

Self-Learning based training device for people with Alzheimer's and Visually challenged

Submitted in partial fulfilment of the requirements for the award of Bachelor of
Engineering degree in Electronics and Communication Engineering

By

RUTH BALAJI (39130382)

SUBHARAJA CHELLAM.A (39130443)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

**SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by
AICTE JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119.**

APRIL 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of RUTH BALAJI (39130382) and SUBHARAJA CHELLAM. A (39130443) who carried out the project entitled "**Self-Learning based training device for people with Alzheimer's and Visually challenged**".

Internal Guide

Dr. A. ARANGANATHAN M.Tech., Ph.D.,

Head of the Department

Dr. T. RAVI M.E., Ph.D.,

Submitted for Viva voice held on: 26.04.23

INTERNAL EXAMINER
EXTERNAL EXAMINER

DECLARATION

We, Ruth Balaji (39130382) and Subharaja Chellam.A (39130443) hereby declare that the Project Report entitled **“Self-Learning based training device for people with Alzheimer’s and Visually challenged”** done by us under the guidance of **“Dr.A.ARANGANATHAN”** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication.

DATE: 26.04.23

PLACE: Chennai

1.



2.



SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and successfully completing it. We are grateful to them.

We convey our thanks to **Dr.N.M.NANDHITHA., M.E., Ph.D. Dean, School of Electrical and Electronics Engineering** and **Dr.T.RAVI., M.E., Ph.D. Head of the department. Dept. of Electronics and Communication Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep gratitude to our project guide **Dr.A.ARANGANATHAN, M.Tech., Ph.D, Associate professor, Dept. of Electronics and Communication Engineering**, for his valuable guidance, suggestions, and constant encouragement for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non- teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Visual impairment is a common symptom of Alzheimer's disease (AD). Recent studies have demonstrated the potential of visual interventions to improve the functioning of AD patients. Therefore, clarification of the profile of visual deficits in AD and possible mechanisms underlying these deficits are needed. Alzheimer's disease (AD) is an irreversible, progressive brain disorder that slowly destroys memory and thinking skills. Alzheimer's is one of the most common causes of Dementia. Dementia means loss of Cognitive functioning – thinking, remembering, and reasoning – and behavioral ability to such an extent that it interferes with daily life. Image processing is widely used in the medical field in order to detect diseases and help doctors in decision-making based on observation. The paper aims to detect Alzheimer's disease at the earliest so that patients can be prevented before irreversible changes occur in the brain. We propose the image processing technique to process the Magnetic Resonance Imaging (MRI) of the brain. We Propose a camera- based detection for visually impaired or Alzheimer's people to identify the texts on the printed labels, or books, names of the objects in real-time, and names of the known person using face detection and Sign languages. To read the texts from books we're using an OCR (Optical Character Recognition) method to convert the Image to Text. Once the images are converted to text. By using Machine Learning we're going to detect the known faces, Real-time objects, and sign language to text. The Predicted output will be Converted to Audio using Text to Speech API.

TABLE OF CONTENT

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	vii
1.	INTRODUCTION	
	1.1. INTRODUCTION TO AI	1
	1.2. ADVANTAGES AND DISADVANTAGES OF AI	1
	1.3. INTRODUCTION TO ALZHEIMER'S	2
	1.4. DOMAINS	2
	PROBLEM STATEMENT	3
	OBJECTIVE	3
2.	LITERATURE SURVEY	5
3.	INVESTIGATION	
	3.1. EXISTING AND PROPOSED METHOD	28
	3.2. DIFFERENCE BETWEEN EXISTING AND PROPOSED METHOD	38
	3.3. ADVANTAGES OF PROPOSED METHOD	36
4.	RESULT AND ANALYSIS	39
5.	CONCLUSION	45
	REFERENCE	46
	APPENDIX	
	A. SOURCE CODE	48
	B. SCREENSHOTS	59
	C. RESEARCH PAPER	60

TABLE OF FIGURE

FIGURE NO.	TITLE	PAGE NO.
Figure 1.1	Global AI market size	2
Figure 3.1	Block diagram	27
Figure 3.2	Face detection system architecture	27
Figure 3.3	LBP operation	29
Figure 3.4	OCR conversion process and system AI architecture	30
Figure 3.5	Real-time object detection using SSD mobile net	31
Figure 3.6	Sign-to-speech using TensorFlow and Keras model	31
Figure 3.7	SSD mobile net architecture	32
Figure 4.1	MRI images	37
Figure 4.2	The input of Facial Recognition	37
Figure 4.3	The output of Facial Recognition	38
Figure 4.4	OCR input	38
Figure 4.5	OCR output	39
Figure 4.6	Real-time object detection	40
Figure 4.7	Sign to Speech	42

CHAPTER 1

INTRODUCTION

1.1. Introduction to Artificial Intelligence

Artificial intelligence (AI) is the ability of machines to replicate or enhance human intellect, such as reasoning and learning from experience. Artificial intelligence has been used in computer programs for years, but it is now applied to many other products and services. For example, some digital cameras can determine what objects are present in an image using an artificial intelligence software. In addition, experts predict many more innovative uses for artificial intelligence in the future, including smart electric grids. AI uses techniques from probability theory, economics, and algorithm design to solve practical problems. In addition, the AI field draws upon computer science, mathematics, psychology, and linguistics. Computer science provides tools for designing and building algorithms, while mathematics offers tools for modeling and solving the resulting optimization problems.

1.2. Advantages and Disadvantages of Artificial Intelligence

Advantages: -

Reduction in Human Error, Takes risks instead of Humans,
Helping in Repetitive jobs and Faster Decisions.

Disadvantages: -

High costs of creation, Making humans lazy, Unemployment
and no emotions .

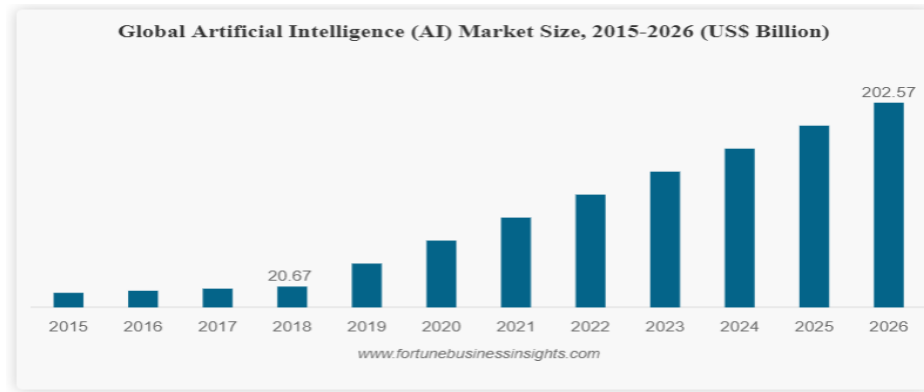


Figure 1.1. Global AI market size

1.3. Introduction to Alzheimer's

Alzheimer's disease is one of more advanced stage of death rate disease in India and it is an irreversible, progressive neurological brain disorder which can be multifaceted disease that slowly break downs the brain cells, which may cause memory loss and thinking capability skill losses, and even though simply tasks mightnot be possible to remember. This is known as dementia. This disease grows progressively worse in situation neurodegenerative type of dementia which can't be diagnose this type of disease in medical assessment medical assessment and which contains patient history MMSE and physical and neurobiological examinationneeded. To exam The MRI table and scanning of both structural and hippocampus and cerebral context and enlargement of different stages of disease can be identified.

1.4. Domains

1. Image Processing

Image processing allows us to transform and manipulate thousands of imagesat a time and extract useful insights from them. It has a

wide range of applications in almost every field. Python is one of the widely used programming languages for this purpose.

2. Machine Learning

Machine Learning is making the computer learn by studying data and statistics. Machine Learning is a step into the direction of artificial intelligence (AI). Machine Learning is a program that analyses data and learns to predict the outcome.

PROBLEM STATEMENT

Alzheimer's disease is a progressive neurologic disorder that causes the brain to shrink (atrophy) and brain cells to die. Alzheimer's disease is the most common cause of dementia a continuous decline in thinking, behavioral and social skills that affect a person's ability to function independently. Alzheimer's patients had to be on 24 hours assistance and monitoring as they can't manage on their own. So, in order to help them, we have proposed a system in such a way that Alzheimer's disease can be identified at an early stage using the image processing technique to process an MRI of the brain. We have added a few more things that could help them to recognize family members using Face recognition and sign language, identifying objects and texts using Object character recognition and the output will be converted to Audio using Text to speech API. So, in order to develop these ideas, we are using a smart and artificial intelligence camera-based detection system for visually impaired and Alzheimer's patients.

OBJECTIVE

The main objective is to predict Alzheimer's in the early stage and also provide them with support using Machine Learning Algorithms and provides solution efficiently. This Concept is designed to suit people with Dementia (Alzheimer's) and people who are visually impaired. We Propose a camera- based detection for visually impaired or

Alzheimer's people to identify the texts on the printed labels, or books, names of the objects in real-time, and names of the known person using face detection and Sign languages. To read the texts from books we're using an OCR (Optical Character Recognition) method to convert the Image to Text. Once the images are converted to text. By using Machine Learning we're going to detect the known faces, Real-time objects, and sign language to text. The Predicted output will be Converted to Audio using Text to Speech API.

CHAPTER 2

LITERATURE SURVEY

The period between 1940 and 1960 was strongly marked by the conjunction of technological developments (of which the Second World War was an accelerator) and the desire to understand how to bring together the functioning of machines and organic beings. For Norbert Wiener, a pioneer in cybernetics, the aim was to unify mathematical theory, electronics, and automation as "a whole theory of control and communication, both in animals and machines". Just before, the first mathematical and computer model of the biological neuron (formal neuron) had been developed by Warren McCulloch and Walter Pitts as early as 1943.

At the beginning of 1950, John Von Neumann and Alan Turing did not create the term AI but were the founding fathers of the technology behind it: they made the transition from computers to 19th-century decimal logic (which thus dealt with values from 0 to 9) and machines to binary logic (which rely on Boolean algebra, dealing with more or less important chains of 0 or 1). The two researchers thus formalized the architecture of our contemporary computers and demonstrated that it was universal machine, capable of executing what is programmed. Turing, on the other hand, raised the question of the possible intelligence of a machine for the first time in his famous 1950 article "Computing Machinery and Intelligence" and described a "game of imitation", where a human should be able to distinguish in a teletype dialogue whether he is talking to a man or a machine. However controversial this article may be (this "Turing test" does not appear to qualify for many experts), it will often be cited as being the source of the questioning of the boundary between the human and the machine.

Two factors explain the new boom in the discipline around 2010.

First of all, access to massive volumes of data. To be able to use algorithms for image classification and cat recognition, for example, it was previously necessary to carry out sampling yourself. Today, a simple search on Google can find millions.

Then the discovery of the very high efficiency of computer graphics card processors to accelerate the calculation of learning algorithms. The process is very iterative, it could take weeks before 2010 to process the entire sample. The computing power of these cards (capable of more than a thousand billion transactions per second) has enabled considerable progress at a limited financial cost (less than 1000 euros per card).

Jongkil Hyun (2021) et al. Face recognition applications are being widely studied owing to their extensive usability in the field of computer vision. However, processing an entire image requires a considerable amount of time. To reduce the processing time, several algorithms that extract only the face from the image during pre-processing are studied. Haar classifiers are extensively used for the hardware implementation of face detection algorithms that improve the processing speed of face classification. This paper proposes a Haar classifier-based face detection architecture that removes unnecessary iterations during classification to further improve the processing speed. The proposed architecture improves the processing speed by 4.46% compared to that of conventional Haar classifier-based face detection architectures, for face detection using a VGA image with 30 faces.

The proposed architecture tends to improve the processing speed as the number of faces in the image increases while matching the detection accuracy of conventional methods. Additionally, this architecture can be widely applied to classification algorithms that are based on iterations.

Keywords—FPGA, hardware architecture, face detection, Haar classifier

I. INTRODUCTION Computer vision and deep learning are becoming increasingly significant with the rapid development of computing

technology. Accordingly, image processing of human faces is being studied for identification purposes and applications pertaining to augmented reality and internet of things [1–3]. However, performing image processing on an entire input image has the drawbacks of difficulties in realtime operation and implementation using small devices; this is because image processing consumes a considerable amount of time and power owing to the large amount of computation involved. To overcome these drawbacks, the process of cropping only the facial area from the entire image is necessary prior to implementation.

Several studies use face detection algorithms to classify only the facial area within an entire image, which is subsequently used as an input for image processing to improve the efficiency of the entire image processing system [4–7]. Face detection algorithms are classified into feature-based [4], appearance-based [5], knowledge-based [6], and template-matching [7] methods. Among these algorithm-based methods, the feature-based method can detect the facial area in real-time because it utilizes low-level features, such as color and shape. The Haar classifier algorithm in the Viola–Jones Object Detection Framework is a feature-based method that detects human faces using the features trained by the AdaBoost machine learning technique [8]. The Viola–Jones algorithm has the advantage of being able to detect not only the frontal face, but also the side of the face or the accessories worn, depending on the learning data; further, this algorithm entails a low computational cost and ensures a high face detection rate. Moreover, because the classification of the Viola–Jones face detection algorithm is relatively simpler than the other algorithms, several studies are attempting to further reduce the power consumption and improve the processing speed by implementing the algorithm on hardware [9].

In one of the studies that implemented the Viola–Jones face detection algorithm on hardware, sub window and an image-resizing method were used to detect multiple faces in one image; further, an architecture that considerably improves the face detection processing speed through the pipelining of classifiers was proposed [9]. The face detection hardware architecture in [9] exhibits a processing speed approximately 35 times faster

than that of the software-based Viola–Jones algorithm. Despite several performance improvements, the processing speed is still limited because the detection operation needs to be performed repeatedly for all the sub-windows in the image. Therefore, we propose a face detection hardware architecture using a novel skip scheme that reduces the total number of iterations is reduced. Consequently, the processing speed of the proposed architecture is improved.

Bharath Tej Chinimilli (2020) et al. The attendance system is used to track and monitor whether a student attends a class. There are different types of attendance systems like Biometric-based, Radiofrequency card based, face recognition based and old paper-based attendance system. Out of them all, a Face recognition based attendance system is more secure and time-saving. There are several research papers focusing on only the recognition rate of students. This research focusing on a face recognition based attendance system with getting a less false- positive rate using a threshold to confidence i.e. euclidean distance value while detecting unknown persons and save their images.

Compare to other euclidean distance-based algorithms like Eigenfaces and Fisherfaces, Local Binary Pattern Histogram (LBPH) algorithm is better [11]. We used Haar cascade for face detection because of their robustness and LBPH algorithm for face recognition. It is robust against monotonic grayscale transformations. Scenarios such as face recognition rate, false-positive rate for that and false-positive rate with and without using a threshold in detecting unknown persons are considered to evaluate our system. We got face recognition rate of students is 77% and its false-positive rate is 28%. This system is recognizing students even when students are wearing glasses or grown a beard. Face Recognition of unknown persons is nearly 60% for both with and without applying threshold value. Its false-positive rate is 14% and 30% with and without applying threshold respectively. Index Terms—Face detection, Face recognition, LBPH algorithm, Harr Cascade. I.

Control of machines and their process with various technologies based on computer software is called automation. In this modern age, these advancements have proven to increase accuracy and also helping us to improve our livelihood. Innovations such as these save lots of labour work. One advancement in the field of automation is the Automated Attendance system replacing the old and traditional attendance marking. The paper-based method of marking attendance is time-consuming and its complexity increases with the increase of overall strength. This case is nullified in this automated version as it saves time and an additional bonus comes with security as it also helps to prevent proxy of attendance.

The objective of our proposed system is to create a face recognition based attendance system with getting a less falsepositive rate in detecting unknown persons by applying a threshold and save their images. We used Haar cascade for face detection because of their robustness and LBPH algorithm for face recognition. It is robust against monotonic grayscale transformations. Our System even detects and saves the images of any unknown person in the class whose Information is not there in the database.

Gurlove Singh (2020) et al. While recognizing any individual, the most important attribute is face. It serves as an individual identity of everyone and therefore face recognition helps in authenticating any person's identity using his personal characteristics. The whole procedure for authenticating any face data is sub- divided into two phases, in the first phase, the face detection is done quickly except for those cases in which the object is placed quite far, followed by this the second phase is initiated in which the face is recognized as an individual. Then the whole process is repeated thereby helping in developing a face recognition model which is considered to be one of the most extremely deliberated biometric technology. Basically, there are two type of techniques that are currently being followed in face recognition pattern that is, the Eigenface method and the Fisherface method. The Eigenface method basically make use of the PCA (Principal

Component Analysis) to minimize the face dimensional space of the facial features.

The area of concern of this paper is using the digital image processing to develop a face recognition system. After the face is detected the main task of face recognition system starts as to identify the known or unknown face and act accordingly. Often people are mistaken by the term face detection whereas face recognition means on the other hand is to authenticate a given face data based on the stored face data in the database. Once the face data matches with the database then the system is authenticated [1][2]. There are different approaches that are being followed in the whole process of face recognition. Each process has its own pros and cons, and has also some limitations which makes it different from other approaches [5].

Shadman Sakib (2020) et al. Diagnosing Brain Tumor with the aid of Magnetic Resonance Imaging (MRI) has gained enormous prominence over the years primarily in the field of medical science. Detection and/or partitioning of brain tumors solely with the aid of MR imaging is achieved at the cost of immense time and effort and demands a lot of expertise from engaged personnel. This substantiates the necessity of fabricating an autonomous model brain tumor diagnosis. Our work involves the implementation of a deep convolutional neural network (DCNN) for diagnosing brain tumor from MR images. The dataset, used in this paper, consists of 253 brain MR images where 155 images are reported to have tumors.

Our model can single out the MR images with tumors with an overall accuracy of 96%. The model outperformed the existing conventional methods for the diagnosis of brain tumor in the test dataset (Precision = 0.93, Sensitivity = 1.00, and F1-score = 0.97). Moreover, the average precision-recall score of the proposed model is 0.93, Cohen's Kappa 0.91, and AUC 0.95.

Therefore, the proposed model can be helpful for clinical experts to verify whether the patient has a brain tumor and, consequently, accelerate the treatment procedure. A tumor results from an uncontrolled division of abnormal cells forming a mass that can hamper the normal functionality of the tissue or organ [1]. Tumors are distinguished with their origin as basis and cell types. The brain manifests early stages of tumor mostly in the cerebrum area, whereas secondary ones (metastatic) find its way to the brain from other parts of the body [2]. Tumors may be malignant (high-grade) which are cancerous or benign (lowgrade). Compared to a benign brain tumor, a malignant brain tumor grows very rapidly and is more prone to invade adjacent tissues. Thus, a primary malignant brain tumor has a dreary prognosis and considerably reduces cognitive function and quality of life [3]. Cancers pertaining to the brain and various nervous system rank tenth in the leading causes of mortality and is considered as the 3rd most prevalent cancer among teenagers and adults [4]. The five-year survival rate for male and female patients with a cancerous brain is 34% and 36% respectively. Gliomas, the most prevalent cerebral tumor type in adults, account for almost 80% of malignant cases [2], [3].

The causes of a brain tumor can be attributed to environmental aspects such as excessive usage of artificial chemicals or genetic factors. Treatment options include radiotherapy, chemotherapy, and surgical procedure. The earlier a brain tumor is detected the higher the chances of survival and the wider the treatment options. Various methods may be employed to diagnose a brain tumor such as MRI scan, BIOPSY and SPECT (Single Photon Emission Computed Tomography) scan. Magnetic Resonance Imaging (MRI) is the most prevalent method due to its non-intrusive imaging modality that provides distinctive tissue contrast. It is also highly conformable in terms of normalization of tissue contrast providing minute details of interest.

However, limitations are faced because of the uncertain, random, and irregular size, shape, and locality of brain tumors. Also, non-autonomous partitioning of the tumors comes at the cost of a lot of time and is a laborintensive, cumbersome, and largely subjective task given the amount of data to be handled thus reducing accuracy. The algorithms applied for segmentation of brain tumor can be categorized as traditional or non-autonomous techniques and techniques pertaining to deep-learning. The former includes regularized non-negative matrix factorization (NMF), Computer-aided Diagnosis (CAD) systems involving computation methods such as K-means clustering and Principal Component Analysis (PCA) and Support Vector Machines (SVM) [6], [7].

Referencing cases scanned by T1-weighted and T2-weighted sequences, Hsieh et al. [8] fuzzy clustering was coupled with region-growing which resulted in a 73% segmentation accuracy. Albarracín et al. [9] evaluated various clustering algorithms for brain tumor segmentation revealing that k-means, fuzzy k-means, and Gaussian mixture models were inferior to 77% accuracy achieved by the Gaussian hidden Markov random field algorithm. Soltaninejad et al. [10] graded brain tumors using SVM as a basis by utilizing aspects of 38 first-order or second- order statistical measurements. Recently, the deep learning (DL) based techniques developed are enticing to researchers as they are automated using a complex set of features directly from data [11]. In the study [12], tumor segmentation was performed using the LinkNet network. At first, all seven training datasets were segmented using Single LinkNet network later, a method for CNN was developed for the autonomous extraction of most prevalent tumor types with no preprocessing requirements. The dice score obtained for a single network was 0.73 and multiple systems were 0.79. The study [13], also use multiple convolutional neural networks for brain tumor segmentation by inputting the 3D MRI image voxels into a two dimensional CNN model. The output is a brain tumor segmentation, not a classification.

However, the number of images used in this method for the training process is relatively less raising acceptability queries for datasets with a large number of images. Le et al.[15] performed studies on the ImageNet dataset. Using the concepts of CNNs, they came up with a high-performance object detector which achieved a 70% relatively improved accuracy compared to earlier researches on the ImageNet dataset. Here, a proposition involving a deep convolutional neural network (DCNN) based on a pre-trained network is made. VGG-16 classification model pre-trained on the ImageNet dataset is utilized. The dataset employed is constituted of two classes of brain MRI images. The model yields better results than conventional methods. The overall article is laid out in the following manner: Section II elaborates details of the methodology propositioned. The experimental results, performance evaluation, and the result comparison with the conventional methods are outlined in Section III. Finally, section IV pertains to the conclusion.

Brijeshwar Dessai (2019) et al, Handwritten Character Recognition is one of the most challenging and demanding area of interest for researchers in domains of pattern recognition and image processing. Many researchers have worked with recognition of characters of different languages but there is comparatively less work carried for Devanagari Script. In past few years, however the work carried out in this direction is increasing to a great extent. Handwritten Devanagari Character Recognition is more challenging in comparison to the recognition of the Roman characters. The complexity is mostly due to the presence of a header line known as shirorekha that connects the Devanagari characters to form a word. The presence of this header line makes the segmentation process of characters more difficult. There is uniqueness to the handwriting styles of every individual which adds to the complexity. In this paper, we propose development of Convolutional Neural Network (CNN) based Optical Character Recognition system (OCR) for Handwritten Devanagari Script which is observed to recognize the characters accurately.

Character recognition deals with recognizing the characters accurately from a document. Character recognition is done either from printed documents or handwritten documents. Character recognition is distinguished into two classes: on-line character recognition and off-line character recognition. In off-line recognition characters are optically captured by a scanner and the writing is available as an image. In on-line recognition the user writes directly on a pen-based input device and curves and styles of writing are identified as written to recognize the character.

Handwritten character recognition is a technique to convert handwritten document into a text file which can be edited. Handwritten character recognition is a very complex task in contrast to printed character recognition as the characters are written in various styles and sizes. Every individual has a unique way of writing. As a result it becomes very tough to achieve 100% accuracy as the accuracy is based entirely on user handwriting styles. Many researches have been carried out in this direction but achieving 100% accuracy is not possible. Optical Character Recognition (OCR) is a method which takes an image as input scans it and transforms it to an editable form. There are various systems in the market which scan printed data and some scan even handwritten data. But when it comes to the accuracy of the output most existing software's fail to provide enough accuracy as needed. The OCR systems which are present mostly make use of machine learning techniques such as support vector machines (SVMs) and artificial neural networks (ANNs) for the recognition of characters.

These classifier models used in the OCR systems require that input images are down sampled to reduce the complexity. This down sampling results in the loss of some important information. SVM classifier for instance provides different recognition accuracy for characters with different font styles. Therefore, in order to build an efficient OCR system which can make use of data in the form of images we propose use of convolutional neural networks in contrast with use of SVMs and ANNs. Convnets have proved to

be very useful in high quality image and video recognition methods. Convolutional Neural Network (CNN) are class of Deep Neural Network which need smaller parameters and are easier to train.

By changing the number of hidden layers and trainable parameters in each layer of CNN their ability to model input dataset can be varied. CNN have lesser trainable parameters in comparison to a fully connected feed-forward network. CNNs have notion of local receptive field, weight replication and subsampling which allows us in having some amount of shift and distortion changes in data. A number of convolutional and subsampling layers are present in CNNs for image processing. These layers are finally terminated by fully connected layer where recognition of data take place. Deep convolutional neural networks have a compact and cheaper mechanism as opposed to standard feed forward network of same depth. But a lot of processing and computation is required to train a CNN, along with a large labelled dataset.

G.Vinod (2022) et al, Real time object detection in traffic surveillance is one of the latest topics in today's world using Region based Convolutional Neural Networks algorithm in comparison with Convolutional Neural Networks. Real-Time Object Detection is performed using Regional Convolutional Neural Networks (N=78) over Convolutional Neural Networks (N=78) with the split size of training and testing dataset 70% and 30% respectively. Regional Convolutional Neural Networks had significantly better accuracy (75.6%) compared to Convolutional Neural Networks (47.7%) and attained significance value of $p=0.041$. Regional Convolutional Neural Networks achieved significantly better object detection than Convolutional Neural Networks for identifying real-time objects in traffic surveillance. Increasingly, traffic control is a critical concern in quickly expanding cities. Due to bad traffic management a lot of manhours are being wasted. By using Regional Convolutional Neural Networks (R-CNN) it detects multiple objects in a single-shot within the input closed circuit television video [1]. An intelligent traffic surveillance system plays a vital role

in today's world to avoid traffic jams and accidents [2]. The real time object detection model can be implemented in real time applications like driverless cars, vehicle tracking systems [3], object detection [4] and vehicle detection systems. Six papers were discovered in the IEEE Xplore database, while five papers were discovered in the Google Scholar database. convolutional neural networks have been developed to address this issue [2].

The deformable convolutional structure was developed to overcome the problem of convolutional neural networks' fixed grid structure [5] sought to increase real-time object detection accuracy. To begin, a subset of the COCO dataset is used to construct a benchmark database. This benchmark database was created primarily to assess the performance of object detection algorithms in real-time object detection. [6] created and implemented a real-time object recognition system for traffic surveillance based on an R-CNN method that shares full-image convolutional features with a detection network, allowing for practically cost-free region recommendations [1] was built and developed for a Region-based Convolutional Neural Network (R-CNN) used in the MATLAB simulation environment for real-time object detection. This includes the fundamentally new R- CNN model, which employs the SVM algorithm for classification. We've already worked on numerous research projects across multiple fields, so we're well-versed in the research process. Research has shown that there is an association between a person's age and his or her ability to learn [7,8, 9, 10]. Now, we're encouraged to continue this project due to the rising popularity of it. Based on literature survey it can be summarized that the existing system found the lack of accuracy, complex, background, noise disturbance, occlusion, low-resolution and time complexity.

Ming Liu (2022) et al, With the reduction of Unmanned Aerial Vehicle (UAV) hardware cost and the development of deep learning algorithm, the real-time object detection algorithm applied in UAV vision has great advantages in many fields. However, due to the limited energy consumption and computing power of embedded devices used in the drones and the

variable object scales and complex backgrounds in the UAV vision restrict the applications in object detection based on the drones. In this paper, we optimized the generation of anchor boxes, introduced a new module to increase the receptive field to improve the detection of small targets, and used adaptively spatial feature fusion in the feature pyramid to increase feature fusion of multi-scale features. At last we pruned the model to make it lighter and faster, and got the Average Precision (AP) of 89.7% for UAVcar aerial images and the speed of 35.7 FPS by running on Neural Processing Units (NPUs), which proves the feasibility of the intelligent object detection algorithm's efficient processing in hardware resource-limited environment.

In recent years, UAV has played an irreplaceable role in many fields including military and civilian fields relying on the virtue of their lightweight, Maneuverability, flexible movement and low energy consumption. Real-Time Object Detection is an important task in applications of UAV, and has been well-explored. With the rapid development of computer vision and artificial intelligence technology, among many target detection algorithms, the method based on deep learning has been widely used because it does not need the Feature Engineering, and has higher precision.

The object detection algorithms based on deep learning are mainly divided into two stage methods and one stage methods. The typical algorithms of the former are R-CNN, as well as the improved Fast R-CNN and Faster R-CNN. The detection speed of these algorithms is too slow to meet the requirements of real-time. One stage algorithm does not need to extract candidate regions first, and the speed is greatly improved. Typical algorithms include Yolo series and SSD. However, the usual object detection methods based on deep learning need a large amount of calculation and large storage spaces for storing intermediate results. Due to the limited load capacity of UAV, for example, the maximum load weight of Inspire 2 professional UAV produced by Dajiang company is 810 g. The UAV platform for car aerial images object detection can only use the embedded system driven by battery and limited computing resources.

Therefore, the limitation of hardware cost and transmission delay is still the primary problem for object detection tasks in UAV vision.

To sum up, the difficulties of deploying deep neural network algorithm to achieve effective object detection based on UAV focus on two aspects [6]: (1) UAV vision has special mission scenarios and object characteristics. Different from the mainstream objection detection, most of these task scenes adopt the conventional horizontal angle of view, and the object scale is larger and the characteristics are more obvious. The content of the drone shots is complex, the size of the target to be inspected is small, and the features are ambiguous, which are not the task type targeted by the mainstream algorithm network design. (2) The computing power resources of UAV platform are limited.

Panca Mudjirahardjo (2018) et al, The first task in motion detection and recognition is moving object extraction. After extract it, we perform representation model. Failure in moving object extraction will contribute an error in detection and recognition process. This extraction is performed by processing of several frames to distinguish foreground from the background. In this paper we developed a simple extraction method which was suitable for real time application. The first stage, we performed moving object detection by using frames subtraction. This yielded boundary label in the motion area. This boundary label couldn't extract the whole moving object. For every row of image, we calculate the boundary density function. The probability of pixels in this density is used to extract them as foreground. The experimental result, our system can extract the moving object in computation time of 71 - 80 ms, video rate of 15 frames per second (fps) and image frame size of 640x480 pixels, which is suitable for real time application. Prior to detect an object and recognize the object or a motion, we need to extract the moving object.

This task is important to reduce the searching time to the object. The conventional method to detect an object is by scanning the whole image with the particular window. This will consume the time. By extracting the moving object, this is considered to reduce the searching time, especially in real time application. In static camera, the simple method to detect a moving object is by subtracting the subsequent frames. The recent frame is subtracted by the previous frame yields the boundary which shows the motion exist. This boundary shows the different of pixel intensity caused by a moving parts. Furthermore, we need to know what the moving object is. This is performed by analyzing pixels inside the motion boundary. Whether the pixels is as foreground or as background. This method is done by [1]-[3]. The other motion detection method by using background model [4][5]. By using this method, it yields the whole of moving object, not only the edge detection. This is a moving object extraction. This method, however, requires several frames more than two frames. Many frames processing will give a better background model. However, this processing is not suitable for real time application. If we use the frame subtraction method, a further process is required to extract the motion object, not only a boundary label. This object extraction is required by motion recognition.

There are several methods to extract the motion object, the simple method is area filling process [1]-[3][14]. By this method, first, they fill the right area and left area of boundary label, then they performed intersection between them. There are some automatic foreground extraction methods, for one frame processing [6-10] or sequence frames processing [11-14]. Tong *et al.* [6] performed field extraction based on the color distance between a pixel and the mean value of the field. They used only hue and saturation components in HSV color space. Huang *et al.* [7] employed a histogram learning technique to detect the background pixels. Color models were learned for foreground and background pixels, using a training set of soccer videos.

The color model was an RGB color histogram with N bins per channel in the RGB color space. Ali et al. [8] employed their algorithm for soccer field and assumed the background color was green color. They determined the background pixel if the color intensity of $G > R > B$. Leibe et al [9] to detect and localize objects considered object categorization and figure-ground segmentation as two interleaved processes. Their approach was a highly flexible learned representation for object shape that can combine the information observed on different training examples in a probabilistic extension of the Generalized Hough Transform. Mudjirahardjo et al [10] extracted the foreground from background by using Euclidean distance function of the determined pixels as foreground and the surrounding pixels. If the distance was less than a threshold, then the pixel was a foreground.

They calculated the distance in RGB and HSV color space. Chakraborty et al. [11] employed a background subtraction method along with a frame differencing method using three consecutive frames were employed for segmentation on the moving objects in the foreground. Durus [7] accomplished the segmentation of background by using the dominant color distribution. They calculated the mean of the peak intensity values of the histogram of several frames. Then they measured each pixel intensity respect to the mean value by using Mahalanobis distance. If the distance less than the variance, then the pixel was supposed to be background. Mudjirahardjo et al. [13] extracted the interest object in dynamic background by using velocity histogram based on Harris corner detector. Lucas-Kanade tracker and shift histogram based on motion history image (MHI).

Chit San Lwin (2020) et al, The recognition of characters from similar character groups has gained popularity because it considers only corresponding similar character groups instead of all possible characters when one character recognized.

In this paper, we present a novel Myanmar optical character recognition (MOCR) system for handwritten text. Firstly, the system uses tensor- based subspace learning (TSL) to reduce dimension space for only relevant features. The system then gathers similar character groups using the *K*-means clustering algorithm. The convolutional neural network (CNN) is adopted to discriminate the input character string against the groups of characters that are the most similar to it. The system finally classifies and produces the results as the recognized characters in the phrase level. The experimental works are conducted to evaluate the results in terms of precision, recall, processing time, and error rate. The results have shown that they are superior to other contemporary MOCR works.

Optical character recognition (OCR) is the process of transforming character images into digital text. This process allows us to translate a document into an editable digital format to be used on electronic devices such as computers, mobile phones, etc. Since many recent years, a lot of researches have investigated OCR transforming process for some languages such as the Thai language [1], Quranic characters [2], Chinese [3, 4], Indic [5, 6], English [7], etc. However, it is still an ongoing work for Myanmar characters to reach a standard model. Traditional OCR technique can be seen in recognition of the printed or handwritten documents. The difference from printed document recognition, handwritten document recognition has more challenges to recognize a character that has many forms of variations such as shapes, sizes, and different writing styles due to the possibility of receiving handwritten styles from different people. Handwritten OCR researches have been presented in offline or online recognition by the studies [1, 3, 4, 7, 8].

Although they had proposed some efficient recognition algorithms and techniques, there are still some gaps for Myanmar handwritten characters to be compatible with them. Therefore, this paper performs the recognition of phrase level Myanmar characters based on its similar handwritten character groups with the following contributions. Firstly, prior to

grouping similar characters, the system reduces the image dimension for more relevant features. It is executed with a tensor-based subspace learning (TSL) technique. Grouping similar patterns of the input characters are executed using the *K*-means clustering algorithm. The system finally feeds *K*-groups feature vectors into a fully connected convolutional neural network to classify the text of the input image. The rest of the paper is arranged as follows. The literature review is discussed in Section 2 and the proposed system is widely explained in Section 3. The experimental works and their results are presented in Section 4 and the paper is finally concluded in Section 5.

Alexander Wong (2018) et al, Object detection is a major challenge in computer vision, involving both object classification and object localization within a scene. While deep neural networks have been shown in recent years to yield very powerful techniques for tackling the challenge of object detection, one of the biggest challenges with enabling such object detection networks for widespread deployment on embedded devices is high computational and memory requirements. Recently, there has been an increasing focus in exploring small deep neural network architectures for object detection that are more suitable for embedded devices, such as Tiny YOLO and SqueezeDet. Inspired by the efficiency of the Fire microarchitecture introduced in SqueezeNet and the object detection performance of the single shot detection micro architecture introduced in SSD, this paper introduces Tiny SSD, a single-shot detection deep convolutional neural network for real-time embedded object detection that is composed of a highly optimized, non-uniform Fire subnetwork stack and a non-uniform sub-network stack of highly optimized SSD-based auxiliary convolutional feature layers designed specifically to minimize model size while maintaining object detection performance. The resulting Tiny SSD possess a model size of 2.3MB (~26X smaller than Tiny YOLO) while still achieving a map of 61.3% on VOC 2007 (~4.2% higher than Tiny YOLO).

These experimental results show that very small deep neural network architectures can be designed for real-time object detection that are well-suited for embedded scenarios. computer vision, as it involves a combination of object classification and object localization within a scene (see Figure 1). The advent of modern advances in deep learning [7], [6] has led to significant advances in object detection, with the majority of research focuses on designing increasingly more complex object detection networks for improved accuracy such as SSD [9], R-CNN [1], Mask R- CNN [2], and other extended variants of these networks [4], [8], [15].

Despite the fact that such object detection networks have showed state-of-the-art object detection accuracies beyond what can be achieved by previous state-of-the-art methods, such networks are often intractable for use for embedded applications due to computational and memory constraints. In fact, even faster variants of these networks such as Faster R-CNN [13] are only capable of single-digit frame rates on a high-end graphics processing unit (GPU). As such, more efficient deep neural networks for real-time embedded object detection is highly desired given the large number of operational scenarios that such networks would enable, ranging from smartphones to aerial drones. Recently, there has been an increasing focus in exploring small deep neural network architectures for object detection that are more suitable for embedded devices. For example, Redmon et al. introduced YOLO [11] and YOLOv2 [12], which were designed with speed in mind and was able to achieve real-time object detection performance on a high-end Nvidia TitanX desktop GPU.

However, the model size of YOLO and YOLOv2 remains very large in size (753 MB and 193 MB, respectively), making them too large from a memory perspective for most embedded devices. Furthermore, their object detection speed drops considerably when running on embedded chips [14]. To address this issue, Tiny YOLO [10] was introduced where the network architecture was reduced considerably to greatly reduce model size (60.5 MB) as well as greatly reduce the number of floating point operations

required (just 6.97 billion operations) at a cost of object detection accuracy (57.1% on the twenty-category VOC 2017 test set). Similarly, Wu et al. introduced SqueezeDet [16], a fully convolutional neural network that leveraged the efficient Fire microarchitecture introduced in SqueezeNet [5] within an end-to-end object detection network architecture.

Given that the Fire microarchitecture is highly efficient, the resulting SqueezeDet had a reduced model size specifically for the purpose of autonomous driving. However, SqueezeDet has only been demonstrated for objection detection with limited object categories (only three) and thus its ability to handle larger number of categories have not been demonstrated. As such, the design of highly efficient deep neural network architectures that are well-suited for real-time embedded object detection while achieving improved object detection accuracy on a variety of object categories is still a challenge worth tackling. In an effort to achieve a fine balance between object detection accuracy and real-time embedded requirements (i.e., small model size and real-time embedded inference speed), we take inspiration from both the incredible efficiency of the Fire microarchitecture introduced in SqueezeNet [5] and the powerful object detection performance demonstrated by the single-shot detection macro architecture introduced in SSD [9].

The resulting network architecture achieved in this paper is Tiny SSD, a single-shot detection deep convolutional neural network designed specifically for real-time embedded object detection. Tiny SSD is composed of a non-uniform highly optimized Fire sub-network stack, which feeds into a non-uniform sub-network stack of highly optimized SSD-based auxiliary convolutional feature layers, designed specifically to minimize model size while retaining object detection performance. This paper is organized as follows. Section 2 describes the highly optimized Fire sub-network stack leveraged in the Tiny SSD network architecture. Section 3 describes the highly optimized sub-network stack of SSD-based convolutional feature layers used in the Tiny SSD network architecture.

Guodong Guo (2018) et al, Recently deep learning has become dominant in face recognition and many other artificial intelligence areas. We raise a question: Can deep learning truly solve the face recognition problem? If not, what is the challenge for deep learning methods in face recognition? We think that the face image quality issue might be one of the challenges for deep learning, especially in unconstrained face recognition. To investigate the problem, we partition face images into different qualities, and evaluate the recognition performance, using the state-of-the-art deep networks. Some interesting results are obtained, and our studies can show directions to promote the deep learning methods towards high-accuracy and practical use in solving the hard problem of unconstrained face recognition.

Deep learning (DL) [1], [2] has recently become dominant in a wide variety of biometrics problems and many other artificial intelligence (AI) areas. One of the greatest successes of DL has been in face recognition (FR) where the accuracies have been improved greatly over the traditional methods [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. We raise a question: Can deep learning really solve the face recognition problem? Or, can we say that, given the great success of DL, the face recognition problem has been solved or almost solve. To answer this question, and get a better understanding of the performance of DL methods in FR, we perform an empirical study with designed experiments accordingly.

In face recognition, it is well-known that the hard problem is unconstrained face matching, in which we believe that the face image quality variations are probably the biggest issue that makes the problem hard. Based on this view, our conjecture is that the face image quality issue may still be a grand challenge even for the recently developed DL methods. In FR with traditional features, it is well-known that the face image quality has a big influence on recognition accuracy; In DL features, however, a large dataset with face images of different quality for each subject, is used to train the deep models. Will the quality still be an issue? In our empirical study, we design the face recognition experiments with matching across

different face image qualities, which is seldom done in an explicit way in previous face recognition approaches. In practice, however, one can meet the cross-quality face matching problem frequently.

For example, in the FBI's interstate photo system (IPS), millions. This was partly supported by a NSF-CITeR grant and a WV HEPC grant of mugshot photos could be matched to face images collected from social media web sites where the wild photos may have a wide variety of qualities. Since there is no existing face database, which is purposely assembled with annotated face image qualities, we partition some recent, public face databases into different face image qualities, using an automated face image quality assessment method. After the quality partition, the face images from the same subject are divided into different qualities, such as low, middle, and high. Then we can perform face recognition experiments across quality changes.

For the deep learning techniques, we select some representatives of the state-of-the-art. To avoid any bias in training and parameter tuning, we adopted the already-trained face models that have reported very high accuracies in the popular face database LFW (labeled faces in the wild) [16]. The contributions of our work include:

An important problem is raised for deep learning, through investigating the impact of face image quality changes on deep learning techniques;

"Annotations" of face image qualities are performed on two public face databases, which is for the *first time* to perform quality partition, to the best of our knowledge. This partition can be useful for examining the face image quality issue in unconstrained face recognition; The design of cross-quality face recognition protocols is useful to discover the real challenges in unconstrained face recognition, rather than simply saying "in the wild".

where a number of high-quality face images may be matched to each other with high accuracies.

An evaluation of the performance of the state-of-the-art deep learning techniques in cross-quality face recognition, disclosing the capability of deep learning methods in cross-quality face matching, an important problem but not well-studied yet, in unconstrained FR.

The paper is organized as follows. In Section II, we introduce the automated annotations of face image qualities on two public databases, using a face image quality assessment method; The protocols of cross-quality face matching are redesigned as well. In Section III, we briefly describe the representative deep models that we used for the evaluation. In Section IV, the FR evaluations are executed. A discussion is given in Section V, and finally we draw some conclusions.

CHAPTER 3

INVESTIGATION

3.1. EXISTING and PROPOSED METHODOLOGY

Existing System:

The features of the existing system called “Alzheimer’s daily companion” are as follows:

Free and immediate advice and tips for dealing with Alzheimer’s and dementia behaviors and situations. 24-hour assistance via toll-free Call or email submission. Access to free Alzheimer’s and other dementia caregiver resources and training materials. The Major Drawbacks of “Alzheimer’s daily companion” are as follows: Since Alzheimer’s people are already being affected by memory loss it becomes very difficult for them to send or check E-Mail or Call Toll-Free numbers for support.

Proposed System:

Our Proposed System aims to detect the Alzheimer’s disease at earliest so that patient can be prevented before irreversible changes occur in brain. We propose the image processing technique to process the Magnetic Resonance Imaging (MRI) of brain. Additionally, as a solution we also help them to recognize their family members and also bring sign to speech concept to support with their daily activities and emergency support using image processing with audio-based feedback.

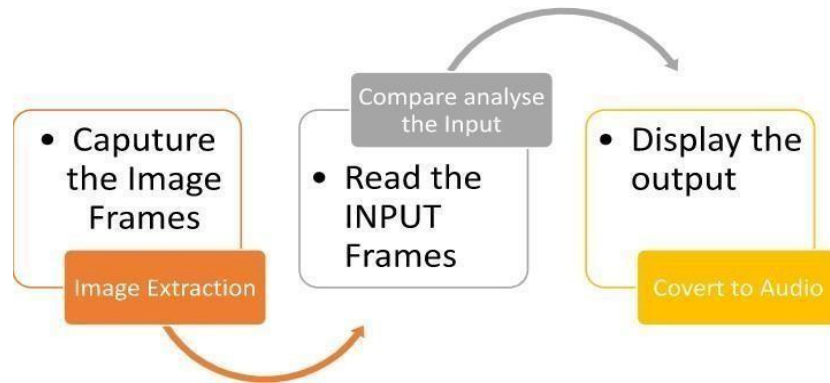


Figure 3.1. Block diagram

Face Detection System Architecture

Local Binary Patterns Histogram (LBPH)

Local Binary Patterns Histogram algorithm was proposed in 2006. It is based on local binary operator. It is widely used in facial recognition due to its computational simplicity and discriminative power.

The steps involved to achieve this are:

Face acquisition, feature extraction, creating dataset and classification

The LBPH algorithm is a part of OpenCV.

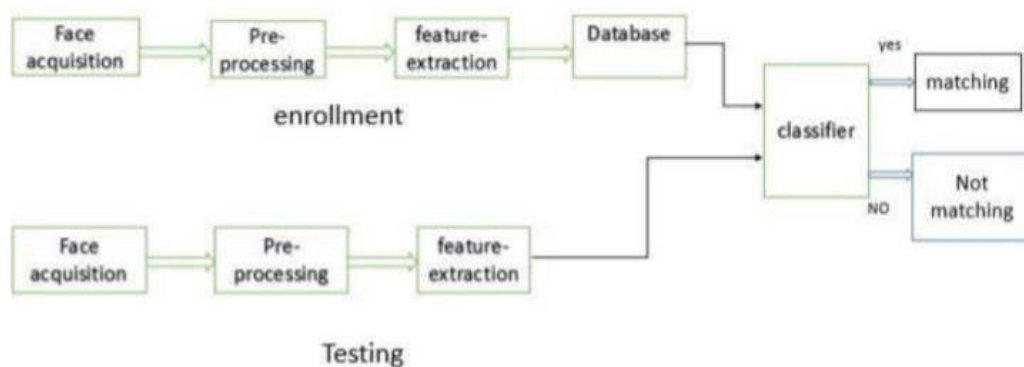


Figure 3.2. Face Detection System Architecture

1. Parameters: the LBPH uses 4 parameters:

Radius: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

Neighbors: the number of sample points to build the circular local Binar pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.

Grid X: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Grid Y: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameter's radius and neighbors.

The image below shows this procedure:

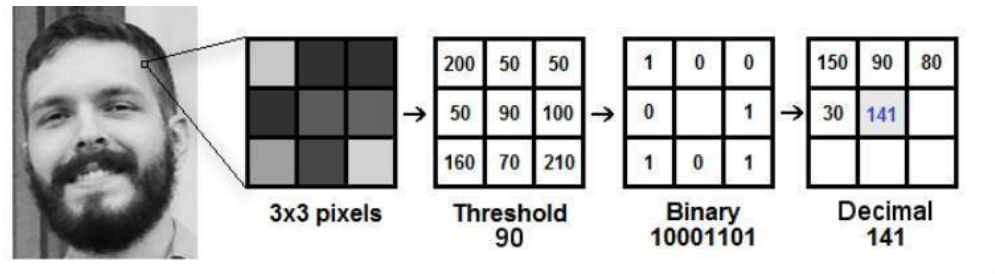


Figure 3.3. LBP operation

Based on the image above, let's break it into several small steps so we can understand it easily:

Suppose we have a facial image in grayscale. We can get part of this image as a window of 3x3 pixels. It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255). Then, we need to take the central value of the matrix to be used as the threshold. This value will be used to define the new values from the 8 neighbors. For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold. Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same. Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image. At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

Note: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.

OCR Conversion process and System AI Architecture

Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and “read” the text embedded in images. Python-tesseract is a wrapper for Google’s Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others. Additionally, if used as a script, Python-tesseract will print the recognized text instead of writing it to a file.

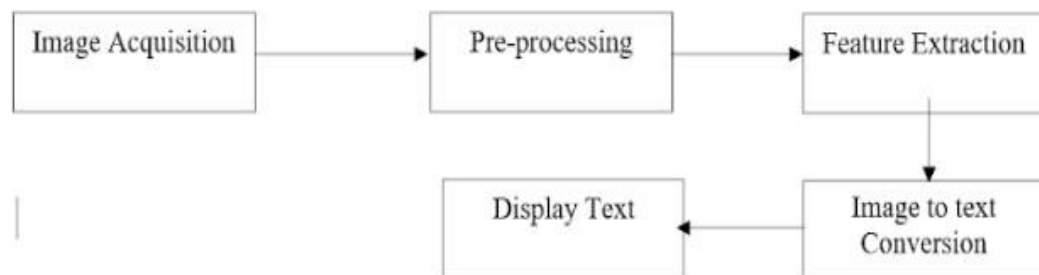


Figure 3.4. OCR conversion process and system AI architecture

Real-time Object Detection using SSD Mobile net

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Well-researched domains of object detection include face detection and Pedestrian Detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance.

Object detection as the term suggests is the procedure to detect the objects in real world. For example, dog, car, humans, birds etc. In this process we can detect the presence of any still object with much ease.

another great thing that can be done with it is that detection of multiple objects in a single frame can be done easily. For Example, in the image below the SSD model has detected mobile phone, laptop, coffee, glasses in a single shot. It detects different objects in a single shot.

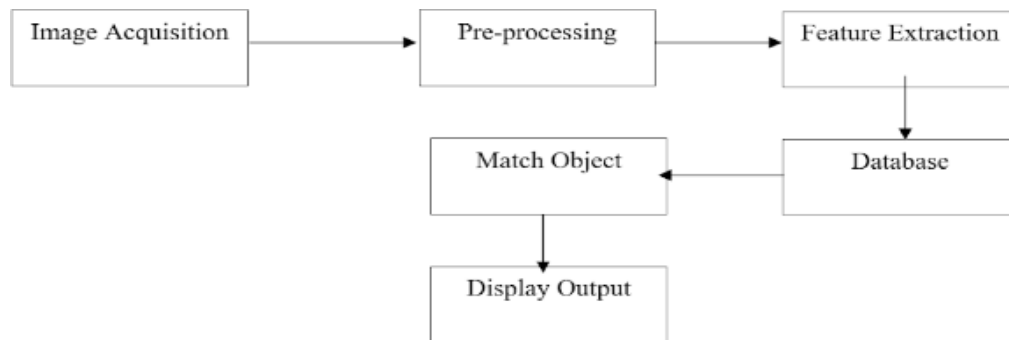


Figure 3.5. Real-time Object Detection using SSD Mobile net

Sign to Speech using TensorFlow and keras Pretrained Model

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming.

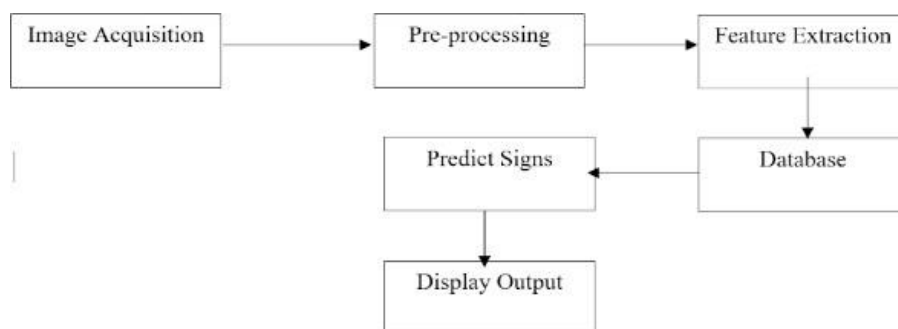


Figure 3.6. sign to speech using Tensorflow and keras pretrained model

SSD Mobile Net Architecture

The SSD architecture is a single convolution network that learns to predict bounding box locations and classify these locations in one pass. Hence, SSD can be trained end-to-end. The SSD network consists of base architecture (MobileNet in this case) followed by several convolution layers:

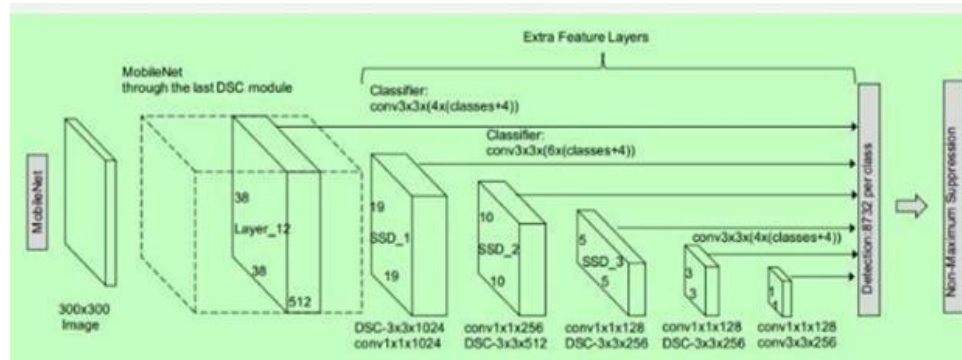


Figure 3.7. SSD Mobile net architecture

Text to Speech

gTTS (*Google Text-to-Speech*), a python library and CLI tool to Google translate's text to speech API. Write spoken mp3 data to a file, a file like object for further audio manipulation or stdout or simply pre generate google translate TTS request URLs to feed to an external program.

Features

Customizable speech-specific sentence tokenizer that allows for unlimited lengths of text to be read, all while keeping proper intonation, abbreviations, decimals and more. Customizable text pre-processors which can, for example, provide pronunciation corrections.

Audio Player

Audio player is a cross-platform Python 3 package for playing sounds (mp3, wav). It provides the key features of an audio player, such as opening a media file, playing (loop/block), pausing, resuming, stopping, and setting the playback volume.

Hardware

A camera is an optical instrument that can capture an image. Most cameras can capture 2D images, with some able to capture 3D images. At a basic level, most cameras consist of sealed boxes (the camera body), with a small hole (the aperture) that allows light through to capture an image on a light-sensitive surface (usually a digital sensor or photographic film). Cameras have various mechanisms to control how the light falls onto the light-sensitive surface. Lenses focus on the light entering the camera. The aperture can be narrowed or widened. A shutter mechanism determines the amount of time the photosensitive surface is exposed to light. The still image camera is the main instrument in the art of photography. Captured images may be reproduced later as part of the process of photography, digital imaging, or photographic printing. Similar artistic fields in the moving-image camera domain are film, videography, and cinematography.

Software

Python IDLE

IDLE (Integrated Development and Learning Environment) is an integrated development environment (IDE) for Python. The Python installer for Windows contains the IDLE module by default. IDLE is not available by default in Python distributions for Linux. It needs to be installed using the respective package managers.

IDLE can be used to execute a single statement just like Python Shell and also to create, modify, and execute Python scripts. IDLE provides a fully-featured text editor to create Python scripts that include features like syntax highlighting, autocompletion, and smart indent. It also has a debugger with stepping and breakpoints features.

Advantages

Free and Open-Source, Easy to Learn, Vast Libraries Support, Greater Productivity, Interpreted Language, Portability and Dynamically Typed.

Disadvantages

Poor Memory Efficiency, Slow Speed, Weak in Mobile Computing and Runtime Errors.

Open CV

OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify an image pattern and its various features we use vector space and perform mathematical operations on these features. The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python, and Java interfaces and supports Windows, Linux, Mac OS, Ios, and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

Advantages

It is available free of cost, It is fast, Low RAM usage (approx. 60-70 Mb) and Portable.

Disadvantages

Does not provide the same ease of use and takes a lot of time to display the image

Features of OpenCV Library

The use of the OpenCV library will allow you to:

Read and write images, Capture and save videos, Image processing such as filtering and transformation, Feature detection, Video or image object detection such as human body parts, cars, signage, etc, and Video analysis

3.2. DIFFERENCE BETWEEN EXISTING AND PROPOSED METHODOLOGY

Existing Methodology	Proposed Methodology
24 hours assistance.	Can handle themselves on their own.
Could not Identify Alzheimer's at an early stage.	Early Stage of identification of Alzheimer's.
Couldn't Recognize Family members on their own.	Helps recognize Family members on their own.
There wasn't any Optical Character Recognition.	Helps in recognizing objects and things by their own selves.
Could help them identify printed texts.	Helps in identifying text printed on labels or books.
No usage of Camera	Camera is used
No Image processing technique was proposed to process the Magnetic Resonance Imaging of the brain.	The image processing technique is proposed to process the Magnetic Resonance Imaging of the brain.

3.3. ADVANTAGES OF PROPOSED METHODOLOGY

Early Detection of Alzheimer's using MRI Images, Family Member Recognition for Identifying People in Home, Realtime Object Detection to Identify Objects and Pets, Gesture Recognition for Immediate Audio-based Support also known as a sign to speech, OCR – Optical Character Recognition to Help Read books etc, No Hardware No Disturbance Concept and Ease of Access.

CHAPTER 4

RESULT AND ANALYSIS

Figure 4.1, shows that when you drop in the MRI images of the patient into the medical images uploading box then it analysis the state of the patient. There are four different forms of mental states: Extremely mild delusional, Mildly delusional, Moderately delusional, and Non-demented. As a result, we can determine the patient's exact condition and whether they have Alzheimer's disease based on the provided photographs. The patient's expected type is "NON-DEMENTED," according to this graph.

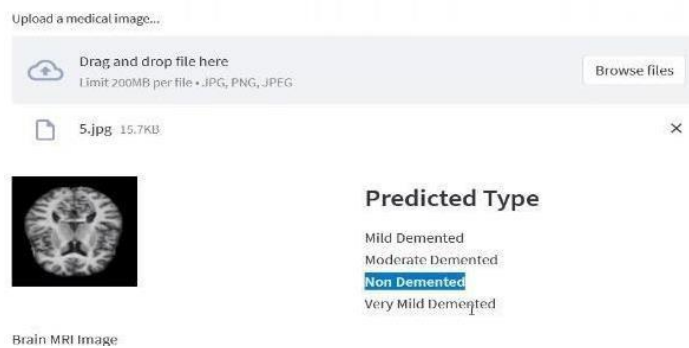


Figure 4.1. MRI image

Figure 4.2, shows that when a person looks in front of the camera the camerareads the person's face and recognizes it.

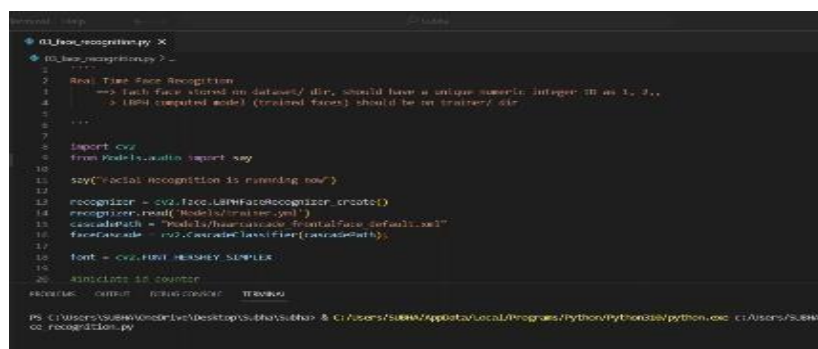


Figure 4.2. The input of Facial Recognition

Figure 4.3, shows that once the face is identified with the database then immediately an audio plays stating the person's name.

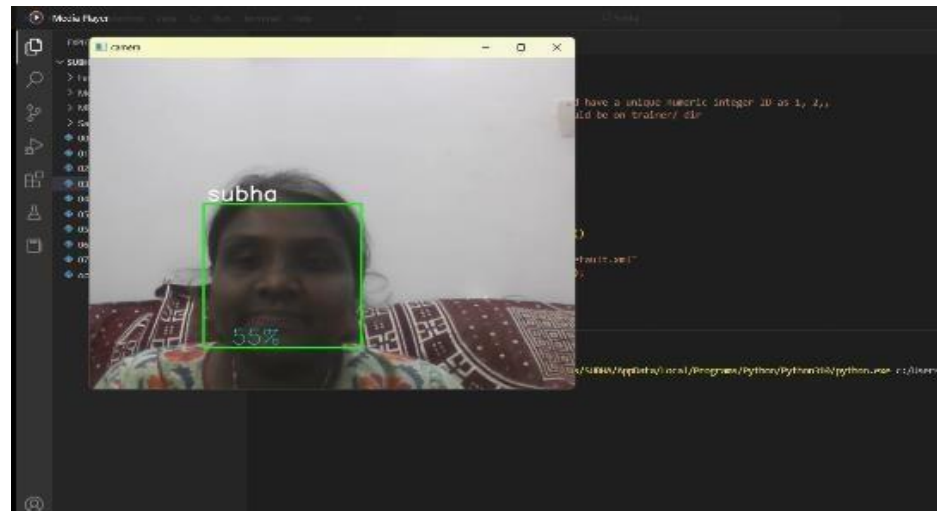


Figure 4.3. The output of Facial Recognition

Figure 4.4, shows that when a word on a book or label is brought in front of the camera and the camera recognizes it.

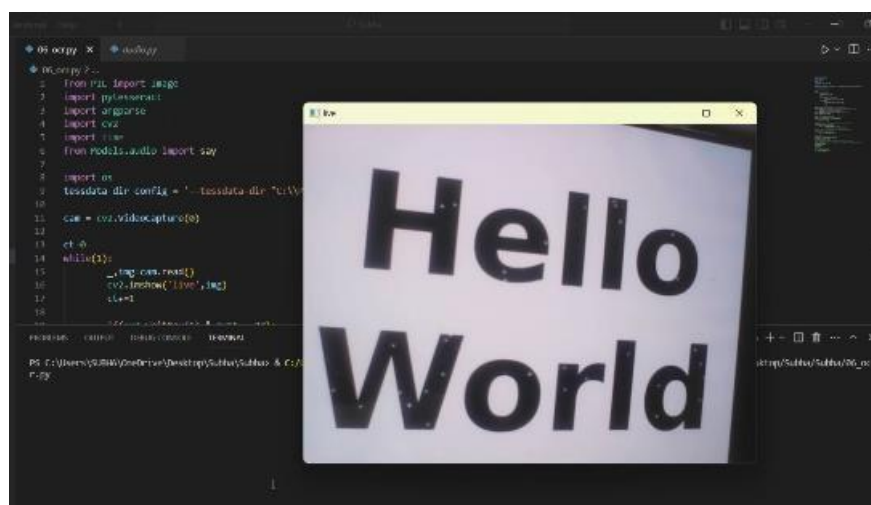


Figure 4.4. OCR input

Figure 4.5, shows that once the word is recognized the output of the word is displayed as well as an audio is heard stating what the word is. For example-“Hello world”.

```

1 from PIL import Image
2 import pytesseract
3 import argparse
4 import cv2
5 import time
6 from pydub.audio import say
7
8 import os
9 tesseract_dir = 'C:\\Program Files\\Tesseract-OCR\\tesseract'
10
11 cam = cv2.VideoCapture(0)
12
13 ct = 0
14 while(1):
15     img = cam.read()
16     cv2.imshow('live',img)
17     ct+=1
18
19

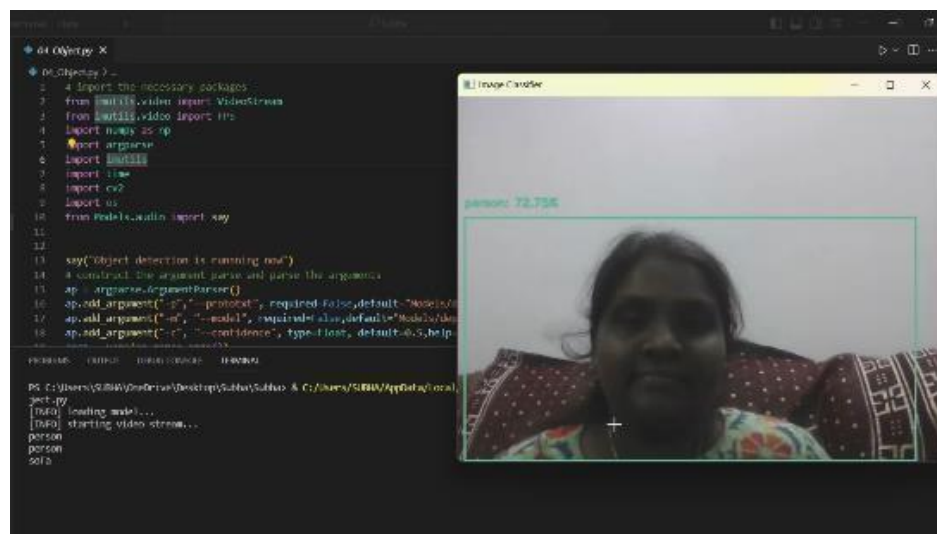
```

PS C:\Users\SUBHA\OneDrive\Desktop\Subha\Subha> C:\Users\SUBHA\AppData\Local\Programs\Python\Python38\python.exe c:\Users\SUBHA\OneDrive\Desktop\Subha\Subha\06.py
Hello
world
PS C:\Users\SUBHA\OneDrive\Desktop\Subha\Subha>

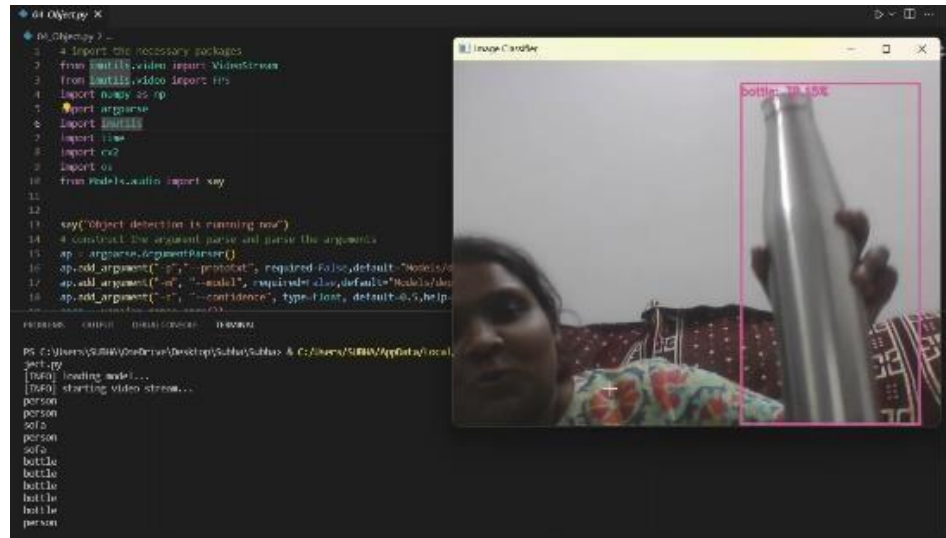
Figure 4.5. OCR output

Figure 4.6, shows that when a person comes in contact with the camera then it immediately detects the person and the audio plays stating it's a person, while it is the same way for a bottle.

So, using this camera we can help Alzheimer's patients recognize family members, real-time objects, Object character recognition, and Sign to speech.



(A)



(B)

Figure 4.6. Real-time object detection

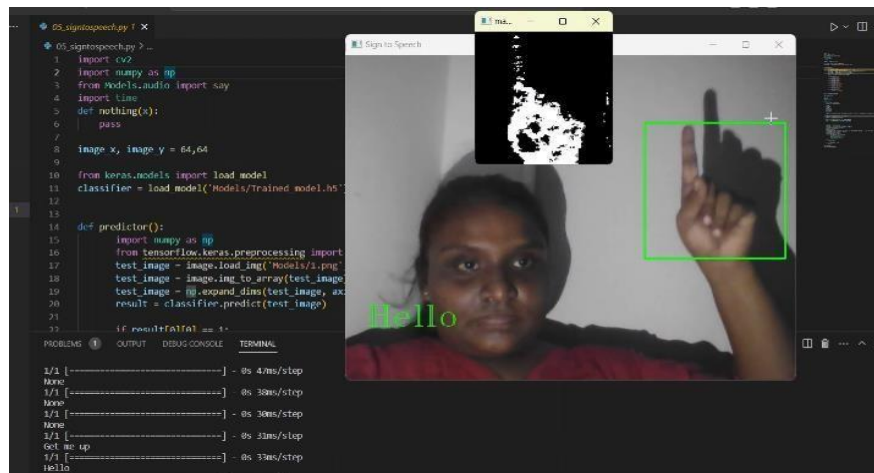
Figure 4.7 A, shows that when a person in front of the camera raises one finger demonstrating a sign, it means “HELLO”.

Figure 4.7 B, shows that when a person in front of the camera raises two fingers demonstrating a sign, it means “HELP”.

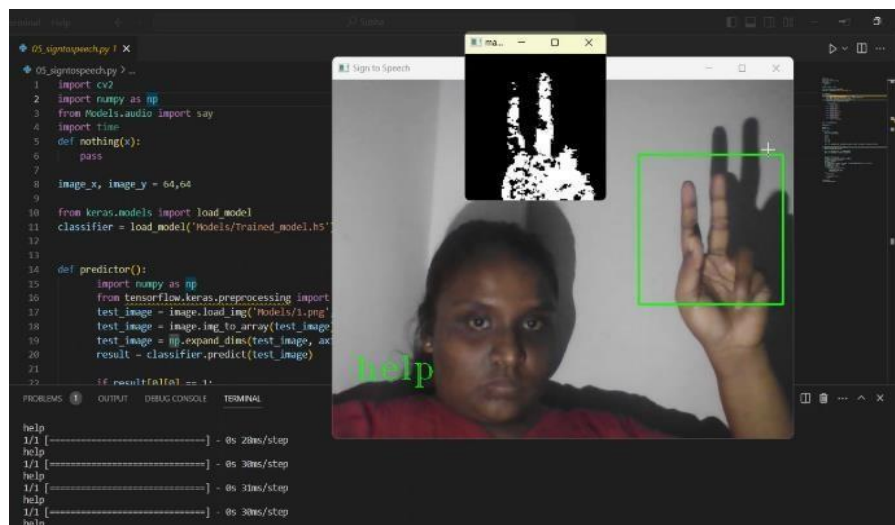
Figure 4.7 C, shows that when a person in front of the camera raises three fingers demonstrating a sign, it means “HOW ARE YOU”.

Figure 4.7 D, shows that when a person in front of the camera raises the thumb, index finger, and pinky finger demonstrating a sign, it means “GET ME UP”.

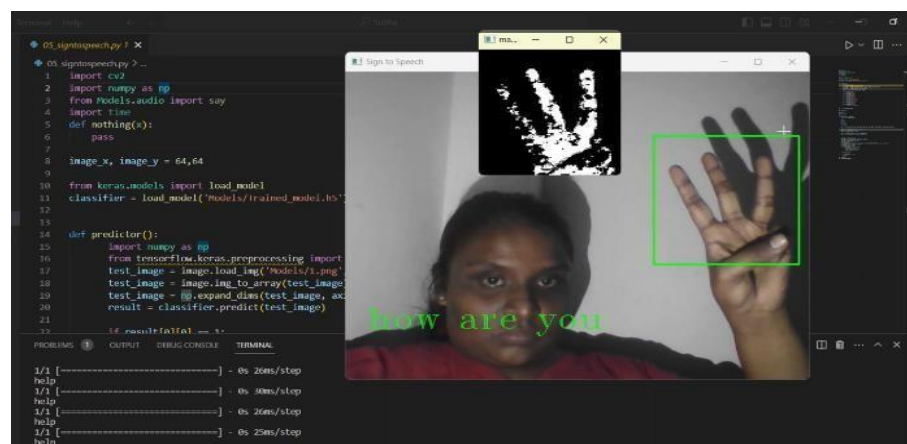
Figure 4.7 E, shows that when a person in front of the camera raises all five fingers demonstrating a sign, it means “SORRY”.



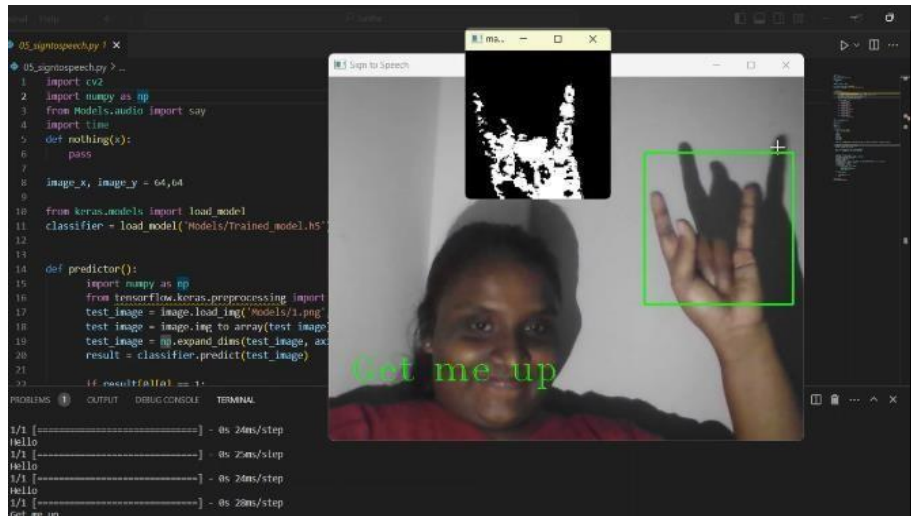
(A)



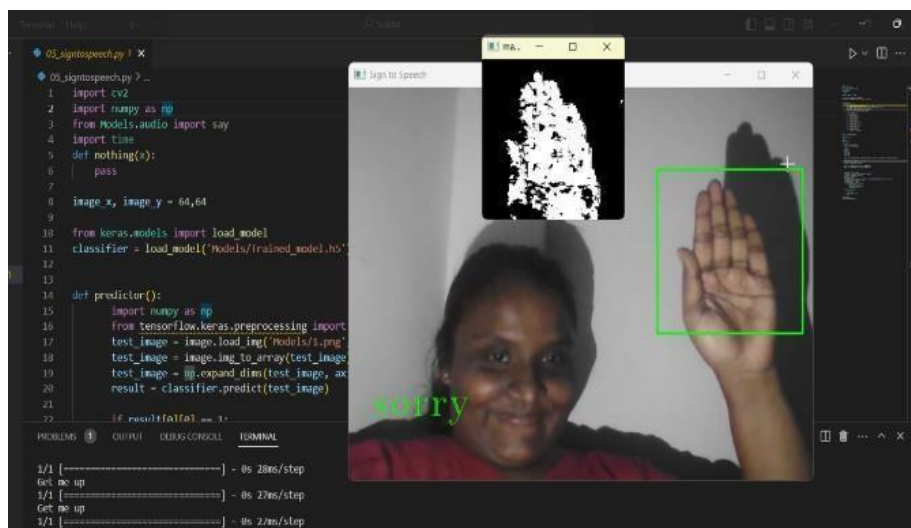
(B)



(C)



(D)



(E)

Figure 4.7 (A-E) Sing to Speech

CHAPTER 5

CONCLUSION

Therefore, in this project, Early Detection of Alzheimer's using MRI Images, Family Member Recognition for Identifying People in the Home, Realtime Object Detection to Identify Objects and Pets, Gesture Recognition for Immediate Audio based Support also known as Signal to speech, OCR – Optical Character Recognition to Help Read books, etc. are achieved and made easy for the patients to utilize so that they don't require anyone to assist them at any time. It is also helpful for doctors to identify Alzheimer's at an early stage and help the patient recover from it before any irreversible changes take place in their brain.

REFERENCES

- [1] Alzheimer's Association, W. Thies, and L. Bleiler, "Alzheimer's disease facts and figures," *Alzheimer's Dementia*, vol. 9, no. 2, pp. 208_245.
- [2] I. E. Jansen, J. E. Savage, K. Watanabe, J. Bryois, D. M. Williams, S. Steinberg, J. Sealock, I. K. Karlsson, S. Hägg, L. Athanasiu, and N. Voyle, "Genome-wide meta-analysis identifies new loci and functional pathways influencing Alzheimer's disease risk," *Nature Genet.*, vol. 51, no. 3, pp. 404_413.
- [3] GBD Dementia Collaborators, "Global, regional, and national burden of Alzheimer's disease and other dementias, 1990_2016: A systematic analysis for the global burden of disease study 2016," *LANCET Neurol.*, vol. 18, no. 1, pp. 88_106.
- [4] F. V. Chirila, T. K. Khan, and D. L. Alkon, "Fibroblast aggregation rate converges with validated peripheral biomarkers for Alzheimer's disease," *J. Alzheimer's Disease*, vol. 42, no. 4, pp. 1279_1294.
- [5] D. P. Veitch, M. W. Weiner, P. S. Aisen, L. A. Beckett, N. J. Cairns, R. C. Green, D. Harvey, C. R. Jack, W. Jagust, J. C. Morris, R. C. Petersen, A. J. Saykin, M. Shaw, A. W. Toga, J. Q. Trojanowski, and Alzheimer's Disease Neuroimaging Initiative, "Understanding disease progression and improving Alzheimer's disease clinical trials: Recent highlights from the Alzheimer's disease neuroimaging initiative," *Alzheimer's Dementia*, vol. 15, no. 1, pp. 106_152.
- [6] S. S. Dominy, C. Lynch, F. Ermini, M. Benedyk, A. Marczyk, A. Konradi, Nguyen, U. Haditsch, D. Raha, C. Griffin, and L. J. Holsinger, "Porphyromonas gingivalis in Alzheimer's disease brains: Evidence for disease causation and treatment with small-molecule inhibitors," *Sci. Adv.*, vol. 5, no. 1, 2019, Art. no. eaau3333.

- [7] S. Salloway, R. Sperling, N. C. Fox, K. Blennow, W. Klunk, M. Sabbagh, L. S. Honig, A. P. Porsteinsson, S. Ferris, and M. Reichert, "Two phase 3 trials of bapineuzumab in mild-to-moderate Alzheimer's disease," *New England J. Med.*, vol.370, no. 4, pp. 322_333.
- [8] K. G. Mawuenyega, W. Sigurdson, V. Ovod, L. Munsell, T. Kasten, J. C. Morris, K. E. Yarasheski, and R. J. Bateman, "Decreased clearance of CNS β -amyloid in Alzheimer's disease," *Science*, vol. 330, no. 6012, p. 17774.
- [9] J. Cummings, G. Lee, A. Ritter, M. Sabbagh, and K. Zhong, "Alzheimer's disease drug development pipeline: 2019," *Alzheimer's Dementia, Transl. Res. Clin.Intervent.*, vol. 5, no. 1, pp. 272_293.
- [10] T. Jonsson, J. K. Atwal, S. Steinberg, J. Snaedal, P. V. Jonsson, S. Bjornsson, H. Stefansson, P. Sulem, D. Gudbjartsson, J. Maloney, and K. Hoyte, "A mutation in APP protects against Alzheimer's disease and age-related cognitive decline," *Nature*, vol. 488, no. 7409, pp. 96_99.
- [11] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154.
- [12] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "FPGA-based face detection system using Haar classifiers," in *ACM/SIGDA Int. Symp. Field- Programm. Gate Arrays*, pp. 103–112.

APPENDIX

A. SOURCE CODE

Facial Recognition

```
""
Real Time Face Recognition
    ==> Each face stored on dataset/ dir, should have a unique numeric integer ID as 1, 2,,
    ==> LBPH computed model (trained faces) should be on trainer/ dir

'''

import cv2

recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainer.yml')
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath);

font = cv2.FONT_HERSHEY_SIMPLEX

#iniciate id counter
id = 0

# names related to ids: example ==> name: id=1, etc
names = ['None', 'kathir', 'id2']

# Initialize and start realtime video capture
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video width
cam.set(4, 480) # set video height

# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)
```

```

img_text = ""
found = set()

while True:

    ret, img =cam.read()
    img = cv2.flip(img, 1) # Flip vertically

    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor = 1.2,
        minNeighbors = 5,
        minSize = (int(minW), int(minH)),
    )

    for(x,y,w,h) in faces:

        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

        id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

        # Check if confidence is less thAN 100 ==> "0" is perfect match
        conf=round(100 - confidence)

        if (conf > 40):
            img_text = names[id]
            confidence = " {0}%".format(conf)

        else:

```

```

img_text = "unknown"

found.clear()

confidence = " {0}%".format(conf)

cv2.putText(img, str(img_text), (x+5,y-5), font, 1, (255,255,255), 2)
cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)


cv2.imshow('camera',img)


k = cv2.waitKey(1) & 0xff # Press 'ESC' for exiting video
if k == 27:
    break


# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
cam.release()
cv2.destroyAllWindows()

```

Optical Character Recognition

```
06_ocr.py
06_ocr.py
1  from PIL import Image
2  import pytesseract
3  import argparse
4  import cv2
5  import time
6  from Models.audio import say
7
8  import os
9  tessdata_dir_config = '--tessdata-dir "C:\\Program Files\\Tesseract-OCR\\tessdata"'
10
11  cam = cv2.VideoCapture(0)
12
13  ct=0
14  while(1):
15      _,img=cam.read()
16      cv2.imshow('live',img)
17      ct+=1
18
19      if(cv2.waitKey(1) & 0xFF ==27):
20          cv2.imwrite('Samples/live.png',img)
21          break
22      elif(ct==500):
23          cv2.imwrite('Samples/live.png',img)
24          break
25
26  # construct the argument parse and parse the arguments
27  ap = argparse.ArgumentParser()
28  ap.add_argument("-i", "--image", required=False,default="Samples/live.png",
29                  help="path to input image to be OCR'd")
30  ap.add_argument("-p", "--preprocess", type=str, default="blur",
31                  help="type of preprocessing to be done")
32  args = vars(ap.parse_args())
33
34  # load the example image and convert it to grayscale
35  image = cv2.imread(args["image"])
36  gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

```

06_ocr.py
06_ocr.py
31     help="type of preprocessing to be done")
32     args = vars(ap.parse_args())
33
34     # load the example image and convert it to grayscale
35     image = cv2.imread(args["image"])
36     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
37
38     cv2.imshow("Image", gray)
39
40     # check to see if we should apply thresholding to preprocess the
41     # image
42     if args["preprocess"] == "thresh":
43         gray = cv2.threshold(gray, 0, 255,
44                             cv2.THRESH_BINARY | cv2.THRESH_OTSU)[1]
45
46     # make a check to see if median blurring should be done to remove
47     # noise
48     elif args["preprocess"] == "blur":
49         gray = cv2.medianBlur(gray, 3)
50
51     # write the grayscale image to disk as a temporary file so we can
52     # apply OCR to it
53     filename = "{}.png".format(os.getpid())
54     cv2.imwrite(filename, gray)
55     #cv2.imshow("Output", gray)
56     # load the image as a PIL/Pillow image, apply OCR, and then delete
57     # the temporary file
58     mytext = pytesseract.image_to_string(Image.open(filename), config=tessdata_dir_config)
59     os.remove(filename)
60     print(mytext)
61     say(mytext)
62
63     # show the output images
64     # cv2.imshow("Image", image)
65     cv2.destroyAllWindows()
66

```

Real-time Object Detection

```
04_Object.py
04_Object.py
1  # import the necessary packages
2  from imutils.video import VideoStream
3  from imutils.video import FPS
4  import numpy as np
5  import argparse
6  import imutils
7  import time
8  import cv2
9  import os
10 from Models.audio import say
11
12
13 say("Object detection is running now")
14 # construct the argument parse and parse the arguments
15 ap = argparse.ArgumentParser()
16 ap.add_argument("-p", "--prototxt", required=False, default="Models/deploy.prototxt.txt", help="path to Caffe 'deploy' prototxt file")
17 ap.add_argument("-m", "--model", required=False, default="Models/deploy.caffemodel", help="path to Caffe pre-trained model")
18 ap.add_argument("-c", "--confidence", type=float, default=0.5, help="minimum probability to filter weak detections")
19 args = vars(ap.parse_args())
20
21 # initialize the list of class labels MobileNet SSD was trained to
22 # detect, then generate a set of bounding box colors for each class
23 CLASSES = ["None", "aeroplane", "bicycle", "bird", "boat",
24            "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
25            "dog", "horse", "motorbike", "person", "plant", "sheep",
26            "sofa", "train", "monitor"]
27 COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))
28
29 # load our serialized model from disk
30 print("[INFO] loading model...")
31 net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])
32
33 # initialize the video stream, allow the camera sensor to warmup,
34 # and initialize the FPS counter
35 print("[INFO] starting video stream...")
36 vs = VideoStream(src=0).start()
37
```

```

04_Object.py
04_Object.py
37
38
39     time.sleep(0.1)
40
41     img_text = ''
42     found = set()
43
44     # loop over the frames from the video stream
45     while True:
46         # grab the frame from the threaded video stream and resize it
47         # to have a maximum width of 400 pixels
48         frame = vs.read()
49         frame=cv2.flip(frame,1)
50         #frame = imutils.resize(frame, width=400)
51
52         # grab the frame dimensions and convert it to a blob
53         (h, w) = frame.shape[:2]
54         blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),
55                                     0.007843, (300, 300), 127.5)
56
57         # pass the blob through the network and obtain the detections and
58         # predictions
59         net.setInput(blob)
60         detections = net.forward()
61
62         # loop over the detections
63         for i in np.arange(0, detections.shape[2]):
64             # extract the confidence (i.e., probability) associated with
65             # the prediction
66             confidence = detections[0, 0, i, 2]
67
68             # filter out weak detections by ensuring the `confidence` is
69             # greater than the minimum confidence
70             if confidence > args["confidence"]:
71                 # extract the index of the class label from the
72                 # `detections`, then compute the (x, y)-coordinates of
73                 # the bounding box for the object

```

04_Object.py

04_Object.py

```
70         if confidence > args["confidence"]:  
71             # extract the index of the class label from the  
72             # `detections`, then compute the (x, y)-coordinates of  
73             # the bounding box for the object  
74  
75  
76             idx = int(detections[0, 0, i, 1])  
77             box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])  
78             (startX, startY, endX, endY) = box.astype("int")  
79  
80             # draw the prediction on the frame  
81             label = "{}: {:.2f}%".format(CLASSES[idx], confidence * 100)  
82             cv2.rectangle(frame, (startX, startY), (endX, endY),  
83                           COLORS[idx], 2)  
84             y = startY - 15 if startY - 15 > 15 else startY + 15  
85             cv2.putText(frame, label, (startX, y),  
86                         cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)  
87  
88             print(format(CLASSES[idx]))  
89             #say(format(CLASSES[idx]))  
90  
91             if(format(CLASSES[idx]) != str(img_text)):  
92                 img_text= format(CLASSES[idx])  
93                 if str(img_text) not in found :  
94                     say(str(img_text))  
95                     found.add(str(img_text))  
96             else:  
97                 found.clear()  
98  
99             cv2.imshow("Image Classifier", frame)  
100             if cv2.waitKey(1) & 0xFF == 27:  
101                 break  
102  
103     # do a bit of cleanup  
104     cv2.destroyAllWindows()  
105     vs.stop()  
106
```

Sign to Speech

```
05_signtospeech_py X
05_signtospeech_py
37
38 cam = cv2.VideoCapture(1)
39
40 img_counter = 0
41
42 img_text = ''
43 found = set()
44
45 while True:
46     ret, frame = cam.read()
47     frame = cv2.flip(frame,1)
48
49     l_h=0
50     l_s =70
51     l_v=0
52     u_h = 175
53     u_s = 225
54     u_v = 255
55
56     img = cv2.rectangle(frame, (425,100),(625,300), (0,255,0), thickness=2, lineType=8, shift=0)
57
58     lower_hsv = np.array([l_h, l_s, l_v])
59     upper_hsv = np.array([u_h, u_s, u_v])
60     imcrop = img[102:298, 427:623]
61
62     hsv = cv2.cvtColor(imcrop, cv2.COLOR_BGR2HSV)
63     mask = cv2.inRange(hsv, lower_hsv, upper_hsv)
64
65     img_name = "Models/1.png"
66     save_img = cv2.resize(mask, (image_x, image_y))
67     cv2.imwrite(img_name, save_img)
68     img_text = predictor()
69     print(img_text)
70     cv2.putText(frame, img_text, (30, 400), cv2.FONT_HERSHEY_TRIPLEX, 1.5, (0, 255, 0))
71     cv2.imshow("Sign to Speech", frame)
72     cv2.imshow("mask", mask)
73     if(str(img_text) != 'None'):
```

05_signtospeech.py X

05_signtospeech.py

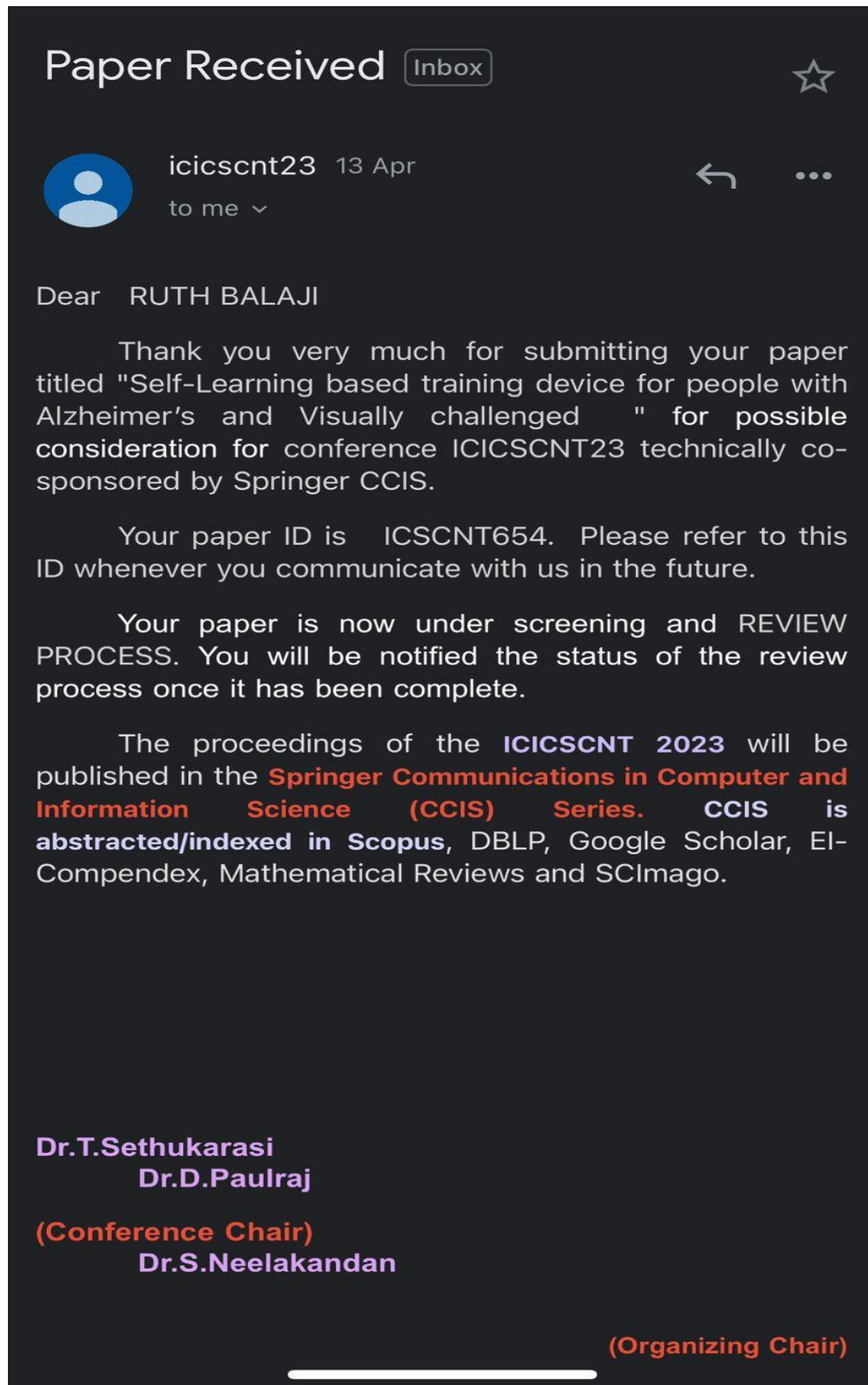
```
1  import cv2
2  import numpy as np
3  from Models.audio import say
4  import time
5  def nothing(x):
6      pass
7
8  image_x, image_y = 64,64
9
10 from keras.models import load_model
11 classifier = load_model('Models/Trained_model.h5')
12
13
14 def predictor():
15     import numpy as np
16     from keras.preprocessing import image
17     test_image = image.load_img('Models/1.png', target_size=(64, 64))
18     test_image = image.img_to_array(test_image)
19     test_image = np.expand_dims(test_image, axis = 0)
20     result = classifier.predict(test_image)
21
22     if result[0][0] == 1:
23         return 'None'
24     elif result[0][1] == 1:
25         return 'Hello'
26     elif result[0][2] == 1:
27         return 'help'
28     elif result[0][3] == 1:
29         return 'how are you'
30     elif result[0][4] == 1:
31         return 'Get me up'
32     elif result[0][5] == 1:
33         return 'sorry'
34     elif result[0][6] == 1:
35         return 'i am sleepy'
36
37
```

05_signtospeech_py X

05_signtospeech_py

```
54     u_v = 255
55
56     img = cv2.rectangle(frame, (425,100),(625,300), (0,255,0), thickness=2, lineType=8, shift=0)
57
58     lower_hsv = np.array([l_h, l_s, l_v])
59     upper_hsv = np.array([u_h, u_s, u_v])
60     imcrop = img[102:298, 427:623]
61
62     hsv = cv2.cvtColor(imcrop, cv2.COLOR_BGR2HSV)
63     mask = cv2.inRange(hsv, lower_hsv, upper_hsv)
64
65     img_name = "Models/1.png"
66     save_img = cv2.resize(mask, (image_x, image_y))
67     cv2.imwrite(img_name, save_img)
68     img_text = predictor()
69     print(img_text)
70     cv2.putText(frame, img_text, (30, 400), cv2.FONT_HERSHEY_TRIPLEX, 1.5, (0, 255, 0))
71     cv2.imshow("Sign to Speech", frame)
72     cv2.imshow("mask", mask)
73     if(str(img_text) != 'None'):
74         if str(img_text) not in found :
75             say(str(img_text))
76             found.add(str(img_text))
77             #time.sleep(2)
78     else:
79         found.clear()
80
81     if cv2.waitKey(1) == 27:
82         break
83
84     cam.release()
85     cv2.destroyAllWindows()
86
```

B. SCREENSHOTS



C. RESEARCH PAPER

SELF-LEARNING-BASED TRAINING DEVICE FOR PEOPLE WITH ALZHEIMER'S AND VISUALLY CHALLENGED

RUTH BALAJI

B.E STUDENT

*Department of Electronics and
Communication Engineering*

*Sathyabama Institute of Science and
Technology, Chennai – 600119, India*

ruthbalaji073@gmail.com

SUBHARAJA CHELLAM. A

B.E. STUDENT

*Department of Electronics and
Communication Engineering*

*Sathyabama Institute of Science and
Technology, Chennai – 600119, India*

subharaja.ak@gmail.com

Dr. A. ARANGANATHAN

M.E., Ph.D

*Department of Electronics and
Communication Engineering*

*Sathyabama Institute of Science and
Technology, Chennai – 600119, India*

aranganathan.etc@sathyabama.ac.in

Abstract— Alzheimer's disease frequently exhibits symptoms of visual impairment (AD). Visual interventions have the potential to help AD patients operate better, according to recent studies. Determining the profile of visual abnormalities in AD and potential processes driving these deficits are therefore necessary. Alzheimer's disease (AD) is a long-term, irreversible brain disorder that gradually impairs one's ability to think and remember things. Alzheimer's disease is one of the most frequent causes of dementia. Dementia is the loss of cognitive functioning, which includes thinking, remembering, and reasoning, as well as behavioral ability to the point where it affects day-to-day activities. In the medical industry, image processing is frequently used to identify disorders and support clinicians' observation-based decision-making. The objective of the study is to detect Alzheimer's disease as early as possible, allowing patients to receive treatment before their brains undergo irreversible changes. To process the brain's Magnetic Resonance Imaging (MRI), we suggest the image processing method. We propose a camera-based detection system that enables individuals with vision problems or Alzheimer's disease to read written labels or books, recognise the names of things in real-time, and recognise the names of well-known individuals using face identification and sign languages. To read the texts from books, we convert the image to text using an OCR (Optical Character Recognition) approach. After text has been created from the images. By using machine learning, we will be able to convert sign language, recognised faces, and live objects into text. The projected output will be converted to audio using the Text to Speech API.

I. Introduction

Artificial intelligence is the capacity of robots to replicate or enhance human intelligence, including logic and experience-based learning (AI). Artificial intelligence is now used in a variety of different products and services, despite having been used in computer programmes for a very long time. To recognise the objects in an image, certain digital cameras, for instance, use artificial intelligence algorithms. Researchers predict that in the future, artificial intelligence will be used in smart energy grids and many other cutting-edge applications. AI uses techniques from probability theory, economics, and algorithm design to

resolve problems in the real world. The field of AI also makes use of mathematics, psychology, languages, and computer science. Tools for modeling and solving the resulting problems are provided by mathematics, and tools for creating and constructing algorithms are provided by computer science.

Alzheimer's disease frequently exhibits symptoms of visual impairment (AD). Visual interventions have the potential to help AD patients operate better, according to recent studies. Determining the profile of visual abnormalities in AD and potential processes driving these deficits are therefore necessary. Alzheimer's disease (AD) is a long-term, irreversible brain disorder that gradually impairs one's ability to think and remember things. Alzheimer's disease is one of the most frequent causes of dementia. The loss of cognitive abilities, such as thinking, remembering, and reasoning, as well as behavioral skills to the extent that it interferes with day-to-day activities is known as dementia. In the medical industry, image processing is frequently used to identify disorders and support clinicians' observation-based decision-making.

The brain shrinks and brain cells die as a result of Alzheimer's disease, a neurologic degenerative illness. The most common form of dementia, Alzheimer's disease affects a person's ability to operate independently and is characterized by a continuous decline in cognitive, behavioral, and social skills. Alzheimer's patients can't take care of themselves and need regular assistance. We have created a technique that will help them by employing image processing to analyze a brain MRI in order to find early signs of Alzheimer's disease. They may now be able to recognize their family members using face recognition and sign language, detect items and text using object character recognition, and convert the output to audio utilizing text-to-

speech API thanks to a few new additions. So, we are using an innovative and artificial intelligence camera-based detection system for visually impaired and Alzheimer's patients in order to develop these ideas.

II. Proposed System

In this proposed system we have 4 configurations that are being performed, they are: -

1. Family Facial Recognition
2. Optical Character Recognition
3. Real-Time Object Detection
4. Sign to Speech using Tensor Flow and Keras Pre-trained Model

FAMILY FACIAL RECOGNITION –

It involves two steps that is enrollment and testing. These have three common steps: Face acquisition, pre-processing, and feature extraction. The only extra step in enrollment is that after the three pieces of information are processed, they are collected as a database. The enrollment and testing together are made as classifiers, in which we have two parts that are once after the patient recognizes the face of the family member and if it is matching with the database, we get an output of “Yes” which means it is matched but if it is not matched then we get an output of “No” that is not matched. This facial recognition recognizes the family member’s face and captures the facial identity and stores the data and helps the Alzheimer’s patient by wearing this camera to help them recognize the family member.

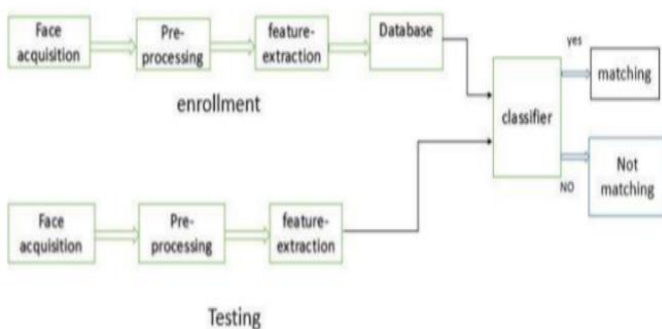


Figure 1. System Architecture of Family Facial Recognition

OPTICAL CHARACTER RECOGNITION –

On wearing the camera, the first step involved is Image Acquisition i.e., the act of obtaining an image from a source. After obtaining the image then the image is pre-

processed and after that, the features are extracted from the image. The extracted image is then translated to text as the following stage, and the text is then presented as the last step. This method helps Alzheimer’s to identify objects and read labels on books or objects.

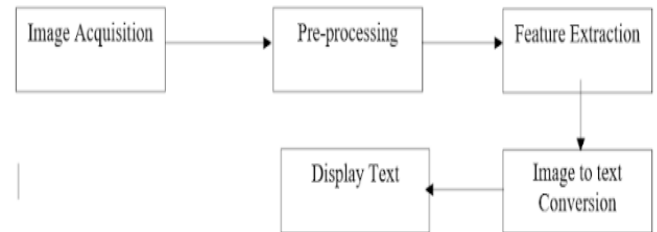


Figure 2. System Architecture of Optical Character Recognition

REAL-TIME OBJECT DETECTION-

On wearing the camera, the first step involved is Image Acquisition i.e., the act of obtaining an image from a source. After obtaining the image then the image is pre-processed and after that, the features are extracted from the image. The next step is that all the database is collected and stored and it checks if the object is recognized and matched. The last step is that the output is displayed, whether the recognized object is matched and found in the database or not. This method helps Alzheimer’s patients to recognize real-time objects like balls, cars, glasses, etc. The output is in the form of text converted to Audio.

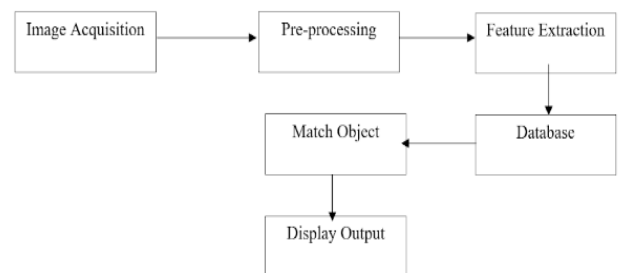


Figure 3. System Architecture of Real-time Object Detection

SIGN TO SPEECH USING TENSOR FLOW AND KERAS PRE-TRAINED MODEL-

On wearing the camera, the first step involved is Image Acquisition i.e., the act of obtaining an image from a source. After obtaining the image then the image is pre-

processed and after that, the features are extracted from the image. The next step is that all the database is collected and stored. After which the predicted signs are recognized and checked if it is matched with the database and the output is displayed. This helps Alzheimer's patients to identify sign languages and it converts them to text and finally to audio for the patients.

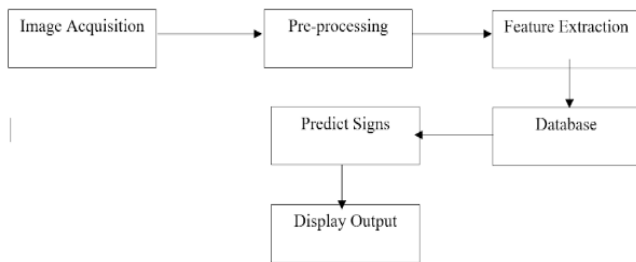


Figure 4. System Architecture of Sign to Speech

3.1. Hardware components

An optical device that can record an image is a camera. The majority of cameras can capture 2D images, and some can even capture 3D ones. The majority of cameras are essentially sealed boxes with a tiny hole (the aperture) for light to pass through so that they can be used to take pictures on surfaces that are sensitive to light (usually a digital sensor or photographic film). Cameras employ a number of strategies to control the amount of light that reaches the light-sensitive surface. Lenses focus on the light entering the camera. You can either make the aperture larger or smaller. The length of time the photosensitive surface is exposed to light is controlled by a shutter mechanism. The still-image camera is the main tool in photography. Images that have been taken can later be reproduced using photography, digital imaging, or photographic printing. Cinematography, videography, and filmmaking are related artistic disciplines in the world of moving-image cameras.

3.2. Software components

An integrated development environment (IDE) for Python is called IDLE (Integrated Development and Learning Environment). The IDLE module is by default included in the Python installer for Windows. Linux Python distributions do not

by default include IDLE. It must be set up utilizing the appropriate package managers. Similar to Python Shell, IDLE enables the creation, editing, and execution of Python programs as well as single-statement execution. The fully featured text editor in IDLE, which offers features like syntax highlighting, autocompletion, and smart indent, may be used to write Python scripts. Moreover, it contains a debugger with step and breakpoint capabilities.

For image processing, machine learning, and computer vision, people use the enormous open-source package known as OpenCV. Today, it contributes significantly to real-time operation, which is essential in contemporary systems. With it, one can search for faces, objects, and even human handwriting in images and movies. When used with a number of libraries, like as NumPy, Python can process the OpenCV array structure for analysis. To identify an image pattern and its many features, we use vector space and mathematical operations on these features. The first version of OpenCV was 1.0. Because it is offered under a BSD license, OpenCV is free for both academic and business use. It features C++, C, Python, and Java interfaces and supports Windows, Linux, Mac OS, iOS, and Android. When OpenCV was created, real-time applications for increased processing effectiveness were the main focus. Everything is written in C/C++ that has been optimized to take advantage of multi-core processing.

III. Result and Analysis

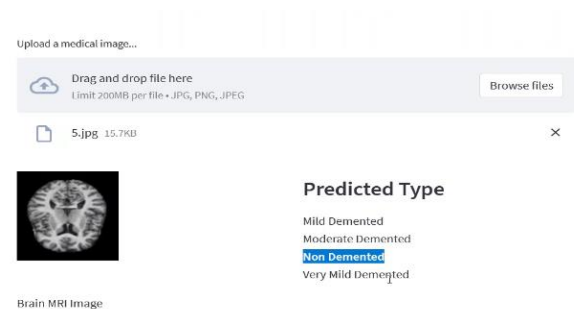


Figure 5. MRI images

Figure 5, shows that when you drop in the MRI images of the patient into the medical images uploading box then it analysis the state of the patient. There are four different forms of mental states: Extremely mild delusional, Mildly delusional, Moderately delusional, and Non-demented. As a result, we can determine the patient's exact condition and whether they have Alzheimer's

disease based on the provided photographs. The patient's expected type is "NON-DEMENTED," according to this graph.

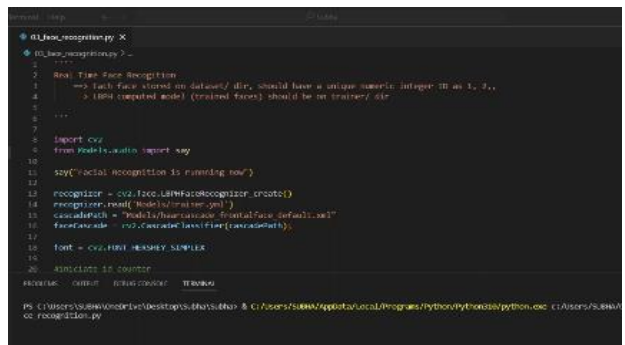


Figure 6. The input of Facial Recognition

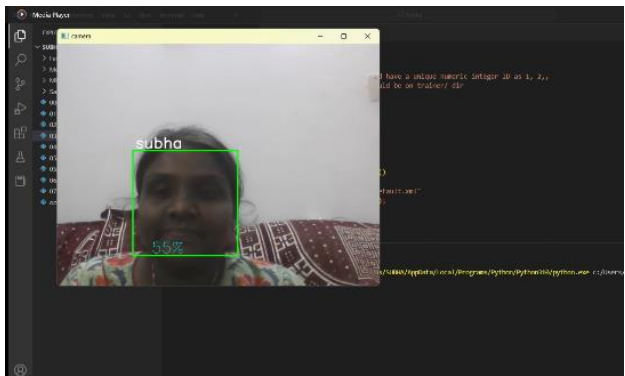


Figure 7. The output of Facial Recognition

Figure 6, shows that when a person looks in front of the camera the camera reads the person's face and recognizes it.

Figure 7, shows that once the face is identified with the database then immediately an audio plays stating the person's name.

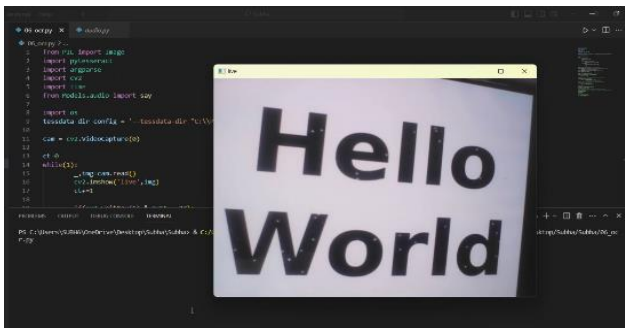


Figure 8. OCR input

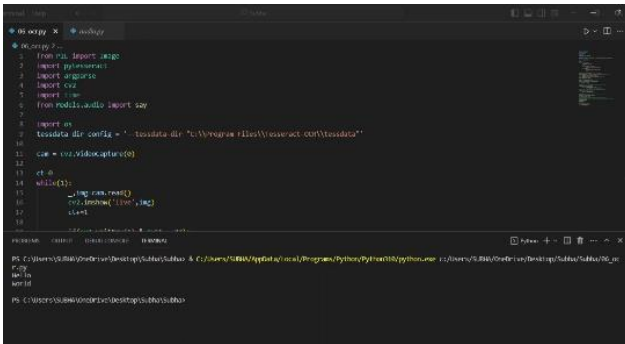


Figure 9. OCR output

Figure 8, shows that when a word on a book or label is brought in front of the camera and the camera recognizes it.

Figure 9 shows that once the word is recognized the output of the word is displayed as well as an audio is heard stating what the word is. For example- "Hello world".

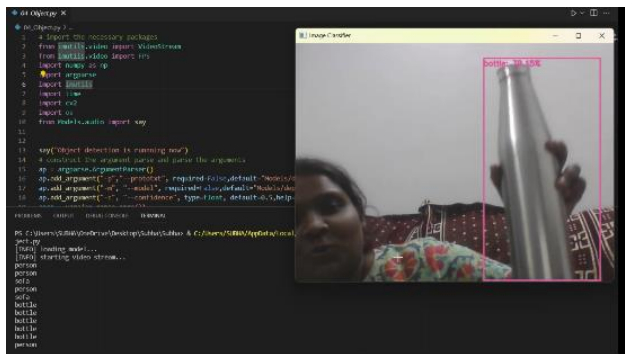
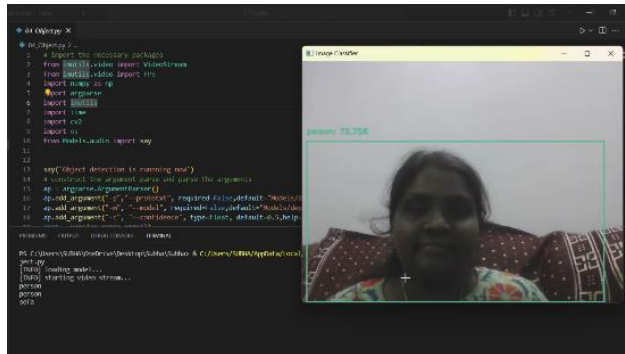
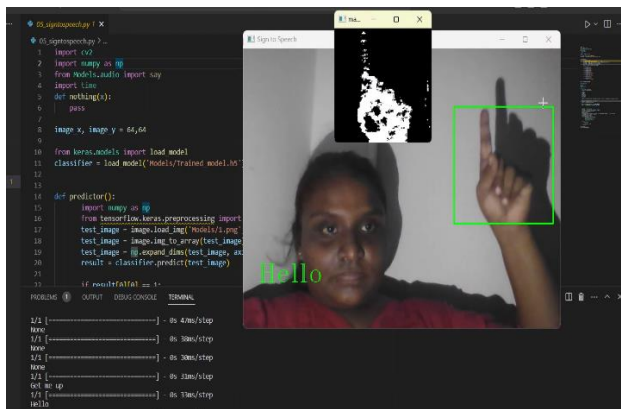


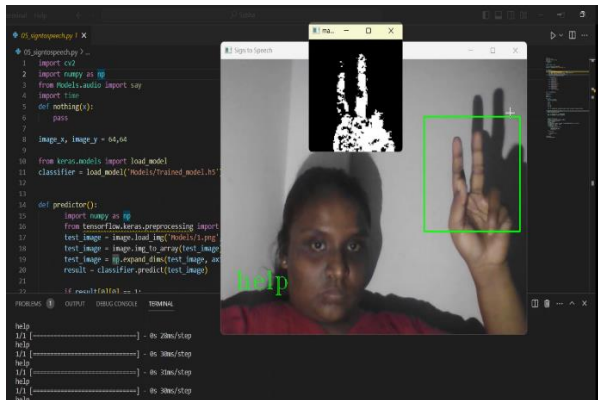
Figure 10. Real-time Object detection

Figure 10, shows that when a person comes in contact with the camera then it immediately detects the person and the audio plays stating it's a person, while it is the same way for a bottle.

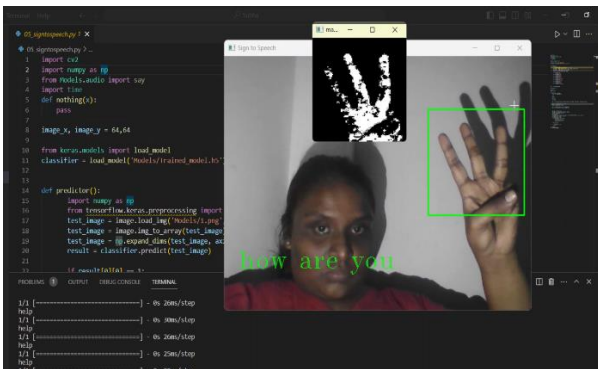
So, using this camera we can help Alzheimer’s patients recognize family members, real-time objects, Object character recognition, and Sign to speech.



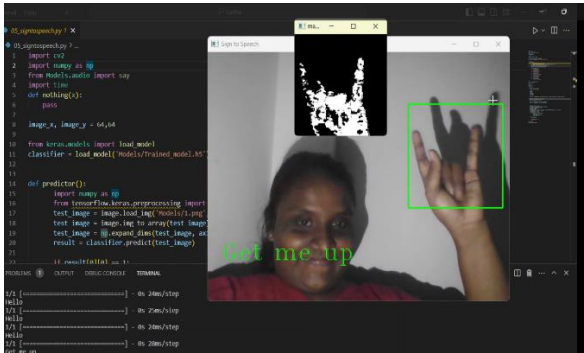
(A)



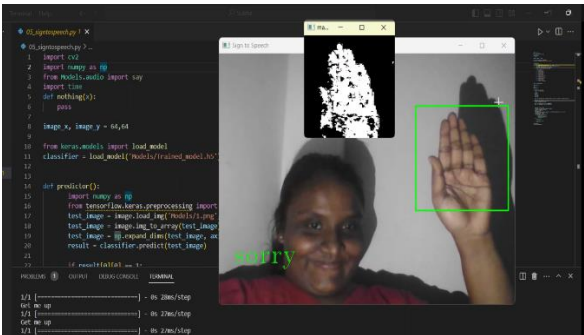
(B)



(C)



(D)



(E)

Figure 11 (A – E). Sign to Speech Output

Figure 11 A, shows that when a person in front of the camera raises one finger demonstrating a sign, it means “HELLO”.

Figure 11 B, shows that when a person in front of the camera raises two fingers demonstrating a sign, it means “HELP”.

Figure 11 C, shows that when a person in front of the camera raises three fingers demonstrating a sign, it means “HOW ARE YOU”.

Figure 11 D, shows that when a person in front of the camera raises the thumb, index finger, and pinky finger demonstrating a sign, it means “GET ME UP”.

Figure 11 E, shows that when a person in front of the camera raises all five fingers demonstrating a sign, it means “SORRY”.

IV. Conclusion

Because of this, we can determine the type of Alzheimer's in our research, Early Detection of Alzheimer's Using MRI Imaging, by looking at the uploaded images. Family Member Recognition for Identifying People in the Home, Realtime Object Detection to Identify Objects and Pets, Gesture Recognition for Immediate Audio based Support also known as Signal to speech, OCR – Optical Character Recognition to Help Read books, labels, and words, etc. are achieved and made easy for the patients to utilize so that they don't require anyone to assist them at any time. It is also helpful for doctors to identify Alzheimer's at an early stage and help the patient recover from it before any irreversible changes take place in their brain. So, this project is very much helpful for both the patients as well as the doctors to deal with Alzheimer's.

REFERENCES

- [1] The Alzheimer's Association, W. Thies, and L. Bleiler's "Alzheimer's disease facts and numbers." *Alzheimer's Dementia*, volume 9, issue 2, pages 208–245.
- [2] "Genome-wide meta-analysis identifies new loci and functional pathways influencing Alzheimer's disease risk," *Nature Genet.*, vol. 51, no. 3, pp. 404–413; I. E. Jansen, J. E. Savage, K. Watanabe, J. Bryois, D. M. Williams, S. Steinberg, J. Sealock, I. K. Karlsson, S. Hägg, and L.
- [3] F. V. Chirila, T. K. Khan, and D. L. Alkon, J. Alzheimer's Disease, vol. 42, no. 4, pp. 1279–1294. Fibroblast aggregation rate converges with validated peripheral biomarkers for Alzheimer's disease.
- [4] L. A. Beckett, N. J. Cairns, R. C. Green, D. Harvey, C. R. Jack, W. Jagust, J. C. Morris, and R. C. Petersen Understanding disease progression and improving Alzheimer's disease clinical trials: Recent highlights from the Alzheimer's disease neuroimaging initiative, *Alzheimer's Dementia*, vol. 15, no. 1, pp. 106–152, A. J. Saykin, L. M. Shaw, A.W. Toga, J. Q. Trojanowski, and the Alzheimer's Disease Neuroimaging Initiative.
- [5] Two phase 3 studies of bapineuzumab in patients with mild-to-moderate Alzheimer's dementia were published in the *New England Journal of Medicine* in 2004. Salloway, Sperling, Fox, Blennow, Klunk, Sabbagh, Honig, L. S., Porsteinsson, Ferris, and Reichert are some of the authors.
- [6] Reduced clearance of CNS β -amyloid in Alzheimer's disease, W. Sigurdson, V. Ovod, L. Munsell, T. Kasten, J. C. Morris, K. E. Yarasheski, and R. J. Bateman, *Science*, vol. 330, no. 6012, p. 17774.
- [7] In J. Cummings, G. Lee, A. Ritter, M. Sabbagh, and K. Zhong, *Alzheimer's Dementia*, *Transl. Res. Clin. Intervent.*,

vol. 5, no. 1, pp. 272–293, they write, "Alzheimer's disease drug development pipeline: 2019."

- [8] T. Jonsson, J. K. Atwal, S. Steinberg, J. Snaedal, P. V. Jonsson, S. Bjornson, H. Stefansson, P. Sulem, D. Gudbjartsson, J. Maloney, and K. Hoyte found evidence that an APP gene mutation protects against Alzheimer's disease and age-related cognitive decline.
- [9] M. J. Jones and P. Viola, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154.
- [10] "FPGA-based face detection system with Haar classifiers," in *ACM/SIGDA International Conference on Field-Program Gate Arrays*, pp. 103–112.