HISTOGRAM OF ORIENTED GRADIENTS (HOG) FOR CAR DETECTION

Submitted by:

Drumil Haresh Ved 210962006
Punith 210962012

Computer Vision Lab Project
CSE 3181

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MANIPAL INSTITUTE OF TECHNOLOGY,
MANIPAL ACADEMY OF HIGHER EDUCATION
NOV 2023

Using HOG + SVM for Car Detection

Drumi Haresh Ved, Putnith
Manipal Institute of Technology, Karnataka, India
drumilhv@gmail.com , npunith125@gmail.com

Abstract:- This paper uses a Histogram of Oriented Objects to detect the number of cars on a given street/highway. Considering the traffic on roads and the economic constraints on the hardware when implementing such monitoring systems on a scale, we propose a low resource-intensive algorithm that can detect cars with an accuracy of up to 80%. We use HOG for the feature descriptor and Support Vector Machine for classification if it's a car or not. Then, we try to make it compatible with high resource-intensive algorithms like YOLO v4 and Har cascades.

Keywords: Histogram of Oriented Objects, Support Vector Machine, YOLO v4, Har Cascades, Car Detection.

I. Introduction:

We intend to use HOG to detect cars on streets and count them; this project tries to find the number of cars in a given video, and then, frame by frame, we find out which car was detected. This method does not use any of the Machine/ Deep learning algorithms to extract features from the images/frames; we only use machine learning to classify the cell as a positive or negative class that is car found or not found. This project is done to reduce the intensive use of resources like power and/or hardware like GPU/CPU/TPU. This project was done keeping in mind that the scalable solution is the affordable one. Even though we were not wholly able to reach the expectation of reducing the cost of detection, we have greatly learned how to minimize it. What optimizations can be done, such as fewer bins or no image processing, could help reduce the cost to nearly 20% of Deep Neural Networks with compatible accuracy.

II. Literary Review

SVM is one of the best classifiers compared to its counterparts like decision trees or random forest because the research data shows that it can be up to 20 % more accurate with no added baggage of resources/compute[8]. HOG can be very inexpensive (in terms of computing) to find the features of an image, whereas SVM can be cheap while making a dilation given the features of the image and pre-trained model, and combining both would give the best of both worlds. SVM in itself can be computed extensively if left to do regularization of parameters, so pre-trained the model is used, and HOG itself gives excellent results on local objects and shapes[5]. Xiangyu Zhang, Ling Zhang, and Xin Lou have done optimizations that reduce the power consumed to process and detect a 1080 p video from 250mW to about 55 mW by not preprocessing the image and passing the raw Bayer image to do the detection; this paper deals with human detection. They used a gradient Pyramid and optimized quantization of scaling factors, which helped reduce space by 4.5 times and compute by 2.5 times[2]. One of the significant optimizations done to reduce compute and memory was to reduce the number of bins in which the gradients and magnitudes were put from 9 to 4. This resulted in massive savings of space and compute without hampering the model's

accuracy significantly[2]. This paper focuses on enhancing human detection using Histogram of Oriented Gradients (HOG) and aims to improve recognition precision. It introduces three procedures for speeding up the matching process: pixel matching, cell matching, and block matching. Among these, block matching stands out as it successfully reduces matching time while maintaining a high detection rate. Additionally, the paper suggests that combining HOG with scale-invariant feature transformation (SIFT) can further enhance the efficiency and robustness of object detection algorithms[7]. Some of the solutions we saw were economically affordable. Still, the models proposed by M. Komorkiewicz, M. Kluczewski, and M. Gorgon are more research-oriented, which suggests that instead of using an integer for the storage of features, we use floating point data type (specifically float32) to store the features that can give more precision while storing the features, and possibility minimal truncation of features due to data type[3]. A two-step vehicle detection algorithm is proposed, combining HOG and Haar features using a coarse-to-fine strategy. Initial segmentation with Haar features and SVM reduces the region of interest, while multi-scale processing and integral image calculation enhance target detection in complex urban environments[6]. Haar Cascade is a highly accurate object detection technique that employs machine learning with positive and negative image samples for training. It excels in object detection with 20% higher precision than other classifiers. Its key advantage lies in the efficient computation of simple rectangular Haar features, enabling real-time applications, while a staged approach minimizes false positives, enhancing detection speed and accuracy[9]. This work utilizes YOLOv4, an advanced object detection algorithm, for traffic surveillance using a custom Indian road traffic dataset. YOLOv4, an improvement over YOLOv3, incorporates CSP darknet-53 and spatial pyramid pooling, offering high detection accuracy, precise bounding box localization, and efficient multi-scale prediction. However, the approach's high accuracy comes at a high GPU cost, prompting the need for optimization in future research to make it more feasible and scalable, with hardware and software requirements including a PC with at least 8GB RAM, 500GB ROM, a 64-bit processor, and a GPU for faster computation[4]. This approach addresses the challenge of small object detection in complex backgrounds, such as aerial views of cars from unmanned aerial systems. HOG-based techniques may produce many false alarms due to object-background resemblance. To mitigate this, the method combines HOG-based detection with a novel causal Markov random field (MRF) classifier, leveraging their complementary information. It employs a two-stage pipeline approach, where the first stage (HOG-SVM) focuses on high recall with some false positives, and the second stage (causal MRF) is designed to eliminate most of these false positives, resulting in improved performance compared to a discriminative-only approach, as demonstrated on standard datasets[10].

III. Methodology

a. Dataset and preprocessing data:

We use the GTI vehicle image database to train our SVM. This dataset contains both positive (cars) and negative (noncar) samples. The specification of the dataset is as follows.

No. of car images: 8792
No of non-car images: 8968
Image shape: (64, 64, 3)

1.1 Negative Samples

1.2 Positive Samples (cars)

b. Preprocessing data:

All the images were resized, and their color space was changed to RGB from whatever color space they were in. Then, histograms for all the images were made, divided into 32 bins, with intensity ranging from 0 to 256.

c. Histogram of Oriented Objects

Input Image: Start with the input image, which is typically the image containing pedestrians or objects you want to detect.

Image Resizing: Resize the input image to a fixed size of 128x64 pixels. This specific dimension, as suggested by the authors, is used for pedestrian detection, and it was chosen to achieve better results on tasks like pedestrian detection.

Gradient Calculation: Calculate the gradients of the image. Gradients are computed to capture the changes in intensity or color values within the image. For each pixel in the image, the gradient in the x-direction (Gx) and the gradient in the y-direction (Gy) are calculated. These gradients are calculated using the following formulas for each pixel (r, c):

Gx = I(r, c) - I(r, c - 1)

Gy = I(r, c) - I(r - 1, c)

Magnitude Calculation: Once the Gx and Gy values are obtained for each pixel, calculate the magnitude of the gradient and the angle of the gradient for each pixel using the following formulas:

$$\text{Magnitude (Mag)} = \sqrt{G_x^2 + G_y^2}$$

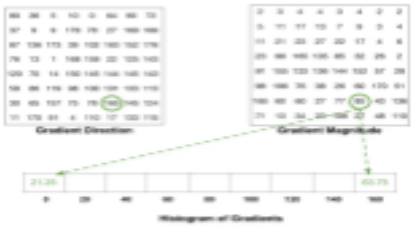
$$\text{Angle (Theta)} = \tan^{-1}\left(\frac{G_x}{G_x}\right)$$

The magnitude represents the strength of the gradient at each pixel, and the angle represents the orientation of the gradient.

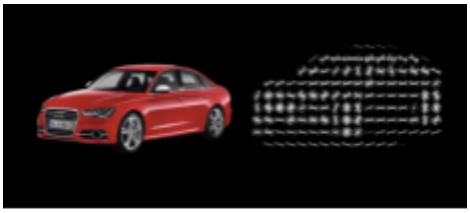
Block Processing: To capture local patterns and structures within the image, the image is divided into small blocks 2, in our case, 8x8 pixels.

Histogram Calculation: For each block, a histogram of gradient orientations is computed. This histogram represents the distribution of gradient angles within the block. The magnitudes of gradient vectors are used to vote into the corresponding bins of the histogram based on their angles.

Feature Extraction: The final HOG features for the entire image are obtained by concatenating the histograms from all the blocks. These HOG features capture the local gradient information and represent the object or region within the image.



1.3 allotting bins



1.4 Histogram of Oriented Objects

The features from the images were extracted and stored in an array, which will primarily be used to train SVM, so all the 17,754 images were put into the hog for collecting the features and then fed into the SVM for training.

d. Training the SVM Model:

All the feature vectors of all 17000 images were appropriate, with labels fed to the library SVM function, and it was trained.

e. Predicting the Images:

The code first resizes the input image and then iterates over different sliding window positions and scales to search for potential car objects. Each frame is scaled, and smoothening is applied before it goes further for detection, as these steps can help increase the model's performance. For each window, it extracts the Histogram of Oriented Gradient (HOG) features, spatial color features, and histogram color features. These features are then scaled and passed through the trained classifier to make a prediction. If the classifier predicts that a car is present in the window, a bounding box is drawn around the detected car in the draw_img frame, and its coordinates are added to the bbox_detection_list. If visualization is enabled, the box_vis list is also updated with the detected bounding box positions.

f. Predicting in Video:

Here, a new video clip is created by mapping the process_image function to each frame of the input video. The process_image function is expected to take an input frame and return an output frame with detected objects and tracking information.

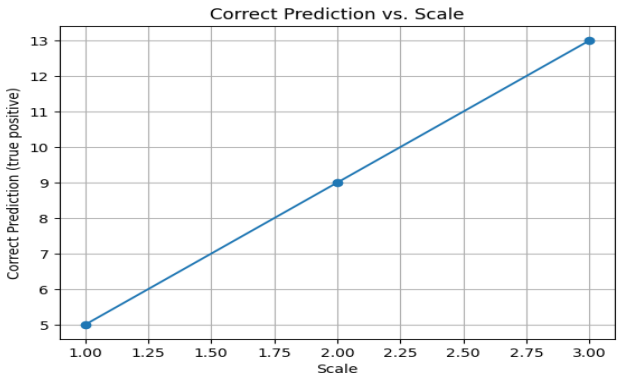
IV. Results

1.5 Final result of Hog + SVM on a test sample

The results of this study are majorly focused on different operations on the video.

a. Effects of Scaling:

Scale	Correct Prediction (true positive)
1	5
2	9
3	13



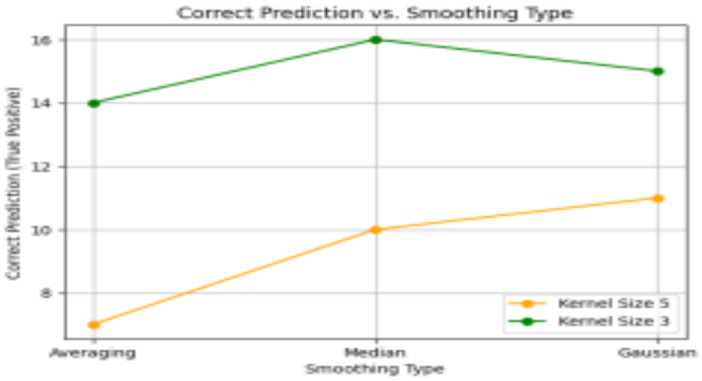
This chart and table is the only depiction of True Positive out of 22.

Observation: Scale 3 gave the best results because the dataset was also resized multiple times, and the cars that were smaller in size became the predicted class.

b. Effects of Smoothning:

Kernel Size 5 and 3:

Smoothing Type	Kernel size 5	Kernel size 3
Averaging	7	14
Median	10	16
Gaussian	11	15

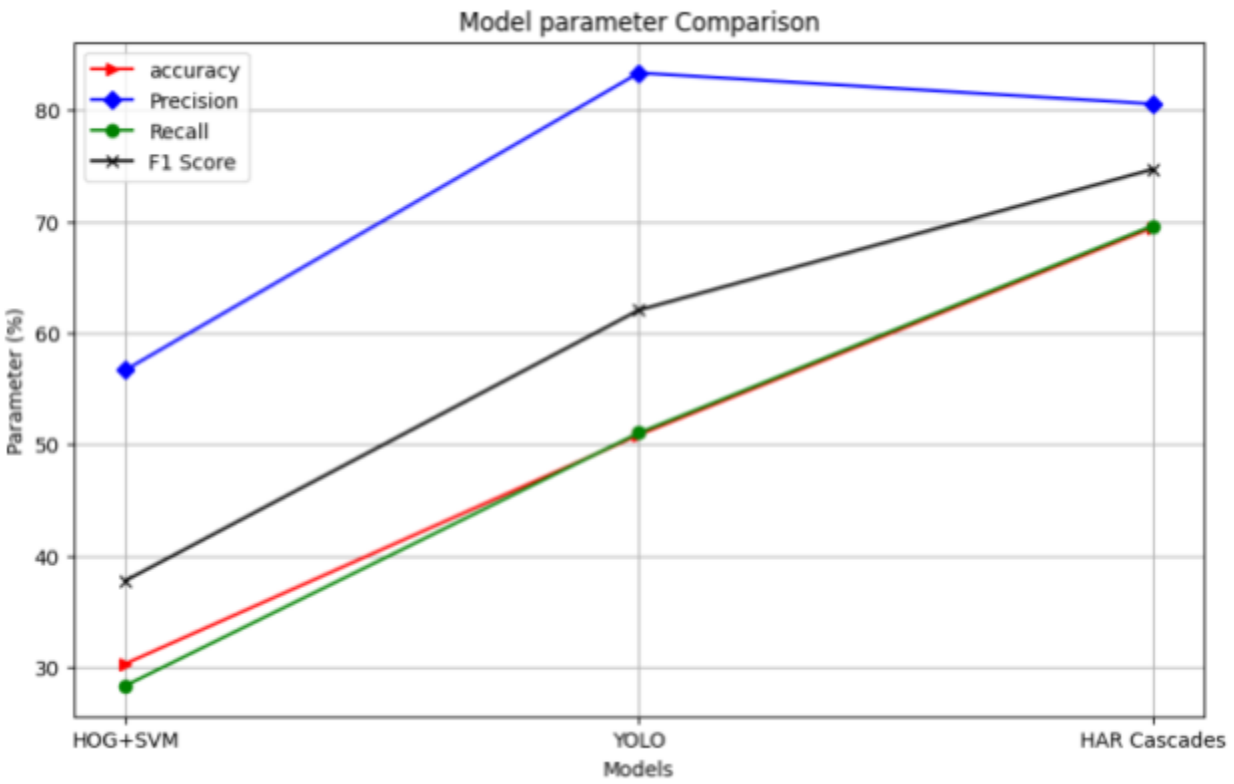


This chart and table is the only depiction of True Positive out of 22.

Observation: The smoothening operation can help us reduce noise from the image, resulting in noiseless images leading to sharper gradients. The 5 x 5 kernel performs poorer compared to the 3 x 3 kernel because 3 x 3 is just enough to remove noises and give better gradients, whereas 5 x 5 can also cause smoothening of the edges, which could lead to poorer performance.

Comparison to other Models:

	HOG+SVM	YOLO	HAR Cascades
Accuracy	**30.30%**	**50.84%**	**69.43%**
Precision	**56.66%**	**83.33%**	**80.55%**
Recall	**28.33%**	**51.02%**	**69.60%**
F1	**37.76%**	**62.04%**	**74.67%**



V. Drawbacks and Future Works:

Our models are of low accuracy. A lot can be done, but one of the major bottlenecks in achieving high accuracy is the lakh of sizable datasets. The major problem is when the video is scaled to a particular size, the dimensions of objects in it change drastically, and this leads to poor accuracy. Many papers do address this issue and take an approach to use Scale-invariant Feature Tracking (SIFT) in cascade to HOG + SVM for object detection; we plan to use SIFT to enhance the accuracy of the model.

VI. Conclusion

Car detection models like YOLO have higher accuracy than HOG+SVM, sometimes up to 98% accuracy and loss of less than 0.1%. These deep learning models run on resource-intensive machines that are not affordable at scale, whereas HOG + SVM can run on minimal resources like Raspberry Pi with little RAM, single-threaded CPU, and no GPU. If done correctly, HOG + SVM can reach accuracies of up to 80% on a low resource-intensive device at speeds comparable to deep learning models.

VII. References

1. R. Kawamoto et al., "A 1.15-TOPS 6.57-TOPS/W DNN Processor for Multi-Scale Object Detection," 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), Genova, Italy, 2020, pp. 203-207, doi: 10.1109/AICAS48895.2020.9073858.
2. X. Zhang, L. Zhang and X. Lou, "A Raw Image-Based End-to-End Object Detection Accelerator Using HOG Features," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 1, pp. 322-333, Jan. 2022, doi: 10.1109/TCSI.2021.3098053.
3. M. Komorkiewicz, M. Kluczewski and M. Gorgon, "Floating point HOG implementation for real-time multiple object detection," 22nd International Conference on Field Programmable Logic and Applications (FPL), Oslo, Norway, 2012, pp. 711-714, doi: 10.1109/FPL.2012.6339159.
4. U. P. Naik, V. Rajesh, R. K. R and Mohana, "Implementation of YOLOv4 Algorithm for Multiple Object Detection in Image and Video Dataset using Deep Learning and Artificial Intelligence for Urban Traffic Video Surveillance Application," 2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT), Erode, India, 2021, pp. 1-6, doi: 10.1109/ICECCT52121.2021.9616625.
5. S. Bougharriou, F. Hamdaoui and A. Mtibaa, "Linear SVM classifier based HOG car detection," 2017 18th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Monastir, Tunisia, 2017, pp. 241-245, doi: 10.1109/STA.2017.8314922.
6. Yun Wei, Qing Tian, Jianhua Guo, Wei Huang, Jinde Cao, Multi-vehicle detection algorithm through combining Harr and HOG features, Mathematics and Computers in Simulation,2019, Pages 130-145, ISSN 0378-4754
7. Terayama, Masahiro & Shin, Jungpil & Chang, Won-Du. (2009). Object Detection using Histogram of Oriented Gradients.
8. KMN Syed Ali Fathima1 DR.K. Merriliance, "SVM with Hog Based on Classification Using Vehicle's Different Viewpoints "
9. S. Gharge, A. Patil, S. Patel, V. Shetty and N. Mundhada, "Real-time Object Detection using Haar Cascade Classifier for Robot Cars," 2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2023, pp. 64-70, doi: 10.1109/ICESC57686.2023.10193401.
10. S. Madhogaria, P. M. Baggenstoss, M. Schikora, W. Koch, and D. Cremers, "Car detection by fusion of HOG and causal MRF," in IEEE Transactions on Aerospace and Electronic Systems, vol. 51, no. 1, pp. 575-590, January 2015, doi: 10.1109/TAES.2014.120141.
11. Histogram of Oriented Gradients explained using OpenCV Histogram of Oriented Gradients explained using OpenCV. Url: https://learnopencv.com/histogram-of-oriented-gradients/
12. Support Vector Machines. Url: https://scikit-learn.org/stable/modules/svm.html
13. YOLOv4 .Url : https://paperswithcode.com/method/yolov4
14. OpenCV Haar Cascades. Url: https://pyimagesearch.com/2021/04/12/opencv-haar-cascades/