Computer Vision Project Presentation

Title: SignTrack: Traffic Sign Detection using YOLO

INDRAPRASTHA INSTITUTE *of* INFORMATION TECHNOLOGY DELHI

Group Members:
Aarya Gupta
Abdullah Shujat
Sargun Singh Khurana

Problem Statement and Scope

Problem Statement :

Develop a real-time Traffic Sign Detection system using YOLOv8 to enhance road safety and support autonomous navigation. The system aims to accurately detect and recognize traffic signs in images and videos using deep learning.

Scope:

Input: Traffic sign images & video streams from dashcams, surveillance cameras, or vehicle sensors.

Dataset: 4,969 samples, split into Train, Validation, and Test sets.

Output: Annotated images/video frames with bounding boxes, labels, and confidence scores.

Target Users:

Autonomous Vehicles & ADAS Systems – For real-time sign recognition, ensuring compliance with traffic rules.

Traffic Monitoring Authorities – To track violations using CCTV & IoT-based monitoring.

Urban Planners & Smart City Developers – To analyze traffic compliance and improve infrastructure planning.

Related Work and Baseline Methods

Baseline Models for Comparison

To evaluate **YOLO v8**, we compare against:

- **SVM + HOG:** Traditional machine learning benchmark for traffic sign detection.
- **CNN-based detection:** Standard deep learning approach widely used in image recognition.

Identified Gaps in Existing Approaches

- **SVM + HOG:** Limited generalization under varying lighting conditions and occlusions.
- **CNN-based Models:** High accuracy but computationally expensive, making them less suitable for real-time applications.
- **Hybrid GIS Models:** Require extensive **geo-tagged data**, which limits scalability and adaptability to new environments.

Datasets and Evaluation Metrics

Dataset Used: Traffic Sign Detection Dataset:

- Contains images of traffic signs and surrounding environments. Used for initial model training and validation of YOLOv8. Helps assess preliminary detection performance.

Evaluation Metrics:

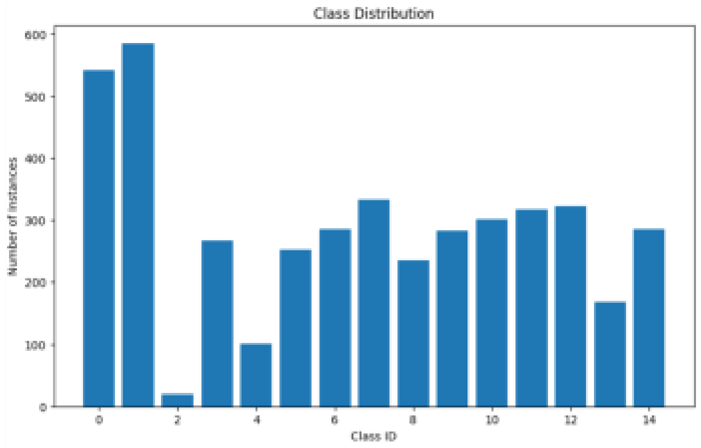
Precision: 94.2% overall

- **High:** Stop Signs (**97.3%**), Speed Limit Signs (**92-100%**)
- **Lower:** Red Light (**88.1%**), Green Light (**82.6%**)

Recall: 90.6% overall

- **High:** Stop Signs (**98.8%**), Speed Limits (**94.1-99.1%**)
- **Lower:** Red Light (**69.4%**), Green Light (**74.1%**)

- **mAP@0.5: 95.7%** – Strong detection performance
- **mAP@0.5-0.95: 82.97%** – Highlights challenges in precise localization



Name of Classes: Green Light, Red Light, Speed Limit 10, Speed Limit 100, Speed Limit 110, Speed Limit 120, Speed Limit 20, Speed Limit 30, Speed Limit 40, Speed Limit 50, Speed Limit 60, Speed Limit 70, Speed Limit 80, Speed Limit 90, Stop

Web Application Architecture

A client-server architecture was adopted for the web application:

- **Backend**: A RESTful API was developed using the Flask framework in Python. Its primary responsibilities are to handle incoming prediction requests, process images using the YOLOv8 model, and serve the frontend application

- **Frontend:** A dynamic and interactive user interface was built using the React JavaScript library. It allows users to interact with the system by uploading images and viewing the detection results.

- **Communication:** The frontend communicates with the backend via HTTP requests. Image data is typically sent to the backend API (e.g., as a base64 encoded string within a JSON payload), and the backend responds with the processed image and detection details, also usually in JSON format.

Backend API Implementation (Flask)

The Flask backend (app.py) serves as the core logic hub:

• **Model Loading:** The fine-tuned YOLOv8 model (weights/best.pt) is loaded into memory upon application startup for efficient inference.

• **Prediction Endpoint:** A primary API endpoint (e.g., /predict) was implemented to:

1. Receive image data uploaded from the frontend.
2. Decode the image data (e.g., from base64) into a format compatible with OpenCV and the model.
3. Perform traffic sign detection inference using the loaded ultralytics YOLOv8 model.
4. Process the model’s output (bounding boxes, class labels, confidence scores).
5. Utilize OpenCV (cv2) to draw the bounding boxes and labels onto the input image.
6. Encode the annotated image (e.g., back to base64) and structure the results (annotated image, detected classes) into a JSON response.
7. Send the JSON response back to the frontend.

• **Frontend Serving:** The backend is also configured to serve the **static built files** (HTML, CSS, JavaScript) of the React frontend application, typically from a designated folder (e.g., a static or build folder).

Frontend User Interface (React)

The frontend, located in the **src/public**, provides the user interaction layer:

• **User Interaction:** It presents an interface (developed using React components) allowing users to easily select or drag-and-drop an image file for analysis.

• **API Communication:** Upon user action (e.g., clicking a ”Detect” button), the frontend uses JavaScript fetch API or libraries like axios to send the selected image data to the backend’s /predict endpoint.

• **Result Display:** It receives the JSON response from the backend and dynamically updates the UI to display the returned annotated image, clearly showing the detected traffic signs and their bounding boxes.

• **Build Process:** The React code is built into optimized static assets (HTML, CSS, JS) using npm run build. These static files are then served by the Flask backend.

Planned Enhancements

With the core application structure in place, future work will focus on:

• **Model Refinement:** Continue optimizing YOLOv8 hyperparameters and potentially fine-tuning the architecture to improve accuracy, particularly for traffic lights and precise bounding box localization.

• **Data Strategy:** Actively expand the training dataset with more diverse scenarios (various lighting, weather, partial occlusions) focusing on under-represented classes. Implement advanced data augmentation techniques.

• **Real-time Video Processing:** Enhance the web application to support real-time detection from video streams (e.g., webcam or uploaded video files), improving UI responsiveness for continuous feedback.

Conclusion: The integrated system provides a solid foundation. Future efforts will prioritize enhancing the core model's accuracy for difficult cases, expanding data diversity, enabling real-time video analysis, and validating performance in realistic scenarios.

Website Demo Work

Contributions

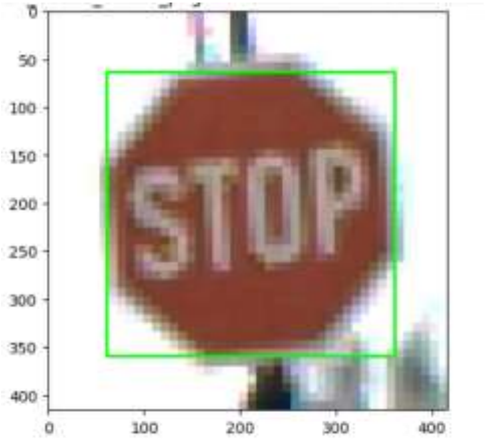
- **Aarya Gupta**: Model Training, Performance Evaluation & Analysis, Exploratory Data Analysis (EDA), Backend API Development (Flask)

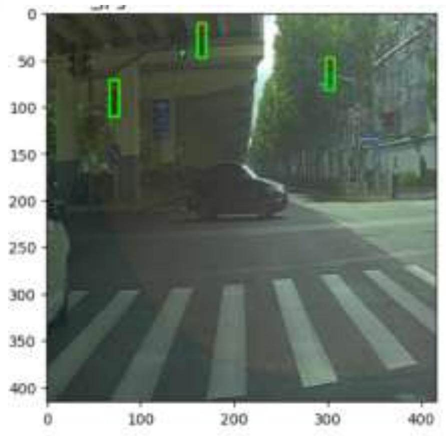
- **Abdullah Shujat**: Frontend UI Development (React), Dataset Collection & Sourcing, Data Preprocessing & Augmentation, Literature Review, Presentation Preparation

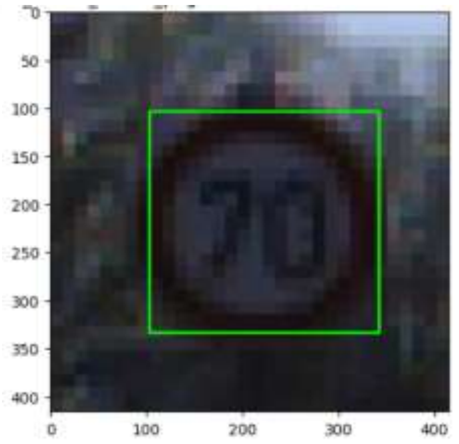
- **Sargun Singh Khurana**: Research and Testing, System Integration (FE/BE) UI Testing, Full System Debugging, Report Writing, Presentation Preparation.

BACKUP SLIDES

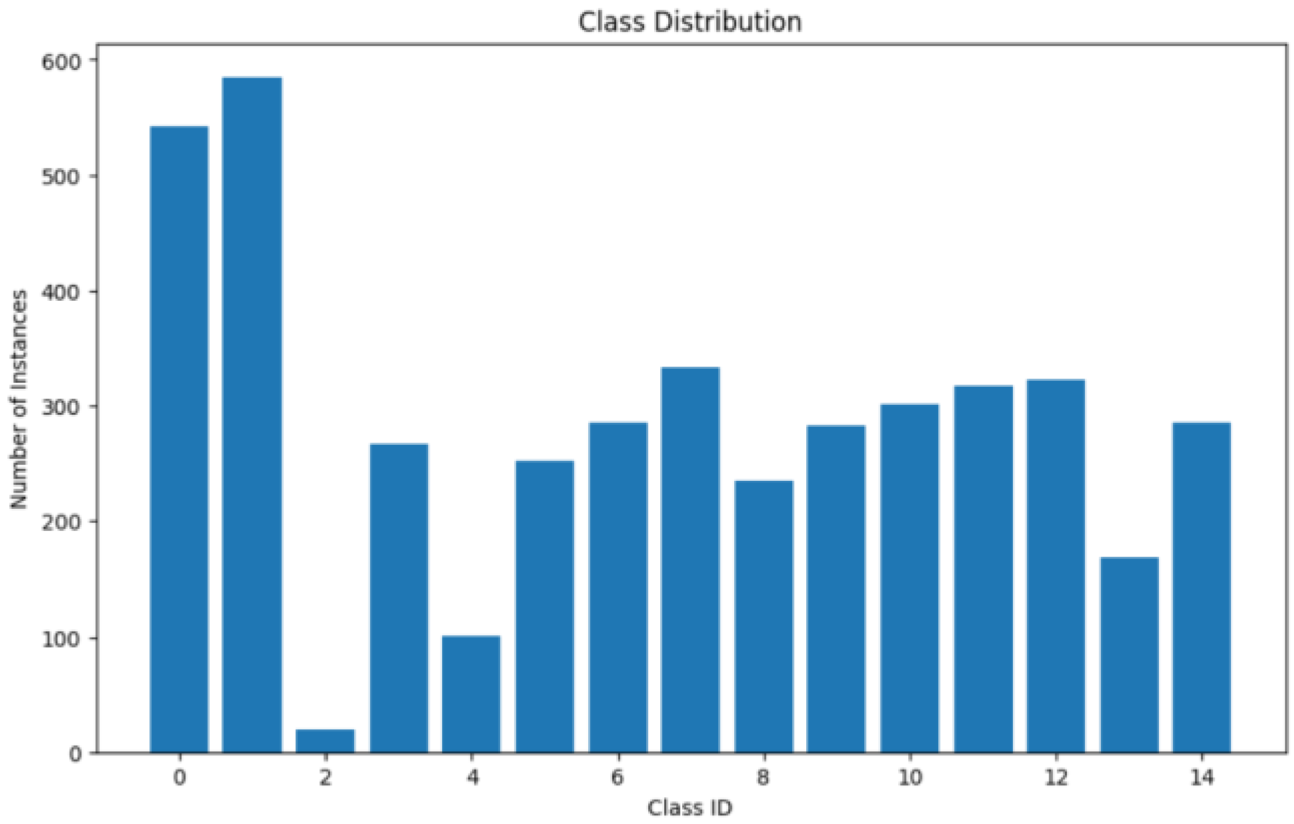
Sample Images



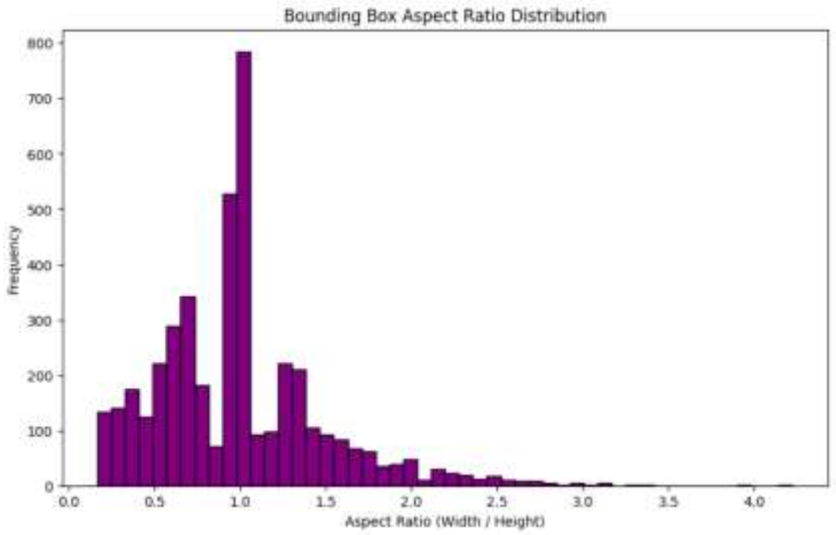


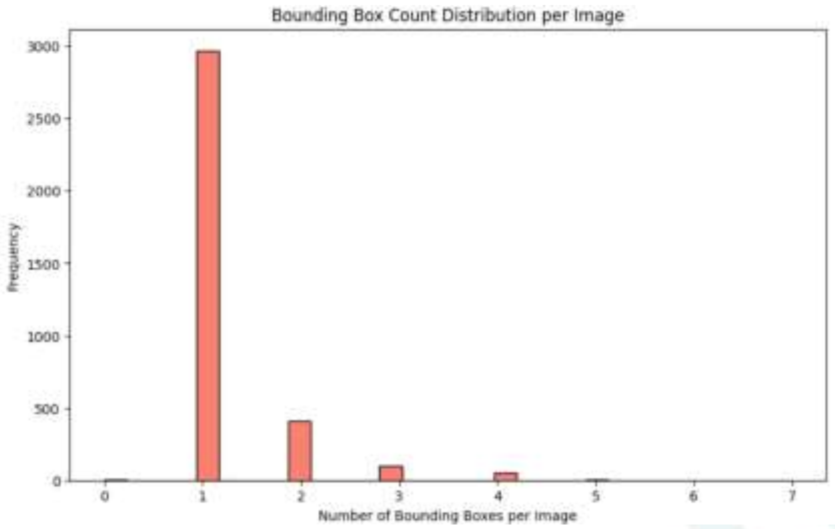


EDA - Datasets

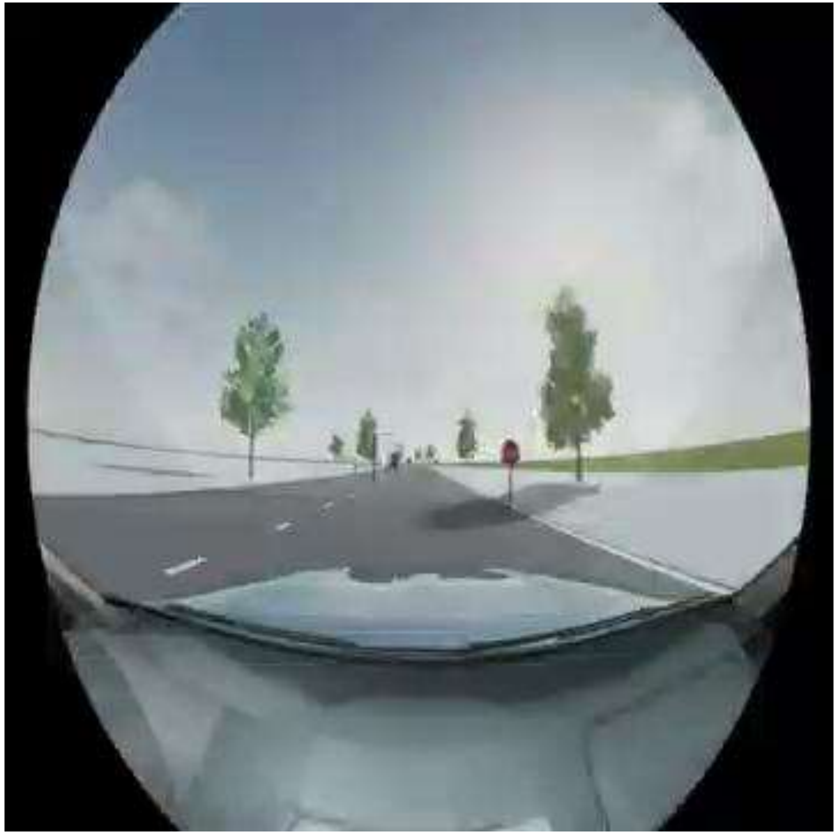


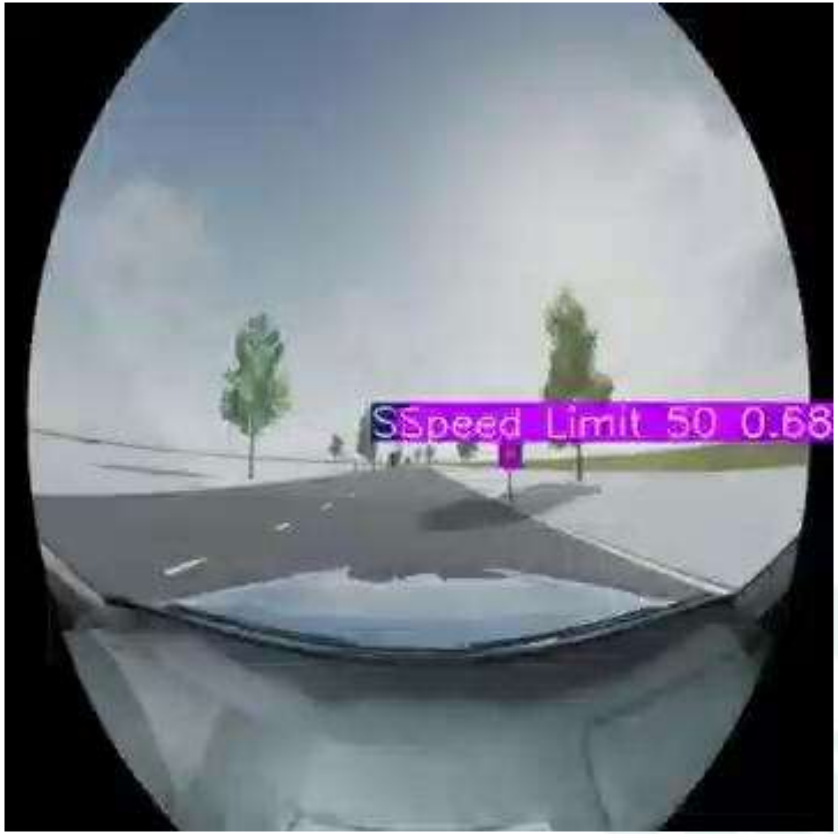
EDA - Datasets



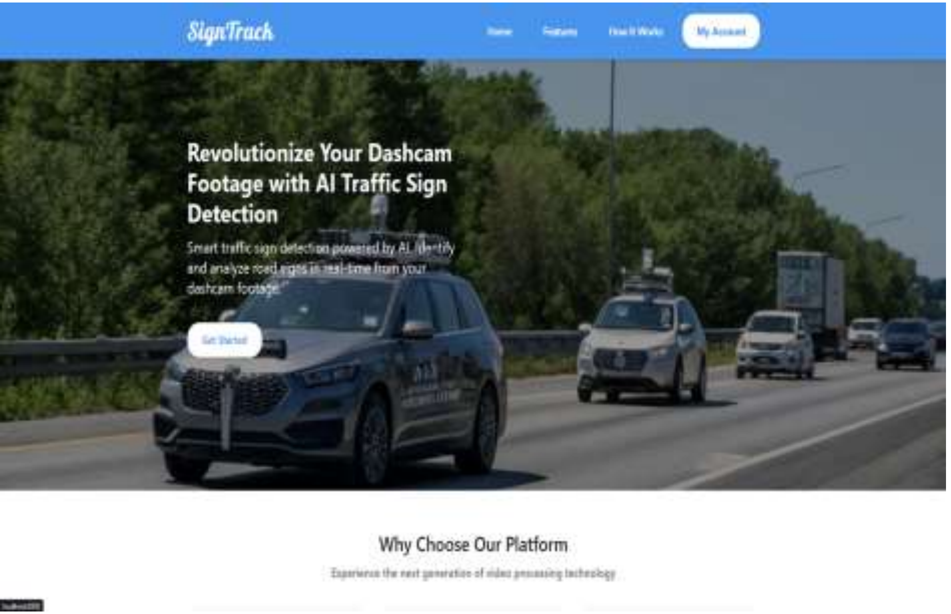


Input Test Video and Patched Result Video





Website Screenshots



SignTrack

Home · Features · How It Works · My Account

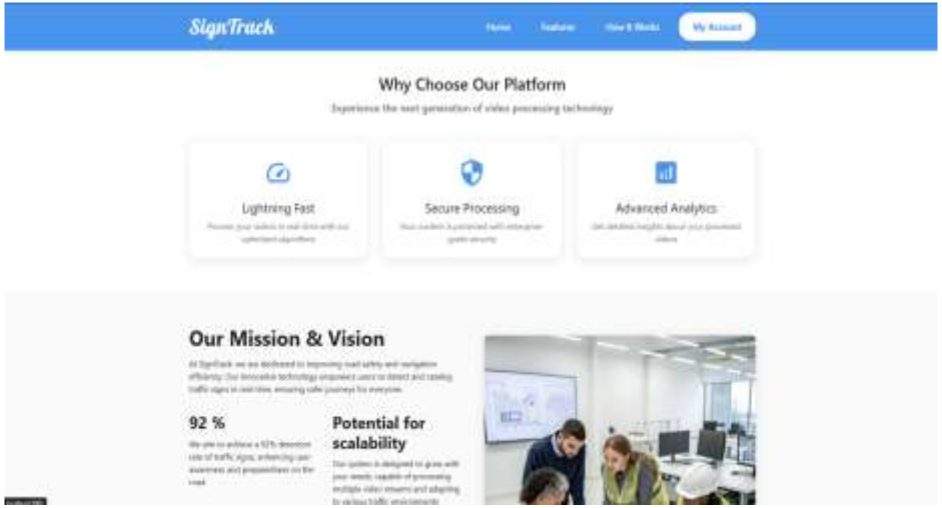
Revolutionize Your Dashcam Footage with AI Traffic Sign Detection

Smart traffic sign detection powered by AI. Identify and analyze road signs in real-time from your dashcam footage."

Get Started

Why Choose Our Platform

Experience the next generation of video processing technology



SignTrack

Home · Features · How It Works · My Account

Why Choose Our Platform

Experience the next generation of video processing technology

Lightning Fast

Process your videos in real-time with our optimized algorithms

Secure Processing

Your content is protected with enterprise-grade security

Advanced Analytics

Get detailed insights about your processed videos

Our Mission & Vision

At SignTrack we are dedicated to improving road safety and navigation efficiency. Our innovative technology empowers users to detect and catalog traffic signs in real-time, ensuring safer journeys for everyone.

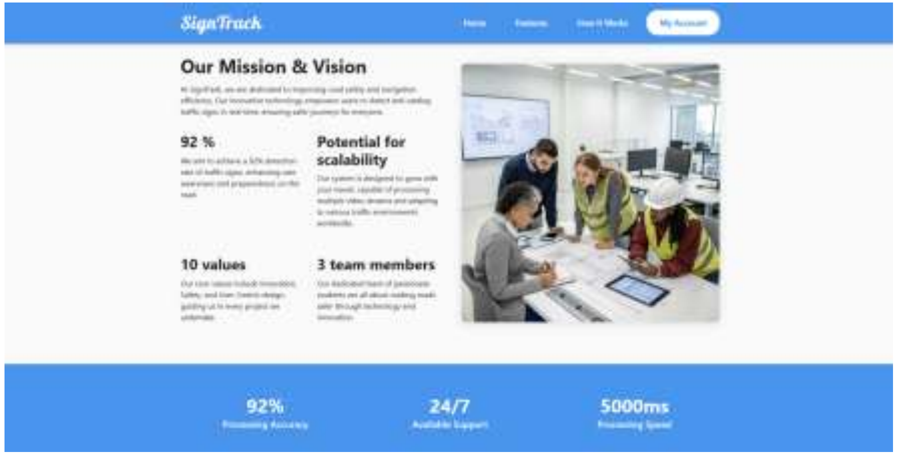
92 %

We aim to achieve a 92% detection rate of traffic signs, enhancing user awareness and preparedness on the road.

Potential for scalability

Our system is designed to grow with your needs, capable of processing multiple video streams and adapting to various traffic environments

Website Screenshots



Our Mission & Vision

At SignTrack, we are dedicated to improving road safety and navigation efficiency. Our innovative technology empowers users to detect and catalog traffic signs in real-time, ensuring safe journeys for everyone.

92 %

We aim to achieve a 92% detection rate of traffic signs, enhancing user awareness and preparedness on the road.

Potential for scalability

Our system is designed to grow with your needs, capable of processing multiple video streams and adapting to various traffic environments worldwide.

10 values

Our core values include innovation, Safety, and User-Centric design, guiding us in every project we undertake.

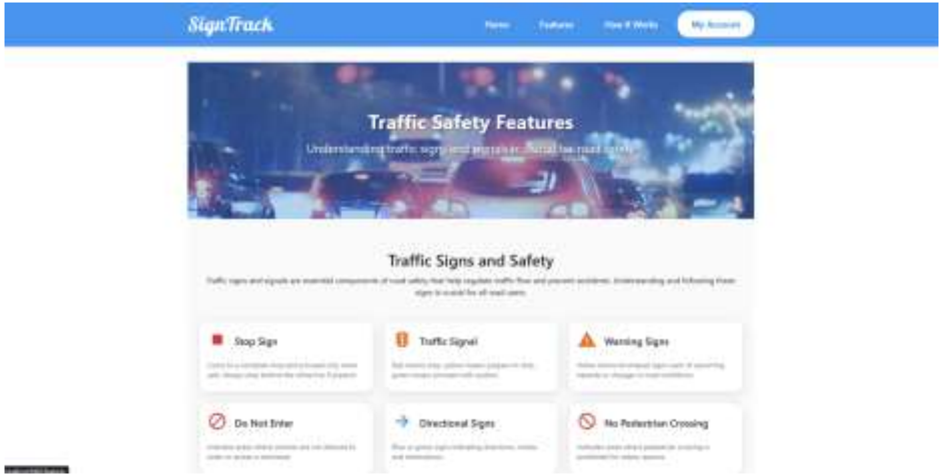
3 team members

Our dedicated team of passionate students are all about making roads safer through technology and innovation.

92% Processing Accuracy

24/7 Available Support

5000ms Processing Speed



Traffic Safety Features

Understanding traffic signs and signals is crucial for road safety

Traffic Signs and Safety

Traffic signs and signals are essential components of road safety that help regulate traffic flow and prevent accidents. Understanding and following these signs is crucial for all road users.

Stop Sign

Come to a complete stop and proceed only when safe. Always stop behind the white line if present.

Traffic Signal

Red means stop, yellow means prepare to stop, green means proceed with caution.

Warning Signs

Yellow diamond-shaped signs warn of upcoming hazards or changes in road conditions.

Do Not Enter

Indicates areas where vehicles are not allowed to enter or access is restricted.

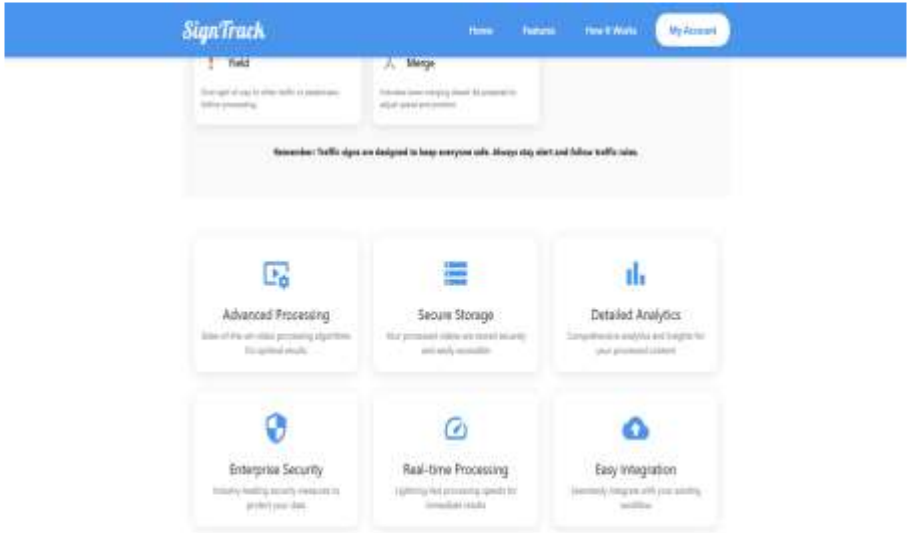
Directional Signs

Blue or green signs indicating directions, routes and destinations.

No Pedestrian Crossing

Indicates areas where pedestrian crossing is prohibited for safety reasons.

Website Screenshots



Yield

Give right of way to other traffic or pedestrians before proceeding.

Merge

Indicates lanes merging ahead. Be prepared to adjust speed and position.

Remember: Traffic signs are designed to keep everyone safe. Always stay alert and follow traffic rules.

Advanced Processing

State-of-the-art video processing algorithms for optimal results.

Secure Storage

Your processed videos are stored securely and easily accessible.

Detailed Analytics

Comprehensive analytics and insights for your processed content.

Enterprise Security

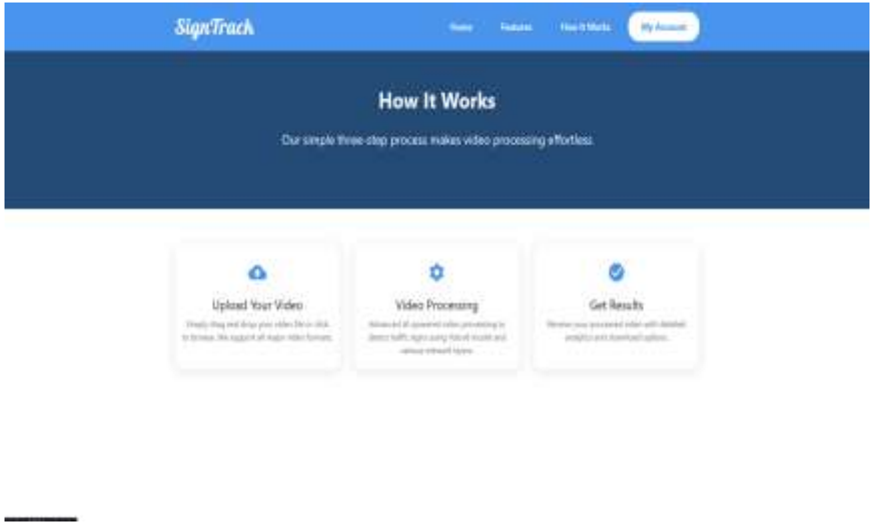
Industry-leading security measures to protect your data.

Real-time Processing

Lightning-fast processing speeds for immediate results.

Easy Integration

Seamlessly integrate with your existing workflow.



How It Works

Our simple three-step process makes video processing effortless.

Upload Your Video

Simply drag and drop your video file or click to browse. We support all major video formats.

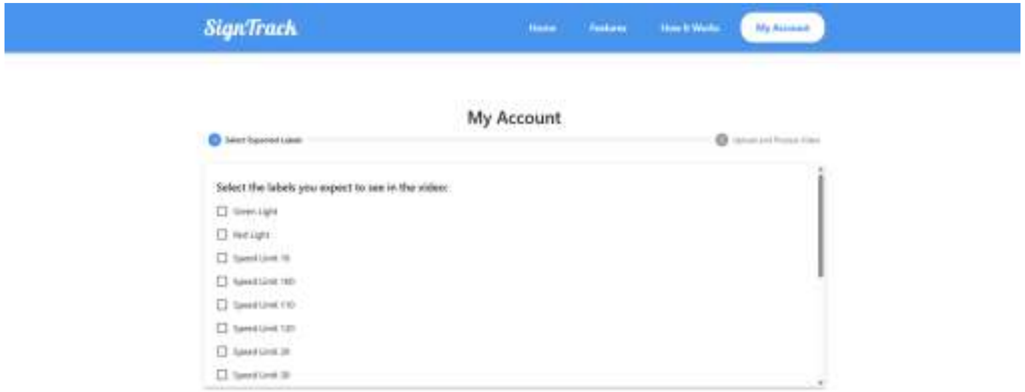
Video Processing

Advanced AI-powered video processing to detect traffic signs using [illegible] models and various network layers.

Get Results

Review your processed video with detailed analytics and download options.

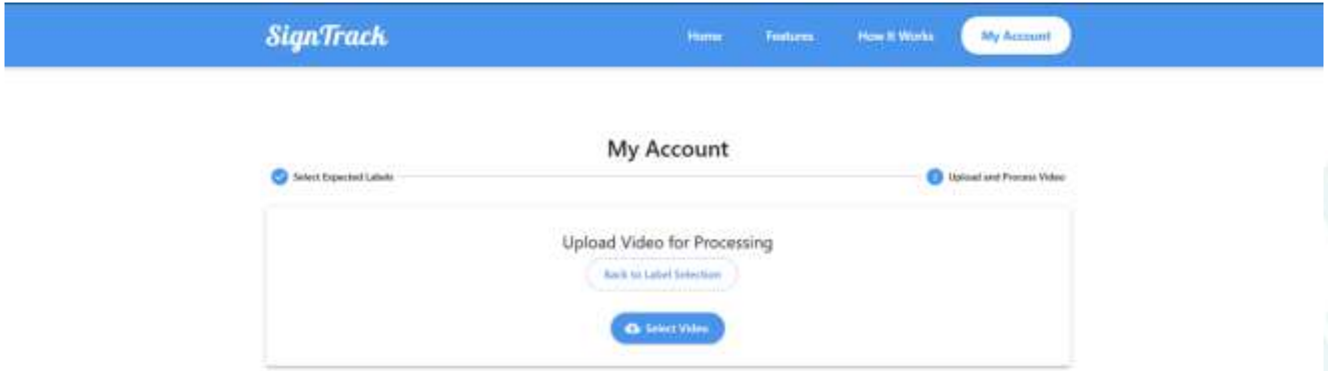
Website Screenshots



My Account

Select the labels you expect to see in the video:

- ☐ Green Light
- ☐ Red Light
- ☐ Speed Limit 10
- ☐ Speed Limit 100
- ☐ Speed Limit 110
- ☐ Speed Limit 120
- ☐ Speed Limit 20
- ☐ Speed Limit 30



My Account

Upload Video for Processing