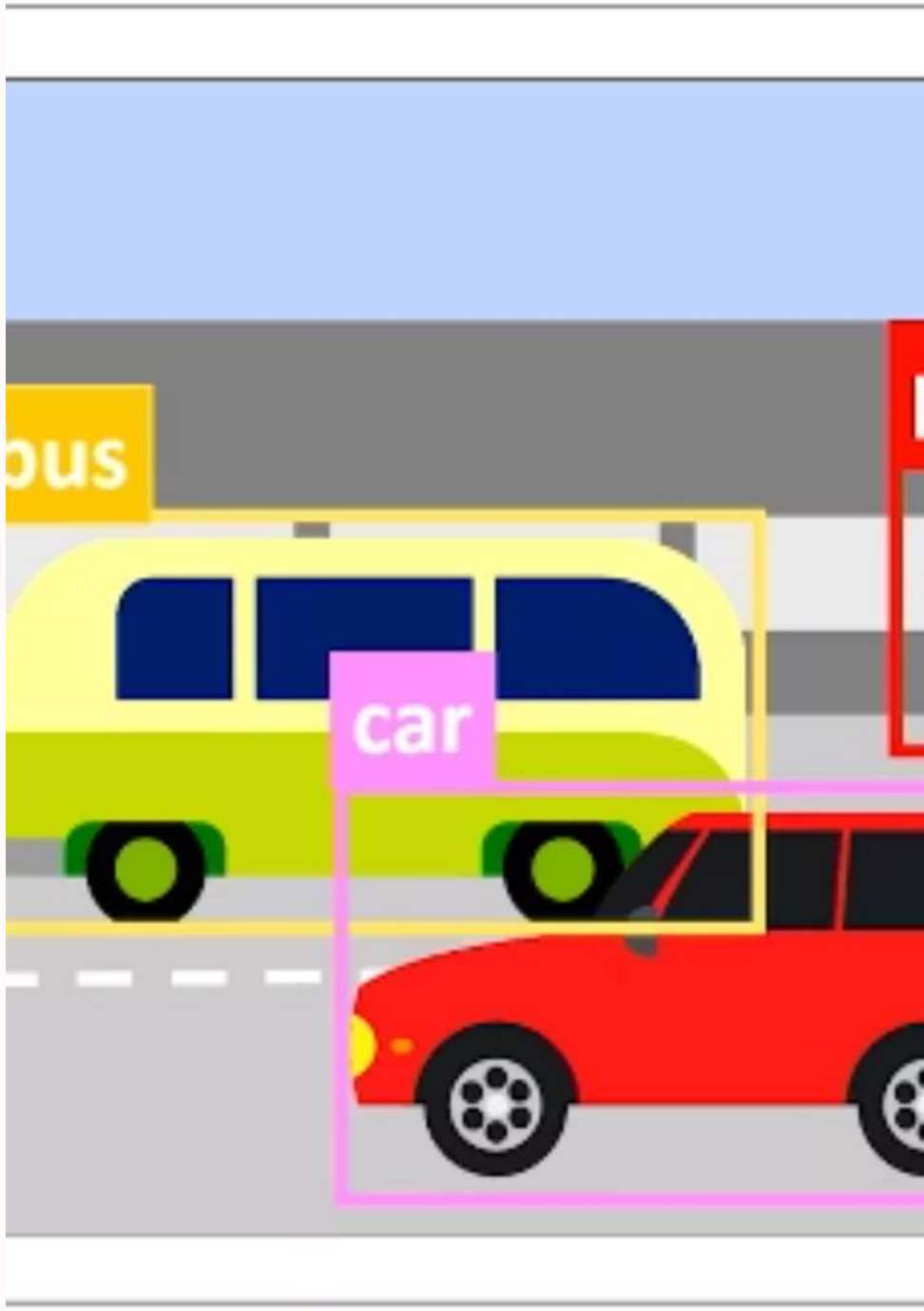
Object Detection in Images using YOLOv5

This mini project explores object detection using the YOLOv5 model.

Presented By- TechnoGeeks

Kiran Yadav (202410116100102)
Gulsan Kumari (202410116100079)
Dolly Prajapati(202410116100069)
Khushi Kumari (202410116100101)



What is Object Detection?

Key Computer Vision Task

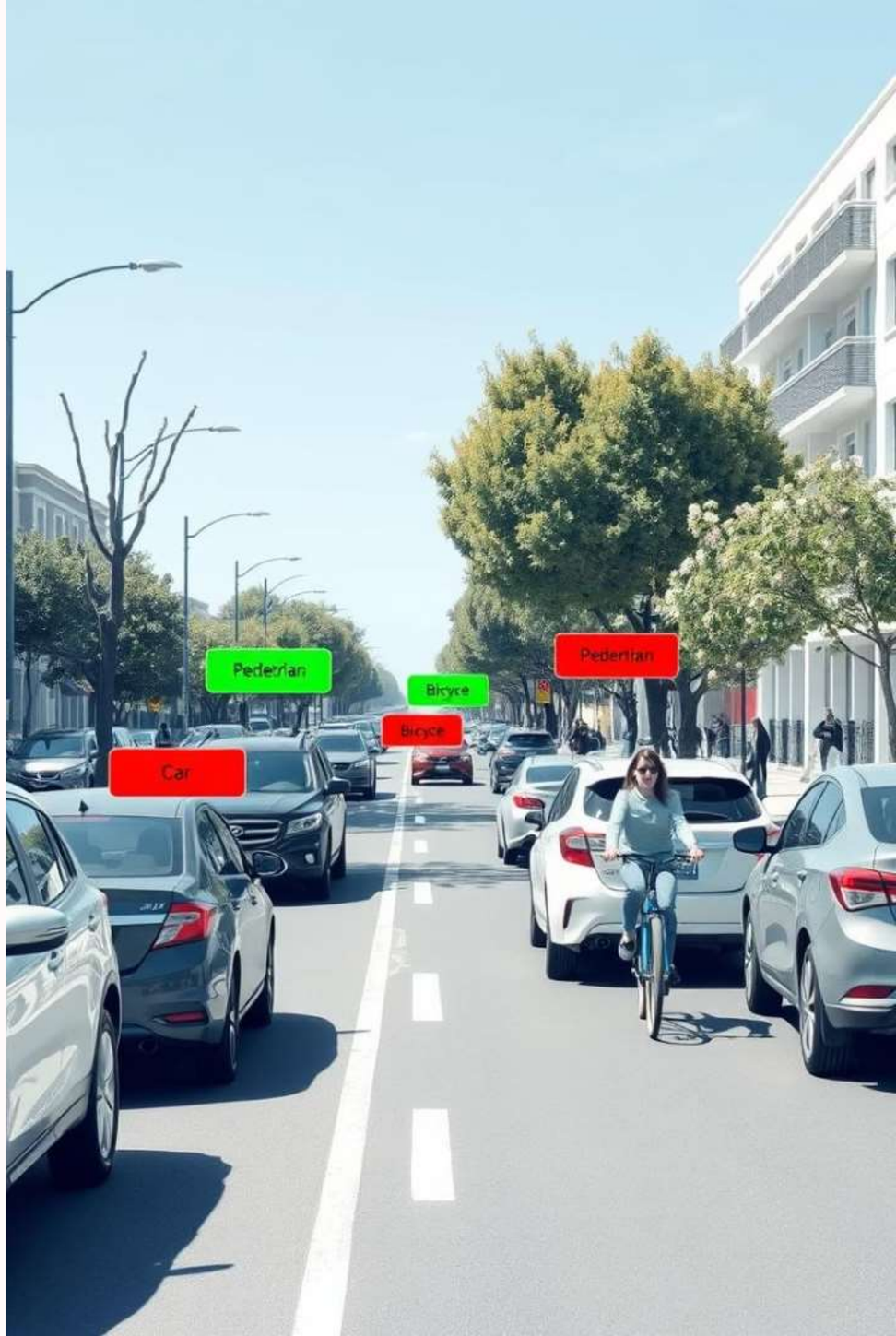
Detects and locates objects in images with bounding boxes.

Object Identification

Determines what objects are present in the scene.

Use Cases

- Self-driving cars
- Security systems
- Medical imaging



Technologies & Libraries

Google Colab

Cloud platform to run Python code seamlessly.

Automation Tools

YOLOv5 Toolkit (from Ultralytics) -Official GitHub codebase for the YOLOv5 model.

Python Libraries

- torch: Deep learning framework
- opencv: Image processing
- pathlib, os: File management
- IPython.display: Display outputs
- google.colab.files: Image uploads

Data set

COCO Dataset (Common Objects in Context)

What is YOLOv5?

Unified Detection Model

Detects objects in one forward pass for real-time speed.

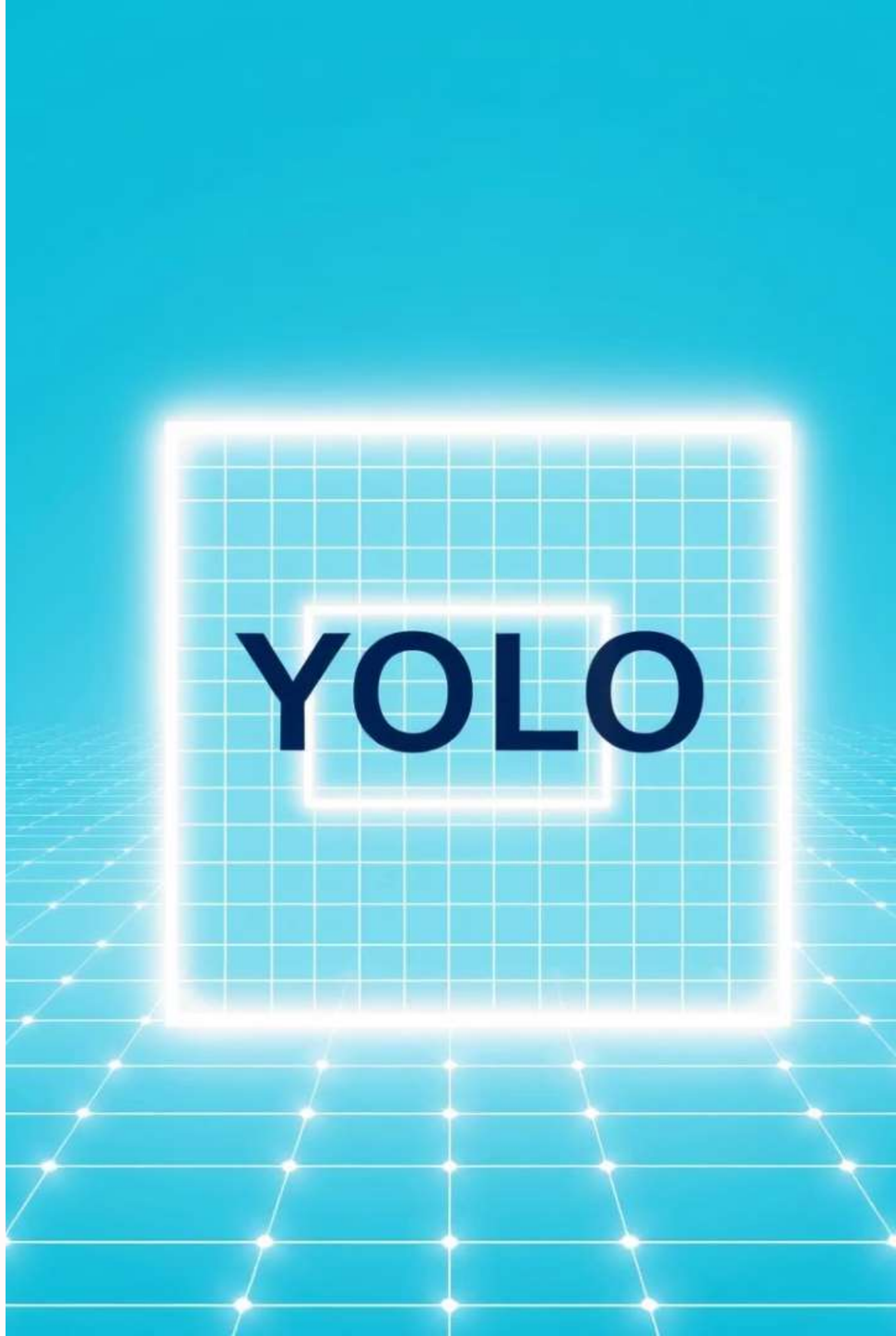
Lightweight & Fast

Efficient architecture making it easy to deploy.

Pretrained on COCO

Recognizes 80+ common objects from a large dataset.

- YOLOv5 breaks the image into grid cells
- Each grid cell is responsible for detecting objects whose center falls in that cell.



YOLOv5 Internal Workflow

1

Image Input

Resized to 640x640 pixels

2

Feature Extraction

Backbone extracts important features

3

Feature Enhancement

Neck refines multi-scale feature maps

4

Prediction Head

Outputs bounding boxes, classes, and confidence

5

Post-processing

NMS removes overlapping duplicates

6

Final Output

Objects detected and labeled on image

YOLOv5 Model Architecture

1

Backbone (CSPDarknet)

- Extracts **basic visual features** from the input image (like edges, shapes, textures).
- Think of it like **a visual scanner** that looks at the image and converts it into useful data

2

Neck (PANet)

- Helps the model understand **features at multiple scales** (small, medium, large objects).
- Combines **low-level details** (edges, colors) with **high-level concepts** (shapes, objects).

3

Head

Takes the output from the neck and **makes predictions**:

- **Bounding boxes**
- **Class labels**
- **Confidence scores**

4

Output

Final detections with labels

Project Workflow

1

Clone Repository

Get YOLOv5 code from GitHub

2

Install Dependencies

Set up required libraries and tools

3

Upload Image

Provide image for detection

4

Run Detection

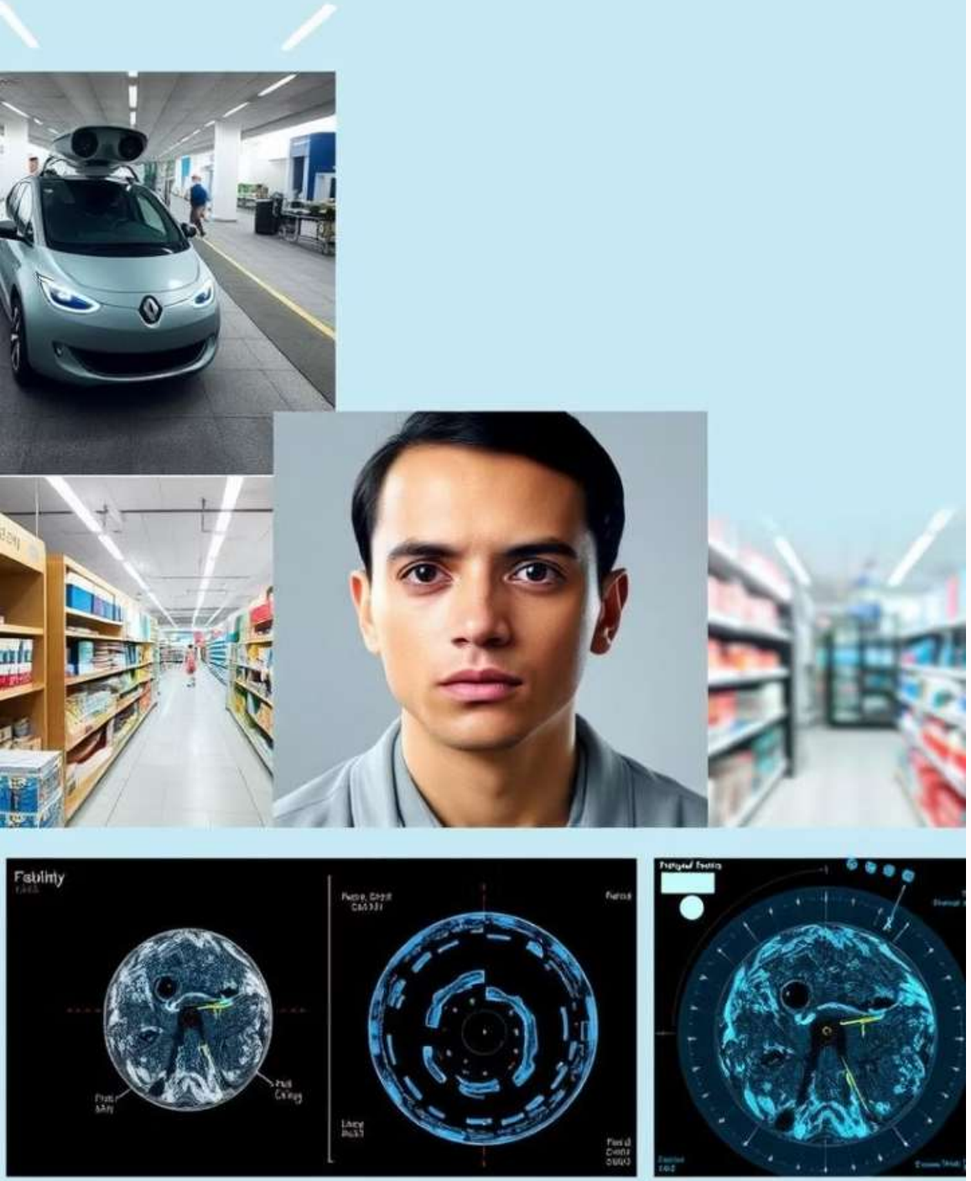
Execute detect.py with model weights

5

Display Output

View detected objects with bounding boxes

OUTPUT



Real-World Applications

Autonomous Vehicles

Pedestrian and obstacle detection for safety

Security Systems

Facial recognition and anomaly detection

Retail Analytics

Counting people and products in stores

Medical Imaging

Identifying conditions from scans automatically

THANK YOU