**PROJECT REPORT :**

**Traffic Sign Recognition and Detection with YOLO**

**ANOUK Zakariae** **Supervised by P. KHAMJANE Aziz**

[anouk.zakariae@etu.uae.ac.ma](mailto:anouk.zakariae@etu.uae.ac.ma) [akhamjane@uae.ac.ma](mailto:anouk.zakariae@etu.uae.ac.ma)

Department of Mathematics & Computer Science, Abdelmalek Essaadi Univ. ENSAH College, Alhoceima, Morocco.

**Abstract**

This project advances traffic sign classification and localization using deep learning, specifically YOLO models. Using the GTSRB (German Traffic Sign Recognition Benchmark) dataset, we developed a dual-model system: one for detection and another for classification. The system achieves high accuracy in both tasks, with the detection model reaching 95.8% precision and the recognition model achieving 92.2% precision. Implementation includes a web-based interface for real-time predictions, demonstrating practical applicability for intelligent vehicle systems.

# Introduction

## 1.1. Background and Motivation



In the context of increasing use of ADAS and driverless cars, the vision systems must integrate traffic sign recognition and detection for safe mobility. These devices need to recognize and understand quite a number of symbols while in motion and sometimes those are difficult such as nighttime, dark areas, or when some perceptions are hidden.

The goal of this project is to ensure that the use of a detection and recognition system of the traffic signs is efficient and dependable. In this manner, it is our goal to also begin to solve a number of issues such as:

* **Traffic sign diversity** They have different shapes, sizes, colors, symbols, and even lettering.
* **Environmental conditions**: Lighting, raining and snowing or even other things in sight.
* **Sign Limitations:** the system needs to be responsive to the situation and hence shall be designed to answer the call swiftly.

## 1.2. Problem Statement and Challenges

• What considerations and procedures should be followed when creating a model which will be able to recognize patterns under different surroundings?

• What measures can be taken to rationalize the performance and expected accuracy in embedded operating systems?

• What is the best way to validate across different geographical datasets?

# Literature Review

## Traditional Methods

* Feature descriptors (HOG) combined with SVM classifiers
* Limited effectiveness with complex variations
* Challenges with environmental conditions

## Modern Approaches

* Convolutional Neural Networks (CNNs)
* YOLO models for object detection
* Improved speed and accuracy over traditional methods

# Methodology

## Dataset Download and Structure :

The GTSRB dataset was used, containing over 50,000 images distributed across 43 classes.

It was organized into the following directory structure to facilitate training and evaluation:

GTSRB/

├── train/

│ ├── images/

│ ├── labels/

├── validation/

│ ├── images/

│ ├── labels/

├── test/

│ ├── images/

│ ├── labels/

├── data.yaml

## Data Organization:

 Images were split into train, validation, and test sets.

 Each set was further divided into two directories:

* **images/**: Stores raw images.
* **labels/**: Contains label files with bounding box coordinates and class annotations in YOLO format.

## Data Augmentation :

To enhance model robustness, additional transformations were applied to the training set, such as:

* Rotation
* Brightness adjustment
* Gaussian blur

## Label Creation and Configuration Files:

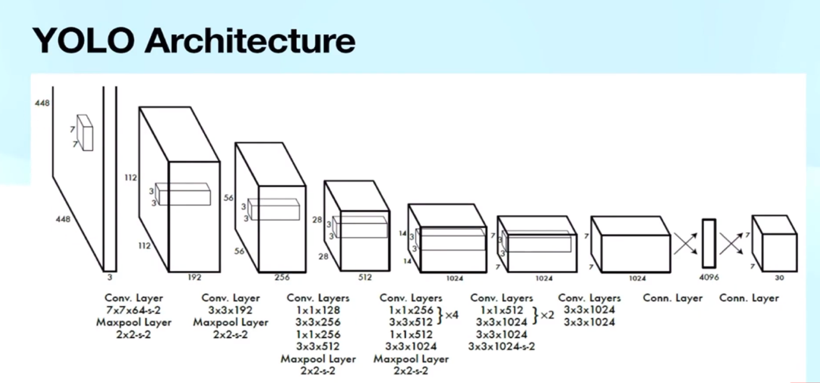
A data.yaml file was generated to specify dataset paths and the number of classes.

## Models and Techniques Used

Two instances of the YOLO model were used:

* **YOLO for detection**: Locates traffic signs by producing bounding boxes.
* **YOLO for recognition**: Classifies traffic signs into 43 categories defined by the GTSRB dataset.

YOLO divides the image into a grid, predicting bounding boxes and class probabilities for each grid cell. The loss function optimizes multiple components, including localization, classification, and confidence in predictions.



## Advantages of YOLO

 **Real-time performance**: YOLO can process multiple images per second, which is critical for embedded applications.

 **Accuracy**: Its unified structure reduces localization and classification errors.

 **Versatility**: Suitable for a wide range of object detection applications.

## **Model Training**

* **YOLO detection model**: Trained to generate bounding boxes around detected traffic signs.
* **YOLO recognition model**: Trained to classify traffic signs into 43 categories.
* Both models were trained on Kaggle using high-performance GPUs.

## **Evaluation**

Model performance was evaluated using standard metrics:

**Precision :** The proportion of correct predictions among all predictions

**Recall :** The proportion of actual objects correctly detected.

**mAP50**

**mAP50-95**

# Implementation

## System Architecture

 **Input of a raw image**: A road image containing one or more traffic signs.

 **YOLO detection model**: Extraction of regions of interest (ROI) by generating bounding boxes around detected objects.

 **ROI extraction**: Regions corresponding to bounding boxes are isolated and prepared for further analysis.

 **YOLO recognition model**: Precise identification of each traffic sign within the isolated ROIs.

 **Output predictions**: Display of coordinates, classes, and associated probabilities for each detected sign.

 **Frontend presentation**: Results are displayed to the user via a web interface

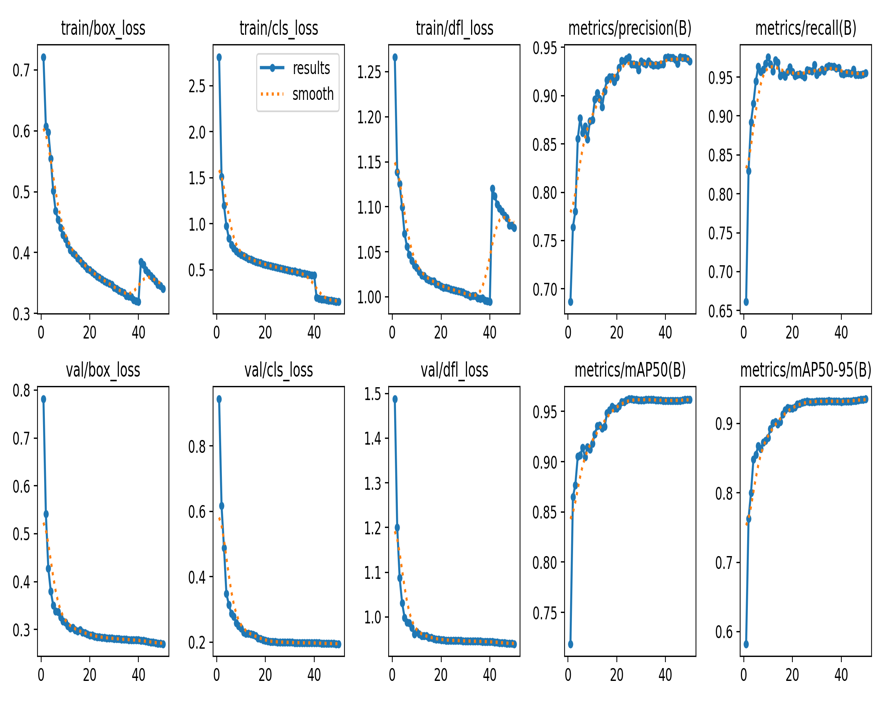
## Technical Requirements

* Frontend: Web-based interface
* Backend: REST API implementation
* Model integration: Combined YOLO models

# Results

## Model Performance Metrics :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **mAP50** | **mAP50-95** |
| YOLO for recognition | 92,2% | 96,4% | 95,8% | 93,2% |
| YOLO for detection | 95,8% | 90,4% | 96,2% | 78,2% |



### Performance Analysis

* **Detection Model Performance**:
  + High precision (95.8%) indicates reliable sign detection
  + Recall of 90.4% shows good coverage of existing signs
  + Strong mAP50 score demonstrates robust localization
* **Recognition Model Achievement**:
  + Balanced precision (92.2%) and recall (96.4%)
  + Exceptional mAP50-95 score of 93.2% showing scale invariance

# Discussion

## Achievements

* Successful integration of dual YOLO models
* Robust performance across various conditions
* Efficient web-based deployment

### Limitations

* The detection model fails when traffic signs occupy approximately 90% or more of the input image, due to YOLO's architectural design being optimized for objects within broader contexts
* Reduced performance in low-light conditions
* Challenges with partially obscured signs
* Dataset geographic limitations

### Challenges Addressed

* Environmental variation handling through data augmentation
* Real-time processing optimization
* Integration of detection and recognition systems

# Conclusion

The project successfully demonstrates the effectiveness of combining two YOLO models for traffic sign recognition and detection. The system achieves high accuracy while maintaining real-time processing capabilities. The web-based deployment enables practical application, though future work is needed to address current limitations.

# Future Directions

* Dataset expansion for geographic diversity
* Embedded system optimization
* Enhanced performance in extreme conditions

# References

[1] Ultralytics. (2024). YOLOv8 Documentation. Retrieved from <https://docs.ultralytics.com/>

* Official documentation for YOLOv8 implementation and usage
* Technical specifications and model architectures
* Training and deployment guidelines

[2] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv:1804.02767.

* Original YOLO architecture and methodology
* Retrieved from <https://pjreddie.com/darknet/yolo/>

[3] Kaggle. (2024). German Traffic Sign Recognition Benchmark (GTSRB).

* Dataset and competition platform
* Retrieved from <https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign>

[4] GitHub. (2024). Ultralytics YOLOv8 Repository.

* Source code and implementation details
* Retrieved from <https://github.com/ultralytics/ultralytics>

[5] FastAPI. (2024). FastAPI Documentation.

* Web framework used for backend implementation
* Retrieved from <https://fastapi.tiangolo.com/>

[6] GTSRB. (2024). German Traffic Sign Recognition Benchmark.

* Official dataset documentation
* Retrieved from <https://benchmark.ini.rub.de/gtsrb_news.html>

[7] PyTorch. (2024). PyTorch Documentation.

* Deep learning framework documentation
* Retrieved from <https://pytorch.org/docs/stable/index.html>

# Appendices

* Complete model configuration files for both YOLO models
* The full data.yaml configuration
* The complete frontend HTML/CSS code
* The full backend Python implementation
* Example images showing the detection and recognition process
* Detailed dataset organization and preprocessing scripts