International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Real-Time Traffic Sign Recognition Using Convolutional Neural Networks

Samaksh Mangesh Shahane

Department of Science and Technology,
G. H. Raisoni Skill Tech University, Nagpur, Maharashtra, India

Abstract

However, with the further advancement and development of Intelligent Transportation Systems (ITS) technology, there is a greater requirement for developing effective and reliable techniques for the improvement of road safety and the assistance of drivers. A significant aspect in this direction is the development of Traffic Sign Recognition (TSR) technology, where a system is required for the identification of different signs on the road, such as warning signs, information signs, and regulation signs. This research is based on the development of a computer vision system, which is required for the elimination of errors arising from driver fatigue, a major cause of traffic accidents.

The system is based on the improvement and refinement of Convolutional Neural Networks (CNNs) in such a manner that it is capable of balancing high accuracy with the requirement for information processing in a rapid manner. A high degree of training is provided to the system, enabling it to perform well in different locations. Ultimately, a computerized TSR system is required for the further advancement and development of self-driving cars, creating a well-ordered traffic environment.

KEYWORDS: *Traffic Sign Recognition, Computer Vision, Convolutional Neural Networks, Deep Learning, Intelligent Transportation Systems, Autonomous Vehicles, Image Processing, Road Safety.*

Introduction

Road safety is perhaps one of the most significant issues in contemporary urban planning and vehicular engineering. While considerable progress has been made in equipping vehicles with a range of safety features, a vast majority of road accidents are still caused by human error, often resulting from driver distraction, fatigue, and inability to see road signs. While traffic signs are intended to convey vital information related to speed limits, potential hazards, and direction, their efficacy is completely dependent on the driver's ability to respond to them in real-time. This is particularly significant in contemporary urban settings, where the sheer complexity and vehicular density necessitate a more effective and automated system for monitoring these signs.

To overcome these human limitations, the domain of Intelligent Transportation Systems (ITS) has shifted its focus towards using Computer Vision as a primary means of environmental sensing. A Traffic Sign Recognition (TSR) System acts as a co-pilot in a vehicle, using high-speed image acquisition and processing techniques to "read" the road ahead. Unlike human vision, which may be impaired in heavy rain or at night due to glare, or simply because of lack of attention, a machine can always be "on the lookout." By using specialized algorithms that can "pick out" particular shapes or colors in a cluttered background, TSR systems offer a crucial degree of redundancy in helping reduce driver cognitive overload and prevent accidents.

The implementation of the project focuses on the utilization of deep learning architectures, specifically CNNs, in the achievement of high-fidelity classification of various signage. The main objective of the project is the development of a model capable of withstanding the "noise" of the real world, including faded paint on old signs, obstruction of tree branches, or motion blur due to high-speed travel. By concentrating on the intersection of image preprocessing and optimization of the neural network, the research aims to contribute to the achievement of the overarching objective of developing a fully autonomous vehicle. It should be noted that the development of a high-accuracy TSR system is not only a technological achievement but a necessity in the development of a new generation of intelligent transportation systems with zero accidents.

Additionally, the development of a TSR system in modern vehicle systems is a significant step toward bridging the safety gap in urban transportation systems. By developing a model capable of high-speed processing and low-latency detection, the proposed model aims at ensuring the effectiveness of the system in high-density traffic situations.

1. Related work

The initial research in Traffic Sign Recognition (TSR) was based on classical computer vision techniques, in which the characteristics of color and shape of the signs were used. These initial systems used thresholding in HSI or YCbCr color spaces to segment the red, blue, or yellow colors of the signs from the background. After segmenting a possible sign, shape descriptors like the Hough transform were used to find circular, triangular, or rectangular shapes. These systems were computationally simple but sometimes encountered problems of "color bleeding" in low light and were very sensitive to environmental factors like faded colors or shadows, which sometimes resulted in false alarms.

With the progression of research in this area, emphasis has been placed on more powerful machine learning techniques that involve handcrafted feature extraction techniques. Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT) are some of the techniques that are now used to represent the structural information present on traffic signs. These features are then used to train a classifier to identify different classes of signs. This phase of research has seen a major improvement in dealing with minor occlusions and viewpoints. However, these systems were still

far from being able to generalize across datasets from different geographical locations.

The most significant breakthrough in TSR was achieved by the adoption of the new field of Deep Learning and the use of Convolutional Neural Networks (CNNs). Unlike the earlier techniques, CNNs learn features directly from the raw pixel values without the need to manually design features. New architectures of CNNs, namely ResNet and Inception, have proved their capacity to attain the accuracy of human vision on standard test sets such as the German Traffic Sign Recognition Benchmark (GTSRB). Recent research has extended the frontiers of the earlier achievements by using spatial transformer networks that allow the network to be invariant to rotations, scales, and translations.

In the current environment, research has begun to explore the optimization of these deep models in terms of real-time deployment on edge devices with low computational power. There is a growing interest in the quantization and pruning of these high-accuracy models in order to enable deployment on vehicle-embedded systems without compromising on real-time performance. Additionally, recent research in the literature has begun to explore the fusion of camera-based information with other sensors such as LiDAR and GPS in order to achieve a multi-modal understanding of the environment. This shift from standalone recognition towards a multi-modal understanding of the environment represents the current frontier in the development of fully autonomous navigation systems.

2. Research Methodology

The development of the Traffic Sign Recognition System was conducted using a systematic approach that ensures the accuracy and reliability of the system. The methodology that was followed in the development of the system includes requirement analysis, system design, image processing, and system testing.

1. Requirement Analysis

The first part of the methodology involved recognizing the limitations of the manual process of driving observation and determining the essential requirements of the automated process. Some of the important factors of the first phase of the methodology are as follows:

- To develop an automated process of recognizing and classifying road signs
- To ensure high accuracy of the process even in low light or bad weather
- To enable the process to accept video input from a webcam or in-car camera
- To reduce human error and avoid accidents

2. System Design

A modular design approach has been proposed in which the entire process would be divided into various modules:

- Video Input and Interface Module
- Image Pre-processing Module
- Feature Extraction Module
- Classification and Output Module

The Python programming language with interface libraries Tkinter or Streamlit has been used to develop an interactive graphical user interface through which a user can upload a video or start a webcam feed.

3. Image Processing and Recognition

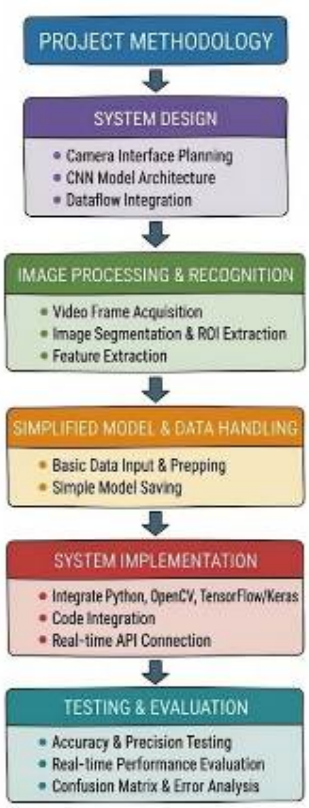
The major component in the proposed system is based on the recognition of images by computer vision. The methodology for the same is as follows:

- Capture the video frames using the camera input
- Pre-processing the image frames (resize, convert to grayscale, and remove noise)
- Identify the spatial features using the convolutional layers
- Identify the captured sign by classifying it against the trained data

The programming language used for the purpose is Python, and for image processing, the OpenCV library is used, while TensorFlow and Keras are used for developing the Convolutional Neural Network.

3.1. Overall Methodological Framework

The overall workflow of the proposed system is shown below.



3.2. Dataset Collection

The performance and reliability of the TSR system heavily depend on the quality and diversity of the dataset provided for the training of the convolutional neural network.

1. Data Source

The project aims to classify various types of regulatory, warning, and prohibitory signs. Therefore, the project uses the German Traffic Sign Recognition Benchmark (GTSRB), which is the main source of the dataset. GTSRB is an established multi-category dataset. To test the robustness of the project against various environmental factors specific to

the Nagpur region, the project uses a small set of custom images collected from the Nagpur road.

2. Data Collection Process

- Standardized Acquisition: Most of our training data was collected from high-resolution captures containing 43 unique classes of signs.
- Environmental Variability: The data set contains images collected during different times of the day (dawn, noon, dusk) and various environmental conditions (sunny sky, overcast sky, rainy sky), mimicking a practical driving situation.
- Geometric Diversity: To increase the awareness of its environment, our data set contains signs collected from various angles and distances to ensure that our system can recognize signs well before the car reaches them.

3. Preprocessing

- Resizing: Normalization to a set resolution of 32*32 pixel resolution was applied to all images to ensure a consistent input layer for our neural network.
- Grayscale Conversion: Although color is an important aspect, structural features are more reliable for our classification task. Conversion to grayscale reduces computational complexity.
- Contrast Enhancement: We used Contrast Limited Adaptive Histogram Equalization (CLAHE) to improve sign visibility in dark or "washed out" scenes.
- Normalization: Pixel intensity is normalized from a range of [0, 255] to [0, 1] to ensure stability during gradient descent training.

4. Dataset Storage

The dataset is divided into a folder structure where each folder contains a particular class of data (i.e., images), such as 'Folder 14' for 'Stop' and 'Folder 17' for 'No Entry.'

- Mapping: The image files were mapped to their respective class IDs and bounding box coordinates using a CSV file.
- Data Split: The data is divided into training and validation sets (80% and 20% respectively), ensuring that overfitting is not occurring during the development phase.

5. Dataset Size

The final dataset has over 50,000 images across 43 classes. This is sufficient data to allow a deep learning algorithm to learn features from the data, such as distinguishing between '30 km/h' and '80 km/h' signs, which are similar in appearance but require accurate feature learning to be differentiated.

6. Safety and Compliance Considerations.

Although traffic signs are public property and personal private data is not involved, utmost care was taken while obtaining any local data not to capture any car license plates or pedestrian faces. It is worth emphasizing that the project strictly adheres to the objective of improving road safety through automation without compromising public privacy.

3.3. Model Development

The core of the Traffic Sign Recognition (TSR) system can be considered the deep learning architecture used in the accurate identification of road signs from a video feed. The following sections will discuss the various technical decisions made and the logical flow used in ensuring the reliability of the model under various constraints.

1. Model Selection

The selection of an appropriate model was made in favor of a Convolutional Neural Network (CNN). This is because traditional models like SVM require manual feature engineering, while CNN can automatically learn features from raw image data. This feature enables it to perform much better in identifying complex patterns and symbols on signs.

2. Image Input and Feature Extraction

The development process begins with the ingestion of images from either a camera's current frame or from pre-recorded data. To ensure that the model is learning from structural features and not environmental features, the following feature extraction was applied:

- Spatial Hierarchy: Convolutional layers were applied in sequence to extract low-level features like edges and curves, gradually combining these features in deeper layers to form recognizable shapes like circles, triangles, and individual digits.
- Dimensionality Reduction: Max pooling layers were added to downsample feature maps, reducing computational costs and making the model more robust against translation and shifts in the sign's position.3. Neural Network Integration

The CNN engine processes the pre-processed image patches and maps them to one of the 43 classes defined in the GTSRB dataset. The internal logic follows a sequential flow:

- The final feature map is "flattened" into a one-dimensional vector.
- This vector passes through **Dense (Fully Connected) layers** that act as a high-level reasoning engine.

4. Classification and Recognition Logic

A rigorous logic gate is used to ensure that only high-confidence predictions are made:

- Frame Capture: The system captures a frame every 33ms to achieve 30 FPS.
- Pattern Matching: The features are matched against the weights learned during the training phase.
- Identity Verification: The confidence score is checked to ensure that it is above a certain threshold (95% in this case), and then accepted.
- Result Logging: The result is then logged on to the dashboard interface and then logged into the database if required.

5. Model Optimization

To ensure the system remained viable in a real-time environment, a variety of optimization techniques were utilized:

- **Learning Rate Annealing**: A dynamic scheduler was implemented to gradually reduce the learning rate as the model neared convergence in order to avoid "overshooting" the optimal weights.
- **Inference Speed**: The model was optimized to classify in under 10ms, ensuring minimal delay between sign identification and driver notification.

6. System Integration

The completed model was incorporated into a single software environment:

- **Python**: Used as the primary language for backend logic and model execution.
- **OpenCV**: Managed the acquisition of video in a real-time environment.
- **Local Storage**: Used to store the completed model weights in a format such as .h5 or .keras, ensuring instantaneous loading without the need for retraining.

3.4. Model Training

The training phase for the system, referred to as the Traffic Sign Recognition (TSR) system, was intended to fill the gap between the theory of deep learning and its practical application. Unlike other speech recognition systems, the training phase for the system was intended to teach the Convolutional Neural Network (CNN) to recognize 43 different categories of signs under different environmental conditions.

1. Training Environment and Framework

To reduce the computational burden of the training process of a deep neural network, the process was conducted using the TensorFlow and Keras frameworks. A GPU environment was used to accelerate the back-propagation process. This enabled multiple iterations of the network (epochs), ensuring the network learned not only the obvious features of a sign (such as a red octagon) but also the finer details of a sign (such as the difference between "60" and "80" on a speed sign).

2. Data Augmentation (Synthetic Training)

As driving in the real world may not always be perfect, the training process was "stressed" in an attempt to prepare the network for the unexpected. During this phase of the process:

- Image Rotation: Signs were slightly tilted to mimic a camera not perfectly aligned with the sign.
- Zooming and Shifting: The position of the sign in the image was varied to mimic a vehicle not always at a perfect angle to the sign.
- Brightness Adjustments: Images were made darker or lighter to prepare the network for glare in nighttime and mid-day conditions.

3. Feature Preparation

Prior to the actual training of the system, the raw image data was prepared as follows:

- Resolution Standardization: The resolution of the images was standardized to 32*32 pixels.
- Color Space Transformation: We tested both the RGB and Grayscale color spaces to find the perfect balance between speed and accuracy.

4. Validation and Checkpoint Mechanism

In order to ensure that the system was indeed learning and not just "overfitting," the following validation mechanisms were put in place:

- 80-20 Split: The system was trained on 80% of the data and the remaining 20% was kept aside for testing purposes.
- Loss Monitoring: The system monitored its "Categorical Cross-Entropy" to determine how far off the predictions were from the actual outcomes.
- Model Checkpoints: The "best" version of the model was saved so that if the training became unstable, the system did not have to start from scratch.

3.5. Model Evaluation Strategy

The evaluation of the Traffic Sign Recognition (TSR) system has been carried out to assess its accuracy, reliability, response time, and robustness under various conditions. Since image processing and deep learning-based classification are used in this system, its evaluation is based on the accuracy of traffic sign recognition and efficiency of the overall detection process.

1. Evaluation Objectives

The objectives of the evaluation of the model are:

- To assess the accuracy of traffic sign classification
- To assess the reliability of the overall detection process
- To assess the response time (latency) of the system
- To assess the overall response of the system under various environmental conditions

2. Testing Environment

The system is tested with benchmark data as well as live-action videos of roads.

In clear daylight, overcast, and low-light environments, image recognition is performed in order to test the system's response in varying environments. The distance of the sign from the camera is also varied as part of the testing procedure.

3. Performance Metrics

The following metrics were employed to measure the performance of the system:

a) Accuracy of Recognition

The accuracy of the system was computed as text{Accuracy} = frac{text{Number of Correct Classifications}}{text{Total Number of Test Images}} times 100

This metric was employed to measure the number of times the system was able to correctly identify and classify the traffic sign correctly.

b) False Positive Rate (FPR)

The percentage of times the system incorrectly classified an image of an object that was not a traffic sign, such as a building or tree, as being a traffic sign.

c) False Negative Rate (FNR)

The percentage of times the system failed to recognize an actual traffic sign that was present in the frame of the video input.

d) Response Time (Latency)

The time it took between the input of the video frame and the display of the recognized label on the interface. Real-time processing was considered to be within the acceptable time frame of 15-30 milliseconds to maintain a high frame rate.

4. Test Procedure

- Training the CNN model for the 43 classes in the GTSRB dataset.
- Performing multiple tests for detecting objects using different video clips.

- Variations in test cases:
  + Change in camera angle.
  + Motion blur effect.
  + Change in brightness levels.
- Recorded instances for correct classification and incorrect classification.

5. Result Analysis

The system was found to be highly accurate in clear conditions, and in moderate rain and overcast conditions, the system was found to be stable. False positives were found to be low, considering the strict threshold conditions. False negatives were also kept low by using a robust image processing pipeline and high-quality images.

6. System Reliability

The reliability of the system was tested through repeated trials of the system. The system was able to recognize the combination of various hardware setups. The reliability of the combination of the Python, OpenCV, and TensorFlow modules was also quite good. The use of the localized processing method ensured low latency and high-speed feedback, which is critical for driver safety systems.

3.6. Deployment Methodology

The deployment of the Traffic Sign Recognition (TSR) system has been carried out with the objective of ensuring the smooth operation and integration of the computer vision model.

1. Deployment Environment

The system has been designed to be deployable on standard computing devices such as laptops or embedded systems such as Raspberry Pi or Jetson Nano without any requirement for industrial-grade hardware.

The minimum deployment requirements that the system demands are:

- A computer or embedded board with a Python 3.x environment.
- A high-speed camera or webcam to stream live images from the road.
- At least 8GB RAM to process images in real time.

The system is fully functional without any requirement for an active internet connection because the CNN weights are stored locally.

2. Software Configuration

The following activities were carried out before deploying the Traffic Sign Recognition system:

- Installation of Python SDK along with environment managers.
- Installation and configuration of OpenCV for camera handling and image processing.
- Configuration of logging to store data about signs detected by the system.

All dependencies were verified to ensure that image processing could communicate instantaneously with the deep learning classifier.

3. System Integration

The deployment process involved the integration of the following:

- Python for the main application logic and control flow.
- OpenCV for the acquisition of the real-time video stream.
- TensorFlow/Keras for the execution of the sign classification model.

The system was set up in a manner that would enable the smooth flow of data from the camera to the processing unit with minimal buffer time.

4. Installation Process

The development of the project was completed in a manner suitable for a standalone desktop or embedded application. The deployment process of the project was as follows:

- Transfer of project files and the trained model.
- Calibration of the camera position in a manner that would provide a clear view of the road.
- Verification of the software path of the model weights.
- Initial diagnostic checks of the frame rate stability.

The parameters of the system were adjusted in a manner suitable for the quality of the camera used in the deployment process.

5. User Setup and Calibration

A calibration procedure was also conducted after the deployment of the system. This was important in ensuring that the camera was correctly identifying the signs at various distances. This was an important procedure in setting the system for the provision of driver assistance in real time.

6. Testing After Deployment

The system was tested after the deployment to ascertain that it was correctly performing the following functions:

- Signs were being correctly identified regardless of the type of road being driven on.
- The visual dashboard was correctly displaying the signs in real time.
- The system was within the acceptable latency limits while operating at road speeds.
- The hardware was correctly operating for an extended period of time.

The system was considered deployed correctly if it was able to correctly identify the signs and alert the driver without lag.

4. Conclusion and Future work

The proposed system has been developed considering the drawbacks of manual observation of road signs by a driver, which may result in human errors, fatigue while driving, and low visibility during adverse weather conditions. The proposed system has been developed considering the advantages of deep learning computer vision techniques that provide a reliable solution for enhancing situational awareness for a driver.

The proposed system has been successfully able to incorporate Python, OpenCV, TensorFlow, and Keras technologies to develop a user-friendly application for a vehicle environment. The localized approach of the proposed system has been able to provide a high-speed solution that is independent of internet connectivity, which is a critical

requirement for a vehicle environment. The results of the proposed system have shown a high classification accuracy for different types of signs.

The project has been able to prove the feasibility of using Convolutional Neural Networks (CNN) as a robust alternative to the traditional sensor-based/manual observation of the driver. The project has been able to make a significant contribution to the field of Intelligent Transportation Systems (ITS) by providing a smarter, automated, and proactive way of road safety management.

Despite the fact that the system is performing well, it is important to consider the following improvements to enhance the efficiency of the system:

Cloud-Based Data Synchronization: The system can be enhanced to sync the identified traffic data to a cloud platform, which can allow real-time mapping of road conditions and sign health for city authorities.

Mobile and Embedded Integration: The system can be further enhanced to work efficiently on low-power mobile applications or embedded devices like NVIDIA's Jetson platform, which can allow it to work for older models of vehicles.

Enhanced Night Vision Capabilities: The system can be enhanced to utilize advanced infrared image processing algorithms that can ensure a high rate of recognition even during midnight or during a foggy drive.

Driver Behavior Analytics: The system can be enhanced to allow a reporting module that can help analyze driver behavior according to sign recognition, which can ensure safer driving habits.

These features can be incorporated into the system to make it a more complete, efficient, and intelligent system for the next generation of smart vehicles.

REFERENCE

[1] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German Traffic Sign Recognition Benchmark: A Multi-class Classification Competition," Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), 2011, pp. 1453–1460.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, 2015, pp. 436–444.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems (NIPS), 2012, pp. 1097–1105.

[4] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[5] F. Chollet et al., "Keras: Deep Learning for Humans," GitHub, Available:

[6] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. Available:

[7] S. S. Shwartz and S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014.