of them, in: International Conference on e-Health and Bioengineering, 2020, pp. 1–5.
[10] F. Lan, G. Zhai, W. Lin, Lightweight smart glass system with audio aid for visually impaired people, in: IEEE Region 10 Conference, 2015, pp. 1–4.
[11] B. Jiang, J. Yang, Z. Lv, H. Song, Wearable vision assistance system based on binocular sensors for visually impaired users, IEEE Int. Things J. 6 (2019) 1375–1383.
[12] N. Kumar, S. Sharma, I.M. Abraham, S. Sathya Priya, Blind assistance system using machine learning, in: J.I.-Z. Chen, J.M.R.S. Tavares, F. Shi (Eds.), International Conference on Image Processing and Capsule Networks, Springer International Publishing, Cham, 2022, pp. 419–432.
[13] Y. Bouteraa, Smart real time wearable navigation support system for BVIP, Alex. Eng. J. 62 (2023) 223–235.
[14] Z. Xie, Z. Li, Y. Zhang, J. Zhang, F. Liu, W. Chen, A multi-sensory guidance system for the visually impaired using YOLO and ORB-SLAM, Information 13 (2022).
[15] M. Mukhiddinov, J. Cho, Smart glass system using deep learning for the blind and visually impaired, Electronics 10 (2021).
[16] U. Masud, T. Saeed, H.M. Malaikah, F.U. Islam, G. Abbas, Smart assistive system for visually impaired people obstruction avoidance through object detection and classification, IEEE Access 10 (2022) 428–13 441.
[17] K. Xia, X. Li, H. Liu, M. Zhou, K. Zhu, Ibgs: a wearable smart system to assist visually challenged, IEEE Access 10 (2022) 810–77 825.
[18] F. Ashiq, M. Asif, M.B. Ahmad, S. Zafar, K. Masood, T. Mahmood, M.T. Mahmood, I.H. Lee, Cnn-based object recognition and tracking system to assist visually impaired people, IEEE Access 10 (2022) 819–14 834.
[19] N. Kumar, A. Jain, Smart navigation detection using deep-learning for visually impaired person, in: International Conference on Electrical Power and Energy Systems, 2021, pp. 1–5.
[20] W.-J. Chang, L.-B. Chen, C.-Y. Sie, C.-H. Yang, An artificial intelligence edge computing-based assistive system for visually impaired pedestrian safety at zebra crossings, IEEE Trans. Consum. Electron. 67 (2021) 3–11.
[21] A. Bhattacharya, V.K. Asari, Wearable walking aid system to assist visually impaired persons to navigate sidewalks, in: Applied Imagery Pattern Recognition Workshop, 2021, pp. 1–7.
[22] R. Khalid, M.W. Iqbal, N. Samand, M. Ishfaq, R. Rashed, S. Rafiq, Traffic light issues for visually impaired people, J. Jilin Univ. 371 (391) (2022) 3–11.
[23] K. Manjari, M. Verma, G. Singal, N. Kumar, QAOVDetect: a novel syllogistic model with quantized and anchor optimized approach to assist visually impaired for animal detection using 3D vision, Cogn. Comput. 14 (2022).
[24] M. Bala, D. Vasundhara, H. Akkineni, C. Moorthy, Design, development and performance analysis of cognitive assisting aid with multi sensor fused navigation for visually impaired people, J. Big Data 10 (2023).

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: common objects in context, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), European Conference on Computer Vision, Springer International Publishing, Cham, 2014, pp. 740–755.
[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD: single shot multibox detector, in: Lecture Notes in Computer Science, Springer International Publishing, Cham, 2016, pp. 21–37.
[27] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint, arXiv:1409.1556, 2014.
[28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, MobileNetV2: inverted residuals and linear bottlenecks, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.
[29] Y. Pachipala, M. Harika, B. Aakanksha, M. Kavitha, Object detection using tensorflow, in: International Conference on Electronics and Renewable Systems, 2022, pp. 1611–1619.
[30] M. Yasir, M. Nababan, Y. Laia, W. Purba Robin, A. Gea, Web-based automation speech-to-text application using audio recording for meeting speech, J. Phys. Conf. Ser. 1230 (2019).
[31] T. Giannakopoulos, pyAudioAnalysis: an open-source python library for audio signal analysis, PLoS ONE 10 (2015).
[32] P.A.A. Cherukuri, N.T.D. Linh, S.K. Vududala, R. Cherukuri, Keras convolutional neural network model in sequential build of 4 and 6 layer architectures, in: D.-T. Tran, G. Jeon, T.D.L. Nguyen, J. Lu, T.-D. Xuan (Eds.), Intelligent Systems and Networks, Springer Singapore, Singapore, 2021, pp. 465–472.
[33] V. Svetnik, A. Liaw, C. Tong, J.C. Culberson, R.P. Sheridan, B.P. Feuston, Random forest: a classification and regression tool for compound classification and QSAR modeling, J. Chem. Inf. Comput. Sci. 43 (2003) 1947–1958.
[34] K. Fawagreh, M.M. Gaber, E. Elyan, Random forests: from early developments to recent advancements, Syst. Sci. Control Eng. 2 (2014) 602–609.
[35] X.S. Yang, Particle swarm optimization, in: Nature-Inspired Optimization Algorithms, Academic Press, 2021, pp. 111–121.
[36] S. Sengupta, S. Basak, R.A. Peters, Particle swarm optimization: a survey of historical and recent developments with hybridization perspectives, Mach. Learn. Knowl. Extr. 1 (1) (2019) 157–191.
[37] P. Wang, E. Fan, P. Wang, Comparative analysis of image classification algorithms based on traditional machine learning and deep learning, Pattern Recognit. Lett. 141 (2021) 61–67.
[38] M. Rajesh, B.K. Rajan, A. Roy, K.A. Thomas, A. Thomas, T.B. Tharakan, C. Dinesh, Text recognition and face detection aid for visually impaired person using Raspberry Pi, in: International Conference on Circuit, Power and Computing Technologies, 2017, pp. 1–5.