DE-41 (EE) M. Shayan Sajid, Rizwan Qadeer Malik, M. Awais Malik

AI BASED SMART ASSISTANCE SYSTEM FOR VISUALLY IMPAIRED PERSONS

COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND
TECHNOLOGY RAWALPINDI
2023

NUST COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING

AI BASED SMART ASSISTANCE SYSTEM FOR VISUALLY IMPAIRED PERSONS

<u>A PROJECT REPORT</u>

DE-41 (EE)

Submitted by

NS Muhammad Shayan Sajid
NS Rizwan Qadeer Malik
NS Muhammad Awais Malik

BACHELORS IN ELECTRICAL ENGINEERING
YEAR 2023

PROJECT SUPERVISOR

Dr. Ahmad Rauf Subhani

NUST COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING
PESHAWAR ROAD, RAWALPINDI

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

DEDICATION

We would like to dedicate this Final Year Project to our supervisor, Dr. Ahmad Rauf Subhani, whose guidance, support, and expertise have been instrumental in shaping this project endeavor. Their unwavering commitment to my academic and personal growth has been truly inspiring throughout this journey.

Dr. Ahmad Rauf Subhani has consistently provided invaluable insights, constructive feedback, and encouragement, pushing us to explore new avenues and challenging us to exceed our own expectations. Their profound knowledge, patience, and dedication have fostered an environment of learning and exploration, enabling us to develop and refine our ideas.

CERTIFICATE OF APPROVAL

It is to certify that the project “AI Based Smart Assistance System for Visually Impaired Persons” was done by NC Muhammad Shayan Sajid, NC Rizwan Qadeer Malik and NC M. Awais Malik, under the supervision of Dr. Ahmad Rauf Subhani.

This project is submitted to Department of Electrical Engineering, College of Electrical and Mechanical Engineering (Peshawar Road Rawalpindi), National University of Sciences and Technology, Pakistan in partial fulfillment of requirements for the degree of Bachelors of Engineering in Electrical Engineering.

Students:

1- Muhammad Shayan Sajid

NUST ID: ______________________ Signature: _____________________

2- Rizwan Qadeer Malik

NUST ID: ______________________ Signature: _____________________

3- Muhammad Awais Malik

NUST ID: ______________________ Signature: _____________________

APPROVED BY:

Project Supervisor: Dr. Ahmad Rauf Subhani

Signature: __________________ Date: ___________________

DECLARATION

We hereby declare that no portion of the work referred to in this Project Thesis has been submitted in support of an application for another degree or qualification of this of any other university or other institute of learning. If any act of plagiarism found, we are fully responsible for every disciplinary action taken against us depending upon the seriousness of the proven offence, even the cancellation of our degree.

1- Muhammad Shayan Sajid

NUST ID: ______________________ Signature: _____________________

2- Rizwan Qadeer Malik

NUST ID: ______________________ Signature: _____________________

3- Muhammad Awais Malik

NUST ID: ______________________ Signature: _____________________

COPYRIGHT STATEMENT

- Copyright in text of this thesis rests with the student author. Copies (by any process) either in full, or of extracts, may be made only in accordance with instructions given by the author and lodged in the Library of NUST College of E&ME. Details may be obtained by the Librarian. This page must form part of any such copies made. Further copies (by any process) of copies made in accordance with such instructions may not be made without the permission (in writing) of the author.

- The ownership of any intellectual property rights which may be described in this thesis is vested in NUST College of E&ME, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of the College of E&ME, which will prescribe the terms and conditions of any such agreement.

- Further information on the conditions under which disclosures and exploitation may take place is available from the Library of NUST College of E&ME, Rawalpindi.

ACKNOWLEDGEMENTS

First and foremost, we are thankful to Allah Almighty, the most merciful and the most beneficent, for the blessings, courage, and the intellect He has bestowed us with, because of which it became possible for us to accomplish such remarkable goal.

We would like to express our deepest gratitude to our supervisor, Dr. Ahmad Rauf Subhani for his dedicated guidance, invaluable advice, and constant encouragement throughout our bachelor's study. We are also indebted to him for the efforts he has devoted to serious consultations of this project. His enthusiasm and insights in many research problems have provided me with a source of thoughts and actions.

We are thankful to our parents and the family for their constant support and encouragement at each thick and thin. Moreover, we are grateful to all our fellows, friends who supported us intellectually and emotionally. Many thanks for all those who contributed in it, without all their support and appreciation we would be lacking a major source of inspiration and motivation. We are hopeful that this project will add value to the people's life and would be developed further to adopt this technology on the massive level acting as a benchmark for future technologies of Pakistan.

ABSTRACT

The aim of the project is to facilitate visually impaired persons. The visually impaired population faces numerous challenges in their daily lives, relying heavily on assistance from others to perform simple tasks.

This project involves creating an AI-based system that provides assistance to visually impaired individuals. The system is designed to be compact and easy to use, and can recognize common objects, perform facial recognition, and convert text to speech.

The proposed system consists of a wearable device equipped with a camera, microphone, and tactile input/output interface. The camera captures the visual input, which is then processed using computer vision algorithms to identify objects, people, and text in the user's vicinity. Simultaneously, the microphone is used to communicate with the user through voice.

The project also aims to explore the domains of signal processing and machine learning, while also enhancing teamwork and project management skills. If successful, the system has the potential to significantly improve the lives of visually impaired individuals by providing them with greater independence and accessibility.

SUSTAINABLE DEVELOPMENT GOALS (SDGs)

- SDG 3: Good Health and Well-being: Your system can improve the health and well-being of visually impaired individuals by providing them with better access to information, assistance, and support, thus enhancing their overall quality of life.

- SDG 4: Quality Education: By incorporating educational features into the system, such as access to digital learning materials, textbooks, and educational resources, you can contribute to inclusive and equitable education for visually impaired individuals.

- SDG 17: Partnerships for the Goals: Collaborating with relevant stakeholders, such as NGOs, assistive technology experts, and visually impaired communities, can help in the development and implementation of the AI-based Smart Assistance System, fostering partnerships for achieving the SDGs.

TABLE OF CONTENTS

CH 1: INTRODUCTION

1.1. Background

An important issue affecting millions of individuals globally is visual impairment. People who are visually impaired frequently have trouble independently obtaining information, navigating their environment, and doing daily duties. Although numerous assistive technologies have been created to solve these issues, there is still a critical need for more sophisticated and intelligent systems that can offer whole support to those who are blind or visually impaired.

Recent developments in artificial intelligence (AI), notably in the areas of computer vision and natural language processing, have created new opportunities for developing intelligent support systems that are specifically suited to the requirements of people who are visually impaired. In order to provide real-time analysis and interpretation of the environment, AI technologies can make use of picture recognition, object detection, and text-to-speech conversion, providing users with audio feedback and guidance.

White canes, guiding dogs, and screen readers are examples of current assistive technologies for the blind. These instruments have advantages, but they also have certain drawbacks. White canes and guide dogs primarily help with mobility but offer little insight into the surroundings in terms of things, text, or people. On the other hand, screen readers are dependent on text-based information and are unable to understand visual cues or identify things.

By creating an AI-based smart assistance system that is specially tailored for visually impaired persons, the proposed final-year project seeks to close these gaps. The device can analyze and interpret visual data in real-time while providing audio feedback to the user by utilizing AI and computer vision techniques. With the use of this method, people who are visually impaired can better grasp their surroundings, identify objects, read text, and communicate with their surroundings.

Modern computer vision algorithms will be used by the system to find and recognize people, objects, and text that are nearby to the user. The visual data will be transformed into audible feedback using natural language processing algorithms, which can then be transmitted via earbuds. The device will also feature a tactile input/output interface that will let users interact with it through touch gestures and haptic feedback.

The creation of this AI-based smart support system has the potential to significantly enhance the independence and standard of living of people who are blind. The technology can improve the user's capacity to traverse unfamiliar environments, recognize items, and access information by offering real-time audio feedback and thorough environmental analysis. Additionally, by allowing customization and adaptation, the personalized user profile feature will make sure that the system satisfies each user's preferences and demands.

Comprehensive user studies including people who are visually impaired will be carried out to assess the efficiency and usefulness of the AI-based smart support system. The evaluation will pay particular attention to elements like precision, user satisfaction, and the system's overall influence on the user's everyday activities.

The overall goal of this final year project is to use AI and computer vision technologies to their full potential in order to create a sophisticated smart support system that specifically solves the difficulties experienced by people who are blind. The system intends to empower visually impaired people, provide them with more independence, and improve their access to information and the outside world by utilizing AI algorithms.

1.2. Project Objectives

The project has following main objectives:

- Implement computer vision algorithms like vision transformers to enable real-time object detection, recognition, and text extraction from the user's surroundings.
- Utilize Google text to speech library which use natural language processing techniques to convert visual information into audio feedback, providing users with real-time auditory cues about their environment.
- Designing the device to be compact and lightweight, it aims to ensure that visually impaired individuals can comfortably wear or carry the device without causing excessive strain or discomfort.
- Design and integrate a wearable device with a camera, microphone, and a Wi-Fi module ESP32 which will send the camera footage to cloud for processing and receiving output from the cloud to facilitate user interaction with the smart assistance system.

1.3. Project Benefits

The project will have the following benefits:

- The system can provide assistance with tasks such as navigation, object recognition, and text reading. This can help visually impaired persons to move around more easily and independently.
- The system can help visually impaired persons to participate more fully in society. They will be able to go to work, shop, and socialize more easily.
- The system can help visually impaired persons to avoid obstacles and hazards. This can help to reduce the risk of accidents and injuries.
- The system's ability to convert visual information into audio feedback facilitates access to information for visually impaired individuals. It enables them to read text, recognize objects, and comprehend their surroundings more effectively.

- With the smart assistance system, visually impaired individuals can perform tasks more efficiently and effectively. By leveraging computer vision and natural language processing algorithms, the system can provide instant information and guidance, enabling users to accomplish tasks with improved speed and accuracy.

CH 2: LITERATURE REVIEW

2.1. Smart Glasses for Visually Impaired People in Indoor Environment [1]

The paper titled "Smart Guiding Glasses for Visually Impaired People in Indoor Environment" presents a novel assistive technology aimed at improving navigation and mobility for visually impaired individuals. The proposed solution utilizes smart glasses integrated with advanced sensing and computing capabilities to provide real-time guidance and obstacle detection within indoor environments. The system's key components include a pair of smart glasses, depth sensors, and a computer vision algorithm. The smart glasses are equipped with a camera and a display, enabling the wearer to receive visual information. The depth sensors help in capturing the spatial information of the surroundings, allowing the system to perceive the environment and detect obstacles accurately.

The computer vision algorithm plays a crucial role in analyzing the visual data captured by the smart glasses. It processes the images in real-time, extracts relevant information, and generates auditory or tactile feedback to assist the visually impaired user. The feedback can be in the form of audio instructions or vibrations, providing guidance on navigation and alerting the user about obstacles or hazards in their path.

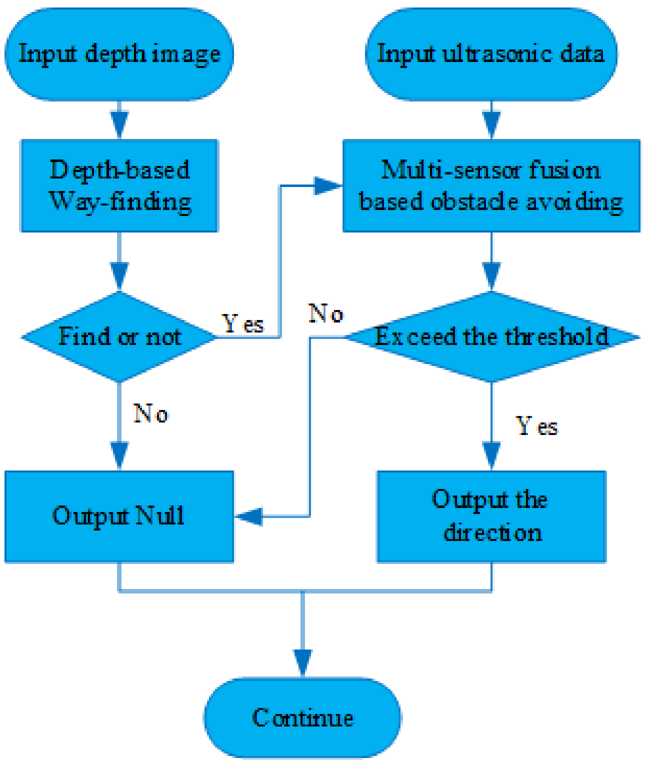


Fig. 1. The workflow of the proposed algorithm.

To validate the effectiveness of the proposed system, the authors conducted experiments in various indoor environments. They recruited visually impaired individuals as participants and evaluated the system's performance based on accuracy, speed, and user satisfaction. The results showed promising outcomes, with the smart guiding glasses successfully assisting users in navigating indoor spaces and avoiding obstacles. The paper also discusses potential future directions for improvement and expansion of the system. These include enhancing the real-time processing capabilities, incorporating additional functionalities, and extending the system to outdoor environments.

In conclusion, the paper presents an innovative approach to assist visually impaired individuals in indoor environments using smart guiding glasses. The system combines camera, depth sensors, and computer vision algorithms to provide real-time feedback, aiding users in navigation and obstacle avoidance. The experiments demonstrate the system's feasibility and its potential to enhance the mobility and independence of visually impaired individuals.

2.2. A Unique Smart Eye Glass for Visually Impaired People [2]

In this paper, special smart glasses are described by using those with vision impairments can travel more easily. Using an ultrasonic sensor and a microcontroller, it can precisely detect the obstruction and quantify the distance. Through a headset, information that has been gathered from the surroundings is transmitted to the blind individual. The information from the internet is collected via the GSM/GPRS SIM900A module. When visually impaired people are in danger, a switch that is attached to the system sends an SMS to the subject's guardian with the subject's position, time, and temperature. Visually challenged people can navigate both interior and outdoor environments by wearing smart glasses.

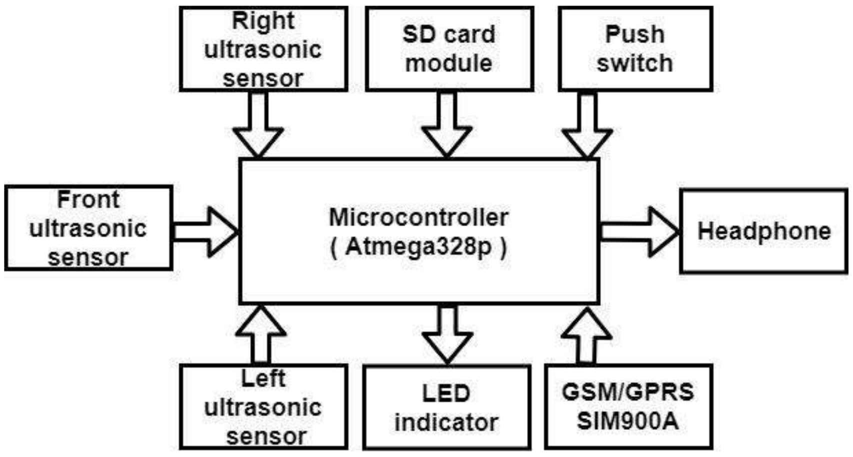


Fig. 2. Block Diagram for the proposed model

Fig. 2 displays the block diagram of our suggested model. Three ultrasonic sensors, an SD card module, a headphone, an LED indication, a switch, a GPRS SIM900A module, and an atmega328p microprocessor are all included in the system. For perfect detection, the wearable device has three ultrasonic sensors located on the left side, the front side, and the right side. This method allows for the detection of obstacles from all three sides [8]. The sensors measure the obstacle distance and relay the value to the microcontroller when an obstruction within a

5meter range approaches the blind. The Atmega328p microcontroller has a 16 MHz clock frequency and an 8-bit programmable integrated circuit. Here, an Atmega328p microprocessor processes the data from ultrasonic sensors. The Atmega328p microcontroller and the SD card module can communicate with each other. To train visually challenged users, voices are stored in the SD card module and played through headphones in accordance with the ultrasonic data. The system has a switch that can be activated in an emergency to perform tasks like sending an SMS to the subject's guardian when they are in danger. Such information as location, temperature, and time are gathered from the internet through the GPRS SIM900A module. The microcontroller processes all of the data from the GPRS SIM900A module. When the switch is hit, an SMS that includes the time, temperature, and location can be sent to the subject's guardian. The LED indicator is used to indicate day or night for the safety of the blind. It is managed by using the GPRS module to obtain the time from the internet. When there is no light in the street at night, the LED automatically turns ON so that others passing by may recognize the presence of blind people. Although the device helps the blind perceive the impediment, the LED also makes the blind person more understandable to others. As a result, a person who is blind or visually challenged can travel wherever without encountering any problems.

CH 3: METHODOLOGY

3.1. The EasyOCR Model

EasyOCR is a state-of-the-art OCR model that has gained popularity due to its simplicity and effectiveness in text recognition tasks. It is an open-source OCR framework that combines deep learning techniques with pre-trained models to achieve accurate and efficient text extraction from images.

EasyOCR is designed to handle various languages and supports a wide range of fonts and writing styles. It provides a user-friendly interface, making it accessible to both developers and non-technical users. The model offers straightforward integration with different programming languages, allowing easy implementation in various applications.

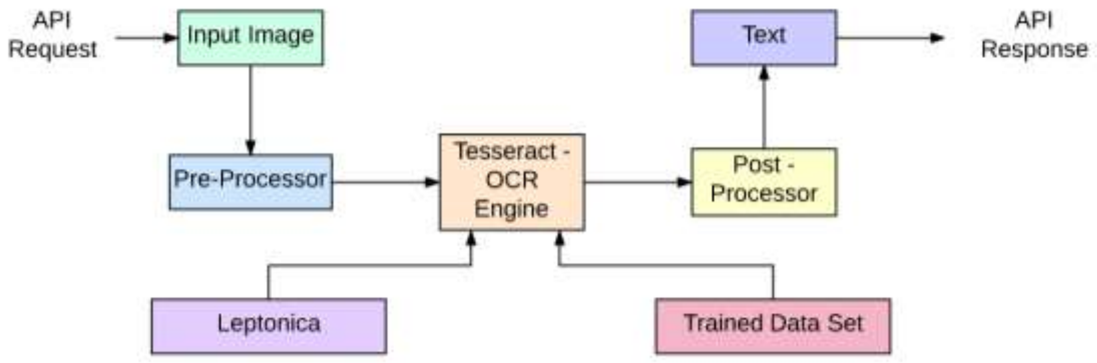


Fig. 3. Optical Character Recognition Process Flow

Working Principles of EasyOCR

EasyOCR utilizes a deep learning architecture, typically based on convolutional neural networks (CNN) and recurrent neural networks (RNN), to perform text recognition. The working principles of EasyOCR can be summarized as follows:

3.1.1. Image Pre-processing

Before feeding the image to the OCR model, it undergoes pre-processing steps such as resizing, normalization, and enhancement to improve the input quality and reduce noise or distortion. For the proper working of the model, it is necessary that we provide the model with images that are good in quality and resolution but in real world we are not always in an ideal condition, so we need some preprocessing techniques that make sure that our image quality is good. Now we will explain them one by one.

3.1.2. Image Resizing

The resizing technique in EasyOCR typically involves using interpolation algorithms such as bilinear interpolation or bicubic interpolation to scale the image up or down. These interpolation methods estimate pixel values based on neighboring pixels to preserve image details during the resizing process.

3.1.3. Image Enhancement

Various image enhancement techniques can be employed in EasyOCR depending on the quality of the image, including:

3.1.4. Histogram Equalization

Histogram equalization is a technique used in image processing to enhance the contrast of an image by redistributing the pixel intensities. The goal is to stretch the intensity range of the image to cover the entire available range, making the image appear more visually pleasing. This technique adjusts the image's histogram to improve contrast and enhance details.

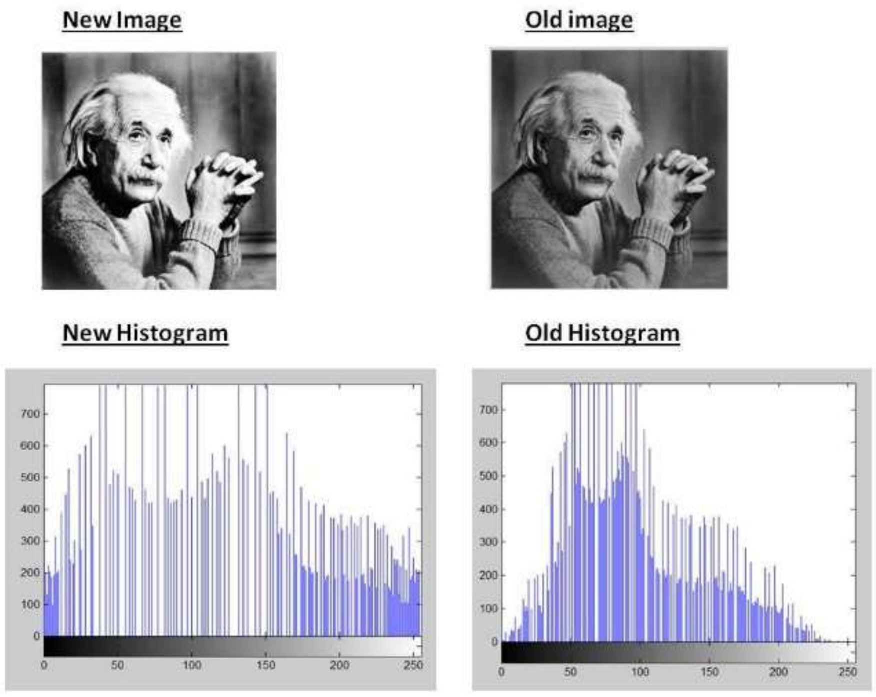


Fig. 4. Histogram Equalization

3.1.5. Adaptive Histogram Equalization:

This technique is the extension of the histogram equalization but in this case instead of computing the histogram of complete picture we compute the histogram of different regions of the image to handle variations in lighting conditions.

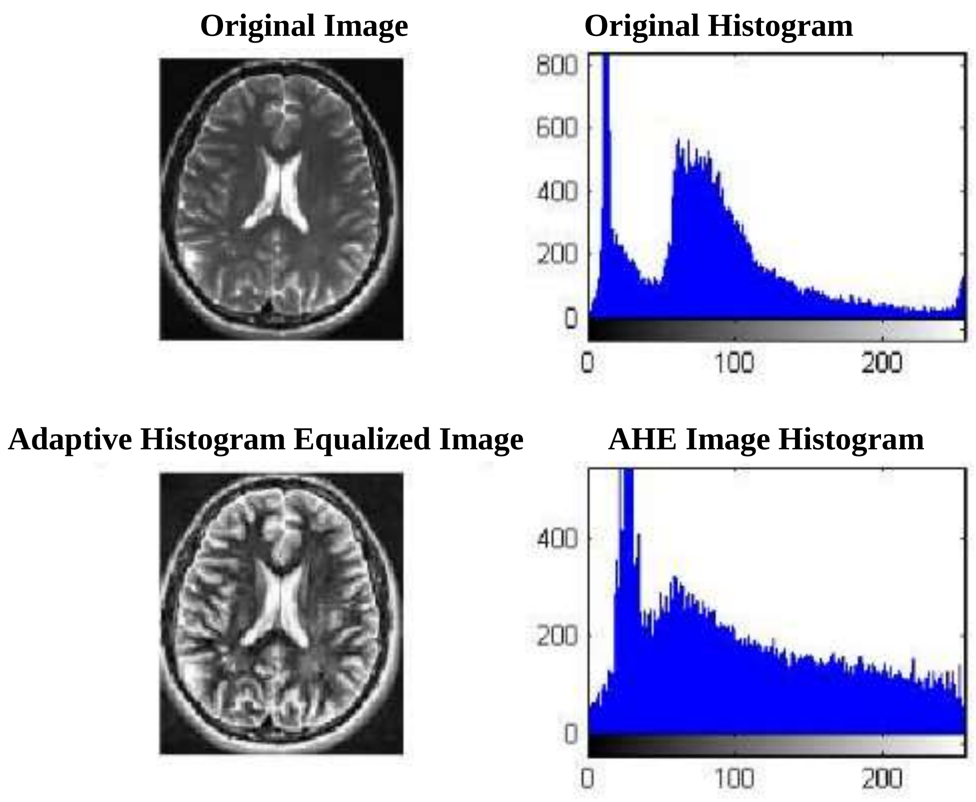


Fig. 5. Adaptive Histogram Equalization

3.1.6. Binarization

Binarization is the process of converting a grayscale or colour image into a binary representation, where each pixel is classified as either foreground (text) or background. Different binarization algorithms are used in EasyOCR to convert the image into a binary representation. These algorithms include:

3.1.7. Adaptive Thresholding

Adaptive thresholding is a technique used in image processing to segment an image into foreground and background regions by determining an optimal threshold value for each pixel based on its local neighborhood. So, instead of using a global or single threshold value the basic approach is to divide an image into small regions or windows and calculate a threshold value for each window independently. This threshold value is then used to classify the pixels within that window as foreground or background. Threshold values are computed locally based on the image's varying characteristics, such as local mean or local median.

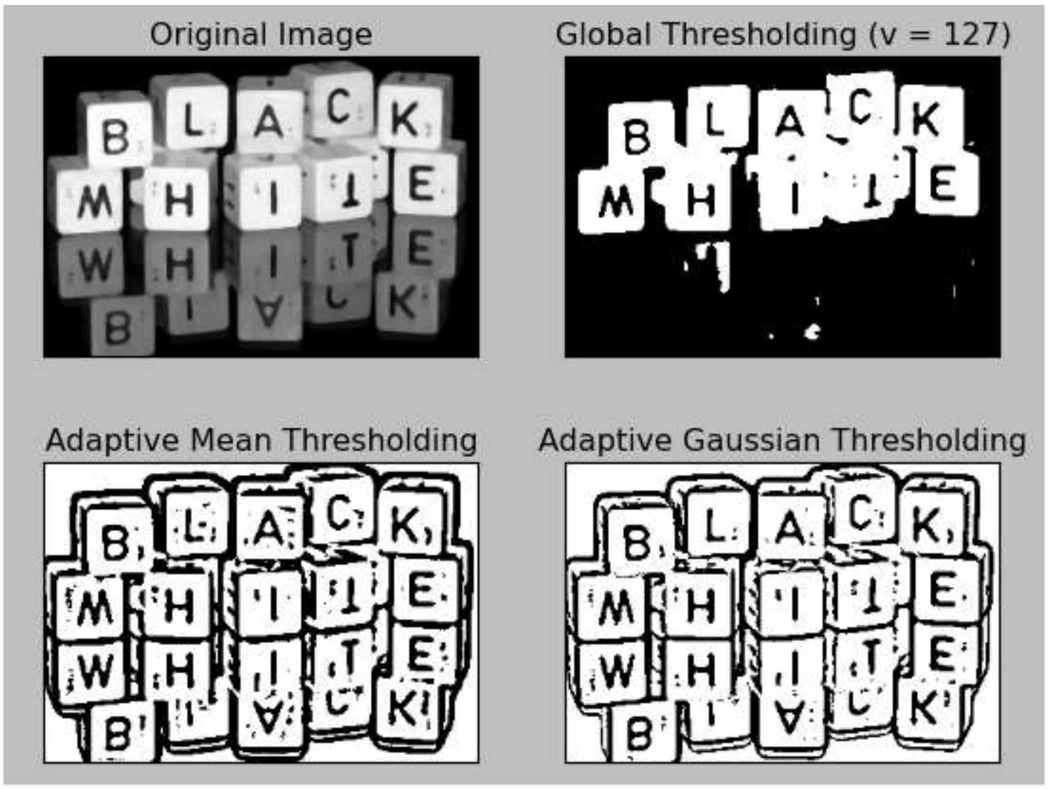
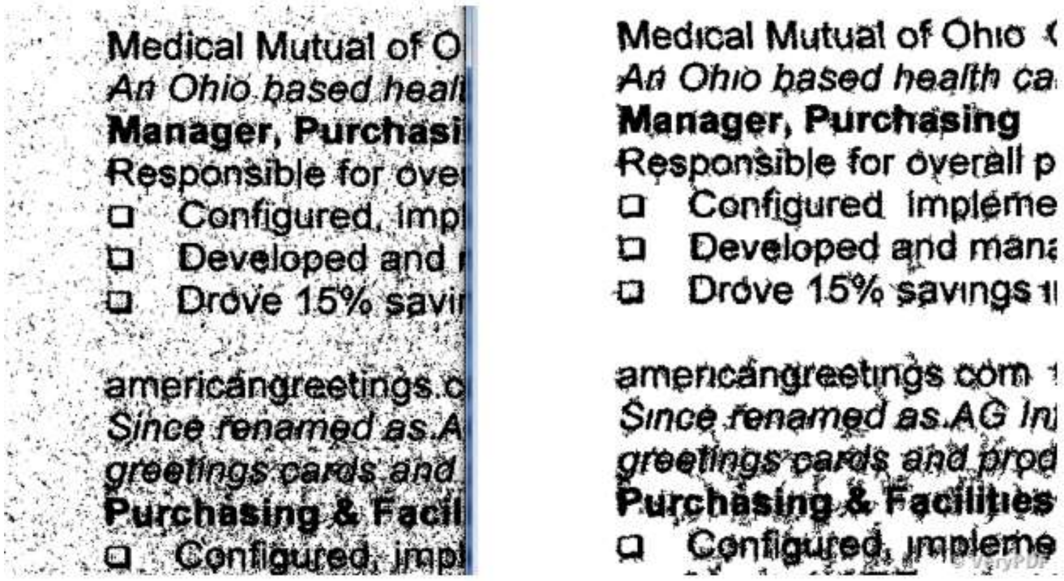

Original Image
Global Thresholding (v = 127)
Adaptive Mean Thresholding
Adaptive Gaussian Thresholding

Fig. 5. Adaptive Thresholding

3.1.8. Noise Removal

EasyOCR utilizes various noise removal techniques, depending on the type of noise present in the image following are some noise removal techniques:



Medical Mutual of O
An Ohio based heal
Manager, Purchasi
Responsible for ove
❑ Configured, impl
❑ Developed and
❑ Drove 15% savi

americangreetings.c
Since renamed as A
greetings cards and
Purchasing & Facil
❑ Configured, impl

Medical Mutual of Ohio
An Ohio based health ca
Manager, Purchasing
Responsible for overall p
❑ Configured impleme
❑ Developed and mana
❑ Drove 15% savings i

americangreetings com
Since renamed as AG In
greetings cards and prod
Purchasing & Facilities
❑ Configured, impleme

Fig. 7. Noise Removal

3.1.9. Median Filtering

In this technique we convolve a filter with image and calculate the median value which is used to replace each pixel's value with the median value of its neighboring pixels, effectively reducing salt-and-pepper noise.

3.1.10. Gaussian Smoothing

In this technique a convolutional filter with a Gaussian kernel is applied to smooth out the image and reduce high-frequency noise.

3.1.11. Morphological Operations

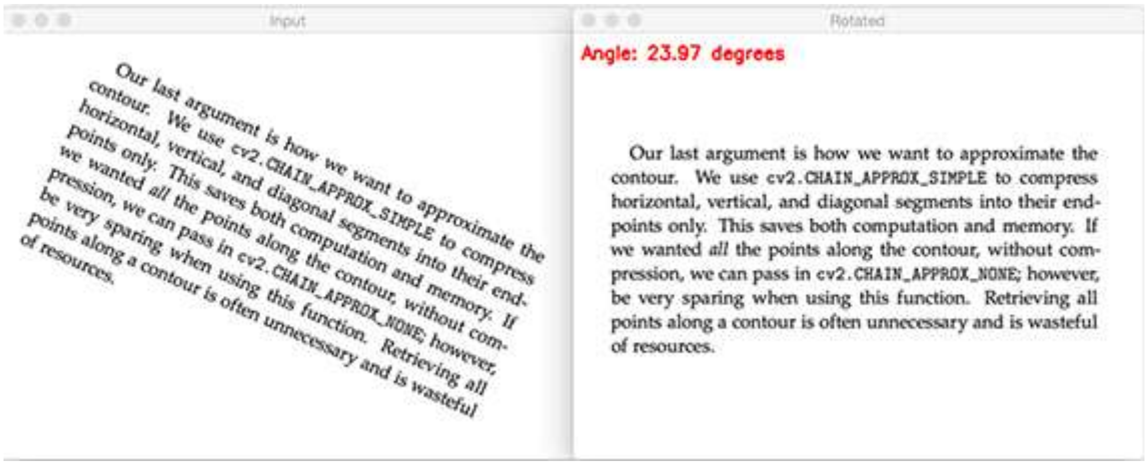
Erosion and dilation operations are used to remove noise or fill gaps in the text regions while preserving their shape.

3.1.12. Skew Correction

Skew correction techniques are employed to rectify any angular distortion or slant in the image caused by camera angle or scanning process. Skew detection algorithms analyze the text orientation and apply geometric transformations, such as rotation or shearing, to align the text regions horizontally or vertically.

3.1.13. Hough Transform

The Canny edge detector is applied to identify the edges within the image. The edges help in detecting the lines representing the text orientation. The Hough transform algorithm is then applied to detect the lines in the image. It accumulates votes for each possible line in the parameter space and identifies the lines with the highest number of votes. Based on the lines detected by the Hough transform, the algorithm analyses their orientations and estimates the skew angle of the text in the image. Using the estimated skew angle, the image is rotated to align the text regions horizontally or vertically, effectively correcting the skew This technique detects lines in the image and estimates the skew angle based on the orientation of the lines.



Our last argument is how we want to approximate the contour. We use cv2.CHAIN_APPROX_SIMPLE to compress horizontal, vertical, and diagonal segments into their end-points only. This saves both computation and memory. If we wanted *all* the points along the contour, without compression, we can pass in cv2.CHAIN_APPROX_NONE; however, be very sparing when using this function. Retrieving all points along a contour is often unnecessary and is wasteful of resources.

Angle: 23.97 degrees

Our last argument is how we want to approximate the contour. We use cv2.CHAIN_APPROX_SIMPLE to compress horizontal, vertical, and diagonal segments into their end-points only. This saves both computation and memory. If we wanted *all* the points along the contour, without compression, we can pass in cv2.CHAIN_APPROX_NONE; however, be very sparing when using this function. Retrieving all points along a contour is often unnecessary and is wasteful of resources.

Fig. 8. Hough Transform

3.2. Text Detection

EasyOCR employs algorithms to identify and locate text regions within the image. It analyses visual cues such as edges, gradients, and contours to detect areas containing text. EasyOCR utilizes the CRAFT algorithm for text detection. CRAFT is a popular deep learning-based text detection algorithm known for its effectiveness in localizing text regions within an image.

3.2.1. Convolutional Recurrent Architecture for Fast Semantic Segmentation(CRAFT)

The CRAFT algorithm, which stands for "Convolutional Recurrent Architecture for Fast Semantic Segmentation," is a computer vision algorithm used for segmenting objects in images. It was developed to efficiently perform pixel-wise semantic segmentation, which involves assigning a specific label to each pixel in an image to classify it into different object categories.

The CRAFT algorithm employs convolutional neural networks (CNN), fully convolutional networks (FCN), and anchor-based predictions to localize the text regions.

3.2.2. Convolutional Neural Networks (CNN):

CRAFT employs a CNN-based architecture to learn discriminative features from the input image. These features help in distinguishing text regions from the background. CNN stands for Convolutional Neural Network CNN are designed to automatically learn and extract meaningful features from input data through a hierarchical structure of layers. The key component of a CNN is the convolutional layer, which applies a set of learnable filters (also known as convolutional kernels) to the input data. These filters perform convolution operations by sliding over the input data and producing feature maps that highlight different aspects of the input. The subsequent layers in a CNN typically include pooling layers, which reduce the spatial dimensions of the feature maps, and fully connected layers, which perform high-level reasoning and decision-making based on the extracted features.

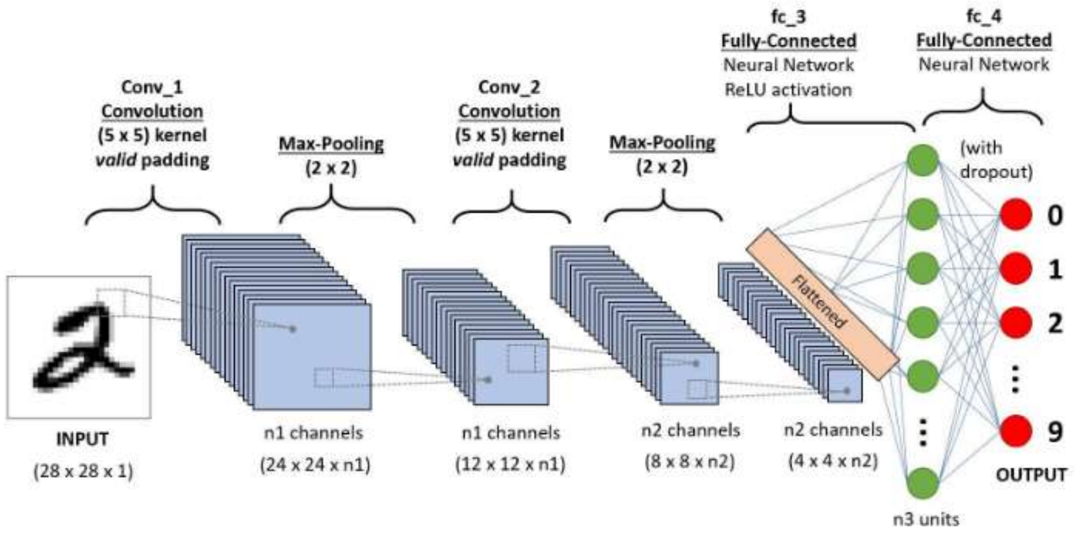


Fig. 9. Convolution Neural Network (CNN)

3.2.3. Fully Convolutional Network (FCN):

The CRAFT model is designed as a fully convolutional network, enabling it to process images of various sizes while maintaining spatial information. This allows for efficient text detection across different image scales. FCN are designed to process input images of arbitrary sizes and produce dense pixel-wise predictions as output. Unlike traditional CNN that are composed of fully connected layers at the end for classification, FCN replace the fully connected layers with convolutional layers to preserve the spatial information. This enables the network to produce dense predictions at each pixel location.

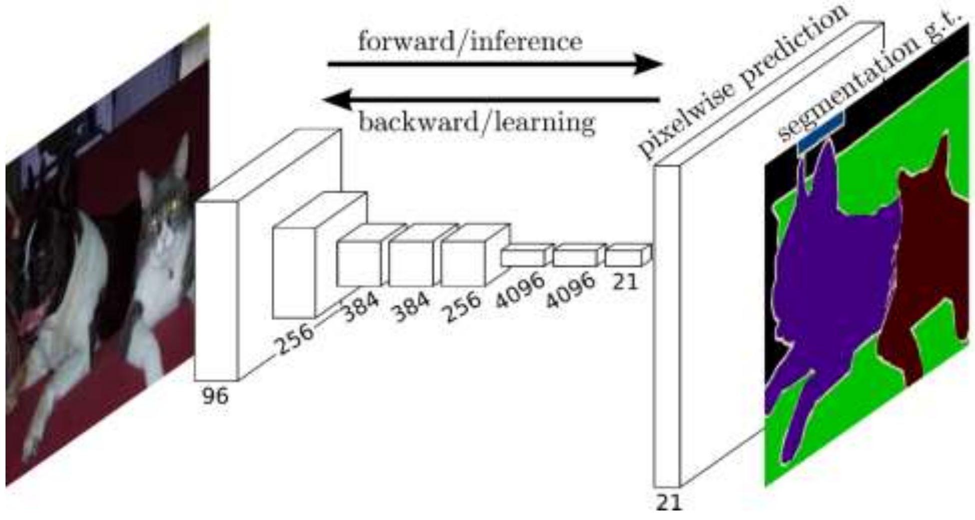


Fig. 10. Fully Convolution Network (FCN)

3.2.4. Anchors and Predictions:

CRAFT utilizes anchor-based predictions, where anchor boxes of different sizes and aspect ratios are defined across the image. The model predicts the likelihood of each anchor containing text, along with the bounding box coordinates.

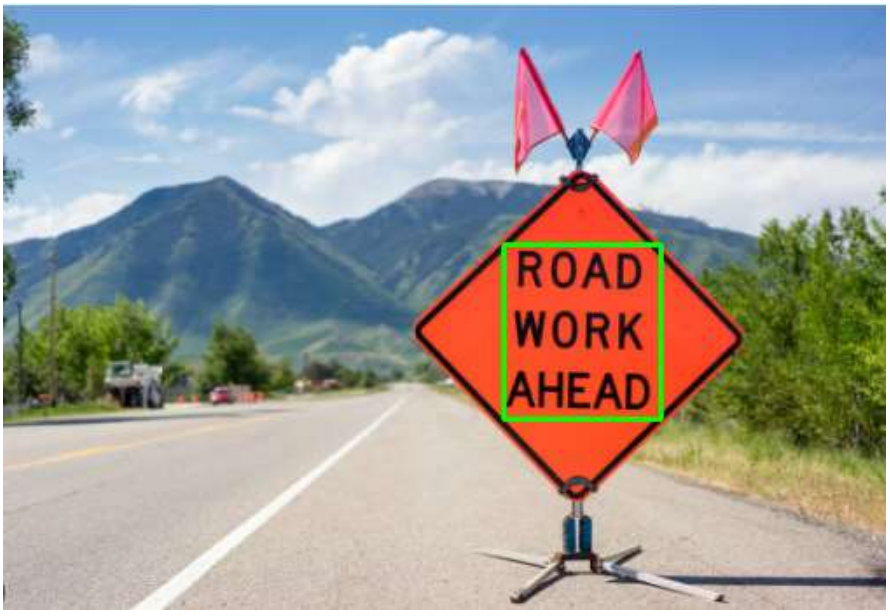


Fig. 11. Anchors and Prediction

3.3. Post-processing

After the initial text detection, post-processing steps are applied to refine and filter the detected text regions. This can include removing false positives, adjusting bounding box coordinates, and grouping nearby text regions into logical entities.

3.3.1. Aspect Ratio Regression:

To handle text regions with varying aspect ratios, CRAFT incorporates aspect ratio regression. This component refines the predicted bounding boxes, ensuring that they tightly enclose the detected text regions.

3.3.2. Multi-Level Feature Fusion:

CRAFT combines features from multiple convolutional layers to capture text information at different scales. By fusing these multi-level features, the algorithm achieves robustness in detecting text regions of different sizes.

Multi-Level feature refers to process of combination different level of features extracted through CNN. These fused features result in a robust representation of the input image, where important features from different levels are combined. This enhances the performance of text detection by leveraging the strengths of features at different scales and levels, leading to improved accuracy and robustness in localizing text regions.

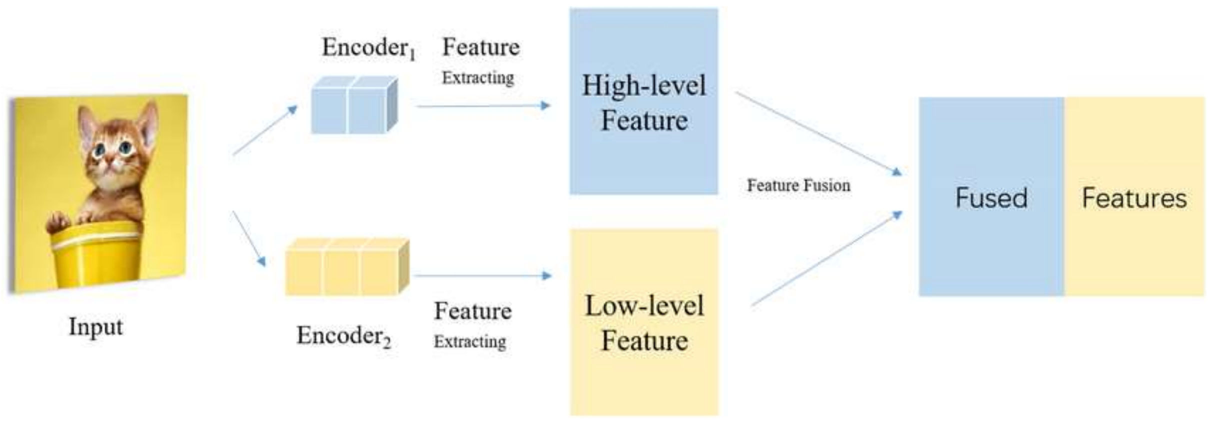


Fig. 11. Multi-Level Feature Fusion

3.3.3. Text Polygon Generation

After obtaining the predicted bounding boxes, CRAFT performs post-processing to filter out false positives and refine the text regions. The algorithm further generates text polygons by estimating the four corner points of each detected text region.

3.4. Text Recognition

Once text regions are detected, EasyOCR uses a combination of CNN and RNN models to recognize and extract the text. The CNN model is responsible for extracting visual features from the text regions, while the RNN model processes the sequence of features to predict the corresponding text.

After text detection, EasyOCR employs a series of algorithms and techniques for the recognition of text. Here is a hierarchical explanation of the different steps involved in the text recognition process:

3.4.1. CNN-Based Text Feature Extraction

The pre-processed text regions are passed through a CNN-based feature extraction model in our case we use VGG_FeatureExtractor model. This model extracts high-level visual features from the text regions, capturing important characteristics for text recognition.

3.4.2. VGG Feature Extractor:

The VGG (Visual Geometry Group) Feature Extractor model is a convolutional neural network architecture that consists of a series of convolutional layers with small 3x3 filters followed by max pooling layers. The main idea behind VGG is to stack multiple convolutional layers with small receptive fields, which allows for the effective extraction of features at different scales and complexities. The network architecture is characterized by its depth, with 16 or 19 weight layers, hence the name VGG-16 and VGG-19. The VGG Feature Extractor model has been pre-trained on large-scale image classification tasks such as the ImageNet dataset, which contains millions of labelled images across thousands of classes.

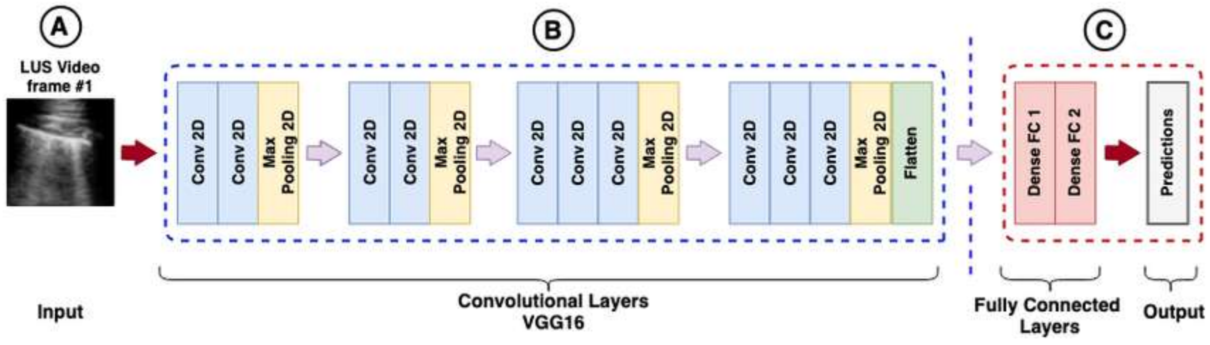


Fig. 12. VGG Feature Extractor

3.4.3. Adaptive Average Pooling

The output of the CNN-based feature extraction undergoes adaptive average pooling, which adjusts the shape of the features to a fixed size. This step ensures compatibility with subsequent layers and accommodates text regions of varying lengths.

3.4.4. Bidirectional LSTM Sequence Modelling

The pooled features are fed into a sequence modelling stage, consisting of bidirectional LSTM layers. The bidirectional LSTM models capture sequential dependencies and context within the text regions, facilitating more accurate recognition.

Bidirectional LSTM (Long Short-Term Memory) models are a type of recurrent neural network (RNN) architecture that can process sequential data in both forward and backward directions. Traditional LSTM models process input sequences in a unidirectional manner, meaning they only consider past information while making predictions. In contrast, bidirectional LSTM models consider both past and future context by incorporating two separate LSTM layers—one processing the input sequence in the forward direction and the other in the backward direction.

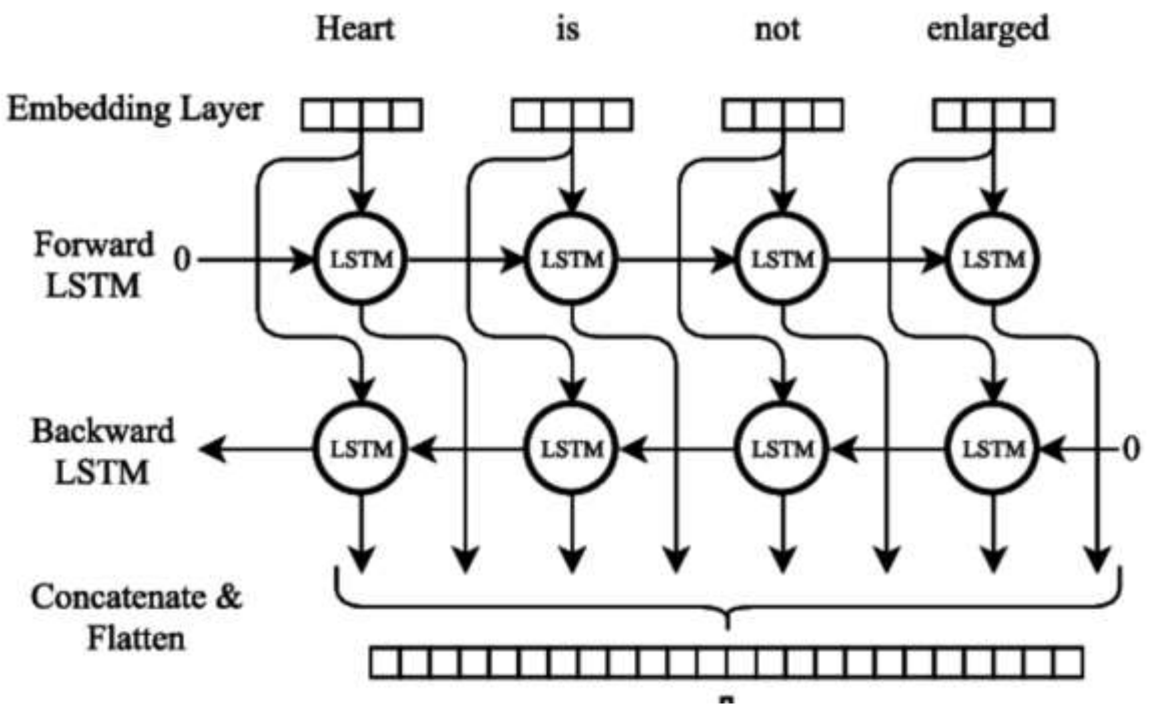


Fig.13. Bidirectional LSTM Sequence Modelling

3.4.5. Linear Transformation and Classification

The output of the sequence modelling stage is passed through a linear layer, which performs a linear transformation on the contextual features. This transformation maps the features to the number of output classes or categories for text recognition. The linear layer enables classification and predicts the recognized text labels.

3.4.6. Language and Font Support

EasyOCR supports a wide range of languages and font styles. It achieves this by training the model on diverse datasets that encompass different languages, writing styles, and fonts.

3.4.7. Training and Fine-tuning EasyOCR

EasyOCR is typically trained on large-scale annotated datasets containing images with corresponding ground truth text. The training process involves optimizing the model's parameters and learning representations of characters and words that maximize the recognition accuracy.

Fine-tuning allows customization of the EasyOCR model to specific use cases or domains. By fine-tuning, the model can adapt to fonts, writing styles, or specialized vocabulary. Fine-tuning involves training the pre-trained EasyOCR model on a smaller dataset containing relevant samples and adjusting the model's parameters accordingly.

3.4.8. Performance Evaluation of EasyOCR

The performance of EasyOCR is assessed based on several evaluation metrics, including recognition accuracy, precision, recall, and F1 score. Performance evaluation is typically conducted on benchmark datasets that include various types of images, text sizes, fonts, and languages.

To measure recognition accuracy, EasyOCR is compared against ground truth text for a given set of test images. The metrics quantify the number of correctly recognized characters or words, as well as any false positives or false negatives.

Additionally, EasyOCR performance can be analyzed in terms of processing speed, memory usage, and compatibility with different hardware platforms.

3.5. Facial Recognition

Our Second objective of this project was to come up with a system that helps the visually impaired to recognize the faces of their colleagues, friends, and family. But the biggest challenge with adding this feature was that if we want to add another person to our dataset then we will need large numbers of pictures of the person we wish to organize. To overcome this issue, we decided to come up with a model that is trained based on share training where we only need to train the last two to three layers and we are good to go. For that we use a pre-train python library face_recognition.

Now, first we need to understand what facial Recognition is and how this facial recognition model is working to help visually impaired to move one step ahead in their life.

3.5.1. Introduction to Facial Recognition

Facial recognition is a technology that involves identifying and verifying individuals based on their facial features. It utilizes machine learning algorithms to analyze and compare facial patterns captured from images or video streams. By leveraging the face_recognition library, an open-source facial recognition framework, the smart assistant system for visually impaired individuals can incorporate facial recognition capabilities.

3.5.2. Working Principles of face_recognition Library

Our face_recognition model provides a range of functions and methods for performing facial recognition tasks. Its working principles can be summarized as follows:

3.5.2.1. Face detection

The face recognition library utilizes the HOG method to detect faces in images. HOG analyses the distribution of gradients in an image to identify regions with facial features.

3.5.2.2. Histogram of Oriented Gradients (HOG)

HOG is a feature descriptor technique used for object detection and recognition tasks. The HOG method calculates and represents the local gradient orientation information in an image. The HOG algorithm divides an image into small cells and computes histograms of gradient orientations within each cell. These histograms represent the distribution of gradient orientations in that region. To capture spatial information and provide robustness to local image variations, adjacent cells are grouped together into larger blocks, and the histograms of these blocks are concatenated to form the final feature vector.

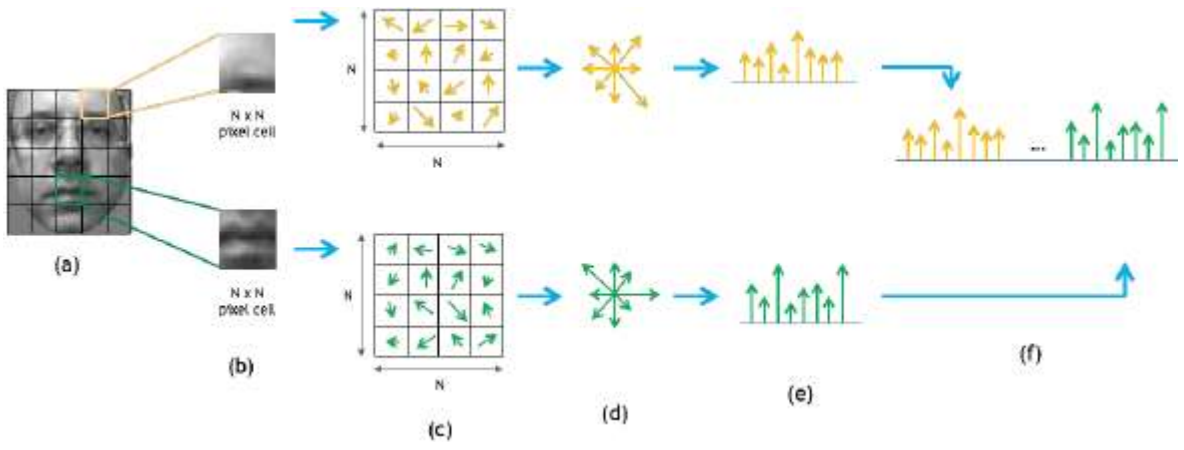


Fig.14. Histogram of Oriented Gradients (HOG)

3.5.2.3. Convolutional Neural Networks (CNN)

In addition to HOG, the library can also use a pre-trained CNN model to detect faces. CNN are deep learning models that can learn complex patterns and features, making them effective for face detection.

3.5.3. Facial feature extraction

3.5.3.1. Facial Landmarks

Once a face is detected, the library can extract facial landmarks. These landmarks are specific points on the face, such as the position of the eyes, nose, mouth, and other facial points. The library employs shape prediction models to estimate these landmarks accurately.

3.5.3.2. Face recognition

The library uses the FaceNet model for face recognition. FaceNet is a deep learning model trained on a large dataset of faces. It can compute a numerical representation (encoding) of a face, also known as a face embedding or feature vector. FaceNet employs a triplet loss function during training to ensure that similar faces have similar embedding and dissimilar faces have distinct embedding.

3.5.3.3. FaceNet

FaceNet is a deep learning model used for face recognition and face verification tasks. The main goal of FaceNet is to map facial images into a high-dimensional space, where similar faces are located close to each other, and dissimilar faces are far apart. FaceNet uses a triple loss function during training. The triplet loss enforces that the distance between an anchor face image and a positive (same person) face image is smaller than the distance between the anchor and a negative (different person) face image. This helps in learning an embedding space where faces of the same person are clustered together. FaceNet uses a deep convolutional neural network (CNN) architecture to extract features from facial images.

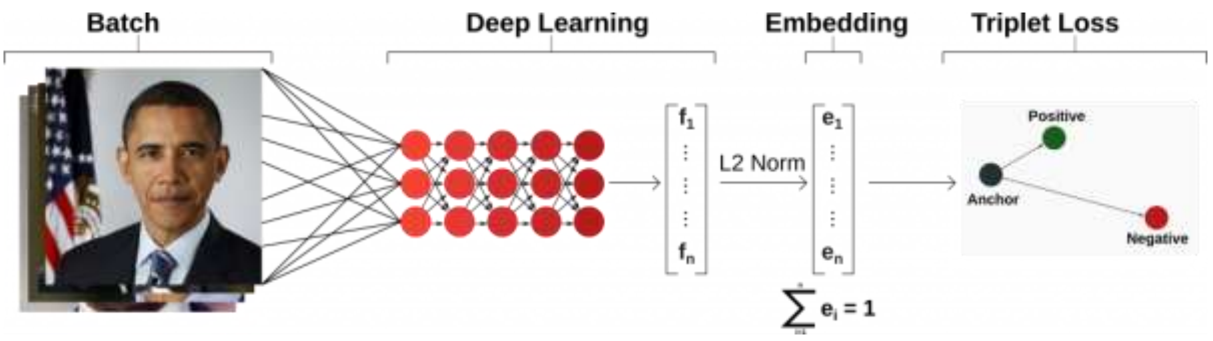


Fig.15. FaceNet

3.5.4. Face comparison and identification

3.5.4.1. Face Embedding Comparison

The library compares the computed face embedding using various distance metrics, such as Euclidean distance or cosine similarity, to determine the similarity between two faces. This comparison is used for tasks like face verification (determining if two faces belong to the same person) or face identification (matching a face against a database of known faces).

3.5.4.2. Face Clustering

The face recognition library offers face clustering functionality, which groups similar faces together based on their embedding. This can be useful for tasks like organizing large face datasets or identifying potential duplicates. Cluster analysis techniques like k-means clustering or hierarchical clustering can be employed to group faces into distinct clusters.

3.5.4.3. K- Means Clustering

K-means clustering is a popular unsupervised machine learning algorithm used for grouping data points into distinct clusters. The goal of K-means clustering is to partition the dataset into K clusters, where K is a predetermined number specified by the user.

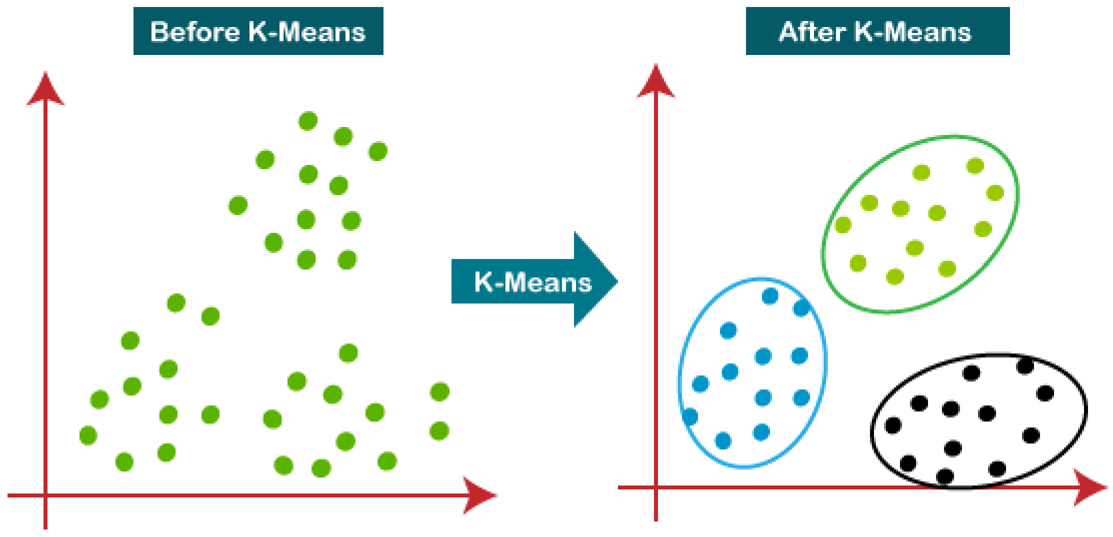


Fig.16. K-Means Clustering

3.5.4.4. Face Landmark Visualization

The library provides utilities to overlay facial landmarks on detected faces, enabling visual inspection and analysis of facial features. The landmarks are typically visualized as dots, circles, or lines, depending on the specific implementation.

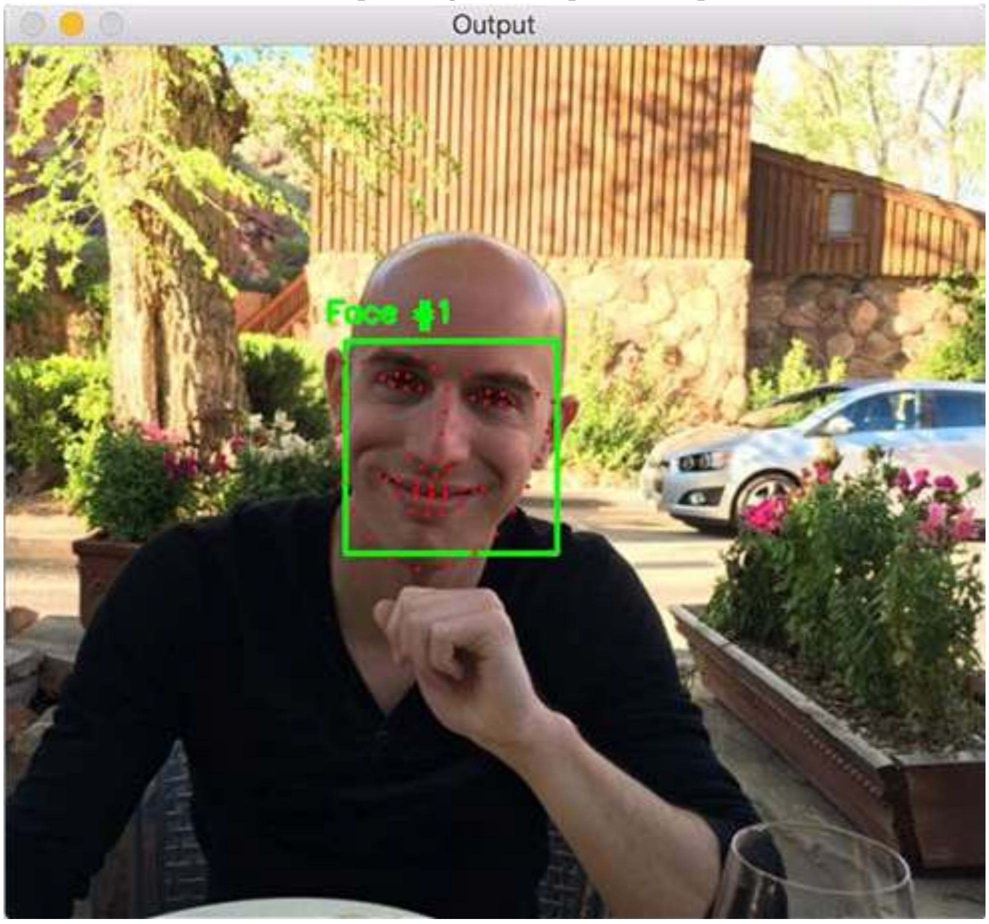


Fig.17. Face Landmark Visualization

3.5.4.5. Training and Fine-tuning in face_recognition Library

The face_recognition model leverages pre-trained deep learning models, such as convolutional neural networks (CNN), that have been trained on large-scale face datasets. These models have learned to extract discriminative facial features, enabling accurate face detection and recognition.

However, fine-tuning the face_recognition library is typically not required for general face recognition tasks. The library's pre-trained models offer robust performance across various scenarios and faces. Nevertheless, if specific requirements or constraints exist, fine-tuning techniques can be explored to optimize the facial recognition system for specific use cases.

3.5.4.6. Performance Evaluation of face_recognition Library:

The performance of the facial recognition system using the face_recognition library can be evaluated through several metrics, including accuracy, precision, recall, and F1 score. Evaluation typically involves testing the system on a dataset with known faces and measuring the rate of correct face identifications.

Performance evaluation should also consider factors such as processing speed, real-time capability, and the system's ability to handle varying lighting conditions, pose variations, and occlusions. These factors contribute to the overall usability and reliability of the facial recognition feature.

By incorporating facial recognition using the face_recognition library, the smart assistant system gains the ability to identify and verify individuals based on their facial features. This feature can have various applications, such as personalized user experiences, enhanced security, and user authentication within the system.

3.6. Object Classification using Vision Transformer (ViT)

We are using Vision Transformer for object detection. Vision Transformer is inspired by the Transformer model which was initially built for natural language processing. Now first we need to understand what exactly this Transformer Model is and how we are going to use it in image classification.

3.6.1. The Transformer Model

The Transformer follows an overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder. They can be seen as the left and right halves of the figure below respectively.

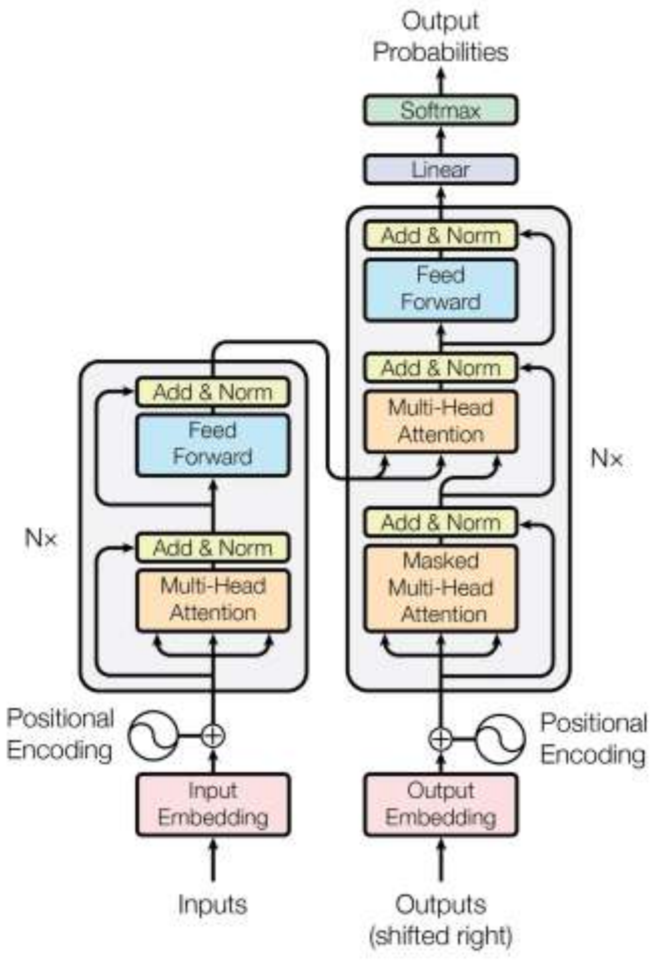


Fig.18. The Transformer Model

3.6.2. Encoder and Decoder Stack

Encoder: The encoder is comprised of a sequence of N = 6 identical layers, each containing two sub-layers. The first sub-layer is a multi-head self-attention mechanism, while the second sub-layer is a simple feed-forward network with position-wise fully connected layers. Residual connections are applied around each of these sub-layers, along with layer normalization. This means that the output of each sub-layer is computed as LayerNorm(x + Sublayer(x)), where Sublayer(x) represents the function executed by the respective sub-layer. To enable these residual connections, all sub-layers, including the embedding layers, generate outputs of dimension d = 512.

Decoder: Similarly, the decoder consists of a stack of N = 6 identical layers. In addition to the two sub-layers present in each encoder layer, the decoder incorporates a third sub-layer, which performs multi-head attention over the output of the encoder stack. Just like in the encoder, residual connections are utilized around each sub-layer, followed by layer normalization. However, there is a modification in the self-attention sub-layer of the decoder stack to prevent positions from attending to subsequent positions. This masking, coupled with the fact that the output embeddings are shifted by one position, ensures that the predictions for a specific position (i) depend solely on the known outputs at positions preceding it.

3.6.3. Self-Attention

An attention function involves the mapping of a query and a collection of key-value pairs to produce an output. In this mapping, the query, keys, values, and output are all represented as vectors. The output is determined by taking a weighted sum of the values, where the weight assigned to each value is calculated using a compatibility function that involves the query and its corresponding key.

The attention mechanism we utilize is referred to as "Scaled Dot-Product Attention". In this approach, the input comprises queries and keys with a dimension of dk, as well as values with a dimension of dv. The dot products between the query and all the keys are computed, and then divided by the square root of dk. Subsequently, a softmax function is applied to obtain the weights corresponding to the values. To streamline the process, the attention function is computed on a set of queries simultaneously, which are grouped together in a matrix called Q. Similarly, the keys and values are packed into matrices denoted as K and V. Finally, the matrix of outputs is computed as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

The attention functions commonly used are additive attention and dot-product (multiplicative) attention. Dot-product attention closely resembles our algorithm, except for the scaling factor of √(1/dk). On the other hand, additive attention calculates the compatibility function using a feed-forward network with a single hidden layer. While these two functions have similar theoretical complexity, dot-product attention proves to be significantly faster and more efficient in terms of space utilization in practical applications. This is due to its ability to leverage highly optimized matrix multiplication code for implementation. Although both mechanisms perform similarly for small values of dk, when dk is larger, additive attention outperforms dot-product attention without scaling. We suspect that as dk increases, the dot products become larger in magnitude, which pushes the softmax function into regions with extremely small gradients. To counteract this issue, we introduce the scaling factor of √(1/dk) to scale the dot products accordingly.

3.6.4. Multi-Head Attention

Rather than conducting a singular attention function with d-dimensional keys, values, and queries, the advantages of applying linear projections to the queries, keys, and values h times were discovered. Each projection employs distinct learned linear transformations to achieve dimensions of dk, dk, and dv for the queries, keys, and values, respectively. Subsequently, the attention function is executed in parallel on these projected versions of queries, keys, and values, producing output values of dimension dv. These output values are concatenated and subjected to another projection, ultimately yielding the final values. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this.

$$MultiHead(Q,K,V) = contact(head_1, \dots, head_h)W^O$$

$$where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are paremeter matrices $W_i^Q \in \mathbb{R}^{d_{model}\times d_k}$, $W_i^K \in \mathbb{R}^{d_{model}\times d_k}$, $W_i^Q \in \mathbb{R}^{d_{model}\times d_V}$ and $W^O \in \mathbb{R}^{hd_v\times d_{model}}$.

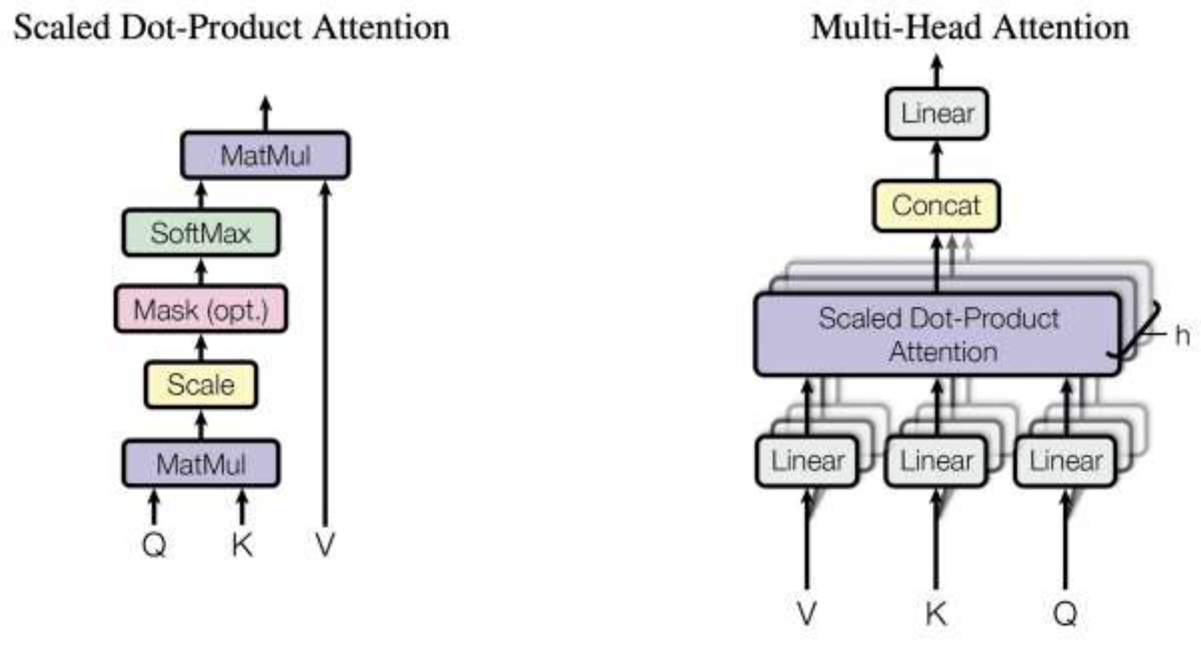


Fig.19. Scaled Dot-Product Attention and Multi-Head Attention

3.6.5. Application of Attention in the Model

- In the "encoder-decoder attention" layers, the queries are sourced from the preceding decoder layer, while the memory keys and values are derived from the encoder's output. This arrangement enables each position in the decoder to attend to all positions within the input sequence. This mirrors the conventional encoder-decoder attention mechanisms found in sequence-to-sequence models.
- The encoder is equipped with self-attention layers, where the keys, values, and queries all originate from the same source—the output of the preceding layer within the encoder. Consequently, each position in the encoder has the ability to attend to all positions within the previous layer of the encoder.
- Likewise, the decoder incorporates self-attention layers that allow each position in the decoder to attend to all positions within the decoder, including and preceding that specific position. However, in order to maintain the auto-regressive property, it is crucial to prevent the flow of leftward information within the decoder. To achieve this, we implement a masking mechanism within the scaled dot-product attention. This involves setting the values corresponding to invalid connections in the input of the softmax function to -1, effectively masking them out.

3.6.6. Position-wise Feed Forward Network

Apart from the attention sub-layers, every layer within our encoder and decoder also includes a fully connected feed-forward network. This network operates independently and uniformly on each position within the layer. It comprises two linear transformations, with a ReLU activation function applied in between them.

Although the linear transformations share the same structure across positions, they employ distinct parameters for each layer. Another way to conceptualize this is by considering them as two convolutions with a kernel size of 1. Both the input and output dimensions are set to d = 512, while the inner-layer has a dimensionality of dff = 2048.

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)\, W_2 + b_2$$

3.6.7. Positional Encoding

In our model, which lacks recurrence and convolution, preserving the order of the sequence requires incorporating information about the relative or absolute position of the tokens. To achieve this, we introduce "positional encodings" into the input embeddings at the lower ends of both the encoder and decoder stacks. These positional encodings possess the same dimension, d, as the embeddings, enabling them to be combined through summation. Various options are available for positional encodings, including learned and fixed alternatives.

From paper: Attention is All You Need

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

3.7. The Vision Transformer:

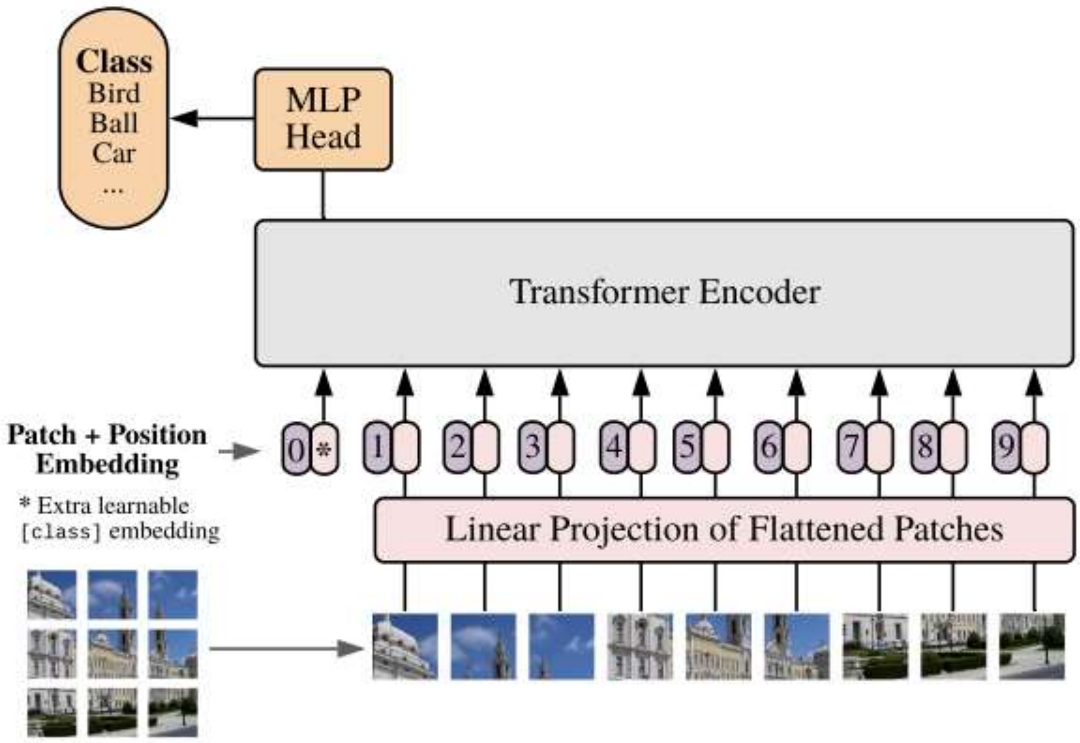

Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0 * 1 2 3 4 5 6 7 8 9
Linear Projection of Flattened Patches

Fig.20. The Vision Transformer

3.7.1. Inductive Bias

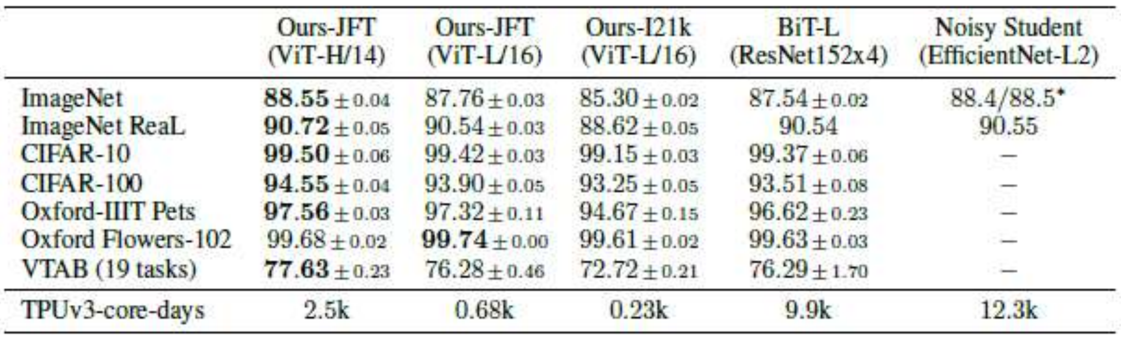
It is important to acknowledge that the Vision Transformer (ViT) exhibits significantly less image-specific inductive bias compared to Convolutional Neural Networks (CNNs). In CNNs, attributes such as locality, two-dimensional neighbourhood structure, and translation equivariance are inherent in each layer throughout the entire model. In contrast, within the ViT architecture, only the MLP (Multi-Layer Perceptron) layers possess locality and translation equivariance, while the self-attention layers operate globally. The utilization of the two-dimensional neighbourhood structure is limited: it occurs initially by dividing the image into patches and during fine-tuning when adjusting the position embeddings for images of different resolutions (as explained later). Apart from these instances, the position embeddings during initialization do not contain any information regarding the two-dimensional positions of the patches, and all spatial relationships between the patches must be learned from scratch.

3.7.2. Hybrid Architecture

In the typical workflow, we first pre-train the Vision Transformer (ViT) using large datasets and subsequently fine-tune it for specific downstream tasks, often on smaller datasets. To facilitate this process, we remove the pre-trained prediction head and replace it with a feedforward layer of dimension $D \times K$, where K represents the number of classes in the downstream task. It has been observed that fine-tuning at higher resolutions can yield benefits (Touvron et al., 2019; Kolesnikov et al., 2020). When inputting higher-resolution images, we maintain the same patch size as in the pre-training phase, resulting in an increased effective sequence length. While the Vision Transformer can handle arbitrary sequence lengths (subject to memory constraints), the pre-trained position embeddings may no longer retain their meaningfulness in this new context. To address this, we perform 2D interpolation on the pre-trained position embeddings, adjusting them based on their locations in the original image. It is important to note that these steps of resolution adjustment and patch extraction are the only instances where manual injection of inductive bias regarding the 2D structure of the images occurs within the Vision Transformer.

3.7.3. Model Comparisons

From paper: An Image is Worth 16x16 Words



	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).

The hierarchical process of image classes in a visual transformer involves the use of various algorithms, techniques, and models. Here are some commonly employed components:

3.7.4. Patch Extraction

This step involves dividing the input image into patches. The specific technique used for patch extraction can vary, but commonly used approaches include non-overlapping grid-based sampling or sliding window methods.

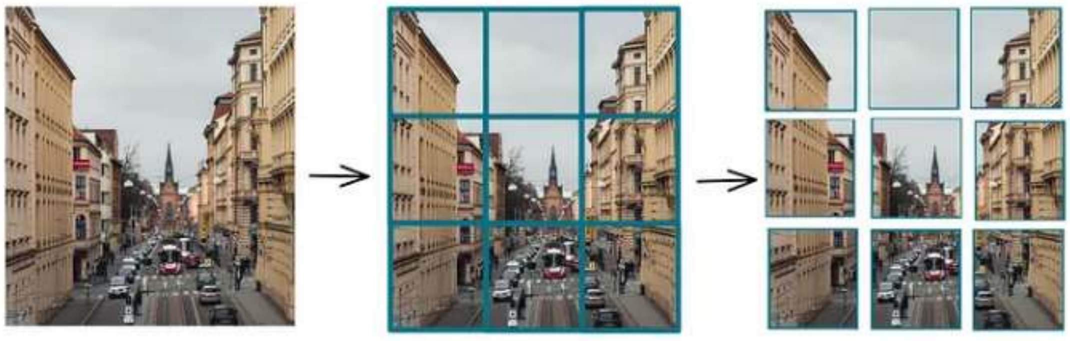


Fig.20. Patch Extraction

3.7.5. Embedding

Embedding the patches into lower-dimensional representations is typically achieved using linear projections or convolutional operations. Common choices for embedding models include convolutional neural networks (CNN) such as ResNet, EfficientNet, or VGGNet.

3.7.6. ResNet

ResNet, short for Residual Network, is a deep learning architecture that has been widely used for computer vision tasks, such as image classification and object detection. ResNet is used to tackle the problem of vanishing gradient that can affect the computation ability of deep learning model. ResNet introduces the concept of residual connection which allow deep network to bypass some layer and pass information quickly. This allows ResNet in the creation of very deep neural networks, reaching depths of over a hundred layers, while still maintaining good performance.

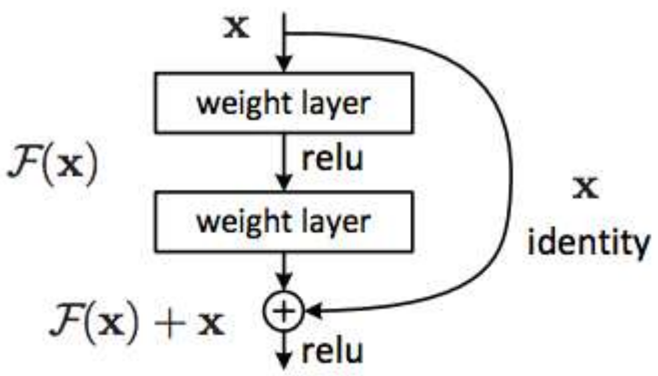

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig.21. Residual Learning: A Building Block

3.7.7. Positional Encoding

To preserve spatial information, positional encoding is applied to the embedded patches. Transformers often utilize sine or cosine functions to encode the positional information. The positional encoding is added to the embedded patches before being processed by the transformer.

3.7.8. Classification Heads

The outputs of the transformer are typically fed into one or more classification heads to predict the image's class labels. Commonly used classification head components include fully connected layers, SoftMax functions, or specialized architectures such as multi-layer perceptron's (MLP) or linear classifiers.

3.7.9. MLP (Multilayer Perceptron)

MLP (Multilayer Perceptron) is a type of artificial neural network widely used in deep learning. It consists of multiple layers of interconnected nodes or artificial neurons. The network structure includes an input layer, one or more hidden layers, and an output layer. Neurons in each layer are connected through weighted connections and apply an activation function to their inputs. MLP learn by adjusting the weights based on input data and desired output using a technique called backpropagation. They are used for tasks like classification and regression and can approximate complex nonlinear functions. MLP have limitations but serve as the foundation for more advanced neural network architectures.

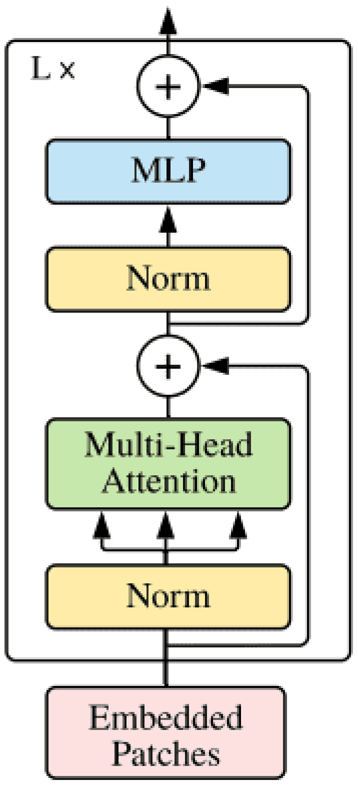


Fig.22. Transformer Encoder

3.7.10. Hierarchical Aggregation

Hierarchical aggregation techniques are often used to capture multi-scale information within the image. This can involve combining the outputs of the classification heads across different levels of the transformer's hierarchy, enabling the model to consider features at multiple scales. Techniques like feature fusion, attention mechanisms, or spatial pyramid pooling can be employed for hierarchical aggregation. But keep in mind that these techniques are used based on the size of the dataset we are using.

CH 4: SOFTWARE

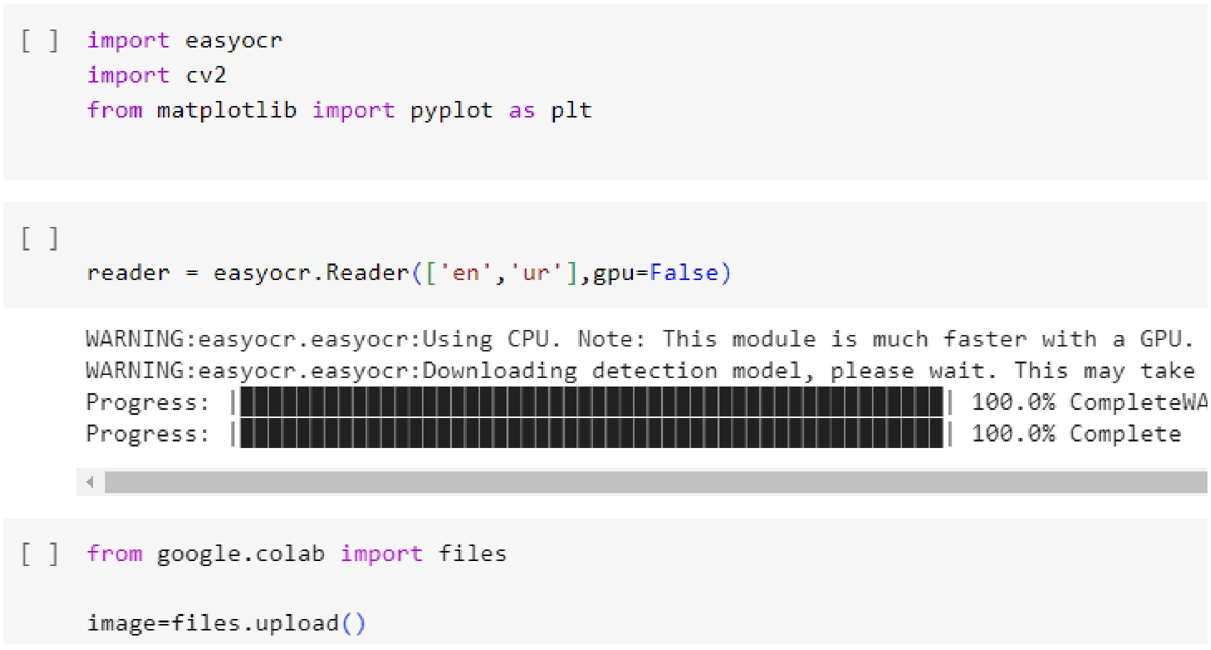
4.1. Project Work

The project comprises of 2 parts:

- In the first part, the code of each feature like EasyOCR, Google text to speech (GTTS), Facial Recognition and The Vision Transformers is implemented on the Google Colab for the test purposes.
- In the second part the code is uploaded to the cloud and it will access using a Wi-Fi module ESP32.

The code of each part is given in the following:

4.1.1. EasyOCR

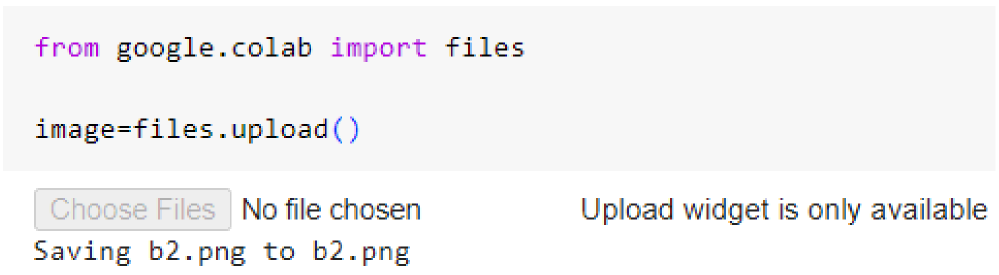


```
[ ] import easyocr
    import cv2
    from matplotlib import pyplot as plt
```

```
[ ]
    reader = easyocr.Reader(['en','ur'],gpu=False)
```

```
WARNING:easyocr.easyocr:Using CPU. Note: This module is much faster with a GPU.
WARNING:easyocr.easyocr:Downloading detection model, please wait. This may take
Progress: |██████████████████████████████████████████████████| 100.0% CompleteWA
Progress: |██████████████████████████████████████████████████| 100.0% Complete
```

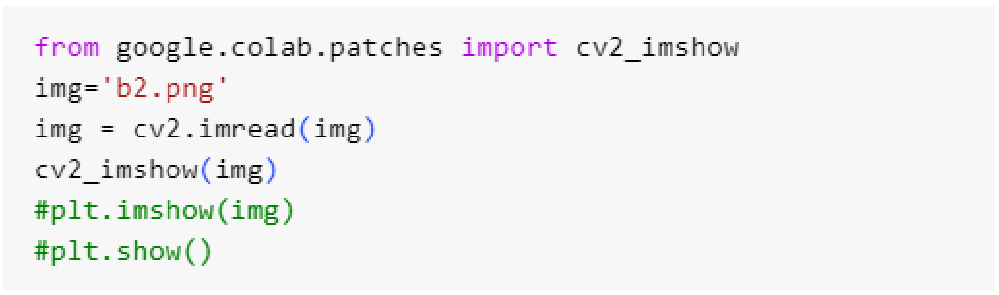
```
[ ] from google.colab import files

    image=files.upload()
```



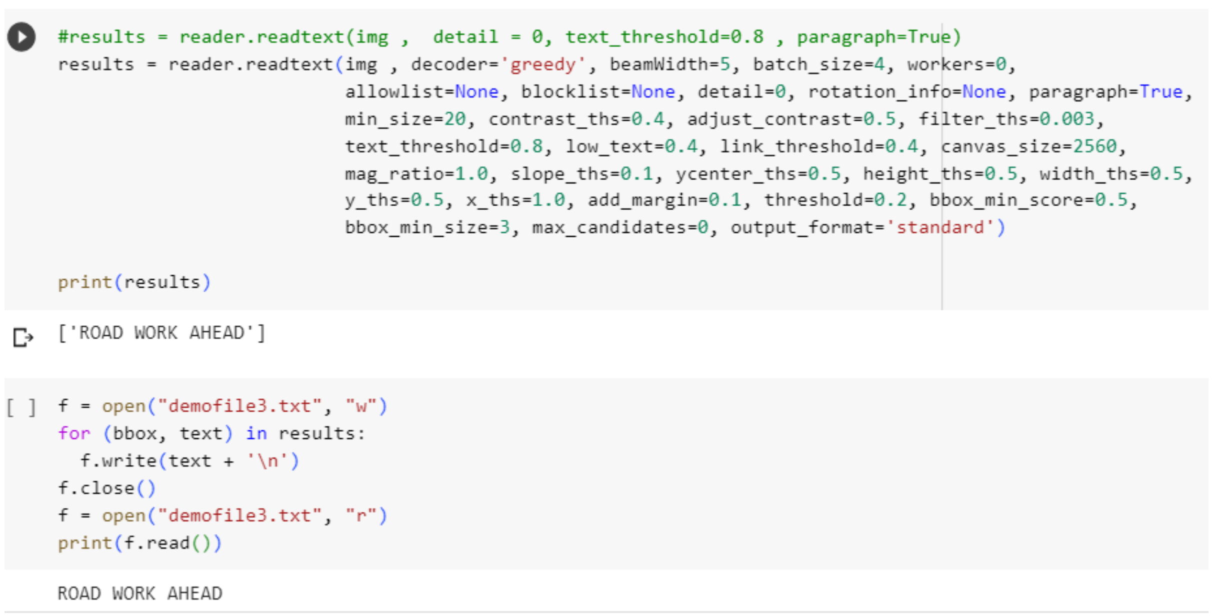
```
from google.colab import files

image=files.upload()
```

Choose Files No file chosen Upload widget is only available

```
Saving b2.png to b2.png
```



```
from google.colab.patches import cv2_imshow
img='b2.png'
img = cv2.imread(img)
cv2_imshow(img)
#plt.imshow(img)
#plt.show()
```

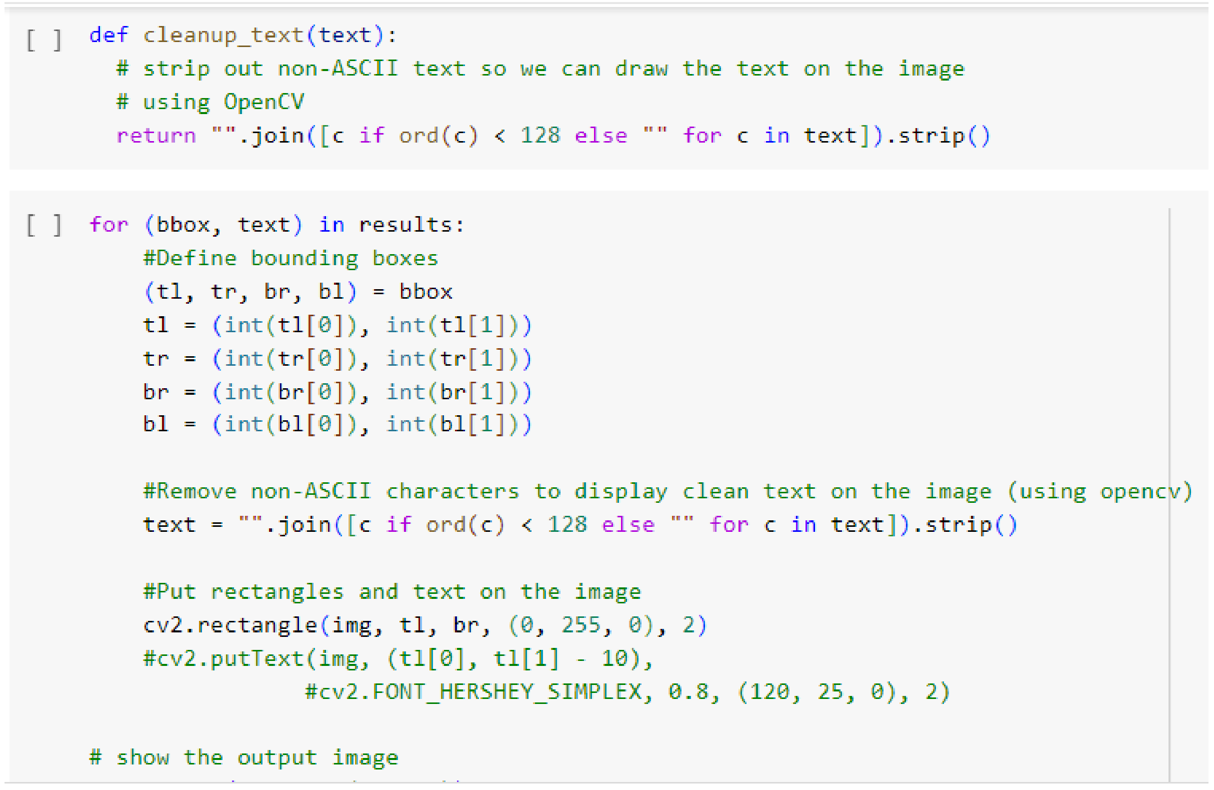


```
#results = reader.readtext(img ,  detail = 0, text_threshold=0.8 , paragraph=True)
results = reader.readtext(img , decoder='greedy', beamWidth=5, batch_size=4, workers=0,
                          allowlist=None, blocklist=None, detail=0, rotation_info=None, paragraph=True,
                          min_size=20, contrast_ths=0.4, adjust_contrast=0.5, filter_ths=0.003,
                          text_threshold=0.8, low_text=0.4, link_threshold=0.4, canvas_size=2560,
                          mag_ratio=1.0, slope_ths=0.1, ycenter_ths=0.5, height_ths=0.5, width_ths=0.5,
                          y_ths=0.5, x_ths=1.0, add_margin=0.1, threshold=0.2, bbox_min_score=0.5,
                          bbox_min_size=3, max_candidates=0, output_format='standard')


print(results)
```

```
['ROAD WORK AHEAD']
```

```
[ ] f = open("demofile3.txt", "w")
    for (bbox, text) in results:
      f.write(text + '\n')
    f.close()
    f = open("demofile3.txt", "r")
    print(f.read())
```

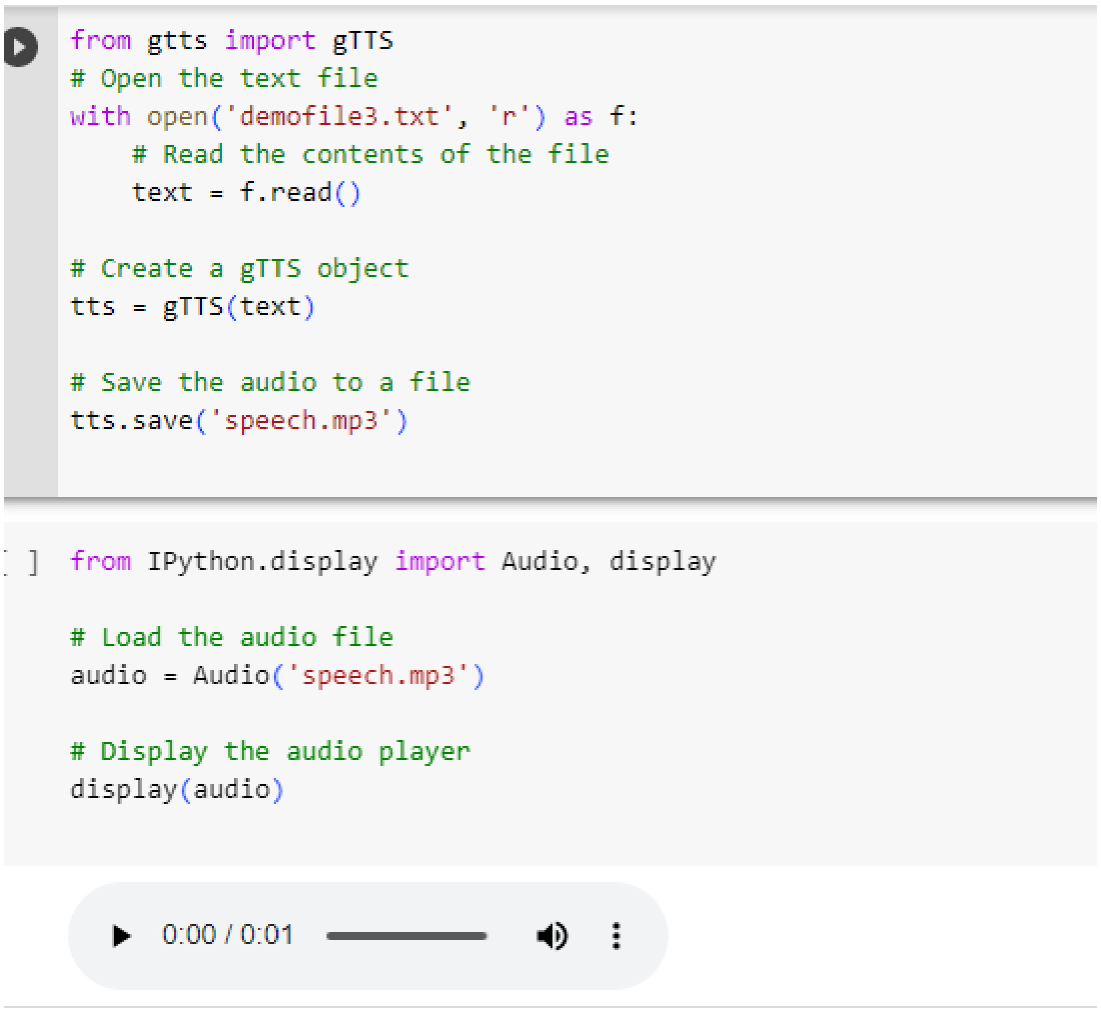
```
ROAD WORK AHEAD
```



```
[ ] def cleanup_text(text):
      # strip out non-ASCII text so we can draw the text on the image
      # using OpenCV
      return "".join([c if ord(c) < 128 else "" for c in text]).strip()
```

```
[ ] for (bbox, text) in results:
        #Define bounding boxes
        (tl, tr, br, bl) = bbox
        tl = (int(tl[0]), int(tl[1]))
        tr = (int(tr[0]), int(tr[1]))
        br = (int(br[0]), int(br[1]))
        bl = (int(bl[0]), int(bl[1]))

        #Remove non-ASCII characters to display clean text on the image (using opencv)
        text = "".join([c if ord(c) < 128 else "" for c in text]).strip()

        #Put rectangles and text on the image
        cv2.rectangle(img, tl, br, (0, 255, 0), 2)
        #cv2.putText(img, (tl[0], tl[1] - 10),
                    #cv2.FONT_HERSHEY_SIMPLEX, 0.8, (120, 25, 0), 2)

    # show the output image
```

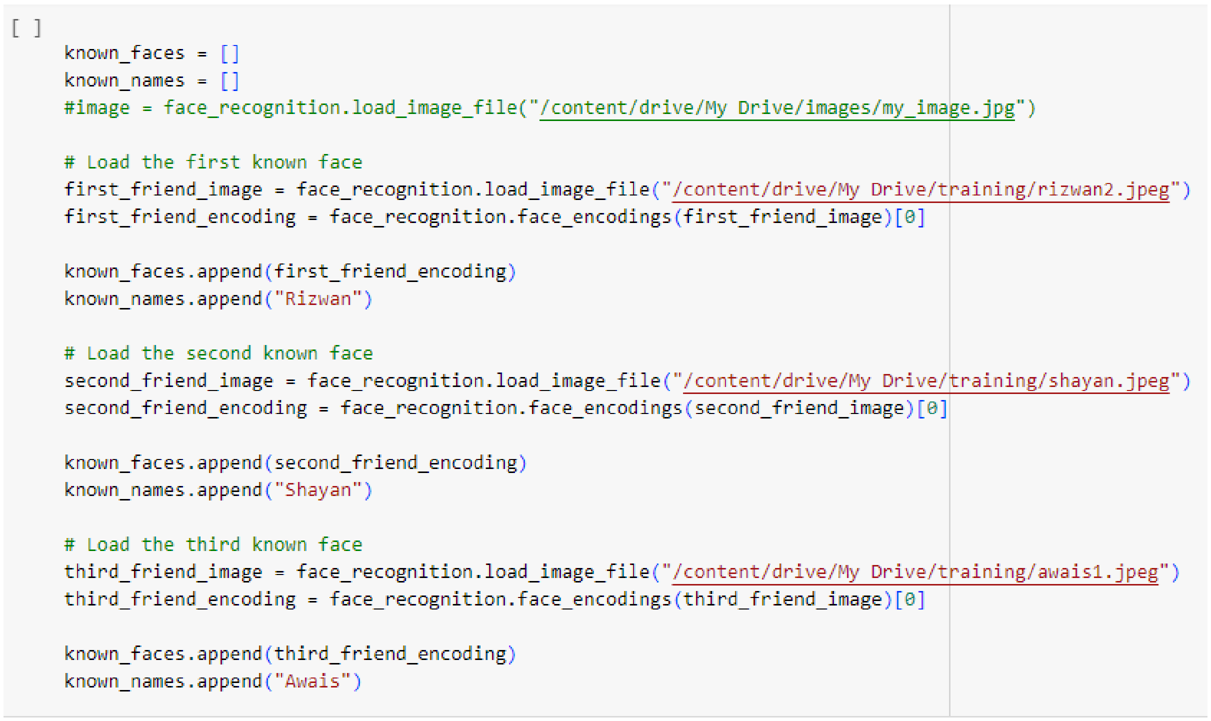
4.1.2. Text to Speech using GTTS



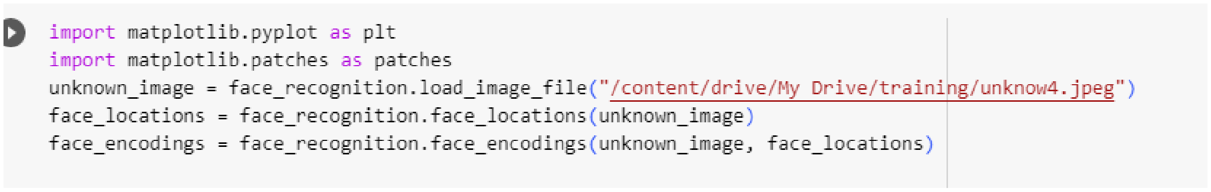
```python
from gtts import gTTS
# Open the text file
with open('demofile3.txt', 'r') as f:
    # Read the contents of the file
    text = f.read()

# Create a gTTS object
tts = gTTS(text)

# Save the audio to a file
tts.save('speech.mp3')
```

```python
from IPython.display import Audio, display

# Load the audio file
audio = Audio('speech.mp3')

# Display the audio player
display(audio)
```

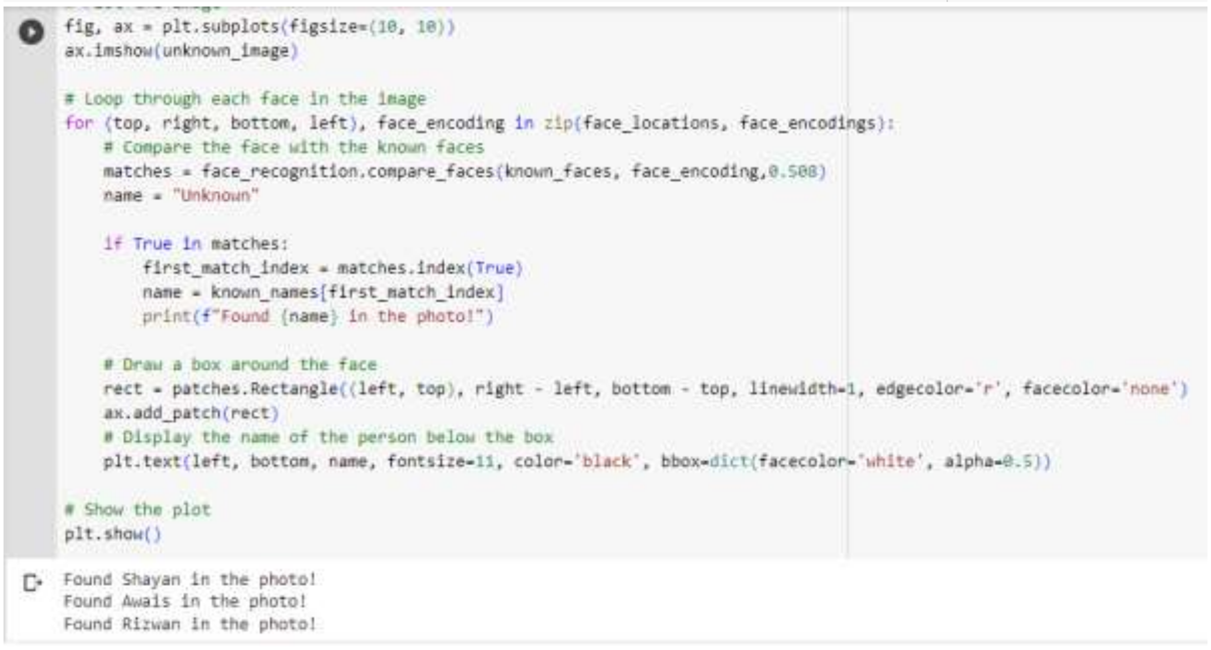
4.1.3. Facial Recognition



```
[ ]
known_faces = []
known_names = []
#image = face_recognition.load_image_file("/content/drive/My Drive/images/my_image.jpg")

# Load the first known face
first_friend_image = face_recognition.load_image_file("/content/drive/My Drive/training/rizwan2.jpeg")
first_friend_encoding = face_recognition.face_encodings(first_friend_image)[0]

known_faces.append(first_friend_encoding)
known_names.append("Rizwan")

# Load the second known face
second_friend_image = face_recognition.load_image_file("/content/drive/My Drive/training/shayan.jpeg")
second_friend_encoding = face_recognition.face_encodings(second_friend_image)[0]

known_faces.append(second_friend_encoding)
known_names.append("Shayan")

# Load the third known face
third_friend_image = face_recognition.load_image_file("/content/drive/My Drive/training/awais1.jpeg")
third_friend_encoding = face_recognition.face_encodings(third_friend_image)[0]

known_faces.append(third_friend_encoding)
known_names.append("Awais")
```

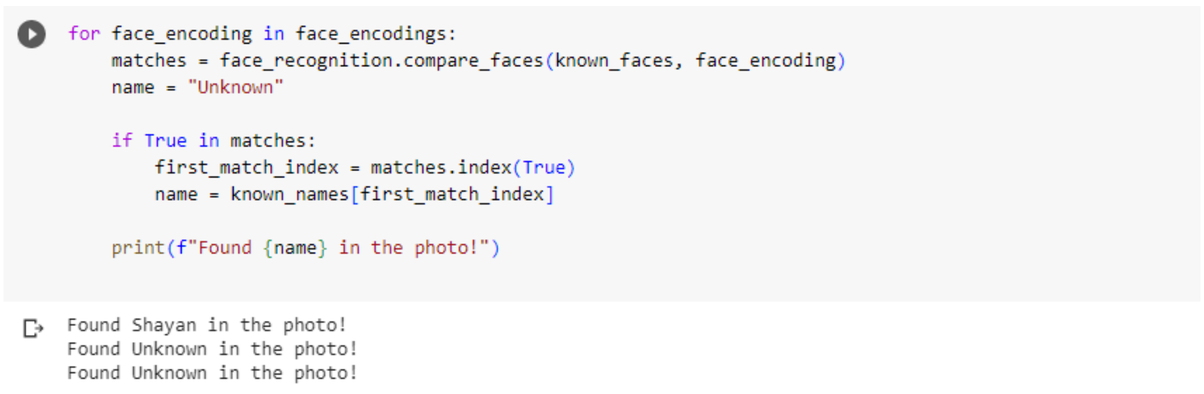


```
import matplotlib.pyplot as plt
import matplotlib.patches as patches
unknown_image = face_recognition.load_image_file("/content/drive/My Drive/training/unknow4.jpeg")
face_locations = face_recognition.face_locations(unknown_image)
face_encodings = face_recognition.face_encodings(unknown_image, face_locations)
```



```
fig, ax = plt.subplots(figsize=(10, 10))
ax.imshow(unknown_image)

# Loop through each face in the image
for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
    # Compare the face with the known faces
    matches = face_recognition.compare_faces(known_faces, face_encoding,0.508)
    name = "Unknown"

    if True in matches:
        first_match_index = matches.index(True)
        name = known_names[first_match_index]
        print(f"Found {name} in the photo!")

    # Draw a box around the face
    rect = patches.Rectangle((left, top), right - left, bottom - top, linewidth=1, edgecolor='r', facecolor='none')
    ax.add_patch(rect)
    # Display the name of the person below the box
    plt.text(left, bottom, name, fontsize=11, color='black', bbox=dict(facecolor='white', alpha=0.5))

# Show the plot
plt.show()
```

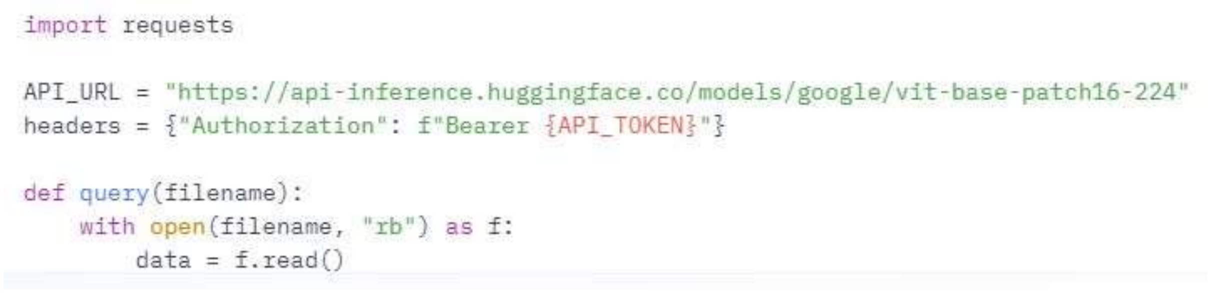
```
Found Shayan in the photo!
Found Awais in the photo!
Found Rizwan in the photo!
```



```python
for face_encoding in face_encodings:
    matches = face_recognition.compare_faces(known_faces, face_encoding)
    name = "Unknown"

    if True in matches:
        first_match_index = matches.index(True)
        name = known_names[first_match_index]

    print(f"Found {name} in the photo!")
```

```
Found Shayan in the photo!
Found Unknown in the photo!
Found Unknown in the photo!
```

4.1.4. The Vision Transformer (ViT)



```python
import requests

API_URL = "https://api-inference.huggingface.co/models/google/vit-base-patch16-224"
headers = {"Authorization": f"Bearer {API_TOKEN}"}

def query(filename):
    with open(filename, "rb") as f:
        data = f.read()
```

4.2. Software Testing

The code of each part is test in the following way:

4.2.1. EasyOCR

We have performed testing on EasyOCR of almost 100 pictures and obtained the accuracy of almost 80 percent. There was fluctuation in the results, but these were due to quality of the pictures because we can’t have the ideal conditions in the real world. Here are some results obtained:

Image 1:

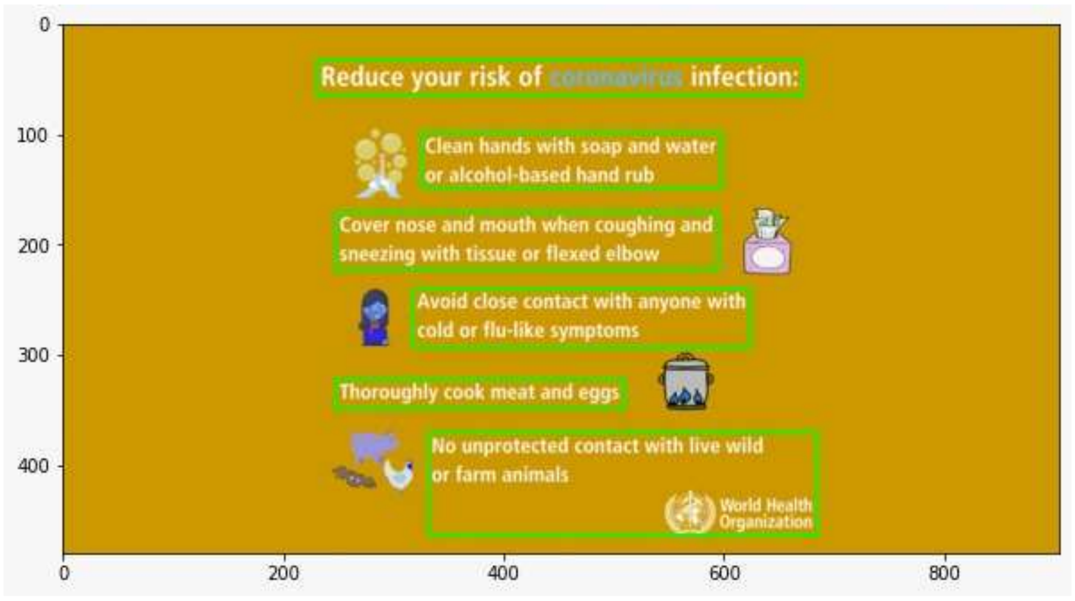
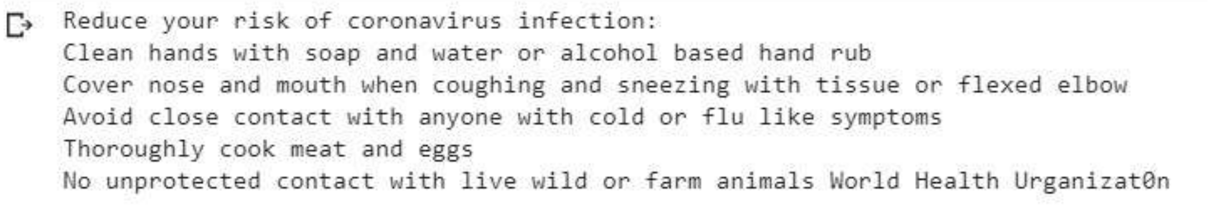

Reduce your risk of coronavirus infection:
Clean hands with soap and water or alcohol-based hand rub
Cover nose and mouth when coughing and sneezing with tissue or flexed elbow
Avoid close contact with anyone with cold or flu-like symptoms
Thoroughly cook meat and eggs
No unprotected contact with live wild or farm animals
World Health Organization
0
100
200
300
400
0
200
400
600
800

Fig.23. Input Image for EasyOCR



```
Reduce your risk of coronavirus infection:
Clean hands with soap and water or alcohol based hand rub
Cover nose and mouth when coughing and sneezing with tissue or flexed elbow
Avoid close contact with anyone with cold or flu like symptoms
Thoroughly cook meat and eggs
No unprotected contact with live wild or farm animals World Health Urganizat0n
```

Fig.24. output Text

Image 2:

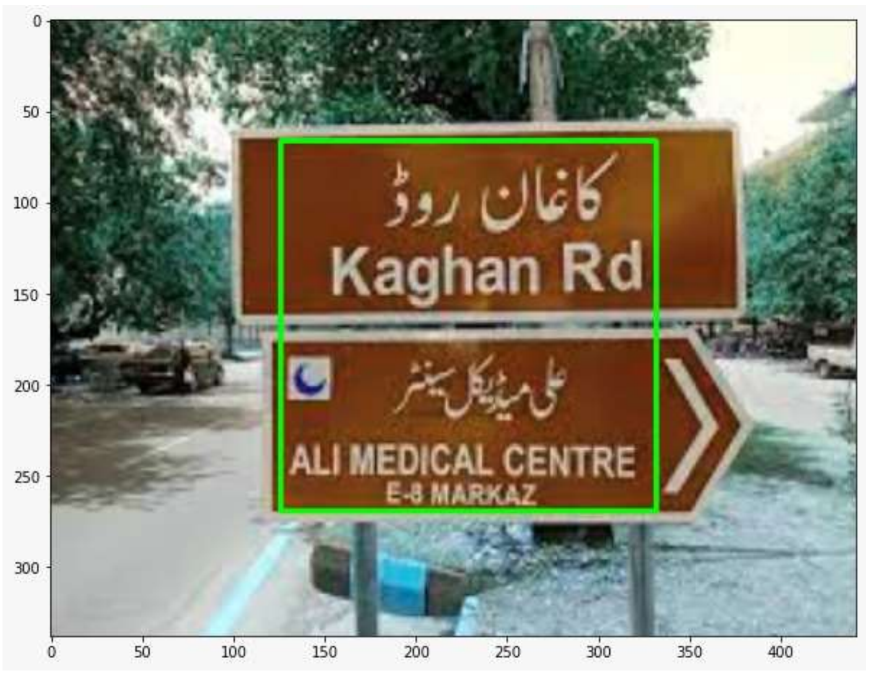
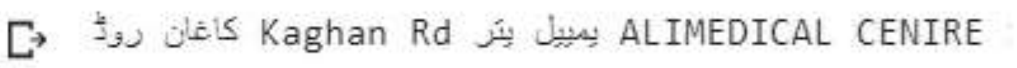


Fig.25. Input Image for EasyOCR



```
کاغان روڈ Kaghan Rd یمبیل بتر ALIMEDICAL CENIRE
```

Fig.26. Output Text

Image 3:

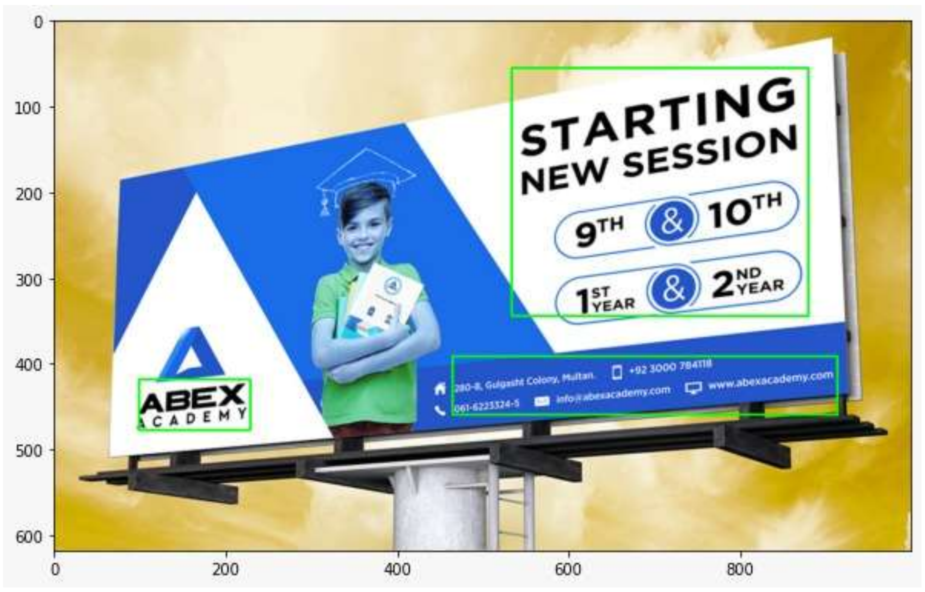
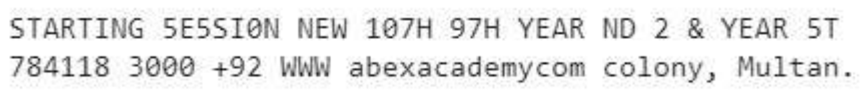


Fig.27. Input Image for EasyOCR



```
STARTING 5E5SI0N NEW 107H 97H YEAR ND 2 & YEAR 5T
784118 3000 +92 WWW abexacademycom colony, Multan.
```

Fig.28. Output Text

4.2.2. Facial Recognition Testing:

Facial Recognition model was also tested on about more than 50 pictures and accuracy of the model was approximately 78 percent. Model performed very well in detecting multiple faces in a single image. But model was not able to perform equally well in finding faces which are at some angle. Following are some results of facial recognition:

Image1:

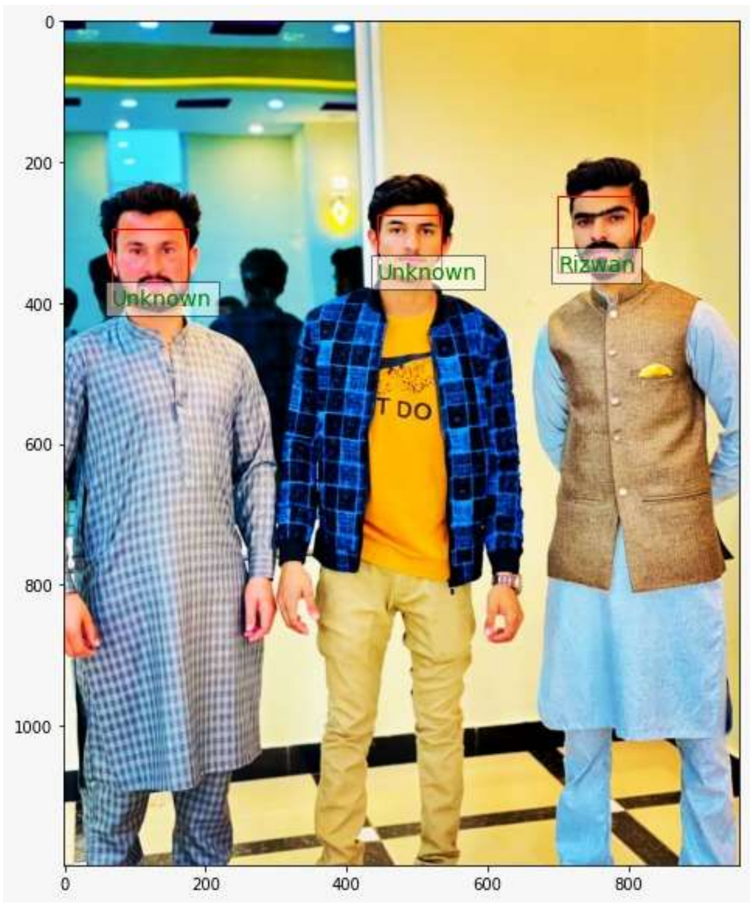


Fig.28. Input Image for Facial Recognition

Image 2:

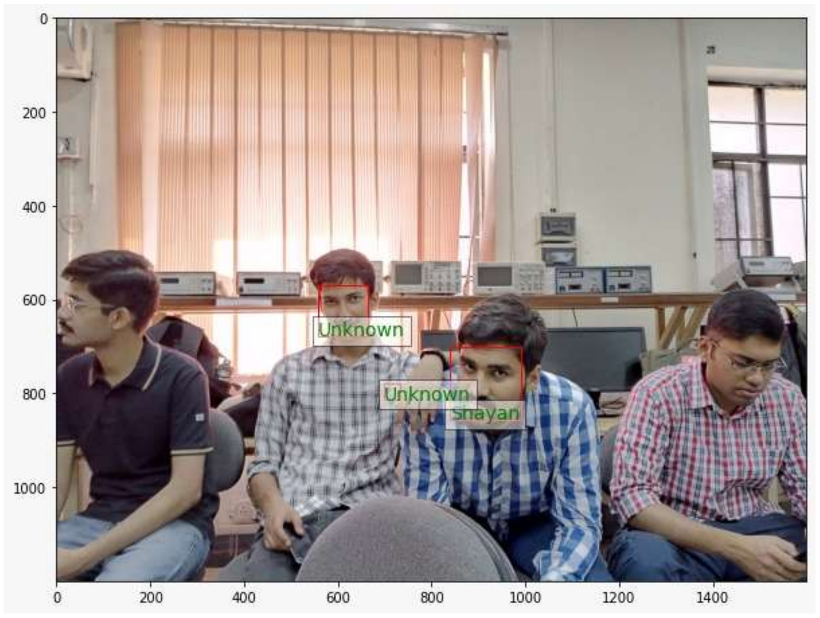


Fig.28. Input Image for Facial Recognition

Image 3:

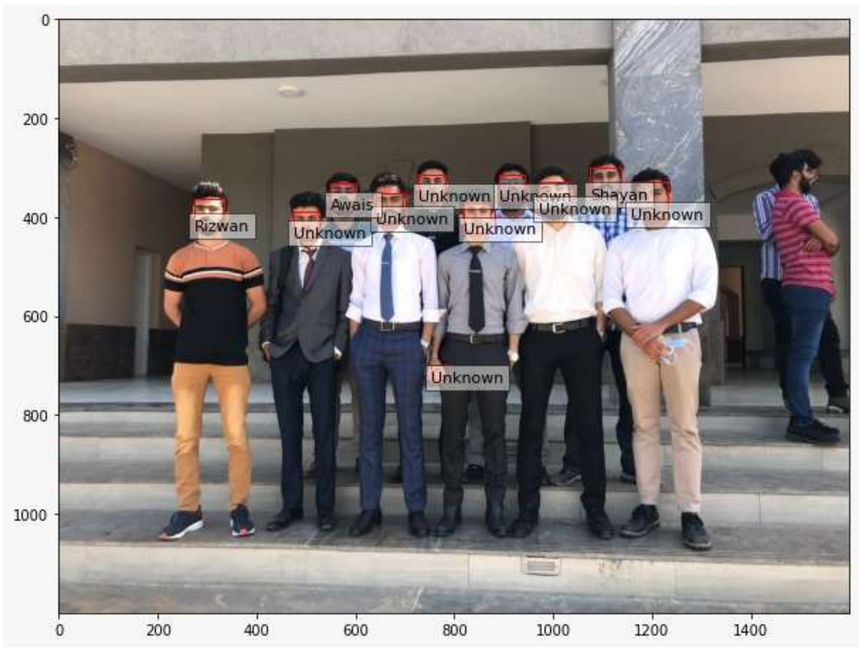


Fig.30. Input Image for Facial Recognition

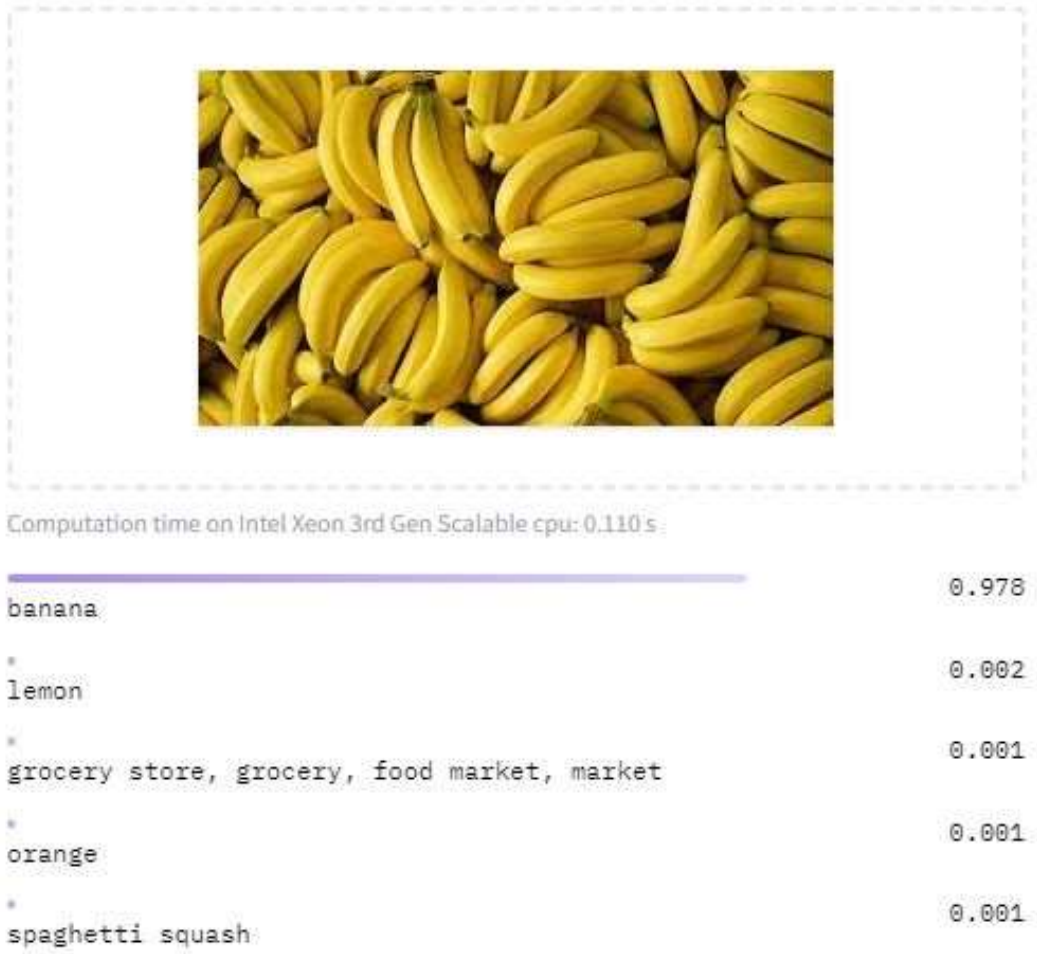
```
Found Shayan in the photo!
Found Unknown in the photo!
Found Unknown in the photo!
```

Fig.31. Output for Facial Recognition

4.2.3. Vision Transformer (ViT) Testing

We have tested the vision transformer for more than 100 pictures has given an accuracy of more than 90 percent. Results are quiet satisfying. Here are few test results:

Image1



Computation time on Intel Xeon 3rd Gen Scalable cpu: 0.110 s

```
banana                                              0.978
lemon                                               0.002
grocery store, grocery, food market, market         0.001
orange                                              0.001
spaghetti squash                                    0.001
```

Fig.29. Input Image for Vision Transformer and Result

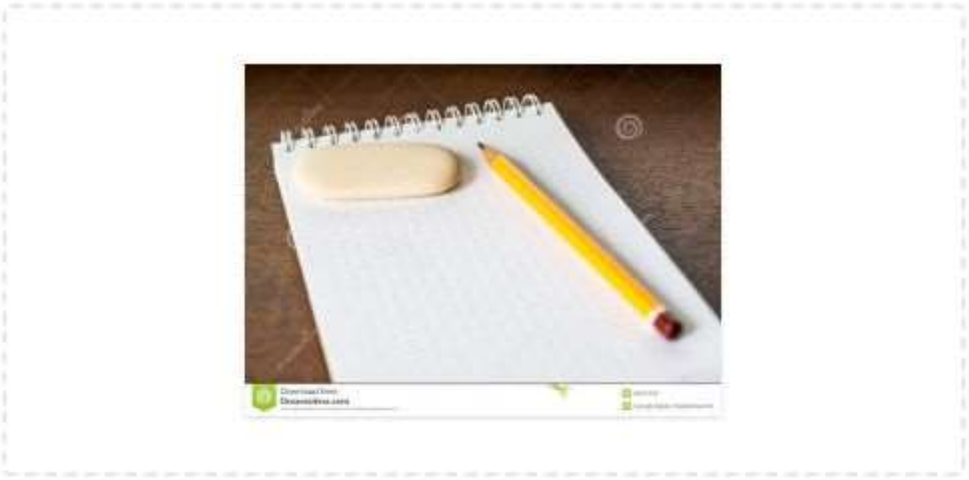
Image 2

Computation time on Intel Xeon 3rd Gen Scalable cpu: 0.093 s

```
                                                    0.987
head cabbage
                                                    0.004
grocery store, grocery, food market, market
                                                    0.004
cauliflower
                                                    0.000
cucumber, cuke
                                                    0.000
broccoli
```

Fig.30. Input Image for Vision Transformer and Result

Image 3

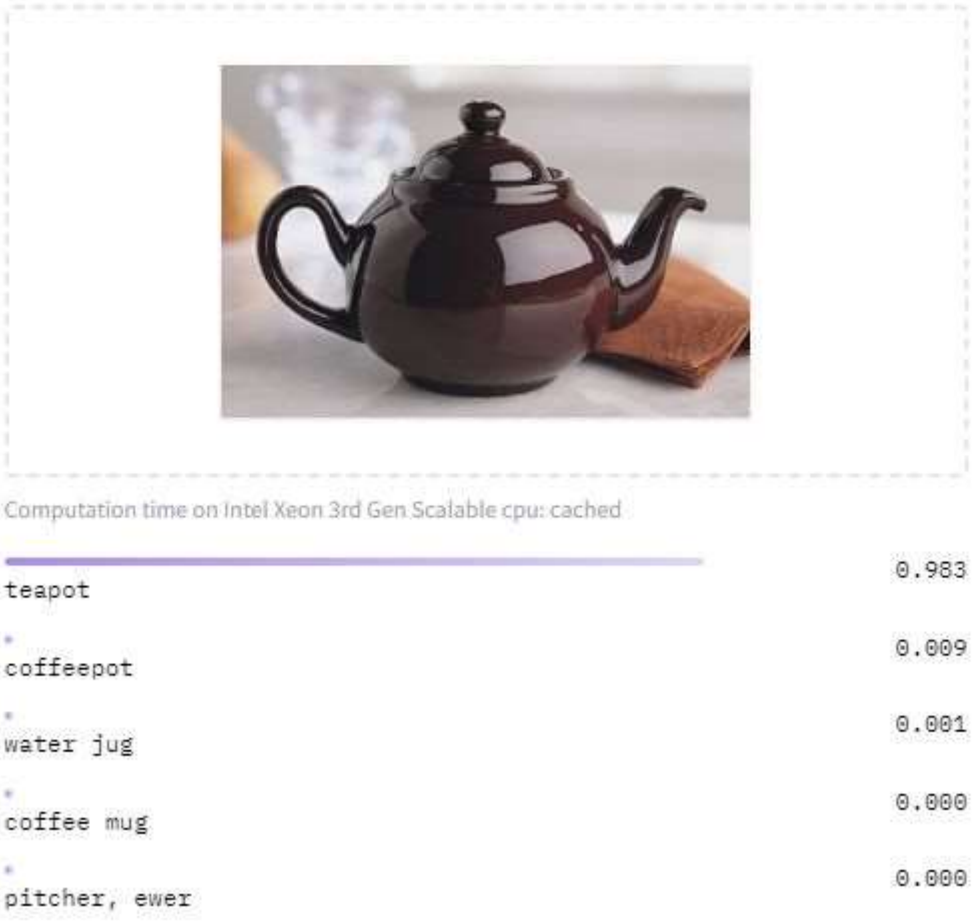


Computation time on Intel Xeon 3rd Gen Scalable cpu: cached

```
                                                  0.982
rubber eraser, rubber, pencil eraser

                                                  0.009
pencil sharpener

                                                  0.005
pencil box, pencil case

                                                  0.002
rule, ruler

                                                  0.001
ballpoint, ballpoint pen, ballpen, Biro
```

Fig.31. Input Image for Vision Transformer and Result

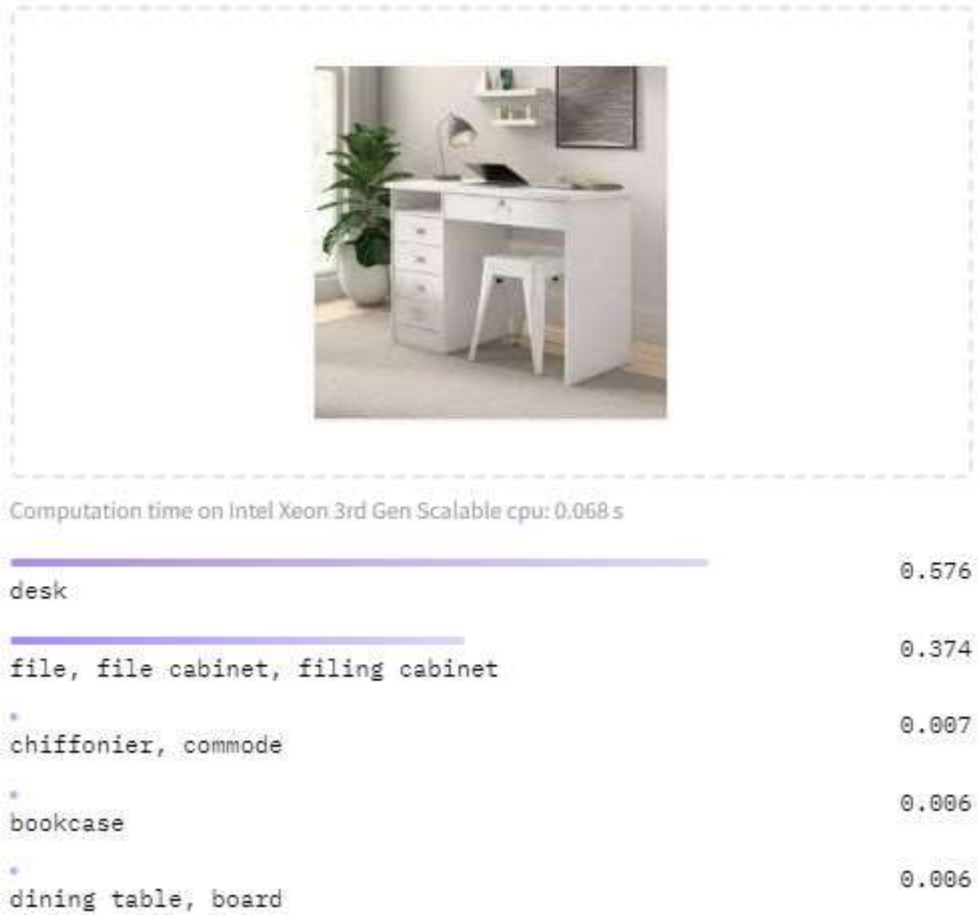
Image 4


Computation time on Intel Xeon 3rd Gen Scalable cpu: cached

```
teapot                0.983
coffeepot             0.009
water jug             0.001
coffee mug            0.000
pitcher, ewer         0.000
```

Fig.32. Input Image for Vision Transformer and Result

Image 5



Computation time on Intel Xeon 3rd Gen Scalable cpu: 0.068 s

```
desk                                              0.576
file, file cabinet, filing cabinet                0.374
chiffonier, commode                               0.007
bookcase                                          0.006
dining table, board                               0.006
```

Fig.33. Input Image for Vision Transformer and Result

CH 5: HARDWARE

5.1. Hardware Implementation

5.1.1. ESP-32 CAM:

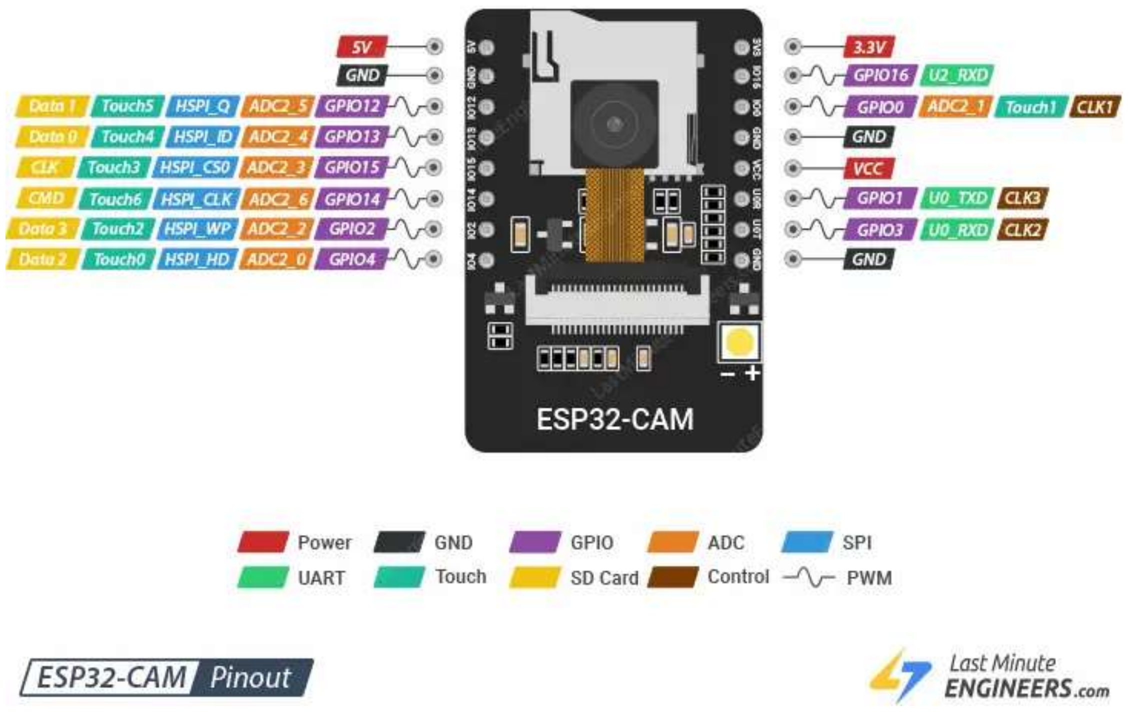


Fig. 34. ESP-32 with Camera Module

We will be using the ESP-32 CAM as our main microcontroller/WiFi module in this project. This microcontroller has a number of features that we will be using. A general overview of them is listed below:

Processor: The processor on this module is built with two 32-bit cores and operates at a 240MHz frequency which means it is able to carry out our main tasks of image parsing and facial recognition.

Memory: The module contains 520kB of memory RAM with an extra 4MB flash attached which makes image processing easy. It also has external 4MB PSRAM to expand storage capacity.

Camera: The camera has a resolution of 2MP which means it can take an image of 1600x1200pi max.

Power Pins: These are available in the configuration of 3.3V and 5V where 5V comes with a voltage regulator. They will be used for powering our circuit.

GPIO Pins: The module has 10 GPIO pins. We will be using 3 of them to enable switching for our 3 modes.

UART Pins: The chip has 2 UART pins. We won't be using these for our project.

MicroSD Card Pins: These pins are used to interface the external microSD card that can be attached to the module. If no card is present, they act as regular GPIO pins.

Touch Pins: The chip has 7 capacitive touch-sensitive GPIO pins. They can be used to detect touch which causes a change in capacitance. We will not be using these in our project.

SPI Pin: The ESP-32 contains a single SPI pin for slave and master modes.

PWM Pins: The chip contains 10 PWM pins. All of them are GPIO. They can be used for motor and LED control. We won't be using for these either.

5.1.2. FTDI Programmer:

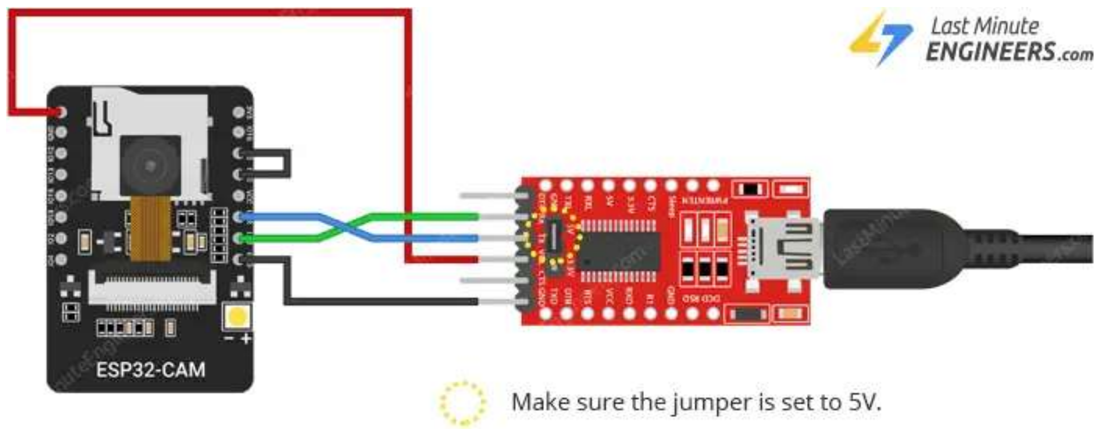


Fig.35. FTDI Programmer

We will be using the FTDI adapter/programmer to code our ESP-32 CAM as it lacks a microUSB port. The ground pins of both are connected and then we provide a 3.3V supply for our module. The adapter has a jumper pin to select between 5V and 3.3V output, we'll keep it at 3.3V. We will disconnect GPIO 0 after uploading the code so that the module goes into run mode instead of upload mode. Now our ESP chip is ready to be programmed.

5.1.3. Web Server:

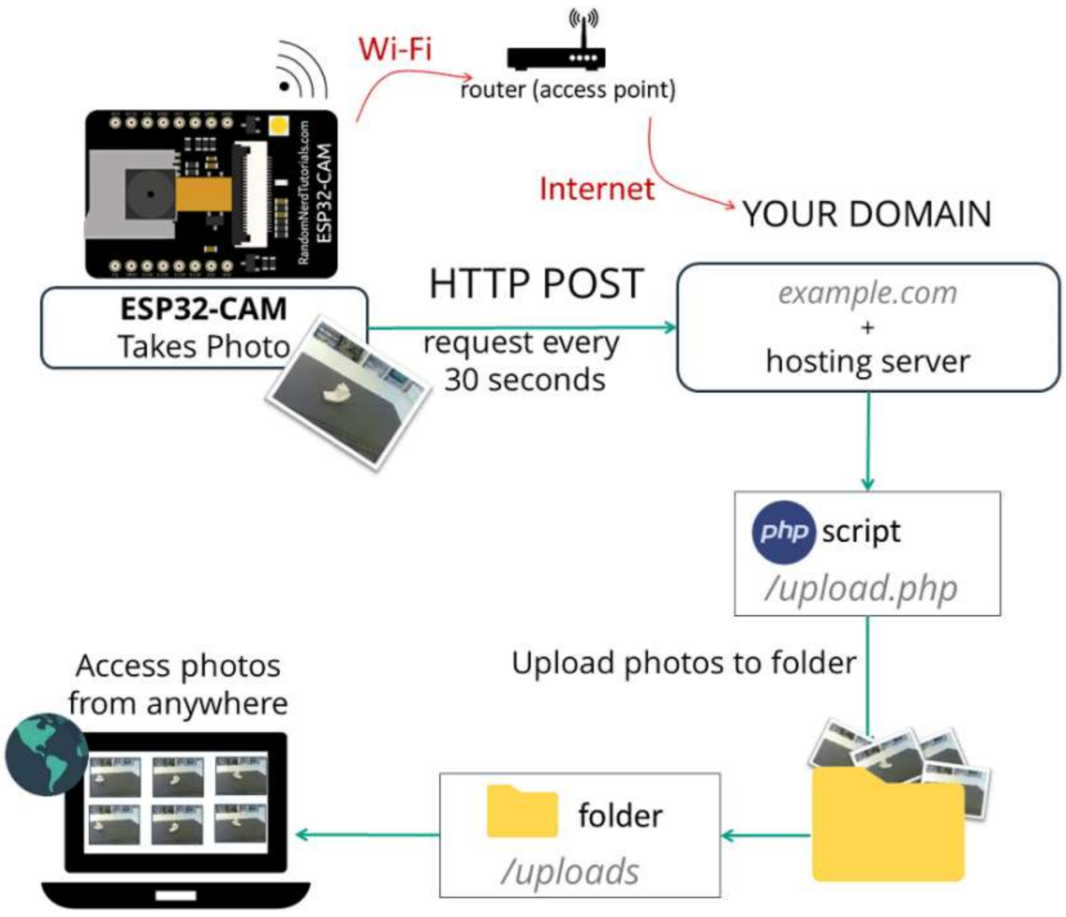


Fig.36. Web Server

We will be using a web server (bluehost) to host our images and then send them further through API calls. Using 3 inputs for our chip, we will use a comparison statement and send them to different folders on the server. A POST code is uploaded to the ESP-32 and a GET one on the server. We set up a watch variable to check for changes on the server. Whenever an image is uploaded, the variable sends a signal and the web server does an API call. Each folder of the server is made using the user-friendly cPanel.

Once our web server has been properly configured and set up, we can move on towards configuring the API calls.

5.1.4. API configuration:

We will be switching to various different APIs to accomplish our tasks from this point onward. Pre-trained models that have already been hosted on different sites are available for inference using their APIs. We will be using these to get our outputs from these models and then saving them on the server. Once the server has confirmed that it has received the outputs they will be sent to a TTS service through yet another API which will get our final output.

5.2. Final Result:

Once we have received the voice signal on our server it will be sent back to the module which will be connected to a pair of earphones. The voice will be transmitted to the user. There are also error commands added to the system in case of a problem to ensure the user knows where the problem lies. In this way, with the press of a button we will be able to communicate what an object is, who is coming or what a text says.

REFERENCES

- [1] Smart Guiding Glasses for Visually Impaired People in Indoor Environment by Jinqiang Bai, Shiguo Lian, *Member, IEEE*, Zhaoxiang Liu, Kai Wang, Dijun Liu
- [2] A Unique Smart Eye Glass for Visually Impaired People, Md. Razu Miah, Md. Sanwar Hussain, Dept. of Electrical and Electronic Engineering, Metropolitan University, Sylhet, Bangladesh.
- https://github.com/google-research/vision_transformer#expected-vit-results
- An image is worth 16x16 words: transformers for image recognition at scale. https://arxiv.org/pdf/1706.03762.pdf
- Attention Is All You Need Ashish Vaswani Noam Shazeer, Niki Parmar, Uszkoreit Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin. https://arxiv.org/pdf/2104.14294.pdf
- Emerging Properties in Self-Supervised Vision Transformers Mathilde Caron1,2 Hugo Touvron1,3 Ishan Misra1 Herve Jegou ´ 1 Julien Mairal2 Piotr Bojanowski1 Armand Joulin1. https://arxiv.org/pdf/2010.11929.pdf
- https://huggingface.co/docs/transformers/tasks/image_classification
- https://github.com/JaidedAI/EasyOCR
- https://sh-tsang.medium.com/review-beit-bert-pre-training-of-image-transformers-c14a7ef7e295

ORIGINALITY REPORT

15%	7%	7%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Md. Razu Miah, Md. Sanwar Hussain. "A Unique Smart Eye Glass for Visually Impaired People", 2018 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE), 2018 Publication	1%
2	www.cs.toronto.edu Internet Source	1%
3	Submitted to Boston University Student Paper	1%
4	Submitted to Indiana University Student Paper	1%
5	Submitted to Frisco Independent School District Student Paper	1%
6	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	1%
7	Submitted to The Robert Gordon University Student Paper	1%