Vision to Voice

A DISSERTATION SUBMITTED TO

SVKM'S NMIMS (DEEMED TO BE UNIVERSITY)
IN PARTIAL FULFILLMENT FOR THE DEGREE OF

MASTERS OF SCIENCE
IN
STATISTICS AND DATA SCIENCE

BY
Aniruddha Shelke 86062300008
Heeta Parmar 86062300012
Sakshi Shinde 86062300041
Suraj Giramkar 86062300052
Parth Sante 86062300067
Indrayani Shinde 86062300068

UNDER THE SUPERVISION OF
Dr Pradnya Khandeparkar

NILKAMAL SCHOOL OF MATHEMATICS, APPLIED STATISTICS AND ANALYTICS

SVKM's Narsee Monjee Institute of Management Studies
(Deemed-To-Be-University)
V.L. Mehta Rd, Vile Parle (West), Mumbai – 400056
November 2024

ACKNOWLEDGEMENT

We, the students of NSoMASA, MSc Statistics and Data Science, Semester 3, acknowledge our gratitude to all the individuals associated with this project.

We would like to pay our respect to our mentor, Dr. Pradnya Khandeparkar, who has imparted us with knowledge, inspiration, and motivation throughout the course of the project. It would not have been possible for us to complete the same without her sincere and affectionate help.

Further, we would like to thank SVKM's Library for allowing us to access the portal through which we got the required research papers, datasets, and study material needed for the project.

At last, we also thank our faculty, parents, and friends who contributed to some extent to complete the same.

INDEX

Abstract
Introduction
Literature Review
Rationale
Aims & Objectives
Data preparation
Methodology
Results and Discussion
Summary and Conclusion
References
Bibliography
Appendix

Abstract

---

*"**The gift of sight opens the door to independence, awareness, and discovery. Without it, every interaction is a journey of trust and adaptation; making technology a crucial guide for those in the dark.**"* Vision plays a significant role in interacting with the world. Visually impaired people face challenges daily. Without sight, their other senses namely hearing, touch are more powerful but they need assistance in crowded situations. Through the help of advanced systems, we can minimize their challenges with the help of models for object detection and speech generation. In this project, **COCO** dataset of – images have been processed to train the model on various common objects. Common objects belonging to 80 different classes are considered. We have utilized the **YOLOv8** model for object detection. **YOLOv8** has revolutionized the field of object detection, enabling accurate and efficient detection of objects in real-time scenarios. In addition to object detection, this project integrates text-to-speech (TTS) functionality to assist visually impaired users in accessing textual information within their surroundings. Using Google Text-to-Speech (gTTS) and pyttsx3, we provide vocal commands that articulate detected objects and their classifications. This TTS capability allows users to receive immediate, vocalized information about both objects and any textual content, further enhancing their understanding of the environment.

Furthermore, we have included the distance approximation technique as an additional feature to enhance the experience. By calculating the relative distance of detected objects, we can inform users about the nearest and the farthest obstacle, making them aware about the dynamic surrounding. Through this combined use of YOLOv8, gTTS, and pyttsx3, the project delivers a comprehensive, AI-powered assistive system that empowers visually impaired individuals to interact with the world in a more autonomous and secure manner.

Introduction

Vision is one of the most basic amongst the five senses of human, providing individuals with the ability to perceive and interpret the surrounding. The ability to see allows people to interpret, understand, and interact with their surroundings with ease, making vision one of the most fundamental senses. For visually impaired individuals, the absence of sight poses significant challenges in performing even the simplest tasks. Simple tasks that individuals with vision perform like handling crowded situations, reading street signs, and preventing hitting an obstacle are done with ease rather than blind individuals. This lack of visual information means that visually impaired individuals must rely heavily on other senses, such as touch and hearing, which have significant limitations in dynamic, ever-changing environments. These limitations make them more dependent on others, limiting their strength, access to information, and overall independence. Blind people usually rely on assistive tools like cane's, guide dogs to do their daily tasks. The Braille system enables tactile reading but is limited to static information and cannot be applied in most public spaces. Although there are advancements in the Braille system, it cannot be employed in a dynamic environment. These existing tools, while helpful, do not address the full range of challenges that visually impaired individuals face, particularly in unfamiliar or crowded spaces. The lack of effective, real-time navigation tools means that blind individuals still face challenges when performing fundamental tasks. The project was designed to offer a holistic solution that goes beyond simple obstacle detection.

By using the latest technologies deploying Artificial Intelligence and Computer vision, we try to offer a solution which can ease their daily challenges faced by them.

Leveraging advanced Computer vision techniques to develop an object detection model for classifying objects in desired classes. Through computer vision, object detection, text-to-speech conversion, and distance approximation, we aim to create an interactive experience that offers immediate feedback and assistance to the user.

The goal is to enable blind individuals to feel more secure and confident as they navigate through public spaces, such as crowded streets, shopping centers, and public transportation. By equipping them with a tool that can provide continuous feedback on their surroundings, we aim to reduce their dependency on others and empower them to make decisions on their own. This project intends to break down barriers by offering a solution that operates in real time, capturing the complexity of dynamic environments and allowing users to respond to them effectively. It is about not only enhancing physical navigation but also granting users the freedom to access information about their surroundings, thereby fostering a sense of independence.

Using the COCO dataset including 80 different classes which covers the common outdoor objects like people, vehicles, etc. In addition, the detected object's text is further converted to audio (text to speech) for effective usage.

Literature Review

For this project, YOLOv8 (You Only Look Once version 8) has been employed as the core object detection model. YOLOv8 offers several advantages for object detection such as Speed and efficiency and accuracy. It is renowned for its ability to process images and videos in real-time, making it ideal for applications that require rapid detection.It delivers high accuracy rates, even on challenging datasets, ensuring reliable object localization and classification.

The development of assistive technologies for visually impaired individuals has gained significant attention in recent years. Various studies have focused on enhancing the independence of visually impaired users through innovative systems that convert visual information into auditory signals. For instance, (S & D, 2018) explored the use of Optical Character Recognition (OCR) algorithms combined with Text-to-Speech (TTS) engines to facilitate the reading of product labels, demonstrating the potential of such technologies in everyday tasks. Additionally, S and D (2018) presented a method for image-to-audio conversion using portable cameras, further emphasizing the importance of accessible solutions for the visually impaired community. The current project builds upon these foundations by integrating object detection and text-to-speech conversion, aiming to provide a comprehensive system that enhances the quality of life for visually impaired individuals.

The literature highlights the challenges faced by visually impaired individuals, with a significant global population affected, particularly in India. Previous works, such as the Smart Glasses Application by (Kadam et al., n.d.), have explored the use of technology to aid the blind by capturing images and processing them for navigation assistance. The proposed system builds on these foundations by integrating TensorFlow's object detection capabilities, emphasizing real-time processing and audio feedback to improve the independence and quality of life for visually impaired users.

Recent advancements in deep learning, particularly in computer vision, have significantly improved object detection, offering innovative solutions for assisting visually impaired individuals. Traditional aids, such as guide canes, often fall short in helping users navigate safely, highlighting the need for more effective technologies (World Health Organization, 2010).

The YOLO (You Only Look Once) algorithm has emerged as a powerful tool for real-time object detection, enabling rapid identification and localization of objects (Najm et al., n.d.). Studies indicate that combining object detection with audio feedback enhances navigation for visually impaired users, providing real-time environmental information. This paper builds on these findings to develop a user-friendly system that leverages deep learning to improve mobility and safety for visually impaired individuals.

Rationale

The rationale for this project lies in addressing the challenges faced by the blind community and developing a system to assist them. Traditional technologies have certain limitations in terms of their effectiveness. By using the new technology, this project aims to develop a mobile application which can detect the obstacle and generate voice to navigate their surroundings. Whether it is identifying nearby pedestrians, detecting an approaching vehicle, or locating objects like chairs, doors, or street signs, YOLOv8's capabilities allow for fast and accurate recognition, thus enhancing the user's confidence in navigating their surroundings. The app can help the visually impaired avoid potential hazards and navigate safely.

The extensive analysis is carried out on the COCO dataset which consists of – images and the YOLOv8 model is used to detect the obstacles and further gTTS and pythonttsx3 is utilized for text to speech conversion.

Aims & Objectives

Aims:

- To assist the visually impaired people by providing them with an effective tool for obstacle detection.
- To improve the experience and increase independence as traditional tools have limitations.
- To leverage computer vision technology to develop an object detection model for classifying objects in desired classes.

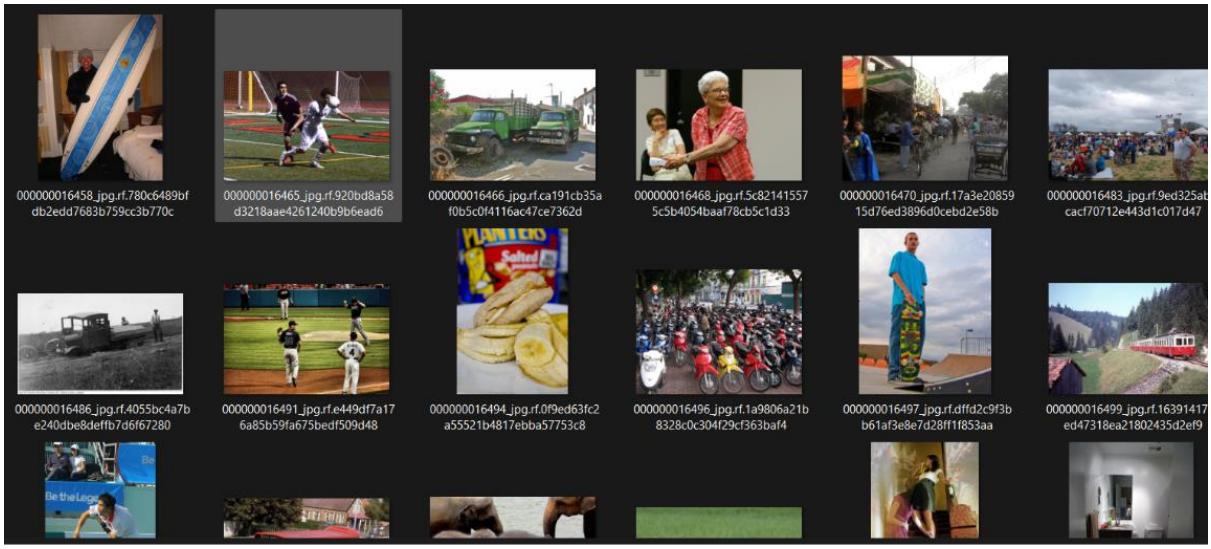
Objectives:

- Develop an object detection model for identifying obstacles which will aid visually impaired individuals.
- Transform the generated text to speech to assist the users for easy accessibility.
- Incorporate distance approximation to provide users with spatial cues about nearby obstacles.
- Voice commands are limited to only the nearest and the farthest object.
- To enable directional guidance which suggests movement to left or right depending on the obstacle present.

About the data

Data was sourced from MS-COCO(Common Objects in Context) 2017, Our data has 2,32,817 images with their annotated text files. The images are categorized into 80 classes. The classes are quite common categories like people, vehicles etc.

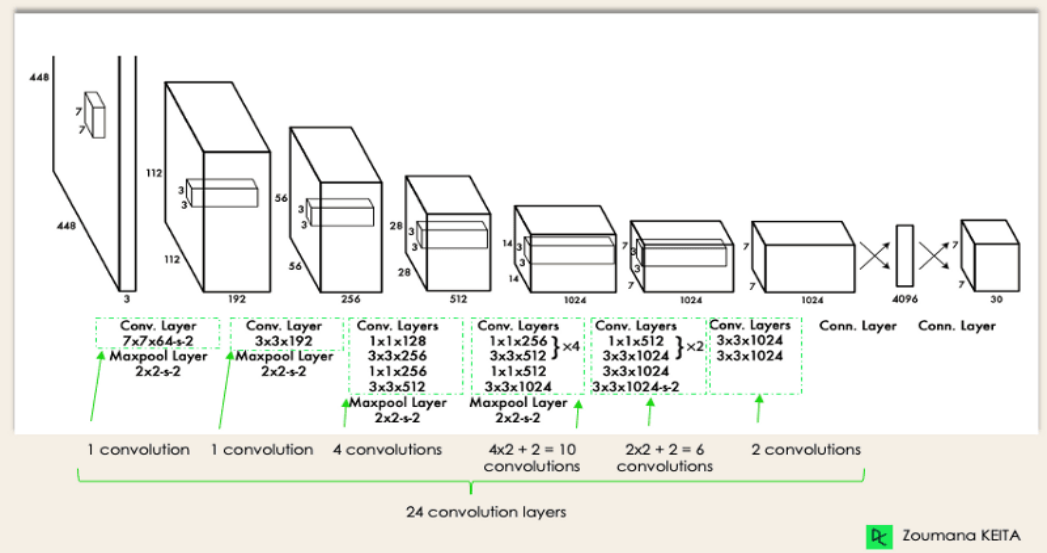
DATA SNAPSHOT:



The images were annotated which were then mapped to their respective classes and a yaml file was created for the same.

Methodology

Yolo(You only Look Once) is an object detection algorithm. It is a convolutional neural network that predicts bounding boxes and class probabilities of an image in a single evaluation.

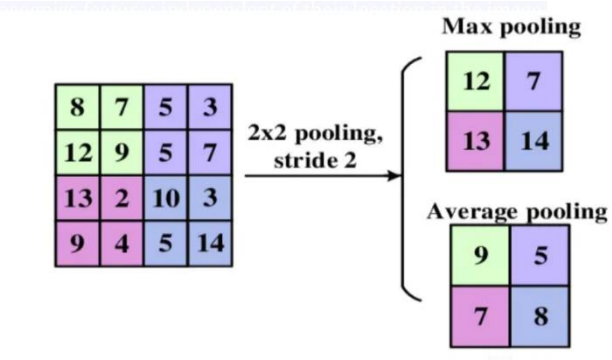


YOLO Architecture and Object Detection Process

The YOLO (You Only Look Once) architecture is a fast, efficient object detection framework designed to localize and classify multiple objects within images in real-time. Modelled similarly to GoogleNet, it comprises 24 convolutional layers, four max-pooling layers, and two fully connected layers. YOLO resizes input images to 448x448 pixels before feeding them through the network, applying a 1x1 convolution to reduce channel count, followed by a 3x3 convolution. The activation function is ReLU, except in the final layer, which uses a linear activation function.

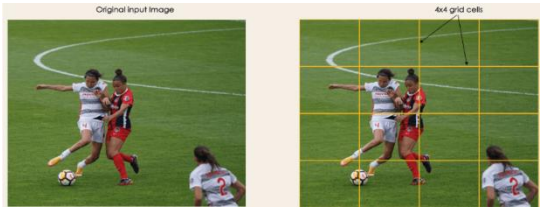
The pooling layer plays a significant role here as it helps to reduce the image size and helps to reduce overfitting.

Pooling in convolutional neural networks is a technique for generalizing features extracted by convolutional filters and helping the network recognize features independent of their location in the image. YOLO models use max pooling.



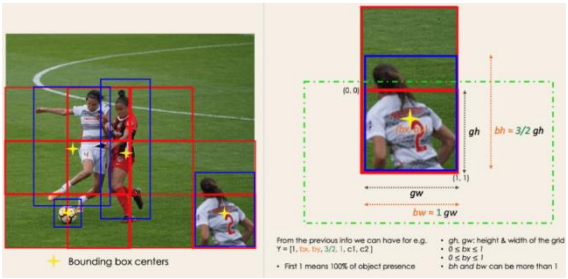
How YOLO Performs Object Detection

1. **Residual Blocks:** The image is divided into an NxN grid (e.g., N=4). Each cell in the grid is responsible for detecting an object within its boundaries, providing predictions for object presence and class probabilities.

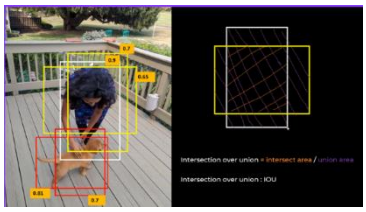


2. **Bounding Box Regression:** For each object, bounding boxes are generated to outline the object's position within a grid cell. Each bounding box is represented by parameters [pc, bx, by, bh, bw, c1, c2], where:

- pc is the confidence score of an object in a grid cell.
- bx, by are the x and y coordinates for the bounding box center.
- bh, bw represent the height and width.
- c1 and c2 are the class probabilities for specific objects like a player or ball.



3.**Intersection Over Union (IOU):** IOU helps eliminate redundant bounding boxes by calculating the overlap ratio (intersection over union) for each box. A threshold (e.g., 0.5) determines which boxes are kept based on their relevance.



4. **Non-Maximum Suppression (NMS):** NMS further refines box selection by retaining only the boxes with the highest confidence score, removing overlapping boxes that might otherwise add noise.

This process makes YOLO highly efficient for real-time applications, allowing for accurate multi-object detection in a single pass through the network.

Text to speech techniques:

Further, for transforming the virtual information into audio i.e. spoken instructions, we have utilized two libraries; Google text to speech (gTTS) and Python library (pyttsx3). By combining these two libraries we have made our model versatile as it can function in both connected as well as offline mode.

gTTS: gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file. The gTTS API supports several languages including English, Hindi, Tamil, French, German and many more. The speech can be delivered in any one of the two available audio speeds, fast or slow.

To install the gTTS API, open terminal and write

pip install gTTS

Pyttsx3:

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to pyttsx3. Engine instance. It is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is female and the second is male which is provided by “sapi5” for windows. It supports three TTS engines :

sapi5 – SAPI5 on Windows

nsss – NSSpeechSynthesizer on Mac OS X

espeak – eSpeak on every other platform

To install the pyttsx3 module, first, you must open the terminal and write

pip install pyttsx3

Distance Approximation:

Distance approximation is the process of calculating the real-world distance of an object based on its size in the image and the camera’s characteristics. This is a critical function in various computer vision applications, such as object detection, robotics, and augmented reality. Below is the detailed breakdown of how distance approximation works in the context of the YOLO-based object detection system.

1. Camera Setup and Calibration:

Before we can approximate distances, the system needs to be calibrated:

- Focal Length: The focal length of the camera is an essential parameter in determining the distance. It is the distance between the camera's lens and the image sensor. In our system, the focal length is fixed at 500 pixels, which is an approximation that represents the camera's zoom level or how much the camera "sees" in terms of the pixel size.

- Real Object Size: The real-world size of objects detected by the camera is predefined. For instance, a person is assumed to have a height of 1 meter, a vehicle like a car has a length of 3 meters, etc. These values are crucial in calculating the distance based on the object's size in the frame.

2. Distance Calculation Formula:

The core formula for distance approximation is derived from the similar triangles' principle, where the size of the object in the real world is related to its size in the image (bounding box), and the camera's focal length. The formula used to calculate the distance is:

Distance = (Real Size of Object * Focal Length) / Bounding Box Dimension

- Real Size of Object: The physical size of the object in meters (e.g., 1 meter for a person, 3 meters for a car).

- Focal Length: A constant representing the zoom level of the camera. In this case, it is 500 pixels.

- Bounding Box Dimension: The size of the object in the image, measured by the bounding box's dimension (either height or width in pixels). This represents how large or small the object appears in the frame.

3. Bounding Box Dimension:

- Bounding Box: When the YOLO model detects an object, it draws a bounding box around the object in the image. The dimensions of this bounding box—either height or width—are used for distance approximation.

  - For objects like people and animals, the height of the bounding box is used to estimate distance.

  - For larger objects like vehicles, the width of the bounding box is used since they tend to be wider than tall.

The bounding box dimensions are in pixels, and they represent how large the object appears relative to its real-world size.

4. Distance Calculation Process:

Step 1: Detecting Objects

YOLO detects objects within a frame, identifies the class (e.g., person, car, dog), and draws bounding boxes around them. The size of each bounding box is then measured.

Step 2: Assigning a Real-World Size

Once the object class is identified, a real-world size is assigned to it. This can vary depending on the object, such as:

- Person: 1.0 meters (height)

- Car: 3.0 meters (length)

- Dog: 0.5 meters (length)

- Bus: 12.0 meters (length)

This real-world size is used in the formula to approximate the distance.

Step 3: Applying the Distance Formula

Using the bounding box dimension (height or width), the real-world size, and the camera's focal length, the distance to the object is calculated. The formula is applied for each detected object.

For example:

- If the bounding box height for a person is 100 pixels, the real size of a person is 1.0 meter, and the focal length is 500 pixels, the distance is calculated as:

Distance = (1.0 * 500) / 100 = 5.0 meters

So, the person is estimated to be 5 meters away from the camera.

5. Directional Guidance:

To enhance the user experience, the system does not just stop at telling the distance. It also guides the user by determining the direction (left or right) based on the position of the object in the frame.

- Frame Center: The center of the camera frame is calculated using the frame width. If the camera's frame width is 640 pixels, the center would be at 320 pixels.

- Object Center: The center of the bounding box is calculated by averaging the x1 (left) and x2 (right) coordinates of the bounding box:

Object Center = (x1 + x2) / 2

- Direction Logic:

  - If the object center is left of the frame center (less than 20% of the frame width), the system will advise to move left.

- If the object center is right of the frame center (more than 80% of the frame width), the system will advise to move right.

Example:

- If the object is detected in the left 20% of the screen, the system says: “Move left.”

- If the object is detected in the right 20% of the screen, the system says: “Move right.”

This logic ensures that the system gives you relevant guidance on where to move based on the relative position of the object in the frame.

6. Thresholding and Alerts:

To avoid constant unnecessary announcements, the system only triggers alerts when:

- The object is within a threshold distance (e.g., 5 meters), indicating that the object is close enough to need action.

- The object’s position relative to the camera has changed significantly (either closer or farther).

This helps prevent redundant alerts and ensures that the user only gets notified when necessary.

7. Why the 20% Threshold for Direction?

The 20% threshold for direction is used to prevent the system from triggering unnecessary direction announcements when the object is near the center of the frame. It is a way of giving more accurate guidance:

- If the object is near the center (within the middle 60% of the frame), the system will not suggest moving left or right unless the object is very close.

- This helps reduce false alarms and ensures that the user is only told to move when necessary.

Results and Discussion

Since we were dealing with the classification problem, the evaluation metrics used to assess the model's performance in this project include Precision, Recall, F1 Score, and the Precision-Recall (PR) Curve. These metrics are crucial for determining the model's accuracy and reliability in object detection.

1. **Precision Curve (P-Curve):**
    - Precision measures the model's accuracy in identifying true positives among all positive predictions.
    - **High Precision** indicates that the model's detections are mostly correct and relevant, with fewer false positives.
    - **Low Precision** suggests a tendency to detect objects that may not be present, leading to higher false positives.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

2. **Recall (R-Curve):**
    - Recall measures the model's ability to detect all relevant objects within an image, essentially identifying true positives.
    - **High Recall** signifies that the model detects most or all objects, minimizing missed detections.
    - **Low Recall** implies the presence of false negatives, where relevant objects go undetected.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

3. **F1 Score (F1 Curve):**
    - The F1 Score is the harmonic mean of Precision and Recall, balancing both metrics, especially when both are equally important.
    - A **high F1 Score** reflects strong performance in correctly identifying objects that are truly present, indicating a balanced model.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

4. **Precision-Recall Curve (PR Curve):**
    - The PR Curve visualizes the trade-off between Precision and Recall, helping in determining the ideal balance for accurate object detection.

- Generally, a model with a higher **Area Under the PR Curve (AUC-PR)** suggests better performance in distinguishing true objects from false ones.

5. **Mean Average Precision at IoU 0.5 (mAP@0.5)**

Mean Average Precision (mAP) is an important metric in object detection that evaluates the accuracy of a model's predictions by measuring how well it detects and locates objects. It aggregates the precision of the model over various recall levels to provide a single score, which represents the model's detection quality.

The **mAP@0.5** metric specifically evaluates the model at an Intersection over Union (IoU) threshold of 0.5:

- **Intersection over Union (IoU):**
  - IoU measures the overlap between the predicted bounding box and the ground truth bounding box for each detected object.
  - It is calculated as

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

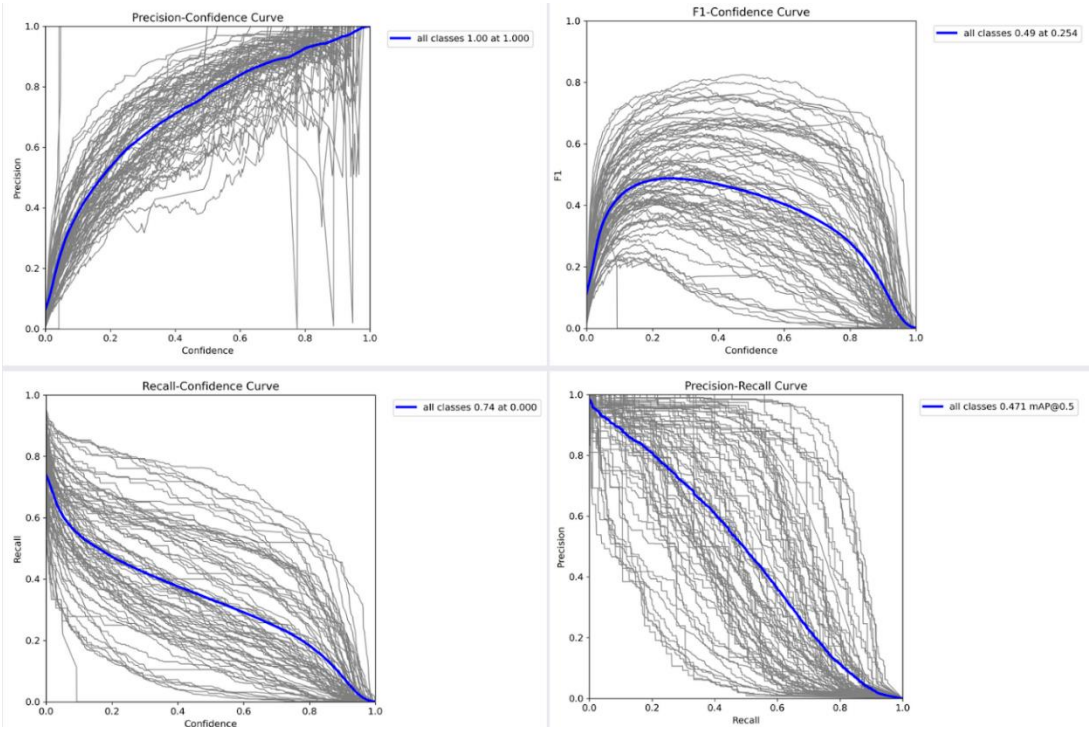
  - A threshold of 0.5 means that a prediction is considered correct if the IoU between the predicted and ground truth boxes is **at least 50%**. In simpler terms, the predicted box should overlap with at least 50% of the actual object's area.
- **Average Precision (AP):**
  - AP represents the precision averaged over recall values for a single class.
  - It is calculated by plotting the precision-recall curve for each class and taking the area under this curve.
  - Higher AP values indicate better performance, as it means the model maintains high precision across a range of recall levels.
- **Mean Average Precision (mAP):**
  - mAP is the mean of AP values across all classes in the dataset.
  - In this project, **mAP@0.5** represents the mean of AP values for all classes when the IoU threshold is 0.5.
  - A higher mAP@0.5 score signifies that the model can accurately detect objects across multiple classes with at least 50% overlap with the ground truth, balancing both precision and recall.

Each metric provides a different perspective on the model's effectiveness, helping to refine and optimize its object detection capabilities.

The dataset contained around 2 lacs of images, but 5000 of them were kept as training data. Following are the evaluation metrics of all the 80 classes that were present in the dataset:

Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%| | 105/105 [00:50<00:0
all	5000	36335	0.603	0.44	0.471	0.33
aeroplane	97	143	0.735	0.713	0.775	0.601
apple	76	236	0.385	0.246	0.187	0.133
backpack	228	371	0.451	0.129	0.161	0.0778
banana	103	370	0.47	0.292	0.311	0.187
baseball bat	97	145	0.595	0.29	0.341	0.187
baseball glove	100	148	0.658	0.439	0.457	0.256
bear	49	71	0.747	0.761	0.776	0.623
bed	149	163	0.601	0.479	0.502	0.375
bench	235	411	0.525	0.238	0.268	0.177
bicycle	149	314	0.658	0.369	0.402	0.23
bird	125	427	0.544	0.361	0.394	0.257
boat	121	424	0.538	0.304	0.357	0.184
book	230	1129	0.445	0.105	0.17	0.08
bottle	379	1013	0.559	0.39	0.42	0.269
bowl	314	623	0.545	0.438	0.472	0.341
broccoli	71	312	0.432	0.365	0.325	0.184
bus	189	283	0.687	0.628	0.678	0.563
cake	124	310	0.501	0.367	0.404	0.263
car	535	1918	0.618	0.487	0.518	0.327
carrot	81	365	0.412	0.288	0.276	0.169
cat	184	202	0.723	0.788	0.765	0.571
cell phone	214	262	0.543	0.331	0.368	0.256
chair	580	1771	0.541	0.3	0.348	0.214
clock	204	267	0.681	0.596	0.625	0.424
cow	87	372	0.651	0.596	0.628	0.439
cup	390	895	0.531	0.372	0.418	0.289
diningtable	501	695	0.472	0.358	0.323	0.209
dog	177	218	0.628	0.624	0.666	0.534
donut	62	328	0.524	0.486	0.446	0.341
elephant	89	252	0.662	0.799	0.76	0.567
fire hydrant	86	101	0.836	0.658	0.713	0.563
fork	155	215	0.501	0.256	0.287	0.197
frisbee	84	115	0.735	0.652	0.734	0.553
giraffe	101	232	0.777	0.793	0.841	0.639
hair drier	9	11	1	0	0	0
handbag	292	540	0.518	0.0955	0.152	0.081
horse	128	272	0.721	0.608	0.664	0.493
hot dog	51	125	0.701	0.392	0.465	0.327
keyboard	106	153	0.617	0.582	0.645	0.477
kite	91	327	0.543	0.544	0.535	0.341
knife	181	325	0.504	0.11	0.135	0.0825
laptop	183	231	0.646	0.615	0.658	0.535
microwave	54	55	0.575	0.545	0.584	0.479
motorbike	159	367	0.648	0.54	0.587	0.36
mouse	88	106	0.744	0.651	0.705	0.524
orange	85	285	0.442	0.372	0.338	0.257
oven	115	143	0.614	0.445	0.479	0.319
parking meter	37	60	0.737	0.42	0.533	0.417
person	2693	10777	0.74	0.637	0.699	0.47
pizza	153	284	0.638	0.585	0.607	0.454
pottedplant	172	342	0.508	0.342	0.343	0.195
refrigerator	101	126	0.607	0.563	0.553	0.422
remote	145	283	0.432	0.177	0.232	0.127
sandwich	98	177	0.476	0.435	0.415	0.31
scissors	28	36	0.743	0.242	0.258	0.205
sheep	65	354	0.527	0.644	0.606	0.414
sink	187	225	0.56	0.436	0.446	0.284
skateboard	127	179	0.721	0.587	0.612	0.42
skis	120	241	0.508	0.27	0.293	0.146
snowboard	49	69	0.426	0.319	0.342	0.231
sofa	195	261	0.507	0.49	0.471	0.334
spoon	153	253	0.435	0.103	0.141	0.083
sports ball	169	260	0.651	0.385	0.44	0.302
stop sign	69	75	0.753	0.587	0.65	0.587
suitcase	105	299	0.482	0.438	0.433	0.289
surfboard	149	267	0.561	0.446	0.46	0.281
teddy bear	94	190	0.657	0.535	0.551	0.37
tennis racket	167	225	0.654	0.578	0.6	0.356
tie	145	252	0.549	0.313	0.348	0.216
toaster	8	9	1	0	0.416	0.272
toilet	149	179	0.643	0.642	0.658	0.544
toothbrush	34	57	0.347	0.14	0.149	0.0867
traffic light	191	634	0.61	0.335	0.387	0.194
train	157	190	0.78	0.726	0.753	0.56
truck	250	414	0.531	0.37	0.404	0.261
tvmonitor	207	288	0.71	0.635	0.685	0.517
umbrella	174	407	0.621	0.478	0.495	0.319
vase	137	274	0.548	0.455	0.421	0.29
wine glass	110	341	0.676	0.299	0.373	0.231
zebra	85	266	0.746	0.794	0.835	0.628

Evaluation Graphs:



1. **Precision-Confidence Curve**

This curve illustrates the relationship between Precision and Confidence across different classes.

- **Precision** measures how many of the detected objects are true positives.
- **Confidence** represents the model's confidence level in its predictions, ranging from 0 to 1.
- The **blue line** represents the overall performance for all classes, showing a general trend where higher confidence leads to higher precision.
- As confidence increases, precision improves, though there are variations across individual classes (represented by the gray lines).

2. **Recall-Confidence Curve**

This curve examines how Recall changes as the confidence threshold varies.

- **Recall** measures the model's ability to identify all true positives.
- As confidence increases, Recall tends to decrease because a higher confidence requirement results in more missed detections (false negatives).
- The **blue line** for all classes shows that a lower confidence threshold allows for higher recall, peaking around 0.74 at 0 confidence, with a gradual decrease as confidence rises.

3. **F1-Confidence Curve**

The F1-Confidence curve shows how the F1 Score varies with changes in the confidence threshold.

- **F1 Score** is the harmonic mean of Precision and Recall, balancing both metrics.

- The **peak of the blue line** indicates the optimal confidence threshold where the model achieves a good balance between Precision and Recall.
- A moderate confidence threshold around 0.25 provides the best F1 score (0.49 for all classes), beyond which the score declines as the model becomes either too lenient or too conservative.

4. **Precision-Recall Curve**

The Precision-Recall (PR) curve visualizes the trade-off between Precision and Recall at different thresholds

- The **blue line** indicates the mean Average Precision (mAP) for all classes at a set threshold of 0.5, which is around 0.471.
- This curve helps to identify the model's capability to maintain a balance between precision and recall over various confidence levels.
- A higher **area under the PR curve** generally suggests better model performance, as it indicates the model effectively distinguishes true positives from false positives while minimizing false negatives.

Limitations of YOLO:

1. **Difficulty with Small Object Detection:**

YOLO, while powerful for real-time detection, has difficulty detecting small objects within an image. This limitation arises because YOLO divides the image into a grid, and smaller objects might occupy only a tiny part of a grid cell, which can result in missed detections or reduced accuracy. For instance, if a model is trained to detect pedestrians and vehicles, it might struggle to detect smaller items like cell phones or keys in the background.

2. **Sensitivity to Occlusions:**

YOLO can have difficulty when objects in the scene are partially obstructed or occluded by other objects. Since YOLO treats each object during detection, it may struggle to accurately identify objects that are only partially visible. For instance, in crowded spaces where people or items are closely packed, YOLO's ability to recognize occluded objects decreases, potentially impacting the reliability of the detection results.

3. **Trade-Off Between Speed and Accuracy:**

Although YOLO is optimized for speed, this often comes at the expense of accuracy. Faster versions of YOLO (e.g., YOLOv8n) sacrifice some level of precision to achieve higher frame rates, which can sometimes result in misclassifications or lower detection quality, especially for complex scenes with multiple overlapping objects.

4. **Lower Performance in Low-Light Conditions:**

YOLO's performance can significantly decline in low-light or poorly illuminated environments. This is because the model relies on visual cues and details that become harder to discern in dark or dim lighting. As a result, objects might be misclassified, or YOLO might fail to detect them entirely.

Future Enhancements:

1. **Improved Object Recognition**: Adding more classes or categories.

Currently, the system may be limited to detecting only a certain set of objects, such as people, vehicles, or basic household items. Expanding the model's object recognition capabilities by adding more classes or categories can significantly enhance its usefulness for visually impaired users.

2. **Multilingual Support for TTS**: Expanding gTTS support to additional languages.

Text-to-Speech (TTS) plays a crucial role in converting detected objects into audio descriptions, helping users interact with the system effectively. However, supporting a single language may limit the accessibility of this technology for a diverse range of users. Expanding gTTS or incorporating additional TTS libraries to support multiple languages would allow the system to cater to a broader demographic. This multilingual capability would not only make the application more inclusive for non-English-speaking users but also promote its use across different regions and communities.

3. **Enhanced Pathfinding**: Upgrading directional guidance to provide more nuanced navigation.

Currently, the system offers basic directional guidance by identifying obstacles on the left or right side of the user. Future upgrades could focus on enhancing this pathfinding ability to provide more detailed and adaptive navigation suggestions. For example, the system could use additional spatial analysis to calculate the optimal path around multiple obstacles or dynamically adjust the guidance based on the user's current speed and movement.

Summary and Conclusion

- **Enhanced Object Detection:** Successfully developed a model using YOLO.

The project successfully developed an object detection model using the YOLO (You Only Look Once) algorithm, which is known for its real-time detection capabilities. YOLOv8 was chosen for its balance between speed and accuracy, allowing the system to quickly identify objects in the user's environment with high precision. This functionality provides visually impaired users with essential information about the objects around them, enhancing their situational awareness and enabling them to make informed decisions in real-time.

- **Audio Feedback Integration:** Implemented audio feedback using Text-to-Speech (TTS) for accessible navigation.

To translate visual data into an accessible form, we implemented audio feedback using Text-to-Speech (TTS) technology. By converting detected objects into spoken words, the system informs users about their surroundings through auditory cues. This integration with libraries such as gTTS and pyttsx3 allows for clear, immediate audio output, making the technology especially helpful in guiding users through daily navigation tasks and helping them avoid obstacles.

- **Distance Approximation:** Integrated a distance estimation module that calculates the distance to detected objects.

To provide a more comprehensive understanding of the surrounding environment, a distance estimation module was integrated. This module calculates the distance between the user and detected objects, giving users not only the location of objects but also their relative proximity. This spatial awareness is critical for safe navigation, as it helps users assess which objects are immediately in their path and require prompt action.

- **Enhanced User Guidance:** Focused on nearest object detection, with directional guidance (left/right).

To further refine user experience and safety, the model is focused on identifying and highlighting the nearest object to the user. Additionally, it provides directional guidance, suggesting whether the user should move left or right based on obstacle positioning. This aspect adds a layer of precision, guiding users around immediate obstacles and facilitating smoother, safer movement in crowded or dynamic environments.

References

---

https://www.datacamp.com/blog/yolo-object-detection-explained

https://medium.com/@draj0718/convolutional-neural-networks-cnn-architectures-explained-716fb197b243

https://medium.com/@juanpedro.bc22/detailed-explanation-of-yolov8-architecture-part-1-6da9296b954e

Bibliography

Kadam, N., Singh, V., Singh, S., & Phalke, A. (n.d.). *OBJECT, COLOUR AND DISTANCE DETECTION SYSTEM FOR VISUALLY IMPAIRED PEOPLE*. www.irjmets.com

Najm, H., Elferjani, K., & Alariyibi, A. (n.d.). *XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE Assisting Blind People Using Object Detection with Vocal Feedback*.

S, B., & D, L. (2018). Image to Audio Conversion using Portable Camera. *Journal of Electrical & Electronic Systems*, *07*(03). https://doi.org/10.4172/2332-0796.1000268

Appendix

Drive link for code file and dataset:

RT3_Vision_to_Voice