VisionAI: Object Detection & Segmentation

Introduction

VisionAI: Object Detection & Segmentation

- Computer Vision: A field of AI that enables computers to interpret and process visual information from the world.
- Segmentation: Dividing an image into meaningful parts (e.g., foreground vs background, objects vs environment).
- Object Detection: Identifying what objects are in an image and where they are located (bounding boxes).

Use Cases:

- - Self-driving cars → detect pedestrians and vehicles
- - Healthcare → detect tumors in MRI/CT scans
- - Security → detect intruders in CCTV footage
- - Retail → automatic product tagging

Project Goal

Problem Statement: Pure object detectors sometimes confuse visually similar objects (e.g., dog vs horse).

Goal: Build a hybrid AI system that:
- Accepts multiple uploaded images
- Detects and segments objects in each image
- Refines object recognition for accuracy
- Displays results in a user-friendly web interface

End Product: A Streamlit app that students, businesses, or researchers can use without writing code.

Dataset & Pretrained Models

COCO Dataset (Common Objects in Context):
- 200K+ images, 80 object categories (person, dog, car, airplane, etc.)
- Provides bounding boxes + segmentation masks
- Powers Mask R-CNN for detection + segmentation

ImageNet Dataset:
- 14M+ images, 1,000 categories (dog breeds, bird species, tools, etc.)
- Used to train ResNet50 classifier

Why Pretrained?
- Training from scratch = millions of images + weeks of compute
- Pretrained models capture general knowledge → we adapt it (transfer learning).

Challenges with Pure Detection

Mask R-CNN Issue:
- Example: Shows a “dog” but labels it as “horse” because COCO dataset has limited context.

Limitations of Detection-only models:
- Class granularity is low (e.g., “dog” but not “Labrador retriever”)
- May confuse similar animals (horse vs dog vs cow)
- False positives: detects something that doesn’t exist

Why Important?
- Misclassification in healthcare/autonomous driving could be dangerous.

Solution: Add refinement step using a stronger classifier.

Hybrid System Architecture

Steps in Pipeline:

1. Input Image → user uploads via Streamlit
2. Mask R-CNN → outputs bounding boxes + masks + initial labels
3. Cropped Regions → each detected object is extracted from the image
4. ResNet50 Classifier → re-analyzes each crop with ImageNet knowledge
5. Final Output → Annotated image + refined labels with confidence

📊 Analogy:

- Mask R-CNN = “Security guard” (spots suspicious regions)
- ResNet50 = “Expert detective” (examines closely and names correctly)

Technical Details

Frameworks:
- PyTorch & Torchvision: Deep learning & pretrained models
- OpenCV & PIL: Image processing
- Streamlit: Frontend for easy deployment

Features of System:
- Multi-file upload → process several images at once
- Confidence threshold slider → control how strict predictions are
- Export → download results individually or as ZIP

Hardware:
- Runs on CPU (slower)
- Faster on GPU (CUDA acceleration)

Deep Dive: Mask R-CNN

Mask R-CNN = Faster R-CNN + Segmentation Head

Key Components:
- Backbone (ResNet+FPN): Extracts deep image features
- RPN (Region Proposal Network): Suggests candidate object regions
- ROI Align: Fixes misalignment in feature extraction

Heads:
- Classification: Which object (dog, person, car)
- Bounding Box Regression: Where it is located
- Segmentation Mask: Pixel-wise object boundary

Strength: Detects what + where + shape

Deep Dive: ResNet50

Residual Neural Network (ResNet):
- Introduced skip connections → solves vanishing gradient problem
- Enables very deep networks (50+ layers)

ResNet50:
- Pretrained on ImageNet (1,000 classes)
- Great at fine-grained classification (e.g., Labrador retriever vs bulldog vs golden retriever)

Why we use it?
- Mask R-CNN good at localizing
- ResNet good at distinguishing
- Together = accurate + robust

Results

Case Study 1: Dog Image
- Mask R-CNN → “horse (0.99)” ✕
- ResNet50 → “Labrador retriever (0.96)” ✅

Case Study 2: Car + Person
- Mask R-CNN detects “person” + “car”
- ResNet confirms categories
- Output: Correctly labeled + annotated

Performance:
- Hybrid pipeline = fewer misclassifications
- Higher accuracy on confusing categories

Applications

Healthcare: Tumor segmentation, X-ray interpretation
Autonomous Vehicles: Pedestrian + vehicle detection
E-commerce: Automated product tagging, visual search
Agriculture: Crop/weed detection, livestock monitoring
Wildlife Monitoring: Species identification in field images
Education: Teaching deep learning & computer vision concepts

Deployment

Streamlit App Features:
- Easy drag-and-drop interface
- Upload multiple files at once
- Shows original + segmented images
- Confidence score for each detection
- Download annotated images / bulk results as ZIP

Why Streamlit?
- Minimal code for web deployment
- Interactive & user-friendly
- Perfect for classroom demos

Conclusion & Future Work

Achievements:

- Combined detection + classification → more accurate predictions
- Fixed misclassification problem
- Delivered interactive app with downloads

Future Enhancements:

- Real-time video input
- Fine-tuning on domain-specific datasets (medical, traffic, retail)
- Mobile-friendly models (e.g., MobileNet-SSD + lightweight ResNet)
- Adding explainability (heatmaps for why model predicted “dog”)