**Kidney-Stone-Ultrasound Object Detection Using YOLOv8**

A YOLOv8-Based Deep Learning and Segmentation-Guided Approach

N CHARAN SAI (RA2311003011343)

V HARSHITH (RA2311003011347)

## **Method 1: Deep Learning Frameworks (YOLOv8 Training)**

### **Step 1: Setup Conda Environment & Install Dependencies**

conda create --name kidney-stone-detection python=3.9 -y

conda activate kidney-stone-detection

conda install -c conda-forge ultralytics opencv matplotlib numpy -y

### **Step 2: Prepare Dataset in YOLO Format**

Ensure the dataset is organized as:

datasets/

├── images/train/ # Training images

├── images/val/ # Validation images

├── labels/train/ # Training labels (YOLO format)

├── labels/val/ # Validation labels (YOLO format)

Create datasets/data.yaml with:

path: ./datasets

train: images/train

val: images/val

nc: 1

names: ['Kidney\_Stone']

### **Step 3: Train YOLOv8 Model**

python -c "

from ultralytics import YOLO

model = YOLO('yolov8s.pt') # Load pre-trained YOLOv8-small

model.train(data='datasets/data.yaml', epochs=50, imgsz=640, batch=16, device='cuda')

"

## 

## **Method 2: Segmentation-Guided Enhancement**

### **Step 1: Preprocess Ultrasound Images**

python -c "

import cv2

import numpy as np

import matplotlib.pyplot as plt

# Load grayscale ultrasound image

image = cv2.imread('kidney\_ultrasound.jpg', cv2.IMREAD\_GRAYSCALE)

# Enhance contrast using Histogram Equalization

equalized = cv2.equalizeHist(image)

# Apply Gaussian Blur for noise reduction

blurred = cv2.GaussianBlur(equalized, (5,5), 0)

# Save the preprocessed image

cv2.imwrite('preprocessed\_ultrasound.jpg', blurred)

# Show the processed image

plt.imshow(blurred, cmap='gray')

plt.title('Preprocessed Ultrasound Image')

plt.show()

"

### **Step 2: Apply Segmentation**

python -c "

# Load preprocessed image

image = cv2.imread('preprocessed\_ultrasound.jpg', cv2.IMREAD\_GRAYSCALE)

# Adaptive thresholding for segmentation

thresh = cv2.adaptiveThreshold(image, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY, 11, 2)

# Morphological operations to refine segmentation

kernel = np.ones((3,3), np.uint8)

morph = cv2.morphologyEx(thresh, cv2.MORPH\_CLOSE, kernel, iterations=2)

# Save and display segmented image

cv2.imwrite('segmented\_ultrasound.jpg', morph)

plt.imshow(morph, cmap='gray')

plt.title('Segmented Kidney Region')

plt.show()

"