

# Machine Learning Approaches for Traffic Sign Detection: Methods and Challenges

**Xuanyu Ren**

Maple Leaf International School,  
Chongqing, China  
guo20080104@gmail.com

## Abstract:

Traffic sign detection (TSD) is a fundamental perceptual task for intelligent transportation systems and autonomous driving; however, performance deteriorates in adverse weather, low light, occlusion, and cross-regional sign variability. This review summarizes how the field has changed from traditional machine-learning pipelines, such as SVM or AdaBoost with color/shape features, to modern deep architectures, such as two-stage R-CNN variants, one-stage YOLO/SSD families, lightweight models for edge deployment, and new Transformer-based detectors. This review sorts of methods into groups, compares them on major benchmarks, and look at their accuracy, runtime, and robustness. The analysis demonstrates that deep learning significantly outperforms conventional methods in terms of precision and scalability. Nevertheless, a speed-accuracy trade-off remains, and models trained within a specific sign system frequently exhibit inadequate generalization to alternative systems (e.g., India and Germany), highlighting the necessity for region-specific data or explicit domain adaptation. Ongoing problems include bad annotations and long-tail categories. The practical advice is provided for deploying on embedded platforms and highlights some promising areas to explore, such as multimodal fusion (camera + LiDAR), augmentation and adaptation for changes in weather and lighting, compact architectures with knowledge distillation, and Transformer pipelines that are optimized for small objects. The review's goal is to give a short, deployment-focused guide for improving reliable TSD in real-world ITS.

**Keywords:** Traffic Sign Detection, Machine Learning, Deep Learning, Intelligent Transportation Systems

## 1. Introduction

As the rise of intelligent transportation systems and

increasingly autonomous vehicles continues, the need for reliable, accurate traffic sign detection (TSD) has also increased. Online implementation of this

method is crucial to guarantee the safety of road users, as well as enabling real-time decisions for automated driving environments. TSD enables autonomous vehicles to understand road regulations, prevent accidents, and comply with a wide range of rules that vary across different driving environments.

Nevertheless, there is still significant difficulty in accomplishing the robust and real-time detection of traffic signs. The varying weather and illumination conditions, distinct traffic signs in different regions, interferences and complex city backgrounds all make it extremely difficult for better performance [1]. The early successes of the traditional computer vision-based techniques such as Histogram of Oriented Gradients (HOG) with Support Vector Machines (SVM), or AdaBoost classifiers were promising [2]. However, they have not been very useful in practice since (1) these can't be applied across a large variety of situations and scale up from zero.

CNN architecture is a game changer for TSD systems in the time of deep learning. More powerful architectures such as Faster R-CNN, SSD [3], YOLOv5 [4] and EfficientDet have significantly improved the detection accuracy and inference speed, at least for standard datasets like GTSRB, GTSDb and Tsinghua-Tencent 100K. More recently, transformer-based models like DETR and Swin Transformer have pushed the envelope further, enabling end-to-end detection with richer contextual image features and improved interpretability [5]. Promoting traffic sign detection technology is not only a technical usher in the old era, as the main motive force does not come from public security. Research on this year's fatalities indicated that integrating sophisticated TSD systems into ADAS and autonomous driving technology contributes to lowering the number of traffic accident deaths. TSD has also become more feasible in low-resource vehicular environments thanks to lightweight, real-time models that can be deployed on embedded edge platforms - such as MobileNet and YOLO-Nano.

Nonetheless, several barriers remain to our work. For example, well-developed countries will often have a regional bias in their traffic signs. This biases all of our training data, which means it attempts to use these models in places like India and Southeast Asia, where there are a lot fewer of those signs by contrast, their performance is easy to disrupt. Even with the development of sophisticated detectors, bad weather and low-light conditions is a persistent problem. It is not only to put forward a new idea at an algorithmic level but also needs a method which deeply depends on various data source inputs; the regional mapping teams should be established all over the world who can provide high-quality, multi-layer tagging map base.

The goal of this paper is to present an in-depth review of

machine learning-based approaches for traffic sign detection, which will serve as a solid basis for both academic and practice when it comes to working on associated topics.

## 2. Literature review

### 2.1 Classical Machine Learning Methods

To the best of our knowledge, there were very few types of D systems (TSDs) in that era which employed or relied upon legacy computer vision and machine learning techniques. Classical algorithms, on the other hand, often used handcrafted features with which they then "soaked" fruit in machine learning classifiers (like SVM or AdaBoost).

One of the alternate approaches is to use HOG along with SVM for feature descriptor and classifier. Such combination of styles works well in other traffic sign detection domains. As researchers [6] demonstrated that it can be effectively applied to their multi-view traffic sign localization task with simple shape signs. AdaBoost itself (initially inspired by the Viola-Jones method) combined with it and at an earlier stage ends up with real-time systems that however aren't robust in different lighting and occluding conditions [7]. For instance, a performance loss is significant for the image processing systems using known techniques under such an adverse environment as smog or rain/snow has been indicated in reference. This is due to problems of image distortion, poor contrast or noise. The traditional feature extraction algorithm is difficult to stably extract the shape and color of the sign, so it cannot accurately identify.

These traditional methods are much too sensitive to environmental conditions. The manually designed features did not generalize to true scene variations, e.g., motion blur, partial occlusion, and non-standard sign formats. These characteristics were hard to achieve with computer processing. Traditional systems are very strong as long as a certain limit of simplicity is respected, but 'decay' dramatically in the presence of real complications.

### 2.2 Early Deep Learning Models

A key deep-learning early adopter in this context was proposed by [8] They proposed a multi-column deep neural network (MCDNN) into the German Traffic Sign Recognition Benchmark and outperformed even classical approaches. The recognition rate of MCDNN that includes scale, rotation and brightness variations is 99% [9]. It's hard to do with your basic model.

The next milestone was the OverFeat mode [10], which combined classification localization detection into a unified CNN framework. It was necessary for that work to

demonstrate that other models of such a type, in principle, were possible. The experiment of Sermanet and Felzenszwalb is the base of transfer learning. They also demonstrated fine-tuning of pre-trained CNNs for traffic sign datasets.

### 2.3 Contemporary Deep Learning Architectures

The newest TSD models use better architecture and training methods to make performance and generalization better. Today, there are two main types of detectors: region-based detectors and single-shot detectors.

Faster R-CNN and Mask R-CNN are two region-based models that are widely used in traffic sign detection pipelines. They have done very well at localization and segmentation. For real-time applications, single-shot detectors like YOLO (You Only Look Once) and SSD (Single Shot Multibox Detector) have become very popular. Redmon et al. released YOLOv2 to get competitive accuracy with much faster inference times. Later, YOLOv3 and YOLOv5 were released to make the results even better [11].

Lightweight models like MobileNet and EfficientDet make it even easier to deploy on edge devices. Tan et al. introduced EfficientDet with compound scaling, attaining high accuracy while minimizing resource usage, rendering it suitable for resource-limited settings such as ADAS [12]. Carion et al. recently released DETR (Detection Transformer), which brought transformers to object detection. Transformers use global attention mechanisms, which makes them work better in structured environments with overlapping signs or occlusions.

### 2.4 Benchmark Datasets

For evaluating and comparing TSD equipment, it is very important to have standardized datasets. The German Traffic Sign Recognition Benchmark (GTSRB) and the German Traffic Sign Detection Benchmark (GTSDb) are examples of such benchmark datasets. They have thousands of annotated traffic sign images taken in real-world German road conditions.

The LISA dataset from the United States has a large variety of different lighting conditions, scales, and occlusions, which come in quite handy for cross-regional research. The Tsinghua-Tencent 100K also has complex urban conditions from Chinese roads: several signs per frame; various kinds of signs.

Despite these improvements, many of the datasets that already exist continue to favor either European or American signs. Sharma et al. point out that for geographical regions like India and Southeast Asia, where road conditions (and so sign formats) differ markedly from those in Western countries, there are hardly any high-quality annotated

datasets. They argue for an increase in the creation of indigenous (regionally specific) datasets in order that overall model performance might improve [8].

## 3. Methodology

### 3.1 Categorization Approach

There are two main types of traffic sign detection methods: traditional machine learning (ML) techniques and deep learning-based techniques.

Traditional ML methods use things like color segmentation, edge detection, and shape descriptors that are made by hand. These are usually used with classifiers like SVM or ELM. These systems perform effectively in controlled settings but encounter challenges with variability in real-world contexts.

Deep learning has largely replaced traditional methods because it can learn hierarchical features directly from raw image data. These models are often grouped by architecture into: Two-stage detectors like Faster R-CNN, which are very accurate but take longer to make predictions; One-stage detectors like YOLOv5 and SSD, which are known for their real-time performance with little loss of accuracy; and Transformer-based models like DHPE-DETR, which use attention and position encoding to improve localization in complex scenes.

This taxonomy helps figure out the best architecture based on deployment needs like speed, accuracy, or hardware limitations.

### 3.2 Performance Metrics

To evaluate TSD systems, a balance between accuracy, efficiency, and robustness is needed. For example, mean Average Precision can be used at different IoU thresholds to measure detection accuracy. Inference Speed is important for real-time systems like ADAS. Robustness is the model's ability to stay accurate in different situations, like when there is occlusion, lighting changes, or background clutter. This can be improved by using techniques like data augmentation and domain adaptation. In terms of datasets, public datasets like GTSRB, GTSDb, LISA, and Tsinghua-Tencent 100K are often used. However, cross-domain generalization is still a problem, especially when models trained on European signs are tested on Indian or Middle Eastern datasets.

### 3.3 Model Comparison

To assess the practical efficacy of diverse traffic sign detection models, a comparative analysis as presented in Table 1 was performed among five representative architectures. These models are classified based on their type,

backbone architecture, classification accuracy, inference speed (frames per second), and the datasets used to train and test them.

**Table 1: Model comparison table used in TSD**

| Accuracy (%)      | Type       | Backbone    | Accuracy (%) | Speed (FPS) | Dataset        |
|-------------------|------------|-------------|--------------|-------------|----------------|
| Fast R-CNN        | Two-stage  | VGG16       | 90+          | 10-12       | GTSDB          |
| Mask R-CNN        | Two-stage  | ResNet-50   | 95           | 6-10        | GTSRB          |
| Refined Mask RCNN | Two-stage  | ResNet-50   | 97           | 8           | Indian Dataset |
| YOLOv5            | One-stage  | CSP-Darknet | 96           | 30 +        | TT100K         |
| ResNet-based CNN  | Classifier | ResNet-18   | 99           | 99          | Kaggle Signs   |

Fast R-CNN and Mask R-CNN are fairly accurate two-stage detectors, but they are not very fast at making decisions, which makes them less useful for autonomous driving in real time. Refined Mask R-CNN, which was made for Indian roads, increases accuracy to 97.08% while keeping speed low. On the other hand, YOLOv5 is a one-stage detector that strikes a good balance between speed (30+ FPS) and accuracy (>96%), making it a great choice for embedded Advanced Driver Assistance Systems (ADAS). ResNet-based CNN classifiers have the highest accuracy (99%), but they are usually used for classification after region proposal, so they can't be directly compared to real-time object detection.

These results show that when choosing a model, it is essential to think about both the application and the hardware limitations. Two-stage detectors are more accurate, but one-stage detectors like YOLOv5 are better for use in real time.

## 4. Results

### 4.1 Comparative analysis based on published benchmarks

In this section, the results are reported from literature, not experiments conducted. Modern YOLOv5 and YOLOv4 families precede older baselines on widely used benchmarks like TT100K [13], CCTSDB2021, and GTSDB, keeping consistency, while still being able to process data in real time. For instance, TRD-YOLO on TT100K shows a mAP@0.5 of about 86.5% and a stable frame rate of about 73 FPS, which shows a good balance between speed and accuracy for small signs [14]. Different design variants, like STC-YOLO, put even more emphasis on handling small objects and complex scenes. They also report higher mAP than vanilla YOLOv5s on TT100K/CCTSDB settings, which shows that targeted fusion/attention and anchor re-tuning work best in small-sign regimes. In bad weather, an Improved YOLOv5 made for complicated sit-

uations shows clear gains on CCTSDB2021. This shows that condition-aware attention and small-object heads can lead to measurable mAP improvements without a lot of extra work [15]. In direct comparisons on GTSDB, one-stage detectors (YOLOv4/YOLOv5) usually get both a higher mAP and a much higher FPS than two-stage Faster-R-CNN. This supports the current preference for one-stage models in real-time driving stacks.

### 4.2 Models' performance across datasets and countries

Regarding the recognition of traffic signs in different countries, cross-dataset tests show that models trained on Indian traffic-sign data do worse when tested on GTSDB (Germany) and vice versa. However, performance improves when models are trained and tested within the same regional distribution or when augmentation or mixture strategies close the gap. Evidence from surveys also shows that combining regional data (like mixing Indian and German sign sets) makes things more stable, but it doesn't completely get rid of cross-country degradation, especially for rare categories and small objects. These findings endorse the utilization of region-specific datasets during training (or explicit domain adaptation) when deployment is aimed at a specific sign system.

### 4.3 Key findings

Firstly, DL models are better than ML in terms of precision and applicability. Recent benchmarking suggests that deep detectors and recognizers (e.g., YOLO and Faster-R-CNN) have outperformed hand-crafted features and classical classifiers in terms of accuracy and robustness for the common traffic sign tasks.

Secondly, region-specific datasets are critical. Empirical cross-country studies show that there is a substantial drop in accuracy when the data undergoes a distribution shift, while training with local or mixed/augmented datasets leads to more robust predictions.

Finally, the compromise between real-time and accuracy

is a concern. One-stage YOLO based detectors generally possess superior speed-accuracy trade-off compared to two-stage object detectors in road scenes. Furthermore, inlaid results (e.g., ETSR-YOLO ~108 FPS on Jetson AGX Xavier) have indicated that fine-tuned small-object pipelines can reach the real-time requirement on-vehicle as well as to be accurate.

## 5. Conclusion

This review investigated how the traffic sign detection transformed from feature-engineered pipelines to deep models. It also evaluated such models on benchmarking datasets as well as in realistic scenarios. Overall, 1-stage CNN detectors, in particular more recent YOLO versions, are best for small, crowded and bad-weather scenes because they have the finest accuracy-throughput trade-off. “Two-stage models, however, are still good for localization but require more processing power. Yet performance doesn’t look so great: models that work well on one sign system in one region often do badly on another, and long-tail categories remain among the most common ways for you to fail.

Design can also be subject to practical limits. Deployment is further complicated by devices that punish the speed–accuracy trade-off. It allows to use lightweight backbones and placeable head small objects so that it can match on-vehicle latency, power budgets. Just as important is the quality of the data: mis-annotations or low-quality annotations and bias of the regions hinder fair evaluation and generalization. In the future, Three directions are followed: (1) to make data and domain strategies more stronger with region-oriented corpora acquisition as well as domain adaptation/augmentation; (2) make efficient, small-object-aware architectures better, including using Transformer components wisely, without adding too much overhead; and (3) make reliability better through multi-modal sensing and standardized robustness testing. These directions give a deployment-focused way to obtain reliable TSD in next-generation ITS and self-driving cars.

## References

- [1] Kumar, H., Mamoria, P., & Dewangan, D. K. (2025). Vision technologies in autonomous vehicles: progress, methodologies, and key challenges. *International Journal of System Assurance Engineering and Management*. <https://doi.org/10.1007/s13198-025-02912-3>
- [2] Maldonado-Bascón, S., Lafuente-Arroyo, S., Gil-Jiménez, P., Gómez-Moreno, H., & López-Ferreras, F. (2007). Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems*, 8(2), 264–278. <https://doi.org/10.1109/TITS.2007.895311>
- [3] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In *European Conference on Computer Vision* (pp. 21–37). Springer. [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
- [4] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934*. <https://arxiv.org/abs/2004.10934>
- [5] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In *ECCV 2020*. [https://doi.org/10.1007/978-3-030-58452-8\\_29](https://doi.org/10.1007/978-3-030-58452-8_29)
- [6] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. <https://doi.org/10.1109/CVPR.2001.990517>
- [7] Sharma, P., Devare, P., Rasal, A., Kale, S., & Jagtap, A. (2024). A Survey: Traffic Sign Recognition System using Learning Techniques. *2024 IEEE 4th International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*. <https://ieeexplore.ieee.org/document/10911592>
- [8] Cireşan, D., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32, 333–338. <https://doi.org/10.1016/j.neunet.2012.02.023>
- [9] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2014). OverFeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*. <https://arxiv.org/abs/1312.6229>
- [10] Redmon, J., & Farhadi, A. (2016). YOLO9000: Better, faster, stronger. *CVPR*. <https://doi.org/10.1109/CVPR.2017.690>
- [11] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10781–10790. <https://doi.org/10.1109/CVPR42600.2020.01080>
- [12] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32, 323–332. <https://doi.org/10.1016/j.neunet.2012.02.016>
- [13] Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-sign detection and classification in the wild. *IEEE CVPR*, 2110–2118. <https://doi.org/10.1109/CVPR.2016.231>
- [14] Chu, J., Zhang, C., Yan, M., Zhang, H., & Ge, T. (2023). TRD-YOLO: A Real-Time, High-Performance Small Traffic Sign Detection Algorithm. *Sensors*, 23(8), 3871. <https://doi.org/10.3390/s23083871>
- [15] Qu, S., Yang, X., Zhou, H., & Xie, Y. (2023). Improved YOLOv5-based for small traffic sign detection under complex weather. *Scientific Reports*, 13, 16219. <https://doi.org/10.1038/s41598-023-42753-3>