Traffic Sign Recognition Based on HOG Feature and SVM

AUTHORS: JIALIN TANG, QINGLANG SU, CHENYU LIN*, YANGJUN WEN, BINGHUA SU, JUQING YANG

VENUE: EITCE '20: PROCEEDINGS OF THE 2020 4TH INTERNATIONAL CONFERENCE ON ELECTRONIC INFORMATION TECHNOLOGY AND COMPUTER ENGINEERING

Presented by:-
Tejas Deshmukh (230150027)
Aryan Gupta (230150003)

Introduction

Problem: Reliable identification of road traffic signs

Relevant applications

Autonomous Driving

Image-based traffic sign recognition is essential for safe and reliable assisted driving and autonomous driving vehicles.

Street-View Integration

Detecting signs from images lets us integrate them into map/street-view platforms (e.g., Google Maps) to improve navigation and situational awareness.

Introduction – Pipeline Flow

1. **Image enhancement**
   - Boost contrast and normalize lighting (RGB→YUV + adaptive histogram equalization) for clearer sign appearance.
2. **Color segmentation**
   - Convert to HSV and threshold hue ranges to isolate red/blue/yellow candidate regions.
3. **Sign localization**
   - Find sign candidates with Hough circle detection and contour/polygon analysis (tri/quad detection).
4. **HOG Feature extraction**
   - Compute gradient-orientation histograms on normalized patches (e.g., 64×64, 8×8 cells) to form robust descriptors.
5. **SVM classification**
   - Train an SVM to classify HOG vectors into traffic-sign categories.

Prior Work

Extensive research has been conducted on **traffic sign detection** using **Deep Convolutional Neural Networks (CNNs)** and **Support Vector Machines (SVMs)**.

Greenhalgh et al. (2012) — German Traffic Sign Detection

Proposed a two-step method:

1. **MSER (Maximally Stable Extremal Regions)**
   Extracted **candidate areas** of traffic signs by identifying stable regions in varying illumination and contrast.

2. **SVM Classifier with HOG Features**
   Trained on features extracted from these candidate regions for precise classification.

This approach achieved **high accuracy** on the **GTSRB (German Traffic Sign Recognition Benchmark) dataset**, proving the effectiveness of combining **MSER + HOG + SVM**.

Prior Work

Rise of Deep Learning

With the rise of **Deep Learning**, researchers adopted **Deep Neural Networks (DNNs)** for **end-to-end recognition**.

DNN-based models automatically learn both **feature extraction and classification**, eliminating manual feature design.

Deep CNN Architectures

German scientists trained **deep CNN architectures** for traffic sign recognition, achieving:

- **Accuracy surpassing average human recognition levels**.
- Improved robustness under **lighting, rotation, and occlusion** conditions.

These advancements laid the foundation for **modern intelligent driving systems**.

But what is different?

The difference lies in the pre-computation part of the pipeline.

01 IMAGE ENHANCEMENT

Transform camera image to YUV using equations below and then applying adaptive histogram equalisation:

Y = 0.299R + 0.587G + 0.114B

U = −0.147R − 0.289G + 0.436B

V = 0.615R − 0.515G − 0.100B

02 COLOR SEGMENTATION

Transform into HSV values and segment based on the commonly used colours in traffic signs(red, yellow, blue).

$$V = \frac{(R+G+B)}{\sqrt{3}} \qquad H = \begin{cases} \theta, B \le G \\ 360° - \theta, B \ge G \end{cases}$$

$$S = 1 - \frac{\sqrt{3}}{V}\min(\mathrm{R}, \mathrm{G}, \mathrm{B}) \qquad \theta = \arccos\left[\frac{(R-G)+(R-B)}{2\sqrt{(R-G)^2+(R-B)(G-B)}}\right]$$

03 SIGN LOCALISATION

- Circle detection: Use the Hough transform to map edge points into parameter space and detect circles robustly
- Triangle/rectangle detection: Extract contours from the binary image, apply polygonal fitting to remove noise, then classify shapes by vertex count (3 → triangle, 4 → rectangle).

HOG Extraction

HOG is a feature descriptor commonly used for target detection in computer vision and image processing. It can well represent the outline and edge information of traffic signs.

The gradient of the traffic sign image in the horizontal and vertical directions of the pixel points (x, y) is:-

$$G_x(x,y) = I(x+1,y) - I(x-1,y)$$
$$G_y(x,y) = I(x,y+1) - I(x,y-1)$$

Horizontal and Vertical Gradients

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

Magitude

$$a(x,y) = \arctan\left(\frac{G_y(x,y)}{G_x(x,y)}\right)$$

Direction

- Divide the image into several cells
- Collect the gradient direction histogram of each pixel point in each cell
- Cells are blocked and normalized
- Feature vectors in all the blocks are concatenated

$$N = \left(\frac{I_w}{C_w} - 1\right) \times \left(\frac{I_h}{C_h} - 1\right) \times B \times H$$

- N = Dimension of HOG feature space
- Iw and Ih = dimensions of image
- Cw and Ch = dimensions of cell
- B = number of cell in each block

SVM Training, Evaluation & Conclusion

SVM Formulation

Map HOG descriptors to a high-dimensional space via kernel methods; training minimises $||\omega||^2/2 + C\sum\varepsilon_i$ to obtain maximum-margin hyperplane.

Hyperparameters & Kernel

Select penalty **C (from 1 to 200)** and **kernel (Linear, Polynomial, RBF, Sigmoid)** to produce optimal trade-off between margin and misclassification.

Validation & Results

Evaluate on the test set and additional sample subsets — model demonstrates robustness under varying illumination and partial occlusion; failure modes include very small targets, severe reflections, and strong motion blur.

Conclusion

HOG + SVM provides a computationally efficient and accurate baseline for traffic sign recognition; remaining challenges motivate future work on small-object detection and blur/reflection mitigation.

GTSRB — German Traffic Sign Recognition Benchmark

1 **Overview**

Real-world traffic-sign dataset for classification and detection.

2 **Scale & Classes**

≈ **50k images**, **43 classes** (multi-condition imaging).

3 **Image & Format**

PPM images, variable sizes (~15×15 to 250×250); each image contains one sign.

4 **Annotations**

CSV files per class with image size, bounding-box (ROI) coordinates and ClassId.

What We Will Demonstrate.

1. **Preprocessing & Enhancement**: Implement RGB→YUV conversion and adaptive histogram equalisation on the Y channel to correct blur/brightness.
2. **Colour Segmentation:** HSV/H-based thresholding to extract red/blue/yellow candidate regions.
3. **Localisation & Shape Analysis(Using OpenCV Python Library):** Detect circular signs via Hough transform and polygonal signs via contour + polygon approximation.
4. **Feature Extraction:** Compute HOG descriptors on resized sign patches (64×64) to form feature vectors for classification.
5. **Classifier Implementation**: Train an SVM on a subset of the GTSRB dataset, but produce the final model using the scikit-learn Python library using the whole dataset.
6. **Evaluation & Analysis**: Report accuracy, per-class recall/precision and confusion matrix using scikit-learn Python library.

Preprocessing and Enhancement (YUV + CLAHE)

- **Impact:** Significantly enhanced feature quality by normalizing brightness and shadow variations across the entire dataset.

Target Isolation: Color Segmentation (HSV)

Target Colors: Segment based on the commonly used colours in traffic signs(red, yellow, blue)

Output: Generated a binary mask that precisely outlines the sign area, drastically reducing the search space for the subsequent shape detection step.

Localization & Shape Analysis

- **Circular Signs:** Isolate round signs (e.g., speed limits, prohibitions) and crop them for feature extraction.
- **Hough Transform:** Used the Hough Gradient Method (HoughCircles) on the grayscale-converted image (derived from the color mask).
- **Polygonal Signs:** Identify non-circular shapes, primarily triangles (Yield, Dangerous Curves) and rectangles (Priority Road).
- **Candidate Generation:** Based on the identified polygonal vertices (e.g., 3 for a triangle, 4 for a square), a bounding box was generated.

Localization & Shape Analysis Challenges

- Canny edge detection, recommended by Sir, gave poor results for noise removal.
- Hence, we stuck to the basic Hough Transform according to the research paper provided.

HOG Extraction

Using skimage library

- **Feature matrix shape: (39209, 1764)**
- **Time elapsed to extract features: 90.968 s (~1.5 mins)**

Our Implementation

- **Feature matrix shape: (39209, 1764)**
- **Time elapsed to extract features: 855.673 s (~14.25 mins)**

Classification: SVM on HOG Features

- **Support Vector Machine (SVM):** Selected an SVC (Support Vector Classifier) for multi-class classification
- **Kernel Selection via CV:** Implemented 5-fold Cross-Validation on the training data to determine the best kernel function.
- **Kernel Comparison:** Evaluated Linear, RBF (Radial Basis Function), and Sigmoid kernels, and selected the kernel that gave the highest mean cross-validation accuracy.
- **Final Model:** The chosen kernel (e.g., RBF) was then used to train the final SVM model on the entirety of the training data.

Kernel	Mean Accuracy	Std Dev	Time (s)
linear	0.8824	0.00384	771.20
rbf	0.9440	0.00383	1632.62
sigmoid	0.5263	0.00926	766.60

Evaluation and Performance Analysis

- **Test Accuracy:** Achieved a global accuracy of 0.95218 on the official GTSRB Data Set.

HOG Implementation Comparison: Efficiency vs. Performance

Metric	Library HOG (Reference)	Our Custom HOG (Implementation)
Feature Extraction Time (s)	90.97	855.67
SVM Training Time (s)	289.82	358.86
Test Accuracy	94.22%	95.22%
Misclassified Samples	453	375

Confusion Matrix of Our Custom HOG(Implementation)

Confusion Matrix of Our Library HOG(Reference)