Traffic sign recognition using local directional histogram of oriented gradients

Prestone Simiyu[1]
Raphael Angulu[2]
Daniel Otanga[3]

[1]sprestone@mmust.ac.ke
[2]rangulu@mmust.ac.ke
[3]otanga@mmust.ac.ke

[123]Masinde Muliro University of Science and Technology, Kenya

https://doi.org/10.51867/ajernet.6.3.54

ABSTRACT

Histogram of Oriented Gradients (HOG) describes an image gradient by calculating vertical and horizontal gradient magnitudes and directions. HOG uses a one-dimensional (1D) centered derivative mask [−1, 0, +1] for horizontal gradient and its rotations at 90º for vertical gradient. This technique only considers four neighboring pixels while calculating image gradient at a particular pixel. Every pixel in an image carries subtle information and therefore all pixels should be considered when deriving image gradient. Therefore, given a pixel p_i, all its $N = 2^{(d+1)}$ neighbors should be considered when calculating the gradient at distance d from p_i. This paper proposes Local Directional Histogram of Oriented Gradients (LD-HOG), which, given pixel p_i, it calculates the gradient at distance d = 1 from p_i by considering all the eight neighbors of p_i. The proposed operator calculates the image gradient at 0º, 45º, 90º and 135^0. These image gradients are used to generate two HOG histograms. Maximum pooling techniques were applied to combine the two histograms. Experimental results on the German traffic sign detection benchmark (GTSDB) dataset show that LD-HOG (average precision = 0.90, average recall = 0.90 and average F1-score =0.90) out performs HOG (average precision = 0.84, average recall = 0.82 and average F1-score = 0.83) in traffic sign recognition. The averages of the two extractors (HOG and LD-HOG) were calculated from experimental results after applying Support Vector Machine (SVM), Random Forest (RF) and Decision Tree (DT) machine learning classifiers. Stratified K-Fold Cross-Validation was done on the proposed LD-HOG using SVM, RF and DT. Validation results show that SVM performed better with 99 percent, followed by RF with 96 percent. DT was had 76 percent.

Keywords: Histogram of Oriented Gradients, Local Directional Pattern, Local Directional Histogram of Oriented Gradients, Traffic Sign Recognition

I. INTRODUCTION

Self-driving vehicles are becoming increasingly common nowadays as a result of improvements in computer vision and auto-mobile technologies. The ability of these vehicles maneuvering safely on busy roads depends heavily on their ability to rapidly and accurately detect and recognize traffic signs, making traffic sign recognition an important element of self-driving vehicles. Traffic sign recognition techniques help self-driving auto-mobiles and drivers to detect, recognize, and understand various road signs like warning signs, prohibitory signs and reservation signs. Traffic Sign Recognition (TSR) techniques improve safety of all road users. Road safety research is attracting enormous attention around the globe because it is indispensable in protecting human life (World Health Organization, 2018).

According to a report by the World Health Organization (WHO, 2018), over 1.35 million people lose their lives on road accidents, annually, and about 20 million to 50 million are disabled or injured. Most of these road accidents are caused by failure to adhere to road rules, driver distraction and failure in detecting traffic signs. To improve safety on the road, various initiatives have been taken among them design and development of Advanced Driving Assistance Systems (ADAS) that automate tasks like traffic sign recognition and lane departure warning systems. Traffic Sign Recognition (TSR) plays a critical role in the robustness and effectiveness of ADAS. Traffic signs have subtle discriminating attributes based on which they are localized, detected and recognized. Based on the shape and color, traffic signs can be categorized into three classes: i). Caution – they are triangular in shape, with red edge; ii). Mandatory – circular in shape, red border/edge, e.g. speed limit signs; iii). Reservation– rectangular in shape with a blue background

Regulatory sign

Warning sign

Informational sign

Figure 1
Examples of Traffic Sign Categories

1.1 Statement of the Problem

TSR techniques use visual information like colour and shape to detect and recognize traffic signs. A number of TSR techniques have been developed Saouli et al. (2018); Jayaprakasha and KeziSelvaVijilab (2019); Yang et al. (2016); Hechri et al. (2014). Nevertheless, performance of computer vision techniques in TSR are adversely affected by factors such as variations in illumination, occlusion, adverse weather conditions, road sign deformation, colour deterioration due to exposure to UV rays among many other factors An et al. (2024); Khalifa et al. (2024). This paper proposes a new TSR technique that includes three stages; i). Region of interest (ROI) detection and extraction ii). Traffic sign group classification iii). Recognition of the traffic sign. A new feature extraction technique Local Directional Histogram of Oriented Gradient (LD-HOG), an improvement of Histogram of Oriented Gradient (HOG) is proposed and used.

1.2 Research Objective

To develop a robust machine learning model for traffic sign recognition

II. LITERATURE REVIEW

2.1 Theoretical Review

Many authors have approached TSR using different approaches. According to Hechri and Mtibaa (2020), TSR tasks can be grouped into three: i). Color and shape based ii). Traditional machine learning approach iii). Deep learning. An important part of TSR is classification that can be done using different ways including machine learning. A lot of research has been on image classification and various models published over the years. Nonetheless, different researches and developers have attempted to solve issues related to TSR (Image classification) including occlusion, overfitting, class imbalance, and computational cost.

2.2 Empirical Review

Hechri and Mtibaa (2020) applied two steps for TSR: first, classifying signs to either triangle or circle using HOG and Support Vector Machine (SVM), and step two application of CNN for image recognition. Their approach performed better than many models, however it is costly computationally. Additionally, the reservation signs were not taken care of by this approach. Another hybrid method was proposed by Asha et al. (2022) Decision Tree (DT) and Random Forest (RF) classifiers were used on features that were extracted using HOG and Gray Level Co-occurrence Matrix (GLCM) shape and texture, respectively. Khalifa et al. (2024) tackled the challenge of occlusion by applying DT for color-based and both DT and RF for texture-based classification. Notably, their approach strengthens feature extraction and representation; however, the model could struggle to handle complex, high-variance traffic environments. In their paper, Li et al. (2022) notes that traffic signs have descriptive and discriminative features easily discovered at a lower dimension. They propose a model that applies a simple feature extraction technique (HOG with feature reduction) with a lower time cost and performs better in real-world scenarios. Their model incorporates Principal Component Analysis (PCA) in the dimension reduction of features on HOG. The reduction is necessary because higher dimensions increase the computational cost. The paper notes that applying feature reduction increases the model's performance in Precision, Recall, F1-measure, and Accuracy. Test time for the proposed model was reduced significantly.

LeCun et al. (1998) introduced Convolutional Neural Networks (CNNs) IN 1998 and has become a staple in image classification field. CNN have the ability to automatically extract and learn features from images. Additionally, they perform well in recognition of multiclass images. However, CNNs require high computational cost especially when working on large dataset. CNNs also struggle when exposed to imbalanced datasets Zhu and Yan (2022). Buda et al. (2018) suggest application of dropout layers to address the issue of overfitting. However, CNN still require a lot of computational cost when exposed to bigger image dataset. Overfitting problem has been approached using ensemble model by combining several decision trees to form a random forest. The output of a model applying random forest and HOG feature extractor presented by Lee et al. (2024) had an improved accuracy compared to SVMs and CNN. Nonetheless, the computational cost was too high mainly because of the complexity of the decision making of the forest.

The model performed better on imbalanced data but struggled on high-dimensional image dataset Approximate Nearest Neighbor (ANN) model proposed by Halder et al. (2024) is an improvement of K-Nearest Neighbors (KNN). Despite KNN having a simple structure and easily implemented. It also struggled with high dimensional data. Both KNN and ANN has a higher computational cost during inference, as it computes distances to all other data points. Notably ANN reduced the computational power but the computational power is still high.

Shi et al. (2024) notes that Recurrent Neural Networks (RNNs) performs well on temporal data. RNN applies sequence-based tasks mainly Long Short-Term Memory (LSTM) to image classification by treating image data as a sequential task. The main challenge with RNN is limited scalability and high training cost Shi et al. (2024); Razavi et al. (2024); Jain et al. (2023). With promised accuracy and precision RNN classification in not done at the pixel level. Introduced by Chen et al. (2025), Fully Convolutional Networks (FCNs) tackles the pixel-level classification problems including semantic segmentation. Experiment results shows that FCNs perform better than CNNs in image classification. However, similar to CNNs, ANN and RNN, FCN require high computational cost when handling large dataset. Additionally, their performance can degrade on noisy data. Another major challenge in image classification is vanishing gradient problem. This problem grows with deeper network.

ResNet was introduced by Razavi et al. (2024) to handle the vanishing gradient problem in deep networks. It introduced skip connection allowing gradient to easily flow through deep architectures. ResNet-based models have achieved significant success in classification tasks. However, they still require substantial computational resources for training, and achieving optimal accuracy on highly diverse datasets can be challenging. DenseNet is an improvement on ResNet that uses fewer parameters and more effective feature reuse by feed-forwarding connections between each layer and every other layer. EfficientNet scales up the network's depth, width, and resolution more effectively by using a compound scaling technique. It greatly decreases the number of parameters while providing excellent performance. However, the model requires a careful balancing of scaling parameters, and adjusting this equilibrium can be computationally costly Zhu and Newsam (2017); Zhang et al. (2019). Table 1 summaries the performance of some common image classification model.

Table 1

Summary of Machine Learning Approaches to Traffic Sign Recognition

Author	**Algorithm**	**Dataset**	**Accuracy Percentage**
Kerim and Efe (2021) (Kerim and Efe, 2021)	ANN	GTSRB	95
Soni et al., (2019)	LBP, HOG, PCA, SVM	TSRD (Chinese)	84.44
(Namyang and Phimoltares, 2020)	HOG, CLD, SVM, Random Forest	Self-collected (Thai)	93.98
Li et al., (2022) (Li et al., 2022)	Color Histogram, HOG, PCA	GTSRB	99.99
Madani and Yusof (2018)	Border Color, Shape, Pictogram, SVM	GTSRB	98.23
Sapijaszko et al., (2019) (Sapijaszko et al., 2019)	DWT, DCT, MLP	BTSD, GTSRB, TSRD	96,95.7,94.9
Aziz and Youssef (2018)	HOG, CLBP, Gabor, ELM	GTSRB, BTSC	99.10, 98.30
Weng and Chiu (2018)	CCL, HOG, SVM	GTSRB	90.85
Wang (2022)	LR, MLP, SVM	GTSRB	97.5,98.88,95.51

Even though deep learning models like CNNs, ResNet, and EfficientNet have transformed image classification, they have some limitations. For example, they typically need a lot of labeled data to train, which may not be available in some real-world applications. Additionally, computationally costly models like DenseNet and ResNet might not be appropriate for deployment on edge devices or in real-time applications where memory and speed are critical factors. Finally, most of these models lack interpretability, especially in CNNs and deep learning-based architectures, making it difficult to understand how decisions are made, which is crucial in sensitive fields like healthcare and autonomous driving. Models that rely on handcrafted features (e.g., SVMs with HOG) tend to perform worse than end-to-end deep learning approaches, especially on complex datasets where feature extraction may not capture all necessary aspects of the data.

Machine learning models for image classification have advanced significantly, more models that are computationally effective, able to generalize to a variety of noisy data, and interpretable for high-stakes applications are still required. Some of these difficulties might be lessened by combining several strategies and semi-supervised learning.

III. METHODOLOGY

The research strategy was simulation and experimental. Simulation and experiments was done in three stages; preprocessing, feature extraction and classification. A white bed was set with the following conditions to enable fair testing and comparison of LD-HOG, HOG and LDP. Python 3.8 with the OpenCV, NumPy, and scikit-learn libraries, Windows 10, Processor: Intel Core i7 (10th Gen), RAM: 16 GB, GPU: NVIDIA GTX 1650 and IDE: VS Code. The Germany Traffic Sign Recognition Benchmark (GTSRB) dataset split at 80 percent and 20 percent was used in training and testing of the three feature extractors respectively. Experiments were carried out to determine which preprocessing technique would be selected for creation of the model. The preprocessing techniques experimented on were interpolation, cropping, filtering, normalization and segmentation. After preprocessing, features were extracted using three descriptors; Histogram of Oriented Gradient, Local Directional Pattern and Local Directional Histogram of Oriented Gradient. Lastly, three classifiers were used to test the performance of the three descriptors namely; support vector machine, random forest and decision tree.

3.1 Local Feature Descriptors

Local Binary Patterns: Texture features have been widely used in image prediction techniques Sedaghatjoo et al. (2024); Panis et al. (2016). Local Binary Patterns (LBP) is an appearance (texture) descriptor able to detect micro-structure patterns like corners/edges, spots, flat regions and lines on human skin Ojala et al. (1996). Local binary patterns is a common texture descriptor used for image classification task, TSR being one of them. Figure **2** shows LBP calculation for a 3 × 3 image region.

Figure 2
Local Binary Pattern

The Local Binary Pattern (LBP) code from figure 2 can be represented using equation 1

$$LBP = 11001011_2 = 203_{10} \quad \text{.....(1)}$$

Where base 2 is the 8-bit binary equivalent of the image while base 10 is the decimal value.
The binary 8-bit representation of the image is derived using equation 2 below

$$LBP_{p,r}(x_c, y_c) = \sum_{n=0}^{N-1} 2^n s(g_n - g_c) \quad \text{.....(2)}$$

And thresholding function τ is defined by equation 3

$$\tau(x) = \begin{cases} 1, & if\ x \geq 0 \\ 0, & otherwise \end{cases} \quad \text{.....(3)}$$

Where $N = p$ is number of neighbouring pixels, r is the distance of neighboring pixels from the central pixel, g_c is Gray-value of central pixel, g_n for $n = 0,1,2, \ldots N-1$ correspond to Gray value of neighbouring pixel on circular symmetric neighborhood of distance $R > 0$ and the function $\tau(x)$ is a threshold function that generates a binary bit for a particular pixel. Joining the 8 bits leads to a binary number. The binary number is converted to decimal and assigned to central pixel as its LBP code. Histogram of LBP encoded image is used to represent micro-pattern structures in an image. This histogram is defined by equation 4

$$H_i = \sum_{x,y} I(f(x,y) = i), i = 0,1,2 \ldots, 2^p - 1 \quad \text{.....(4)}$$

Where p is the number of texture patterns that LBP operator can encode and equation 5 shows how to calculate I

$$I(a) = \begin{cases} 1, if\ a\ is\ TRUE \\ 0\ otherwise \end{cases} \quad \text{.....(5)}$$

Ojala et al. (2002) found that when $p = 8\ and\ r = 1$, about 90% of all texture patterns consist of uniform patterns. A pattern with utmost 2 bitwise transition from 1 to 0 and vice versa is referred to as uniform pattern. For instance, $10000000, 00000000, 0100000\ and\ 11111011$ patterns are uniform while $11010000, 10100101, and\ 10101101$ are not uniform patterns Ojala et al. (1996). Uniform patterns can denote micro-structures like spot, edge, line or flat region. Original LBP was limited in encoding dominant texture features that span a larger region. The descriptor was extended so that it captures features using different neighborhood with different radii Ojala et al. (2002). Pixels evenly distributed around a central pixel along a circular circumference centered at central reference pixel define neighborhood region. Bilinear interpolation of points that do not fall within the pixels is done to allow any radii and any number of sampling pixels. For extraction of rotational invariant features, LBP binary code is circularly rotated until its' minimum value is gotten Maenpaa and Pietikainen (2005). Extended LBP descriptor could encode more features in an image; however, spatial information of the extracted features could not be preserved. Final histogram is found by joining histograms of each cell.

Local Directional Parten

Local binary pattern is sensitive to changes in illumination and image noise. Local Directional Patterns (LDP) was proposed by Jabid et al. (2010a), a robust texture descriptor in respect to noise and non-monotonic illumination variations. Robustness of LDP is depicted in Figure 3

85	32	26
53	50	10
60	38	45
$LBP = 00111000 = 56$		
$LDP = 00010011 = 19$		

81	29	32
38	58	15
65	43	47
$LBP = 00101000 = 40$		
$LDP = 00010011 = 19$		

Figure 3
Comparison of LBP and LDP on an Original and a Noisy Image

The first image is an original image that has its features extracted using LBP and LDP. The same image is exposed to noise like illumination in the second image. From the image it is clear that LBP changed with introduction of noise but LDP is robust and remained with the same value.

Local directional pattern computes 8-bit code for central pixel in a 3×3 image region by comparing directional responses of each neighbouring pixel unlike comparing raw pixel values as in LBP. Prewitt (1970), Kirsch (1971) and Sobel and G (1968) are candidate edge detectors Pratt (1978) that can be used to derive gradient of an image. Among them, Kirsch is robust in detection of directional edges because it considers all eight neighboring pixels Lee (1996) in a 3×3 image region.

Kirsch edge detector

It is a first-order derivative edge detector that derives gradient of an image by convolving each of the 3×3 region with a set of filters (masks). Kirsch defined non-linear edge detector as in equation 6 Pratt (1978)

$$P(x,y) = \max\{1, \max_{k=0}^{7}[|5S_k - 3T_k|]\} \quad \text{.....(6)}$$

Where

$$S_k = P_k + P_{k+1} + P_{k+2} \quad \text{.....(7)}$$

And

$$T_k = P_{k+3} + P_{k+4} + P_{k+5} + P_{k+6} + P_{k+7} \quad \text{.....(8)}$$

Where a in k_a is evaluated as $a = a \, mod \, 8$ and $P_k \sim [k = 0,1,2 \ldots,7]$ are eight neighboring pixels of $P(x, y)$ shown in Figure 4. Image gradient towards a particular direction is found by convolving 3×3 image region with the respective mask M_k shown in Figure 5

P_3	P_2	P_1
P_4	$P_{(x, y)}$	P_0
P_5	P_6	P_7

8 neighbors of pixel $P_{(x, y)}$

M3	M_2	M_1
M_4		M_0
M_5	M_6	M_7

(b) 8 directional Krisch mask positions

Figure 4 (a)
Eight neighbors of pixel P (x, y) and (b) corresponding Krisch Mask positions

$$\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix}$$
East M0

$$\begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix}$$
North East M1

$$\begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$
North M2

$$\begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$
North West M3

$$\begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix}$$
West M4

$$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$
South West M5

$$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix}$$
South M6

$$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$
South East M7

Figure 5
Krisch Edge Response Mask in Eight Directions

For each central pixel X_c, there are eight directional response values. Presence of a corner or an edge exhibits high absolute values. Interest of LDP is to find k significant responses, set their corresponding bit value to 1 and assign 0 to the remaining $8 - k$ bits. The resultant binary string is converted to decimal and assigned to X_c as its' LDP code. The process is done for all pixels in an image to obtain LDP-encoded image. Figure 6 shows process of image encoding using LDP operator.

85	52	26
53	50	10
60	38	45

M_3	M_2	M_1
M_4		M_0
M_5	M_6	M_7

313	97	503
537		399
161	97	303

0	0	1
1		1
0	0	0

$LDP = 00010011_2 = 19_{10}$

Figure 6
Process of encoding an image with LDP operator with $k = 3$ *(a); Image region. (b) Kirsch masks as presented in Figure 5 (c) Result of convolving each pixel in (a) with 8 Kirsch masks. (d) Pick top* $k = 3$ *significant responses, set there corresponding bit to 1 and the rest to 0*

Starting from M_0, the LDP code is shown in equation 9.

$$LDP = 11001000_2 = 19_{10} \quad \text{.....(9)}$$

LDP code shown is derived using equation 10 and 11

$$LDP_k = \sum_{j=0}^{7} b\left((m_j - m_k) \times 2^j\right) \quad \text{.....(10)}$$

$$b(a) = \begin{cases} 1, if\ a \geq 0 \\ 0, otherwise \end{cases} \quad \text{.....(11)}$$

Where m_k is the k^{th} significant response, and m_j is response of Kirsch mask M_j. Local directional pattern operator generates C_k^8 distinct patterns in LDP encoded image. Histogram $H(i)$ with C_k^8 bins is used to represent the input image of size $M \times N$ as represented by equation 12 and 13

$$H(i) = \sum_{m=0}^{M} \sum_{n=0}^{N} f(LDP_k(m,n), i) \quad \text{.....(12)}$$

$$f(p,i) = \begin{cases} 1\ if\ p = i \\ 0\ otherwise \end{cases} \quad \text{.....(13)}$$

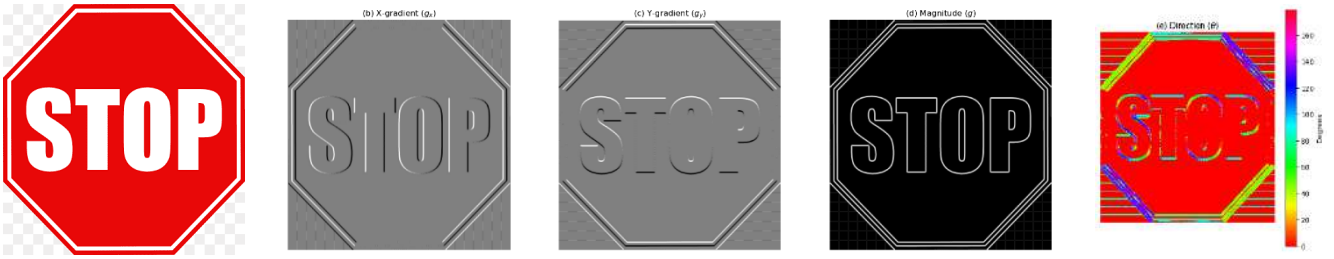
Where $f(p,i)$ is a logical function that compares if the LDP code at location $p(m,n)$ of the LDP-encoded image is equal to current LDP pattern i for all i in the range $0 \leq i \leq C_k^8$. The resultant histogram has dimensions $1 \times C_k^8$ and is used to represent the image. Resultant feature has corners, spots, edges and texture information about an image Jabid et al. (2010b). The limitation of LDP is that not all responses are considered in generating the LDP code. The discarded responses though not among top k, they could contribute into making LDP robust and pattern discriminative. Furthermore, some of top k directional responses could possibly be in particular orientation like west-east.

Histogram of Oriented Gradients

Originally proposed by Dalal and Triggs (2010), Histogram of Oriented Gradients (HOG) computes image gradient by counting occurrences of gradient orientations in local image regions. HOG uses 1D centered derivative mask $[-1,0,+1]$ and $[-1,0,+1]^T$ for vertical (g_x) and horizontal (g_y) directions respectively to calculate image gradient of 64×128 image divided into blocks of 16×16 with $2\,8 \times 8$ cells making up a block. The magnitude (g) and direction of the gradient (θ) are calculated as

$$g_{xy} = \sqrt{g_x^2 + g_y^2} \text{and } \theta_{xy} = \arctan\frac{g_y}{g_x} \quad \text{.....(14)}$$

where g_x and g_y are vertical and horizontal gradients of the image respectively, g is the gradient magnitude and θ is gradient direction. Figure 7 shows the x and y gradients, magnitude (g) and direction (θ) visualization of an input image.



Input Image x-gradient (gx) y-gradient (gy) Magnitude (g) Direction (θ)

Figure 7

Histogram of Gradients Image Responses

At every position (x,y) the image gradient has magnitude and direction. The gradients of colour images are evaluated for every colour channel. The gradient at point (x,y) is the maximum among magnitudes of colour channels and the direction (angle) is one that correspond to the maximum gradient. Histogram of gradients is created in 8×8 cells. The resultant histogram has 9 bins corresponding to angles $0^o, 20^o, 40^o, 60^o \ldots 160^o$. Figure 8 shows a sample histogram for single 8×8 cell where **5** shows number of edges at orientation 120^o.

8.75	2			13		5		26.25
$\mathbf{0^o}$	$\mathbf{20^o}$	$\mathbf{40^o}$	$\mathbf{60^o}$	$\mathbf{80^o}$	$\mathbf{100^o}$	$\mathbf{120^o}$	$\mathbf{140^o}$	$\mathbf{160^o}$

Figure 8

Histogram of Oriented Gradient

If direction θ^o is between any two angles θ_1^o and θ_2^o shown in Figure 8 a voting scheme is employed to determine in which bin does θ^o fall. Generally, if there are **n** entries in direction matrix with θ^o where $\theta\backslash\%20 \neq 0$, the entries are shared between θ_1^o and θ_2^o. The entries at θ_1 and θ_2 will be incremented by

$$\theta_1 = 1 - \left(\frac{|\theta_1^o - \theta^o|}{20}\right) \times n \quad \text{.....(15)}$$

$$\theta_2 = 1 - \left(\frac{|\theta_2^o - \theta^o|}{20}\right) \times n \quad \text{.....(16)}$$

Respectively, sample of this voting scheme is shown in 8 where we show how 35 entries at orientation 165^o are shared between 0^o and 180^o. These histograms are calculated for every 8×8 cell and resultant histograms are concatenated to make up the HOG feature vector. This vector is normalized and used for pattern recognition.

HOG is mostly used for object detection since it differentiates situations where a bright object is against a dark background or a dark object being against a bright background Satpathy et al. (2010) making it mostly suitable for shape description rather than texture feature descriptor. Texture descriptors should capture both image edges and subtle textural information like spots for better surface characterization. HOG is a very high dimensional feature and it only count occurrences of gradient in localized portions of an image without considering the direction of the gradient. A gradient at orientation 0^o could be firing towards east of west with regard to reference pixel. It is important to encode all orientation, magnitude and direction of a gradient to make resultant feature vector more discriminative. Furthermore, HOG only calculates horizontal and vertical gradients of an image with the same weights for the edges. More discriminative image gradient is calculated using filters that consider more directions with different weights for each direction like Kirsch (1971) which considers eight directions or four orientations.

3.2 Local Directional Histogram of Oriented Gradients

The limitation of HOG is that it calculates image gradient using vertical and horizontal edges only. While this approach achieves better results in object detection, more orientations image gradients are needed to improve texture description using HOG. The proposed approach encodes image gradient at four orientations. The image gradients are calculated at $0^o, 45^o, 90^o$ and 135^o unlike HOG operator which calculates image gradient at 0^o and 90^o only.

Local directional HOG uses 1D centered derivative mask $[-1,0,+1]$ and its rotations at $45^o, 90^o$ and 135^o for vertical (g_x), left diagonal (g_{ld}), horizontal (g_y) and right diagonal (g_{rd}) directions respectively to calculate image gradient of 64×128 image divided into blocks of 16×16 with two 8×8 cells making up a block. These filters are shown in figure 9 while figure 10 shows horizontal, vertical, left and right diagonal image gradients and the corresponding magnitudes and angles.

-1 0 +1	**-1** **0** **+1**
X – Gradient Filter	LD- Gradient Filter
-1 **0** **+1**	**-1** **0** **+1**
Y- Gradient Filter	RD- Gradient Filter

Figure 9

Filters used for LDHOG Image Gradients Calculation

Magnitude g_{xy} and angle θ_{xy} are calculated using x and y gradient responses as shown in equation 14. Using left-diagonal and right-diagonal gradient responses, second magnitude g_{di} and angle θ_{di} are calculated as

$$g_{di} = \sqrt{g_{ld}^2 + g_{rd}^2} \text{ and } \theta_{di} = \arctan\frac{g_{ld}}{g_{rd}} \quad \text{.....(17)}$$

Figure 10 shows the gradients visualization for x, y, right and left diagonal directions and their corresponding magnitudes and orientations

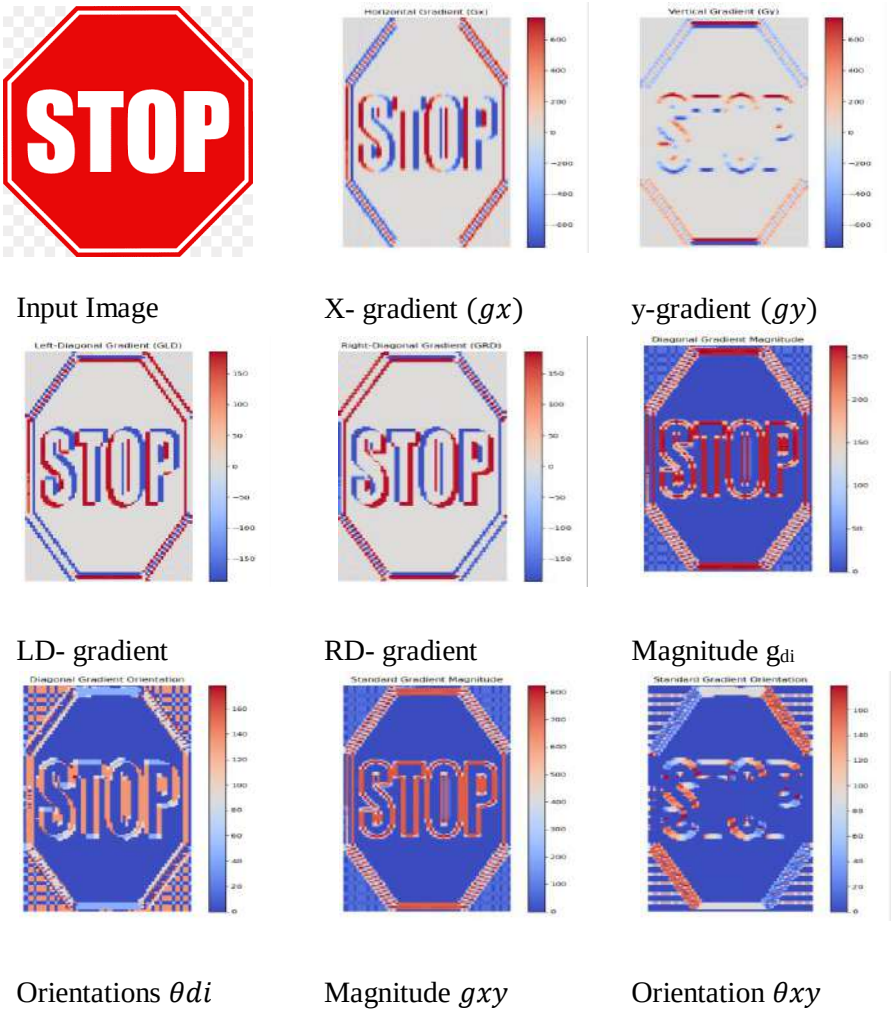


Figure 10
Gradients visualization for x, y, Right and Left Diagonal Directions and their Corresponding Magnitudes and Orientations

Given an image region shown in Figure 11 (a), four image gradients shown figure 11 (b) are calculated.

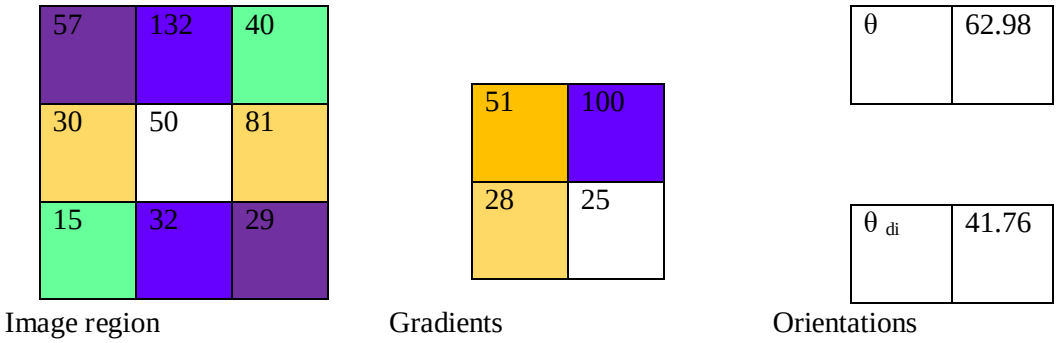


Figure 11
Sample LD-HOG Generation Process

From these gradients, two orientations shown in figure 11 (c) are calculated for current reference pixel using Equation 14 and 17. These angles are entered in their respective bins in two different histograms. The two histograms are then fused and used as a feature vector for TRS. Different fusion techniques can be used to investigated an Image. Fusing techniques include: MAX pooling - Angles in the same bin are compared in the two histograms and the maximum angle is chosen and entered to the respective bin MIN pooling - Angles in the same bin are compared in the two histograms and the minimum angle is chosen and entered to the respective bin

AVG pooling - The average of the two angles is entered to the respective bin. Maximum pooling approaches was repeated on the whole image region and resultant LDHOG is used as feature descriptor for TSR. The image is cropped and resized to 64×128 pixels. Notably, the image can be resized to any dimensions but the aspect ratio must be 1:2. Horizontal, vertical, left-diagonal and right-diagonal image gradients are calculated for each of the colour channels to obtain image gradient for each direction and the channel with highest gradient is chosen as the image gradient. The image is further subdivided into cells of 8×8 pixels. This leads to 8×16 cells. The cells are grouped into blocks where each block consist of 2×2 cells resulting to 7×15 blocks. With each cell having 1×9 histogram feature, the overall feature vector is $7 \times 15 \times 2 \times 2 \times 9 = 3780$. Vertical and horizontal image gradients are used to calculate gradient orientations and their respective magnitude for first local directional histogram. The right-diagonal and left diagonal image gradients are used to calculate gradient and their respective magnitude for second local directional histogram. For each traffic sign, the two histograms obtained are fused and the resultant feature vector used for TSR. Linear Discriminant Analysis (LDA) is used to reduce dimensionality of the final feature vector before the model is learnt.

3.3 Orientation Binning

In this study, 9 bins of 20^o range are used. The histogram contains 9 bins ranging from 0^o to 180^o. If an orientation falls within a bin range, the entry in the respective bin is incremented. If an orientation lies between two bins, a voting scheme shown in Equation 15 and 16 is applied. The choice of number of bins was motivated by previous research Dalal and Triggs (2005); Tan et al. (2014) who found that 9 bin histograms achieve higher accuracies compared to 15 bin histograms.

Figure 12, 13 and 14 shows the bins extracted from the features extracted

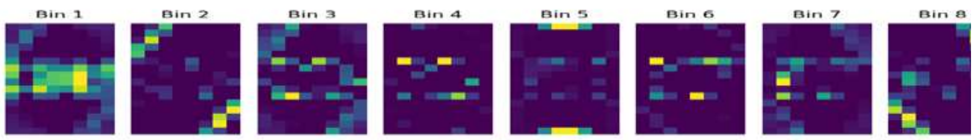


Figure 12
HOG Bins for the Sampled Image

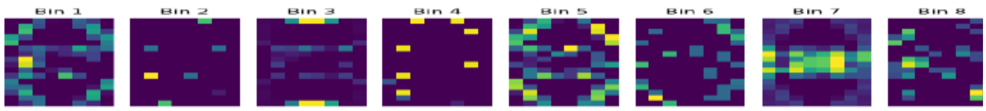


Figure 13
Diagonal HOG for the Sampled Image

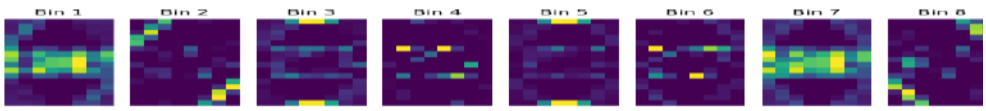


Figure 14
Fused Bins for the Sampled Image

3.4 Normalization

Histogram blocks are normalized using L2-Norm normalization approach as

$$|V|_{L2} = \frac{v}{\sqrt{\sum |v|^2 + \epsilon^2}} \quad (18)$$

Where $|V|_{L2}$ is a normalized feature, $|v|$ is none normalized feature and $\epsilon = 0.00001$ is a small constant used to prevent overflow when $v = 0$. This normalization approach is chosen since it performs better compared to L1 normalization Dalal and Triggs (2005).

3.5 Materials and Experiments

Datasets: The study utilized traffic sign images from German Traffic Sign Detection Benchmark (GTSDB) dataset. Introduced by Stallkamp et al. (2012) in 2012, GTSDB is an open-source dataset that has 43 classes of traffic signs (51,840 images). The Dataset has: training set with 39,209 images and the testing set with 12,631 images. The dataset has a representation of real-world scenarios including different brightness, occlusion and different sizes, and was collected from different locations in Germany. The study applied this dataset in training and testing of LDHOG for TSR. Figure 15 shows a sample of images for each class. The dataset was

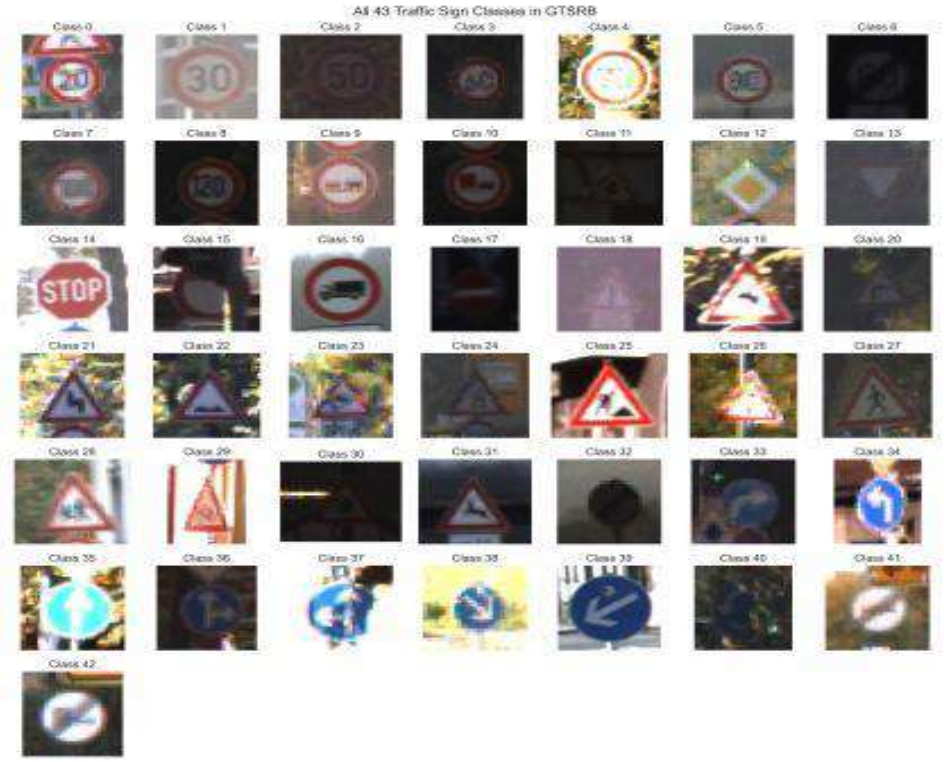


Figure 15

Sampled Images per Classes

Each of the images in Figure 15 were selected because it has different shape, brightens, size and color distributions and it has been widely used in TSR research as shown by figure 16, 17, 18 below.

3.

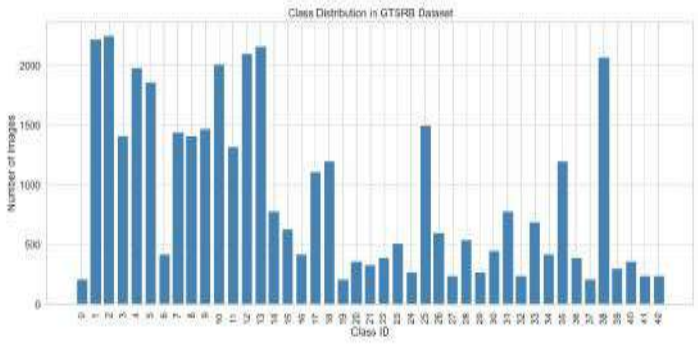


Figure 16

Class Distribution of GTSDB Dataset

[3]The test sample was 20 percent of the dataset. The training sample was 80 percent

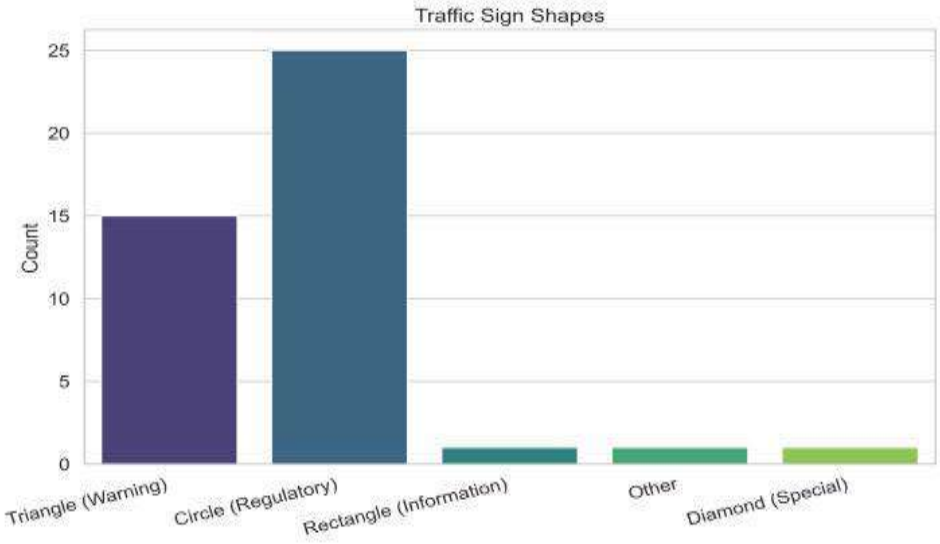


Figure 17
Shape Distribution of GTSDB Dataset

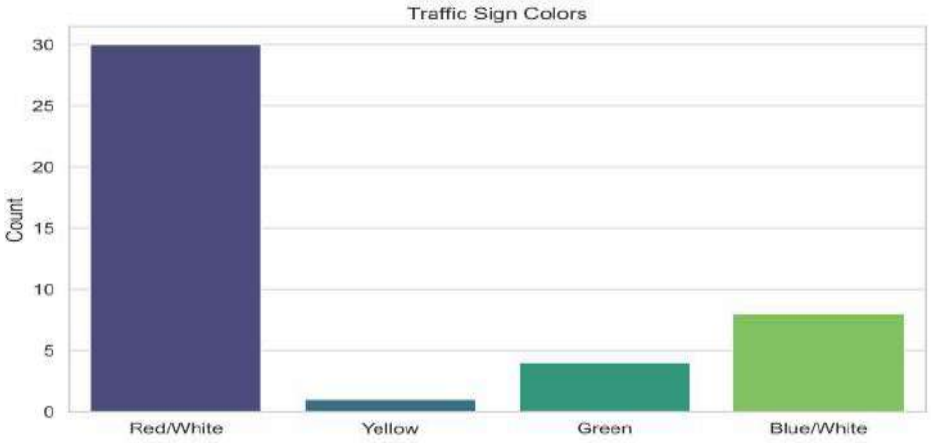


Figure 18
Color Distribution of GTSDB Dataset

3.6 Classification and Regression

TSR can be modelled as a regression of classification problem. Notably, Multiclass classification was performed on The GTSDB dataset targeting 43 classes. Regression is used to model TSR classifier with experiments on SVM, random forest and decision trees as proposed by Hechri and Mtibaa (2020) and Lee et al. (2024). An image is classified into a class Id using these classifiers. for SVM, from experiment, the following values were selected: kernel='rbf', C=10, gamma='scale', class weight='balanced' and probability=True. RBF was selected because of handling non-linearity. Traffic signs have complex shapes/textures that are non-separable in linear space and has outperformed polynomial and sigmoid kernels for vision tasks Patle and Chouhan (2013). The regularization parameter C has wider margins (high probability of misclassification) with smaller values while larger value has a high probability of overfitting. 10 was selected for GTSDB task from experiment results. The kernel co-efficient was set to scale to allow for adaptability of feature scaling and avoids manual tuning pitfalls. GTSRB dataset has different number of images per class as shown in figure 16 causing class imbalance. To solve this class weight is set to balance to prevent biasness

3.7 Validation and Evaluation Protocol

The dataset was imbalanced with warning signs having more classes compared to Mandan dory and reservation. Therefore, stratified K-fold validation protocol is used to validate LD-HOG based TSR on GTSDB datasets. In this validation protocol, the datasets are randomly split in k disjoint subsets of the same size. Then, k iterations of training and validation are performed such that in every iteration, a different fold of data is reserved for validation while the remaining $k-1$ are used to learn a model. The estimated error is the mean of all validation errors. In this study, k was set to 5. Different values of k were tried and 5 was chosen as compromise between subset sizes, validation accuracies and computational efficiency. Additionally, comparison of LD-HOG to HOG was done using precision, recall and F1-Score. Precession measures how reliable are positive predictions. High precision means low false alarm. For TSR, precision

should be high as possible because misclassification can lead to an accident. Precession is calculated using the following equation

$$\text{Precision} = \frac{TP}{TP + FP} \quad \ldots\ldots (19)$$

Where:

TP = True Positives (correctly predicted positives)

FP = False Positives (incorrectly predicted positives)

Recall measures how many actual positives were found. For recall, a high value implies that the model missed a few true instances. Equation 15 shows how recall is calculated.

$$\text{Recall} = \frac{TP}{TP + FN} \quad \ldots\ldots (20)$$

Where:

FN = False Negatives (actual positives predicted as negatives)

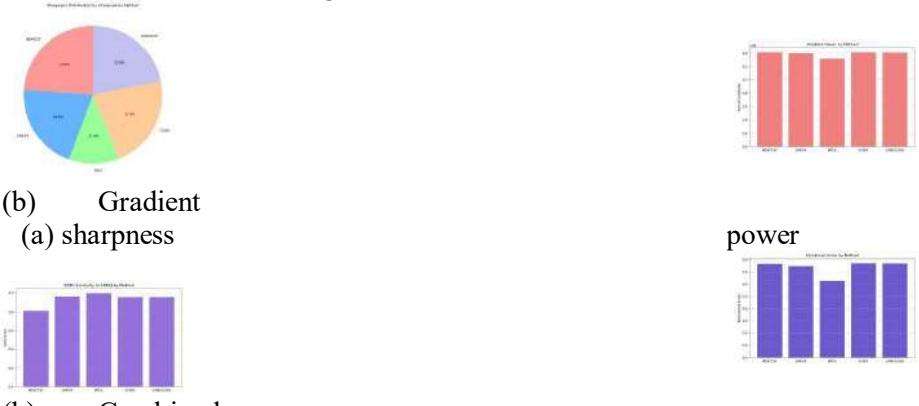
F1-Score measures the harmonic mean of precision and recall. It is the best measure for imbalanced dataset like GTSRB. It is used to prevent misleading high accuracy from imbalanced. F1-score is calculate using the equation

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN} \quad \ldots\ldots (21)$$

IV. FINDINGS & DISCUSSION

Three experiments were performed to measure and evaluate the performance of LD-HOG in comparison to other handcrafted features HOG, LBP AND LDP.

The experiments were done using three classifiers (SVM, Random forest and decision trees) on GTSDB dataset. The first experiment was done on image preprocessing comparing different interpolation methods on different aspects including sharpness, gradient power, Structural Similarity Index (SSIM), Time and Memory. The results of this experiment are shown in Figure **20**.



(a) sharpness (b) Gradient power

(a) SSIM score (b) Combined

Figure 20

Preprocessing Results for Different Interpolation Methods

LanczoS4 interpolation method was selected as the best interpolation method for image preprocessing. The second experiment was to test performance of LDHOG, HOG, LBP and LDP on GTSDB dataset and SVM classifier. The experiment measured precision, recall, f1-score and support. Table 2,3,4,5 and 6 summaries the experiment results.

Table 2

Comparison for LBP, LDP, HOG and LD-HOG on GTSDB Dataset using SVM

Feature Extractor	Precision	Recall	F1-Score
LD-HOG	99	99	99
HOG	96	95	95
LDP	93	92	91

Table 3
Comparison for LBP, LDP, HOG and LD-HOG on GTSDB Dataset using Random Forest

Feature Extractor	Precision	Recall	F1-Score
LD-HOG	95	95	95
HOG	91	89	90
LDP	87	88	88

Table 4
Comparison for LBP, LDP, HOG and LD-HOG on GTSDB Dataset using Decision Tree

Feature Extractor	Precision	Recall	F1-Score
LD-HOG	76	76	76
HOG	65	64	64
LDP	61	62	62

Table 5
Comparison for LBP, LDP, HOG and LD-HOG on GTSDB Dataset using Average

Feature Extractor	Precision	Recall	F1-Score
LD-HOG	90	90	90
HOG	84	82	83
LDP	80	81	80

Table 6
Stratified K fold Validation Results

classifier	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
SVM	98.55	98.64	98.95	98.89	99.02	98.81
RF	94.52	94.80	94.98	94.77	95.28	94.87
DT	76.31	76.33	76.45	76.72	76.70	76.50
Average	89.79	89.92	90.12	90.12	90.33	90.06

V. CONCLUSION & RECOMMENDATIONS

5.1 Conclusion

This paper proposes a new operator, LD-HOG, for encoding image gradient. The proposed operator is an extension of HOG operator. In LD-HOG, a 1D derivative mask [−1, 0, +1] and its rotation at 45^o, 90^o and 135^o is used to calculate two image gradients. The histograms of the two gradients are fused using maximum pooling techniques and used as a feature vector for age estimation. Experimental results on GTSDB dataset show that LD-HOG outperforms HOG features descriptors in TSR. The performance of LD-HOG in TSR demonstrates that using all the 8 neighbors of a pixel in encoding the image gradient at that particular pixel results into robust features for object classification as compared to considering only 4 neighbors. In a digital image, each pixel p_i has $n = 2^{(d+2)}$ neighbors p_0, $p_1 \ldots p_{n-1}$ at distance d from itself. All the neighboring pixels carry subtle information about the gradient of the image at distance d from pixel p_i. Therefore, when encoding image gradient at p_i all the neighboring pixels at distance d should be considered.

5.2 Recommendations

The study recommends the following for future studies. When great robustness to orientation and texture fluctuations is needed, LD-HOG should be taken into consideration as the preferred feature extraction technique in TSR applications. In order to assess LD-HOG's generalizability, it should be tested in more domains as pedestrian detection, face recognition, and vehicle classification, given its exceptional performance on the GTSDB dataset. To further improve feature resilience under different object sizes and resolutions, future studies should look at integrating LD-HOG with multi-scale analytic methodologies.

REFERENCES

An, F., Wang, J., & Liu, R. (2024). Road traffic sign recognition algorithm based on cascade attention-modulation fusion mechanism. *IEEE Transactions on Intelligent Transportation Systems, 25*(11), 17841–17851. https://doi.org/10.1109/TITS.2024.3439699

Asha, J., Giridhran, R., Agalya, K., & Sathya, R. (2022). Traffic sign detection using HOG and GLCM with decision tree and random forest. In *Proceedings of the 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS)* (pp. 879–885). IEEE. https://doi.org/10.1109/ICACRS55517.2022.10029118

Aziz, S., & Youssef, F. (2018). Traffic sign recognition based on multi-feature fusion and ELM classifier. *Procedia Computer Science, 127*, 146–153. https://doi.org/10.1016/j.procs.2018.01.112

Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks, 106*, 249–259. https://doi.org/10.1016/j.neunet.2018.07.011

Chen, C., Li, B., Zhang, H., Zhao, M., Liang, Z., Li, K., & An, X. (2025). Performance enhancement of deep learning model with attention mechanism and FCN model in flood forecasting. *Journal of Hydrology, 658*, 133221. https://doi.org/10.1016/j.jhydrol.2025.133221

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (Vol. 1, pp. 886–893). https://doi.org/10.1109/CVPR.2005.177

Dalal, N., & Triggs, B. (2010). Histograms of oriented gradients for human detection. In *Proceedings of 2005 Institute of Electrical and Electronics Engineers Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 1, pp. 886–893). https://doi.org/10.1109/CVPR.2005.177

Halder, R. K., Uddin, M. N., Uddin, M. A., Aryal, S., & Khraisat, A. (2024). Enhancing k-nearest neighbor algorithm: A comprehensive review and performance analysis of modifications. *Journal of Big Data, 11*(1), 34. https://doi.org/10.1186/s40537-024-00961-3

Hechri, A., & Mtibaa, A. (2020). Two-stage traffic sign detection and recognition based on SVM and convolutional neural networks. *IET Image Processing, 14*(4), 342–350. https://doi.org/10.1049/iet-ipr.2019.0634

Hechri, A., Hmida, R., Abdelali, A. B., et al. (2014). Real-time road lane markers detection for intelligent vehicles. *Advances in Environmental Biology, 8*(7), 2266–2272.

Jabid, T., Kabir, M. H., & Chae, O. (2010b, August). Gender classification using local directional pattern (LDP). In *Proceedings of 2010 20th International Conference on Pattern Recognition* (pp. 2162–2165). IEEE. https://doi.org/10.1109/ICPR.2010.373

Jabid, T., Kabir, M., & Chae, O. (2010a, January). Local directional pattern (LDP) for face recognition. In *Proceedings of 2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*. IEEE. https://doi.org/10.1109/ICCE.2010.5418801

Jain, A., Moparthi, N. R., Swathi, A., Sharma, Y. K., Mittal, N., Alhussen, A., Alzamil, S., & Haq, M. A. (2023). Deep learning-based mask identification system using ResNet transfer learning architecture. *Computer Systems Science and Engineering*. https://doi.org/10.32604/csse.2023.036973

Jayaprakasha, A., & KeziSelvaVijilab, C. (2019). Feature selection using ant colony optimization (ACO) and road sign detection and recognition (RSDR) system. *Cognitive Systems Research, 58*, 123–133. https://doi.org/10.1016/j.cogsys.2019.04.012

Kerim, A., & Efe, M. (2021, April). Recognition of traffic signs with artificial neural networks: A novel dataset and algorithm. In *Proceedings of the 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* (pp. 171–176). IEEE. https://doi.org/10.1109/ICAIIC51459.2021.9415077

Khalifa, A. A., Alayed, W. M., Elbadawy, H. M., & Sadek, R. A. (2024). 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. https://doi.org/10.3390/app14093903

Kirsch, R. A. (1971, June). Computer determination of the constituent structure of biological images. *Computers and Biomedical Research, 4*(3), 315–328. https://doi.org/10.1016/0010-4809(71)90034-6

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998, November). Gradient based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Lee, J., Kim, K., & Lee, K. (2024). Multi-sensor image classification using the random forest algorithm in Google Earth Engine with KOMPSAT-3/5 and CAS500-1 images. *Remote Sensing, 16*(24), 4622. https://doi.org/10.3390/rs16244622

Lee, S.-W. (1996, June). Off-line recognition of totally unconstrained handwritten numerals using multilayer cluster neural network. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 18*(6), 648–652. https://doi.org/10.1109/34.506416

Li, W., Song, H., & Wang, P. (2022). Finely crafted feature features for traffic sign recognition. *International Journal of Circuits, Systems and Signal Processing, 16*, 185–193. https://doi.org/10.46300/9106.2022.16.20

Madani, A., & Yusof, R. (2018). Traffic sign recognition based on color, shape, and pictogram classification using support vector machines. *Neural Computing and Applications, 30*, 2807–2817. https://doi.org/10.1007/s00521-017-3255-0

Maenpaa, T., & Pietikainen, M. (2005). Texture analysis with local binary patterns. In *Handbook of Pattern Recognition and Computer Vision* (pp. xx–xx). World Scientific.

Namyang, N., & Phimoltares, S. (2020, October). Thai traffic sign classification and recognition system based on histogram of gradients, color layout descriptor, and normalized correlation coefficient. In *Proceedings of the 2020 5th International Conference on Information Technology (INCIT)* (pp. 270–275). IEEE.

Ojala, T., Pietikainen, M., & Harwood, D. (1996). A comparative study of texture measures with classification based on featured distribution. *Pattern Recognition, 29*, 51–59. https://doi.org/10.1016/0031-3203(95)00186-2

Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*, 971–987. https://doi.org/10.1109/TPAMI.2002.1017623

Panis, G., Lanitis, A., Tsapatsoulis, N., & Cootes, T. F. (2016). Overview of research on facial ageing using the FG-NET ageing database. *IET Biometrics, 5*, 37–46. https://doi.org/10.1049/iet-bmt.2014.0123

Patle, A., & Chouhan, D. S. (2013). SVM kernel functions for classification. In *2013 International Conference on Advances in Technology and Engineering (ICATE)* (pp. 1–9). IEEE. https://doi.org/10.1109/ICAdTE.2013.6524743

Pratt, W. K. (1978). *Digital image processing*. New York: Wiley.

Prewitt, J. M. S. (1970). Object enhancement and extraction. In *Picture Processing and Psychopictorics* (pp. xx–xx). Academic Press.

Razavi, M., Mavaddati, S., & Koohi, H. (2024). ResNet deep models and transfer learning technique for classification and quality detection of rice cultivars. *Expert Systems with Applications, 247*, 123276. https://doi.org/10.1016/j.eswa.2024.123276

Saouli, A., El Aroussi, M., & Fakhri, Y. (2018). Traffic sign recognition based on multi-feature fusion and ELM classifier. *Procedia Computer Science, 127*, 146–153.

Sapijaszko, G., Alobaidi, T., & Mikhael, W. (2019, August). Traffic sign recognition based on multilayer perceptron using DWT and DCT. In *Proceedings of the 2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 440–443). IEEE. https://doi.org/10.1109/MWSCAS.2019.8881053

Satpathy, A., Jiang, X., & Eng, H.-L. (2010). Extended histogram of gradients feature for human detection. In *Proceedings of 2010 17th IEEE Conference on Image Processing (ICIP)*. IEEE. https://doi.org/10.1109/ICIP.2010.5650070

Sedaghatjoo, Z., Hosseinzadeh, H., & Bigham, B. S. (2024). Local binary pattern (LBP) optimization for feature extraction. *arXiv*. https://arxiv.org/abs/2407.18665

Shi, X., Hao, Z., & Yu, Z. (2024, June). SpikingResFormer: Bridging ResNet and Vision Transformer in spiking neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5610–5619).

Sobel, I., & Feldman, G. (1968). A 3 x 3 isotropic gradient operator for image processing. In *Presented at the Stanford Artificial Intelligence Project (SAIL)*.

Soni, D., Chaurasiya, R., & Agrawal, S. (2019, January). Improving the classification accuracy of accurate traffic sign detection and recognition system using HOG and LBP features and PCA-based dimension reduction. In *Proceedings of the International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM)*. Jaipur, India.

Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012, August). 2012 special issue: 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

Tan, H., Ma, Z., & Yang, B. (2014, June). Face recognition based on the fusion of global and local HOG features of face images. *IET Computer Vision, 8*(3), 224–234. https://doi.org/10.1049/iet-cvi.2012.0302

Wang, B. (2022). Research on the optimal machine learning classifier for traffic signs. In *SHS Web of Conferences*. EDP Sciences.

Weng, H., & Chiu, C. (2018, April). Resource efficient hardware implementation for real-time traffic sign recognition. In *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1120–1124). IEEE.

WHO. (2018). *Global status report on road safety*. World Health Organization.

Yang, Y., Luo, H., Xu, H., et al. (2016). Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent Transportation Systems, 17*(7), 2022–2031. https://doi.org/10.1109/TITS.2015.2509281

Zhang, K., Guo, Y., Wang, X., Yuan, J., & Ding, Q. (2019). Multiple feature reweight DenseNet for image classification. *IEEE Access, 7*, 9872–9880. https://doi.org/10.1109/ACCESS.2018.2890127

Zhu, Y., & Newsam, S. (2017). DenseNet for dense flow. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 790–794). IEEE. https://doi.org/10.1109/ICIP.2017.8296389

Zhu, Y., & Yan, W. Q. (2022). Traffic sign recognition based on deep learning. *Multimedia Tools and Applications, 81*(17), 17779–17791. https://doi.org/10.1007/s11042-022-12163-0