An Audio-Aided Face and Text Recognition System for Visually Impaired

Manjusha Sreedharan, Shalini Mohanraj, and Lakshmi Sutha Kumar

Abstract One of the biggest difficulties for a visually impaired person is to identify people and read text. This limits the visually impaired to interact socially and puts them at risk. In recent years, several deep learning techniques have been used for face and text recognition. This paper proposes a portable system for face and text recognition using the Raspberry Pi 3B+, Raspberry Pi camera module, customized dataset, few push button switches, and earphones. Multi-task cascaded convolutional neural network and support vector machine are used for face detection and face recognition, respectively. The proposed idea involves efficient and accurate scene text detector for text detection and Tesseract optical character recognition (OCR) for text recognition. Finally, converting the text or face labels to speech by e-Speak tool, a process which allows visually impaired people hear the name of the person/text-written in front of them.

Keywords Face recognition · Text recognition · MTCNN · LSTM · Tesseract · SVM

1 Introduction

According to WHO, approximately 285 million people are visually impaired worldwide [1]. Visually impaired people experience many difficulties throughout their life. These include difficulty in communicating with people, perceiving the atmosphere, and confining motion as they are unaware of the dangers in front of them [2]. If the visually impaired person can "see" the world with the support of present technological advancements, they will attain improved independence and freedom. Simple prototype can be designed to support visually impaired using camera to detect and recognize texts and faces.

Face recognition is considered as one of the most important issues because face plays an important role in recognizing people in social life. It should be customized

M. Sreedharan (✉) · S. Mohanraj · L. S. Kumar
Department of Electronics and Communication Engineering, National Institute of Technology Puducherry, Karaikal 609609, India

R. J. Kannan et al. (eds.), *International Virtual Conference on Industry 4.0*,
Lecture Notes in Electrical Engineering 355,
https://doi.org/10.1007/978-981-16-1244-2_2

so that the visually impaired can alter the face dataset according to his/her needs. Face detection is done using deep learning and recognition using support vector machine, and text recognition is done using Tesseract-OCR engine in [3].

Different methods of face detection have been discussed in [4]. It is concluded in [5] that for real-time applications, deep learning methods like multi-task convolutional neural network (MTCNN) are better than conventional methods designed by Viola and Jones. A new method in [6] cascades two processes of face detection and face alignment using multi-task CNN. Face recognition is done using a multi-class SVM classifier in [7]. Python scikit learn libraries are used as mentioned in Scikit documentation (2011) [8]. Text detection is done using an efficient and accurate scene text detector (EAST). Text recognition is done by Tesseract OCR which includes a new neural net called long short-term memory (LSTM)-based OCR engine, which concentrates on line recognition and recognizes character pattern. The recognized face or text is converted to audio output by a text-to-speech (TTS) system that converts text into speech which is explained in [10].

Krishna et al. [11] implementation includes a smartphone that obtains images using the phone's camera and wirelessly transceives to a remote server for recognition. PCA and Microsoft speech engine were used. Chillaron [12] implemented a face detection and recognition application developed using Raspberry Pi hardware and an interface for Android smartphones. The object detection and face recognition are based on the boosted cascade and eigenfaces, respectively. Rani [13] describes a smart glass that can automatically detect, examine, and recognize text and provides voice hints for visually impaired. The prototype was based on the Intel Edison Platform, and recognition was done using Tesseract software.

Aravind and Roshna [14] have constructed a model for text recognition that continuously acquires images, finds textual characters, converts it into speech, and reads out loud. The shortcoming of this model is that the computing device is not mobile which is disadvantageous to the visually impaired. Balduzzist [15] implemented model using a compact computational system that acquires a video and inspects it to detect faces in the frame. The face recognition module uses local binary patterns (LBP) to determine the detected faces. Zhou et al. [9] implemented a text detector called efficient and accurate scene text detector. The model gives the arbitrary location of the textual data and is highlighted by a boundary box in the image.

This paper aims to build a customized camera-based assistant for face and text recognition to help the visually impaired. The idea of the design is in such a way that it initiates with a choice between face or text recognition by the visually impaired. He/she can toggle between both as per his/her necessities. Even though earlier works support visually affected people, most of the prevailing systems use MATLAB as [16]. Handful of the designs use laptops which are not portable. Algorithms used in the previous systems are not accurate and efficient.

The rest of the paper is organized as follows. Section 2 deals with the design of prototype and explanation of the model; a detailed description of the algorithms used in the project is described in Sect. 3; Sect. 4 includes the hardware components used in the design and discusses the experimental implementation of the proposed

system. Based on the design and analyses, conclusions and future scope are provided in Sect. 5.

2 Proposed System

The block diagram in Fig. 1 shows the overall working of the prototype. Face dataset for a single person is developed by taking around 20 photos of him/her in different angles. 80 photos of 4 known persons (friends/relatives of visually impaired) are used as the customized dataset in this paper. During the testing phase, a new photo of the person (any one of 4 persons available in the customized dataset) is taken and recognized. It is classified as unknown if it does not belong to existing dataset. Python programming language is used to program the Raspberry pi. Face recognition and text recognition programs are coded separately and later integrated. The GPIO pins of the Raspberry pi are programmed for mode selection using push buttons.

B1, B2, and B3, three push buttons, used to initiate face recognition, text recognition, and to stop Raspberry Pi, respectively. The user can choose between either face recognition or text recognition as per the requirement. The buttons are connected using the jumper wires from a breadboard to RPi. The camera is switched ON when the button 1 is pressed. During the face recognition session, face detection uses multi-task CNN to detect faces in the scene, and the detected face is passed on for recognition. Face recognition is done by the SVM classifier which recognizes the person (He/She must be one of the persons in the customized dataset for recognition).

Similarly, when the push button selects the text recognition, the camera is switched on for text recognition. When text is detected by EAST detector, it is captured and pre-processed. The pre-processing and recognition are done by Tesseract. Then, the

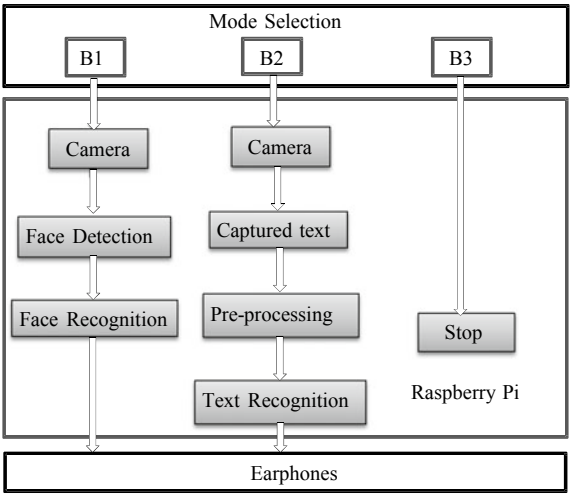

Fig. 1 Block diagram for the face and text recognition system

recognized text is converted to voice. The third button, B3 stops the process and when pressed once more shuts down the Raspberry Pi.

3 Algorithms

3.1 MTCNN

Multi-task CNN, proposed in [19], is used from the integration of face detection and alignment tasks. An image pyramid is formed in which the images are rescaled into many sizes in order to detect faces of all sizes. This is then passed through the cascade of networks- proposal network or P-Net, refine network or R-Net, and output network or O-Net. This then outputs the probability of a face being in the box, the coordinates of the bounding box, and the coordinates of the facial landmarks (locations of the eyes, nose, and mouth). An additional process called non-maximum suppression is performed between these stages to reduce the number of bounding boxes. After the isolation of the image from the background and pre-processing using Dlib, the face needs to be represented in numerical embedding. This is done using pretrained deep neural network OpenFace which is explained next.

3.2 Model Used—nn4.Small2.V1

Two largest publicly available datasets are used for training the current face recognition model as of August 2015. These include FaceScrub and CASIA-WebFace. It uses triplet loss function which is explained in the next section.

3.2.1 Triplet Loss

The embedding is represented by $f(x) \in$ Rd. It embeds an image into a d-dimensional Euclidean space which is constrained into a d-dimensional hypersphere, i.e., $\|f(x)\|2 = 1$. Model is trained such that it learns an image embedding which is done by calculating the L2 distance between image of the same identity (anchor image and positive image) and between different (anchor image and negative image) identities which will be small and large, respectively. Here, a stands for anchor image, p stands for positive image, n for negative image for any identity i, and α stands for least margin.

$$L = \sum_{i=1}^{m} [\left|\left|f(x_i^a) - f(x_i^p)\right|\right|_2^2 - \left\|f(x_i^a) - f(x_i^p)\right\|_2^2 + \alpha]_+ \tag{1}$$

Means max (z, 0) and m is the number of triplets in the training set.

The receiver operating characteristics (ROC) curve in Fig. 1 from [20] shows comparison of ROC curves among recognition algorithms. The ROC curve of the nn4.small2.v1model covers much area than of OpenBR v1.1.0, eigenfaces, OpenFace nn4.small2.v1 folds. This is a smaller announcement of the latest model, nn4.small2.v1, which improves the LFW accuracy from 91.5 to 93.6%. It uses a smaller model and improves the runtime performance. To identify new inputs, the classifier trained with the database entries (embedding) is required. In this paper, a linear support vector machine is used for this purpose.

3.3 Support Vector Machine

The purpose of SVM algorithm is to identify a hyperplane in an N-dimensional space that classifies the data points precisely. Linear SVM is the newest exceptional machine learning algorithm used to solve multi-class classification for extremely large database. For both binary classification and multi-class classification ($k = 2$ and $k \geq 3$, respectively), support vector classification (SVC) is very promising.

3.4 EAST

The Efficient and Accurate Scene Text Detection (EAST) architecture is a two-step process to detect the region of interest. The EAST detector consists of a fully convolutional network (FCN) model that produces word or text-line predictions. The detected text regions can be either rotated rectangles or quadrangles that are subjected to non-maximum suppression, after thresholding to return final results. The architecture was created by taking different sizes of word regions into account. Small word regions that use low-level features are detected in initial stages, while large word regions are detected from the later stage of the network.

3.5 Tesseract OCR

Tesseract software is labeled as one of the most accurate open-source OCR engines. Optical character recognition systems transform a two-dimensional image of text, from its image representation into machine-readable text. It makes use of long short-term memory (LSTM) which is a part of recurrent neural networks. Graves introduced bidirectional LSTM architectures for examining text in both forward and backward directions. These layers are later connected to a single output layer, and thereby, it ensures the most accurate prediction [17].

3.6 Pyttsx

There are several APIs available to convert text-to-speech in Python. One of such APIs is Python text-to-speech (pyttsx). The advantage of pyttsx is that it works offline, unlike other text-to-speech libraries. Rather than saving the text as an audio file, the pyttsx speaks it there. This makes it more reliable. The volume, pitch, and gender of the audio output can be changed as per the needs of the user.

4 Hardware Implementation and Working

The user collects the photos of his friends and relatives. As explained in Sect. 2, this model uses 20 images of four people. These collected photos of each person are saved in separate folders. A Python script which converts these photos into matrices using the Numpy library is written. The nn4.smallv2 model is then loaded. Using this pretrained network, embedding vectors are calculated. From [19], it can be seen that the validate rate of 128 dimensions is the highest, i.e., 87.9% $\pm$ 1.9. Figure 2 shows the squared L2 distance calculated using (1) between anchor-negative and anchor-positive pairs and L2 distance of anchor-negative pair is higher than the L2 distance of anchor-positive pair. After the embedding is calculated, the classifier is trained by them.

The Scikit library in Python with the function of linear SVM classifier (SVM) is used to train the dataset and classification. Therefore, the whole script is divided into four parts. First MTCNN detects the face as explained in Sect. 3.1. The second part is for pre-processing using OpenCV and to find the embedding using Numpy libraries. The third part deals with recognition, and it is done as SVM classifies it. Finally, the last part aims to convert the name of the person (label of the classifier) to voice. The text recognition script consists of three sections. The first section is used for text detection using EAST detector, to produce bounding boxes of these textual content. The second part recognizes the text using Tesseract. The last part converts the recognized text to voice.

An SD card with Raspbian OS and along with the models used for face recognition is loaded in the micro-SD card slot of the raspberry pi. The designed prototype along

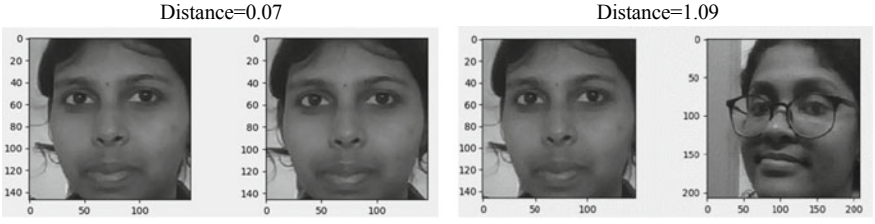

Fig. 2 Distance between similar and dissimilar images

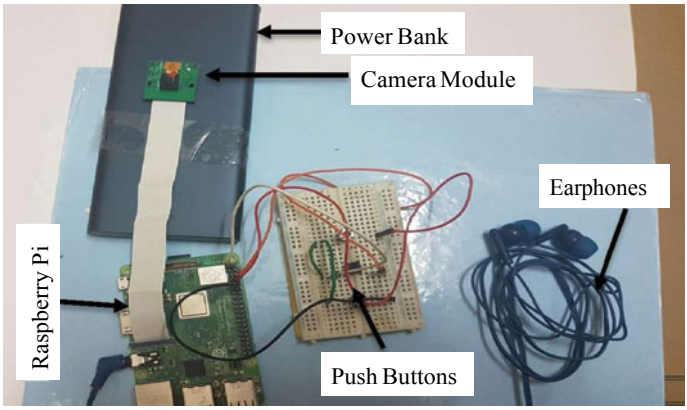

Fig. 3 Hardware setup

with its connections is shown in Fig. 3 where the camera module is connected to the raspberry pi using a ribbon cable. The push buttons are connected to the GPIO pins using jumper wires. The power bank is connected to the raspberry pi power input to start the process. The user can choose the face or text recognition as per needs.

If the first button is pressed, face recognition is started. This is initiated with the switching ON of the camera. The camera then searches the frame for a face. The face detection is done by MTCNN as explained. Each face in the frame is extracted using bounding box regression and face landmark localization. The coordinates of the face are produced as the output. The 128-d vector of the detected face is calculated. In embedding space, Euclidean distances are calculated as explained in Sect. 3.2. The SVC then classifies the face. The label of the classified output and the probability is calculated by Platt's scaling. The next step is to convert the output (label) into audio signals. Thus, the name of the person is given to the Python text-to-speech function. The voice output can be heard using earphones. The stop button is pressed to end the process. It can be started again whenever the user wishes.

When the second button is pressed, the same procedure follows. The camera is turned ON. This can be identified as the red LED is ON. Again, the frame searches for text. The EAST detector produces predictions as explained in Sect. 3.4. These bounding boxes are sent to Tesseract OCR for recognition. The working of Tesseract has been explained in the previous section. Tesseract uses LSTM to predict the next word. Thus, with time, the model gets improved. The user gets the audio using the earphones and can stop the text reading by pressing the stop button.

Four identities have been trained for face recognition. When still images are given to the program, 100% accuracy has been verified as this is a small dataset. In real-time processing, accuracy is represented by the time taken to detect, process, and classify all faces in a frame. Durr [18] suggests that for a video stream, the time for real-time recognition is the time between consecutive frames. We have about 3FPS which is better than [10, 21] which are about 1–2 FPS. The probability of real-time face recognition averages to about 58% as shown in Fig. 4e whereas [21] and [22]

(a) (b) (c)

(d) (e)

Fig. 4 Screenshots of outputs of **a** face detection, **b** face recognition, **c** text detection, **d** text recognition, **e** face recognition labels and respective probabilities

got around 33%. This paper shows 75% improvement in the probability of real-time face recognition as compared to [21] and [22]. For text recognition, despite EAST and Tesseract being the most recommended software, there are always persistent difficulties in analyzing moving scenes; the model used in this paper given improved accuracy in recognition.

5 Conclusions

The goal of this paper is to build a video face and text recognition entity operating in real-time using a low power embedded system. This has been accomplished using Raspberry Pi, Raspberry Pi camera module, few push button switches, and earphones. The camera is used to get the real-time video, push button to select between face and text recognitions and earphones to hear the output. MTCNN and linear SVM are used for face detection and face classification, respectively. Similarly EAST, one of the best text detector models, is used for text detection, and Tesseract OCR is used for text recognition and word prediction. Python text-to-speech is used for conversion of recognized outputs to audio. English is used in the designed prototype out of the 116 available languages. The face recognition accuracy of the prototype reaches 100% for still images. The probability of real-time face recognition averages to about 58% which is much better than the existing systems. For text recognition, the model designed in this paper has given a fairly good accuracy in recognition of signboards.

The major limitation of this model is the recognizing text of smaller font size. Further work can be done to overcome this and to increase real-time accuracy of face recognition. This paper consolidates the most effective methods in the detection and recognition of face and text. These methods have been used to design and implement a novel hardware prototype. This portable device enables visually impaired user to move around without constraints.

References

1. Pascolini D, Mariotti SP (2011) Global estimates of visual impairment: 2010. Br J Ophthalmol
2. Chaudhry S, Chandra R (2015) Design of a mobile face recognition system for visually impaired persons. ArXiv, abs/1502.00756
3. Smith R (2007) An overview of the tesseract OCR engine. In: Ninth International conference on document analysis and recognition (ICDAR). IEEE Computer Society, pp 629–633
4. Aza, IV, Areni IS (2019) Face recognition using local binary pattern histogram for visually impaired people. In: 2019 International seminar on application for technology of information and communication (iSemantic). Semarang, Indonesia, pp 241–245
5. Ma M, Wang J (2018) Multi-view face detection and landmark localization based on MTCNN. In: 2018 Chinese Automation Congress (CAC), Xi'an, China, pp. 4200–4205
6. Zhang K et al (2016) Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process Lett 23(10):1499–1503
7. Zhao L, Song Y, Zhu Y, Zhang C, Zheng Y (2009) Face recognition based on multi-class SVM. In: 2009 Chinese control and decision conference. Guilin, pp 5871–5873
8. Pedregosa et al. (2011) Scikit-learn: machine learning in python. J Mach Learn Res JMLR 12:2825–2830
9. Zhou X et al (2017) EAST: An efficient and accurate scene text detector. In: Proceedings of conference on computer vision and pattern recognition CVPR, pp 2642–2651
10. Rajesh M et al (2017) Text recognition and face detection aid for visually impaired person using Raspberry PI. In: International conference on circuit, power and computing technologies (ICCPCT). Kollam, pp 1–5
11. Krishna S, Little G, Black J, Panchanathan S (2005) A wearable face recognition system for individuals with visual impairments. In: 7th international ACM SIGAC-CESS conference on computers and accessibility. Baltimore, MD, USA
12. Dunai L, Chillarón Pérez M, Peris-Fajarnés G, Lenua I (2015) Face detection and recognition application for android. https://doi.org/10.1109/IE-CON.2015.7392581
13. Rani K (2017) An audio aided smart vision system for visually impaired. In: International conference on nextgen electronic technologies: silicon to software, pp 22–25. https://doi.org/10.1109/ICNETS2.2017.8067889
14. Sasikumar A (2013) A text reading system for the visually disabled. Int J Res Comput Appl Manage
15. Balduzzi L, Fusco G, Odone F, Dini S, Mesiti M, Destrero A, Lovato A (2010) Low-cost face biometry for visually impaired users. In: Biometric measurements and systems for security and medical applications (BIOMS), IEEE workshop, pp 45–52
16. Mareeswari V, Ramaraj V, Uma Maheswari G, Sujatha R, Preethi E (2019) Accessibility using human face, object and text recognition for visually impaired people. Int J Innov Technol Explor Eng (IJITEE) 8(6). ISSN: 2278-3075
17. Graves A, Liwicki M, Fernandez S, Bertolami HB, Schmidhuber J (2008) A novel connectionist system for unconstrained handwriting recognition. IEEE Trans Pattern Anal Mach Intell 31(5):855–868

18. Dürr O, Pauchard Y, Browarnik D, Axthelm R, Loeser M (2015) Deep learning on a Raspberry Pi for real time face recognition. https://doi.org/10.2312/egp.20151036
19. Schroff F, Kalenichenko D, Philbin J (2015) FaceNet: a unified embedding for face recognition and clustering. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR). Crossref. Web
20. Amos B, Ludwiczuk B, Satyanarayanan M (2016) Openface: A general-purpose face recognition library with mobile applications. Technical Reports No. CMU-CS-16-118. CMU School of Computer Science
21. Rosebrock A (2018) Raspberry Pi face recognition pyimagesearch https://www.pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition. Last Accessed 01 Oct 2019
22. Rovai (2018) Real-time face recognition: an end-to-end project. https://www.hackster.io/mjrobot/real-time-face-recognition-an-end-to-end-project-a10826#toc-introduction-0. Last accessed 02 Jan 2020