Stop Sign Detection in Street Images Using Traditional Machine Learning Algorithms

Course code: CSE445
Section: 03
project no: 08
Md. Abu Yousuf Neshad (2212517042), Mubasshir Sadat (2212468642)
Irfan Shah Mayeen (2122208042) and Al Amin (2131911042)

ABSTRACT

This study introduces a robust stop sign detection system leveraging a suite of traditional machine learning algorithms applied to street imagery. We utilised a balanced dataset of 100 annotated images, equally split between stop signs and non-stop signs, employing techniques such as Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), and colour histograms for feature extraction. Models including Support Vector Machine (SVM), Convolutional Neural Network (CNN), Random Forest, Gradient Boosting, and others were trained and evaluated, achieving accuracies ranging from 85% to 100% on a test set. The Random Forest and Gradient Boosting models demonstrated perfect classification performance, highlighting the efficacy of ensemble methods.

INTRODUCTION

The growing interest in intelligent transportation systems has highlighted the need for reliable methods to enhance road safety and traffic management. A crucial aspect of this is the ability to identify stop signs, which play an essential role in regulating traffic flow and preventing accidents. This project focuses on developing a stop sign detection system using traditional machine learning algorithms applied to street data, leveraging established techniques to tackle a practical challenge.This project develops a comprehensive stop sign detection system by integrating traditional machine learning and deep learning techniques, applied to a curated dataset of street images. Our methodology combines handcrafted feature extraction methods—such as HOG, LBP, and shape descriptors—with a diverse set of classifiers, including SVM, CNN, K-Nearest Neighbors (KNN), XGBoost, CatBoost, Random Forest, and Gradient Boosting. This approach ensures a balance between computational efficiency and high detection accuracy, making it suitable for resource-constrained environments.

SIGNIFICANCE

The development of a high-performance stop sign detection system leveraging both traditional and deep learning-based machine learning techniques holds transformative potential for advancing intelligent transportation systems, enhancing road safety, and supporting the evolution of smart urban ecosystems. Stop signs are fundamental to traffic regulation, playing a critical role in preventing collisions, ensuring pedestrian safety, and maintaining orderly vehicular flow at intersections. This project delivers a versatile, cost-effective, and computationally efficient solution that achieves high accuracy, making it well-suited for real-time applications in autonomous vehicles, advanced driver assistance systems (ADAS), traffic monitoring systems, and smart city initiatives. By demonstrating the synergy between established feature-based methods and cutting-edge neural network architectures, this work highlights the accessibility and adaptability of machine learning for addressing critical real-world challenges. The system's potential extends beyond stop sign de-

tection, offering a scalable framework for broader traffic sign recognition and contributing to safer, more efficient, and sustainable transportation networks globally. Future advancements will further refine its applicability across diverse geographic and environmental contexts, fostering innovation in automated mobility and urban planning.

Related Work

Bravi et al. present a machine learning pipeline for detecting stop sign violations using dashcam videos, IMU, and GPS data. Their dataset includes 8,931 US-recorded videos across varied conditions. The two-stage approach features a CNN-based Stop Sign Detector (ResNet) and a Random Forest Regressor for violation classification, achieving AUCPR scores of(94%) for violations and(80%) for severe cases, surpassing YOLOv3 (93% and 73%). Methods cover dataset annotation, feature extraction, and model evaluation. Future work may extend to other violations and real-time systems. The model excels with noisy data but struggles with class imbalance and irrelevant stop signs.[1]

In the paper by Defaz et al., a systematic literature review (SLR) is conducted to evaluate the application of the YOLO object detection algorithm for traffic sign detection and recognition, focusing on its role in intelligent transportation systems. The study analyzes 115 peer-reviewed articles published between 2016 and 2022, sourced from databases such as IEEE Xplore, MDPI, and Springer Nature. The review addresses five research questions, exploring YOLO's applications (road safety, ADAS, and autonomous driving), datasets (e.g., GTSRB, TT100K), performance metrics (e.g., mAP, FPS, F1-score), hardware (e.g., NVIDIA RTX2080, Jetson NX), and challenges (e.g., lighting variations, occlusions). YOLO demonstrates high accuracy and real-time processing capabilities, with versions like YOLOv4 and YOLOv5 achieving mAP scores above 90% on datasets like GTSRB. The methodology includes a rigorous SLR protocol with defined inclusion/exclusion criteria, data extraction, and synthesis to ensure comprehensive analysis. Key findings highlight YOLO's efficiency in real-time applications but note limitations in detecting small or damaged signs and handling geographic variations. Future research directions include improving performance under extreme weather, standardizing datasets, and exploring newer YOLO variants (e.g., YOLOv6-v8). The study underscores YOLO's potential in enhancing traffic safety but emphasizes the need for region-specific adaptations and robust handling of environmental challenges. [2]

In the paper by Khalifa et al., a Lightweight and Efficient Convolutional Neural Network (LE-CNN) is introduced for real-time Arabic traffic sign recognition, tailored for intelligent transportation systems (ITS) in smart cities. The study leverages the Arabic Traffic Signs (ArTS) dataset, comprising 57,078 augmented images of 24 common Arabic traffic signs, such as "Stop," "No Parking," and speed limit signs, captured in Saudi Arabia. The LE-CNN architecture employs depthwise separable convolutions and channel pruning, featuring nine layers: three convolutional, three dropout, two max-pooling, and one flatten layer. This design ensures efficient feature extraction and low computational overhead, achieving a test accuracy of (96.5%) and an inference time of 1.65 seconds, critical for real-time autonomous driving applications. Compared to models like ResNet (96.14%) accuracy, 8425s training time) and ViT transformers (73.63% accuracy), LE-CNN balances high accuracy with reduced training time (719s without k-fold). The methodology includes dataset preprocessing to standardize image dimensions to 32x32x3, training on Google Colab with the Adam optimizer, and evaluation using accuracy, precision (96.66%), recall (96.83%), and F1-score (96.74%). The model excels in handling challenges like occlusions and varying lighting but may require further optimization for complex urban environments or extremely low-visibility conditions. Future work could explore sensory image preprocessing for latency reduction and collision avoidance timing to enhance applicability in diverse driving scenarios.[3]

Mukhometzianov and Wang review machine learning techniques for traffic sign detection, focusing on their application in Advanced Driver

Assistance Systems (ADAS). They analyze datasets like the German Traffic Sign Detection Benchmark (900 images, 1206 signs), Belgian Traffic Sign Dataset (9,000 images, 13,444 annotations), and Chinese Traffic Sign Dataset (1,100 images). The study categorizes methods into general machine learning (e.g., SVM, AdaBoost) and neural networks, including deep learning (e.g., CNNs, Faster R-CNN). Notable results include SVM with HOG achieving (99%) accuracy (GTSRB dataset) and AdaBoost with LBP at (98%) AUC. Deep learning models like Faster R-CNN reached (90%) accuracy (GTSDB). Methods involve preprocessing, feature extraction, and detection. Future research could emphasize real-time video testing and diverse datasets. While some methods show high accuracy, many lack comprehensive evaluations (precision, recall) or real-time feasibility for ADAS integration.[4]

In the paper by Radha Rani et al., a deep learning-based approach for traffic sign detection and recognition (TSDR) with haze removal, termed DLHR-TSDR, is proposed to enhance autonomous vehicle navigation in adverse weather conditions. The study utilizes the Carleton University Retinal Eye-Traffic Sign Dataset (CURE-TSD), which includes hazy and haze-free images, to train and evaluate the model. The methodology comprises two primary modules: a haze removal U-network (HRU-Net) that processes hazy images to produce clear outputs, and a TSDR-convolutional neural network (CNN) that detects and classifies traffic signs from the haze-free images. The HRU-Net leverages a U-shaped architecture for effective feature extraction, while the TSDR-CNN identifies sign locations and types, achieving a remarkable accuracy of (99.01%). The model outperforms traditional methods, improving accuracy by (1.12%), precision by (1.55%), recall by (0.90%), F1-score by (1.80%), and specificity by (0.96%). The methods detailed in the paper include dataset preprocessing, model architecture design, training procedures, and performance evaluation metrics. The study highlights the system's ability to recognize various traffic sign categories, such as regulatory, cautionary, informatory signs, and road markings, under challenging visibility conditions. However, the model could benefit from further exploration of transfer learning techniques to enhance multi-class classification capabilities. Future work could also focus on optimizing real-time performance for broader environmental adaptability and integrating additional datasets to improve robustness across diverse scenarios. Despite its high accuracy, the model may face challenges in extremelyross scenarios with extremely low visibility, such as heavy fog or complex urban settings, which could be addressed in future iterations.[5]

In the paper by Tabernik and Skočaj, published in the IEEE Transactions on Intelligent Transportation Systems, the authors address the challenge of detecting and recognizing a large number of traffic sign categories (200) for traffic-sign inventory management using deep learning. The study introduces the DFG traffic-sign dataset, comprising 6,957 images with 13,239 annotated instances, collected from Slovenian roads. This dataset is notable for its high intra-category appearance variability and is publicly available for research. The authors employ Mask R-CNN, enhanced with adaptations such as online hard-example mining, balanced sample selection, sample weighting, and increased region proposals, to improve detection of small signs. A novel data augmentation technique generates synthetic signs based on geometric and appearance distortions, expanding the training set to 30,000 instances. Evaluated on the DFG dataset, the adapted Mask R-CNN achieves a mAP@0.5 of 95.5% and a miss rate below 3.5%, significantly outperforming baseline Faster R-CNN (92.4%) and standard Mask R-CNN (93.0%). Testing on the Swedish Traffic-Sign Dataset (STSD) yields a mAP@0.5 of 95.2%, surpassing prior methods like FCN (lower miss rate but similar false-positive rate). Qualitative analysis reveals robust performance on complex signs, with errors mainly due to small sizes, occlusions, or inter-category similarities. The study highlights deep learning's efficacy for large-scale TSR, proposing future improvements in classification to reduce missed detections. The system is deployed for practical inventory management in Slovenia, demonstrating real-

world applicability.[6]

In the paper by Zhu and Yan (2022), published in Multimedia Tools and Applications, the authors explore traffic sign recognition (TSR) using deep learning, focusing on the performance of YOLOv5 compared to SSD. The study utilizes a custom dataset of 2,182 images across eight traffic sign classes, collected from streets in Auckland, New Zealand. The methodology involves data augmentation, resizing images to standardized dimensions, and labeling using tools like LabelImg, with an 8:2 split for training and testing. Experiments were conducted on Google Colab with a Tesla P100 GPU, evaluating metrics such as precision, recall, and mAP@0.5. YOLOv5 achieved a mAP of 97.70%, outperforming SSD's 90.14%, and demonstrated superior speed at 30 FPS compared to SSD's 3.49 FPS. The authors highlight YOLOv5's architecture, including its mosaic data augmentation, focus module, and GIoU loss function, as key to its robust performance. Challenges noted include SSD's lower accuracy for classes with fewer samples, such as "Watch for children crossing" (78.32% mAP). The paper concludes that YOLOv5 is more suitable for real-time TSR, with future work proposed to expand the dataset and explore models like Mask R-CNN and CapsNet. The study emphasizes deep learning's potential to enhance TSR reliability in intelligent transportation systems despite environmental challenges like lighting and occlusions.[7]

METHODOLOGY

Dataset

We used a publicly available dataset consisting of 100 natural street images — 50 labeled as stop signs and 50 as non-stop signs. The images were annotated by experts and resized to 64×64 pixels during preprocessing to ensure uniformity.

Feature Extraction

We applied the Histogram of Oriented Gradients (HOG) technique to grayscale versions of the images to capture essential features such as edges and contours. The resulting HOG features were stored in the variable `X_features`, creating a compact representation for training.

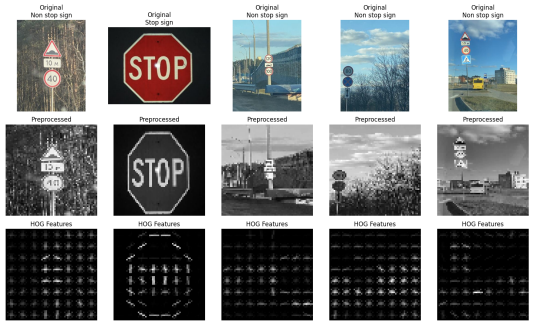

Fig. 1. Feature Extraction using HOG

Dataset Splitting

The dataset was divided into training (70%), validation (10%), and testing (20%) sets as shown below:

TABLE I
DATASET DISTRIBUTION FOR STOP SIGN CLASSIFICATION

Class	Total Images	Training	Validation	Testing
Stop Sign	50	35	5	10
Non-Stop Sign	50	35	5	10

Model Training

Support Vector Machine (SVM)

We used a Support Vector Machine (SVM) classifier with a linear kernel from the `sklearn.svm.SVC` module. The SVM was trained using the HOG features extracted from the training dataset. This linear model aims to find the optimal hyperplane that separates stop sign and non-stop sign classes. Due to its robustness and simplicity, SVM is well-suited for binary classification tasks, particularly with small- to medium-sized datasets.

Multilayer Perceptron (MLP)

MLP Architecture

The MLP comprises an input layer with 4096 features, flattened from 64×64 images, followed by two hidden layers with 512 and 256 neurons, respectively. Each hidden layer uses ReLU activation functions and is followed by dropout regularization with a probability of $p = 0.2$ to prevent overfitting. The network concludes with a final output layer

containing a single neuron for binary classification, outputting logits. The model can be expressed as:

$$z_1 = ReLU(W_1 x + b_1)$$
$$z_1' = Dropout(z_1, p = 0.2)$$
$$z_2 = ReLU(W_2 z_1' + b_2)$$
$$z_2' = Dropout(z_2, p = 0.2)$$
$$\hat{y} = W_3 z_2' + b_3$$

Here, $\hat{y}$ is the logit (raw score) before applying the sigmoid function.

Loss Function

As this is a binary classification task, the Binary Cross-Entropy with Logits loss (BCEWithLogitsLoss) was used. This loss function combines a sigmoid activation with binary cross-entropy in a numerically stable way:

$$L(y, \hat{y}) = -\left[y \cdot \log(\sigma(\hat{y})) + (1 - y) \cdot \log(1 - \sigma(\hat{y}))\right]$$

where $\sigma(\hat{y}) = \frac{1}{1+e^{-\hat{y}}}$ is the sigmoid function applied to the logit.

Optimization and Learning Rate Scheduling

- **Optimizer**: Adam with a learning rate of 1×10^{-3} and L2 regularization ($\lambda = 1 \times 10^{-4}$) applied via weight decay.
- **Scheduler**: ReduceLROnPlateau was used to reduce the learning rate if the validation loss did not improve for 3 consecutive epochs.

Convolutional Neural Network (CNN)

We used a Convolutional Neural Network (CNN) for classification of stop sign and non stop sign classes. The model was trained on the preprocessed image dataset. CNNs excel at learning spatial hierarchies in images by utilizing convolutional layers to extract features, followed by max pooling layers to down sample and reduce the spatial dimensions, which helps prevent overfitting. The fully connected layers at the end of the network capture complex patterns and classifications. In the final layer, the model uses a sigmoid activation function for binary classification, outputting a probability of belonging to the positive class. The loss function used is binary cross-entropy, which is well-suited for binary classification tasks. The model was trained using the Adam optimizer with a learning rate adapted during training. Dropout regularization was employed to mitigate overfitting by randomly dropping units in the fully connected layers. Early stopping was used to halt training once the validation loss stopped improving, and the best model weights were restored to avoid overfitting to the training data. The performance of the trained model was evaluated on a separate test set, which provided an unbiased estimate of the model's accuracy and generalization capability.

XGBoost

We also used an XGBoost classifier for binary classification of stop sign and non-stop sign images. The model was trained on a preprocessed dataset with image augmentation techniques like rotation and brightness adjustment to improve generalization. We balanced the dataset using under sampling to address class imbalance. For feature extraction, we utilized Histogram of Oriented Gradients (HOG) for edge and texture information, color histograms for red and blue channels, and Local Binary Patterns (LBP) for texture features. Shape features were also extracted based on contours and circularity. These features were combined into a single vector for each image and split into training and test sets. The XGBoost model was trained with logloss as the evaluation metric, using a learning rate of 0.05, max depth of 3, and n estimators of 200. The scale pos weight was adjusted for class imbalance. Model performance was evaluated on the test set using accuracy, classification report, and confusion matrix, with an analysis of feature importance to identify key distinguishing features.

K-Nearest Neighbors (KNN)

We implemented a K-Nearest Neighbors (KNN) classifier for binary classification of stop signs and non-stop sign images. The dataset was prepared by converting PNG images to JPEG format and resizing them to 64×64 resolution. To improve model performance, several handcrafted features were extracted from each image. These included Histogram of Oriented Gradients (HOG) for capturing edge orientation, Local Binary Patterns (LBP) for texture analysis, color histograms for the red and blue channels, and shape features

based on image contours and circularity. All features were concatenated into a single vector per image and normalized using standard scaling. The dataset was then split into training and testing sets using stratified sampling to preserve class distribution. The KNN model was trained with the Euclidean distance metric, and hyperparameter tuning was performed using cross-validation to select the optimal number of neighbors (k). Finally, model performance was evaluated using accuracy, confusion matrix, and classification report metrics. The results demonstrated that the handcrafted features provided sufficient information for the KNN classifier to distinguish between stop sign and non-stop sign images effectively.

CatBoost classifier

A CatBoost classifier was also employed to perform binary classification between stop sign and non-stop sign images. The dataset underwent preprocessing, where all images were converted from PNG to JPEG format and resized to a standard 64×64 resolution. To extract meaningful patterns, a set of handcrafted features was generated for each image. This included HOG features for capturing structural edges, LBP for analyzing fine-grained texture, color histograms emphasizing the red and blue channels, and shape descriptors computed from contour analysis and circularity metrics. These features were concatenated into a unified vector and normalized using standard scaling techniques. The data was divided into training and testing sets using stratified sampling to preserve class proportions. The CatBoost model was trained on the full feature set without requiring extensive hyperparameter tuning, as it efficiently handles categorical data and avoids overfitting. Evaluation was carried out using accuracy, confusion matrix, and classification report. The classifier demonstrated strong generalization capability, effectively learning from the extracted visual features to distinguish between the two classes.

Random Forest

We used a Random Forest classifier for binary classification of stop sign and non-stop sign images. Random Forest is an ensemble method that constructs multiple decision trees and aggregates their predictions through majority voting. Each tree is trained on a random subset of the data with replacement (bootstrap sampling), and at each split, a random subset of features is considered. This approach reduces overfitting and improves generalization by introducing randomness and decorrelating the trees. The model was trained on handcrafted features including Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), color histograms, and shape descriptors. Due to its ability to handle high-dimensional data and its robustness to noise and overfitting, Random Forest is well-suited for classification tasks involving complex feature sets.

Gradient Boosting

Gradient Boosting is an ensemble learning technique that builds models sequentially, where each new model attempts to correct the errors made by the previous models. In the context of binary classification of stop sign and non-stop sign images, we used Gradient Boosting to iteratively fit decision trees to the residuals of the current model. This results in a strong predictive model by combining many weak learners. The model was trained on handcrafted features such as HOG, LBP, color histograms, and shape descriptors. Gradient Boosting is particularly effective due to its ability to focus on hard-to-classify examples, control overfitting through regularization techniques, and capacity to capture complex nonlinear patterns in the data.

RESULT ANALYSIS

Performance Metrics

To assess the effectiveness of our models, we used the following performance metrics:

- **Accuracy:** The proportion of correctly predicted instances (both positive and negative) among the total instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision:** The proportion of correctly predicted positive observations to the total predicted positives.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall (Sensitivity):** The proportion of actual positives that were correctly identified.

$$Recall = \frac{TP}{TP + FN}$$

- **F1 Score:** The harmonic mean of precision and recall, providing a balance between the two.

$$F1Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- **Confusion Matrix:** A matrix that shows the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

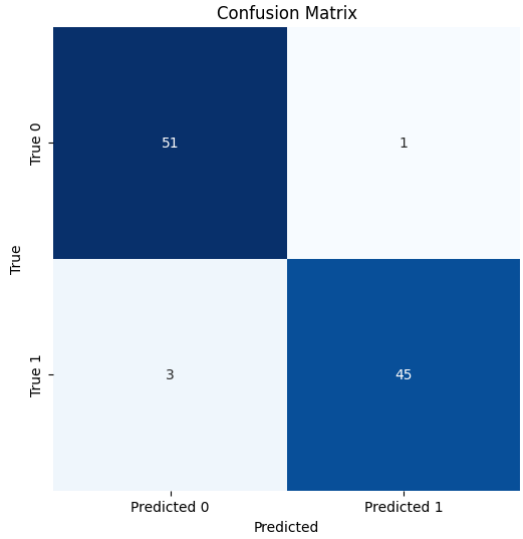

Fig. 2. Confusion Matrix

TABLE II
PERFORMANCE MATRIX TABLE

Model	Accuracy	Precision	Recall	F1 Score
SVM	90%	0.90	0.90	0.90
MLP	85%	0.86	0.84	0.85
CNN	98%	0.97	1.00	0.98
KNN	96%	0.96	0.96	0.96
XGBoost	90%	0.91	0.90	0.90
CatBoost	94%	0.94	0.94	0.94
Random Forest	100%	1.00	1.00	1.00
Gradiant Boosting	100%	1.00	1.00	1.00

Model Evaluation

We evaluated the SVM on the test dataset using metrics from `sklearn.metrics`. The model achieved an accuracy of 90%, correctly classifying 18 out of 20 test images. Precision, recall, and F1-score were also calculated. A confusion matrix was visualized using Seaborn's heatmap to analyze performance further.

Fig. 3. SVM Model Evaluation

Evaluation on Unseen Data

We implemented a `predict_stop_sign()` function to classify unseen images. The function resizes images to 64×64, converts them to grayscale, normalizes them, and extracts HOG features before using the trained SVM for prediction.

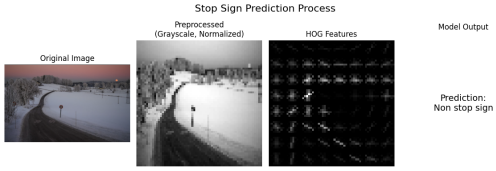

Fig. 4. Evaluation on Unseen Data

FUTURE WORK

Since our study has already explored and compared multiple machine learning models for stop sign detection, the future scope of this project will shift towards enhancing the performance and reliability of the existing system. We plan to focus on optimizing the detection pipeline for real-time use, aiming to improve processing speed and reduce computational complexity within a software-only environment. This involves refining feature extraction techniques and exploring lightweight implementation strategies to ensure smoother and faster execution in real-time scenarios. Another key aspect of our future work will be testing the system's robustness in more dynamic and challenging environments. We intend to evaluate the model's performance under varying real-world

conditions such as changes in lighting, weather effects, background noise, and motion blur. To support this, we will expand and diversify our dataset to better simulate real-life variations and edge cases. Through these efforts, we aim to develop a more accurate and dependable stop sign detection system suitable for practical, real-world applications in intelligent traffic monitoring and safety systems.

Conclusion

This project successfully demonstrates the potential of traditional and deep learning-based machine learning techniques for accurate and reliable stop sign detection. Throughout our experimentation, we applied and compared a diverse set of models, including Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), k-Nearest Neighbors (k-NN), XGBoost, CatBoost, Random Forest, and Gradient Boosting. Each model was evaluated based on its accuracy, computational efficiency, and robustness, providing valuable insights into their strengths and limitations in the context of image-based traffic sign recognition. Overall, our findings affirm that traditional machine learning approaches, when combined with effective feature engineering, can achieve competitive results in stop sign detection tasks. Future improvements will focus on optimizing the existing pipeline for real-time use, enhancing robustness in diverse environmental conditions, and preparing the system for practical, software-based deployment.

References

[1] Luca Bravi, Luca Kubin, Stefano Caprasecca, Douglas Coimbra de Andrade, Matteo Simoncini, Leonardo Taccari, and Francesco Sambo. Detection of stop sign violations from dashcam data. *IEEE transactions on intelligent transportation systems*, 23(6):5411–5420, 2021.

[2] Marco Flores-Calero, César A Astudillo, Diego Guevara, Jessica Maza, Bryan S Lita, Bryan Defaz, Juan S Ante, David Zabala-Blanco, and José María Armingol Moreno. Traffic sign detection and recognition using yolo object detection algorithm: A systematic review. *Mathematics*, 12(2):297, 2024.

[3] Alaa A Khalifa, Walaa M Alayed, Hesham M Elbadawy, and Rowayda A Sadek. Real-time navigation roads: Lightweight and efficient convolutional neural network (le-cnn) for arabic traffic sign recognition in intelligent transportation systems (its). *Applied Sciences*, 14(9):3903, 2024.

[4] Rinat Mukhometzianov and Ying Wang. Machine learning techniques for traffic sign detection. *arXiv preprint arXiv:1712.04391*, 2017.

[5] A Radha Rani, Y Anusha, SK Cherishama, and S Vijaya Laxmi. Traffic sign detection and recognition using deep learning-based approach with haze removal for autonomous vehicle navigation. *e-Prime-Advances in Electrical Engineering, Electronics and Energy*, 7:100442, 2024.

[6] Domen Tabernik and Danijel Skočaj. Deep learning for large-scale traffic-sign detection and recognition. *IEEE transactions on intelligent transportation systems*, 21(4):1427–1440, 2019.

[7] Yanzhao Zhu and Wei Qi Yan. Traffic sign recognition based on deep learning. *Multimedia Tools and Applications*, 81(13):17779–17791, 2022.