# Road Sign Classification Using CNN

## 1. Project Overview

This project aims to build a deep learning model using Convolutional Neural Networks (CNNs) to classify road signs into 30 distinct categories. The ultimate goal is to automate traffic sign recognition, which is essential for road safety and autonomous vehicle navigation. By utilizing CNNs, the system can accurately identify various traffic signs, enabling real-time decision-making in smart transportation systems.

## 2. Objectives

* Automate the classification of road signs using CNNs.
* Enhance model scalability, accuracy, and deployment capability.
* Create a web application for real-time user interaction.
* Evaluate the impact of preprocessing and model tuning on performance.

## 3. Dataset Description

The dataset includes 743 labeled images categorized into 30 classes, such as speed limits, directional signs, and warnings. These images are annotated using an accompanying Excel file. The dataset is fairly small, which presents challenges in achieving high generalization accuracy.

## 4. Data Preprocessing

* To prepare the images for the CNN model:
* All images were resized to 160x160 pixels.
* Keras’s ImageDataGenerator was used to load and label images from directories.
* The dataset was split into training (521), validation (222), and test (285) sets.
* TensorFlow’s AUTOTUNE was implemented to prefetch batches and improve training efficiency.

## 5. CNN Model Architecture

* The CNN model consists of the following layers:
* Convolutional Layers: Extract features from images using ReLU activation.
* MaxPooling Layers: Reduce dimensionality to retain important features.
* Dropout Layers: Prevent overfitting by deactivating random neurons.
* Batch Normalization: Stabilize and speed up training.
* Dense Layers: Map features to output categories.
* Softmax Output Layer: Classify images into 30 categories based on probabilities.

## 6. Model Training Process

* The model was trained using TensorFlow/Keras for 40 epochs. Key training parameters:
* Optimizer: Adam (learning rate = 0.001)
* Loss Function: Sparse Categorical Crossentropy
* Evaluation Metrics: Accuracy and Loss
* Training and validation accuracy/loss were monitored to detect overfitting and assess learning progression.

## 7. Evaluation and Results

* The final results were:
* Training Accuracy: 100%
* Validation Accuracy: 91%
* Test Accuracy: 63%  
  The model performed well on training and validation sets but showed reduced performance on the test set, indicating slight overfitting. A confusion matrix and performance curves helped in detailed analysis.

## 8. Model Deployment

* To demonstrate real-time usability:
* The model was saved in .h5 format.
* A Streamlit-based web application was developed for image upload and prediction.
* Ngrok was used to generate a public URL for remote access to the app.

## 9. Key Findings and Insights

* CNNs are effective in extracting features from image data.
* A small dataset limited generalization capabilities.
* Similar-looking signs often led to misclassification.
* Preprocessing had a significant impact on model accuracy.
* Lack of data augmentation restricted model robustness.
* Training stabilized after approximately 20 epochs.

## 10. Conclusion and Future Work

The project successfully demonstrated the application of CNNs in road sign recognition. A functional model was developed and deployed via a web interface. However, there is room for further improvement.

Future enhancements include:

* Implementing data augmentation techniques.
* Expanding the dataset for better diversity and generalization.
* Experimenting with advanced CNN architectures like ResNet, EfficientNet, or InceptionNet.
* Applying transfer learning using pretrained models such as VGG or MobileNet.
* Performing hyperparameter tuning to optimize performance.