Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

Exploring Object Detection Methods for Autonomous Vehicles Perception: A Comparative Study of Classical and Deep Learning Approaches

Zobeir Raisi*, ValiMohammad Nazarzehi, Rasoul Damani, and Esmaeil Sarani

Electrical Engineering Department, Faculty of Marine Engineering, Chabahar Maritime University, Chabahar, Iran.

Article Info

Article History:
Received 25 February 2024
Revised 08 April 2024
Accepted 05 June 2024

DOI:10.22044/jadm.2024.14241.2529

Keywords:
Vehicle Perception, Pedestrian Detection, Deep learning, classical Machine Learning, Histogram of Oriented, Gradients.

**Corresponding author: zobeir.raisi@cmu.ac.ir (Z. Raisi).*

Abstract

This paper explores the performance of various object detection techniques for autonomous vehicle perception by analyzing classical machine learning and recent deep learning models. We evaluate three classical methods, including PCA, SIFT, and HOG alongside different versions of the SVM classifier, and five deep-learning models, including Faster-RCNN, SSD, YOLOv3, YOLOv5, and YOLOv9 models using the benchmark INRIA dataset. The experimental results show that although classical methods such as HOG + Gaussian SVM outperform other classical approaches, they are outperformed by deep learning techniques. Furthermore, Classical methods have limitations in detecting partially occluded, distant objects and complex clothing challenges, while recent deep-learning models are more efficient and provide better performance (YOLOv9) on these challenges.

1. Introduction

An autonomous car is a vehicle that uses sensors, cameras, radars, or a combination of them to travel between destinations without human supervision. These cars need perception, defined as a description of the visual cognition process for the vehicle. However, visual cognition is incredibly demanding, and this idea is currently one of the most significant open problems within the fields of autonomous driving, machine learning, robotics, and computer vision. Perception technologies are divided into two main categories: classical machine-learning approaches [5, 7, 17, 19, 20, 44, 45] and deep-learning computer-vision approaches [4, 17, 18, 36, 50, 55].

Classical machine learning techniques [7, 17, 19, 20, 44, 45] use data to find the best solution to a particular problem by identifying common features in the data associated with the correct response. These approaches are limited based on the data used to train the system and good performance will be achieved if real-world conditions match the training data. On the other hand, computer vision techniques usually rely on numerical optimization to find the best solution. They have a rational performance and are typically generalizable across various scenarios and conditions. For example, [5, 45] uses machine learning techniques to extract handcrafted HOG features from the input images and SVM to classify pedestrians with rectangular bounding boxes.

Most of the recent approaches [4, 50, 54-57] in autonomous driving are inspired by deep learning architectures such as SSD [26], Faster R-CNN [32], and YOLO [32] and achieved superior performance in several benchmark datasets [7, 23]. Generally, an autonomous perception problem may be classification, localization, or object detection. In an image classification problem, the algorithm decides whether the image is a predefined class or not, and the label is only the indicator of the class category. On the other hand, in an object detection problem, the algorithm predicts the location of the object as well as its presence [22]. To change the image classification to object detection, many works split the image into windows and apply image classification in each window.

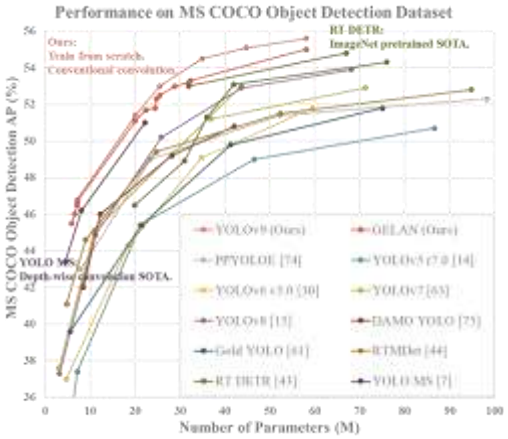

Figure 1. The performances of real-time deep learning object detectors on the benchmark COCO dataset. The image is taken from [43].

Then they slide the windows through the image and repeat the same procedure. As the method suggests, this approach is called Sliding Windows [20]. The field of autonomous perception includes many challenges. These challenges consist of problems in the training set like different poses, low resolution for far objects, variable appearance and clothing, complex backgrounds, etc. Moreover, variations in illumination affect the appearance of an object in an image [17, 18, 20]. Changing the direction of illumination leads to shifts in the location and shape of shadows, changes in highlights, and reversal of contrast gradients. In addition, partial occlusion as a common problem in real-world images is one of the most important difficulties in object detection. This issue happens when a part of the predefined object is blocked by another object. Dollar et al. [8] demonstrated that detection performance drops significantly under partial occlusion and even more drastically under heavy occlusion.

With the advance of deep learning methods several object detectors [3, 18, 49, 52] are proposed that achieved good performance in benchmark datasets [10, 11, 23]. Some methods benefit from two stages of detection as in [33, 51] and some methods [7, 26, 30] detect the objects in the given image in one stage. The Yolo series [2, 21, 30, 31, 41-43] are well-known one-stage pipelines that aim to perform object detection and localization in a single pass through the network. These architectures divide the image into a grid of cells and predict bounding boxes and class scores directly for each cell. Typically, one-stage methods are faster and more suitable for real-time applications due to their single-pass nature. Figure 1 compares recent real-time object detection on the benchmark MS COCO dataset [23].

In this paper, we explore the effect of two methods for pedestrian detection using the INRIA benchmark dataset [7]. We employ the traditional approach, which involves HOG feature extraction paired with SVM classification [6], alongside modern deep learning techniques like Faster RCNN [33], SSD [26], and YOLO [32]. We conduct an ablation study by providing quantitative and qualitative results and comparing the performance of these models and their inference time using the standard and commonly used evaluation metric, namely Mean Average Precision (mAP). The rest of this paper is structured as follows: Section 2 provides a literature review of related work. Section 3 outlines the utilized classical HOG+SVM technique and deep-learning YOLOv3 algorithm. Section 4 discusses experimental results, and Section 5 concludes the paper.

2. Related Work

In this section, we study different classical and deep learning models, discussing the advantages and disadvantages of each. Finally, we select the two best object detection methods from both fields.

2.1. Classical Methods

The classical machine learning methods use handcrafted features to detect objects in the input image. The most well-known algorithms used in the classic machine learning approaches are Haar Wavelets [39], Scale Invariant Feature Transform (SIFT) [25], and Histogram of Oriented Gradients (HOG) [7], which are described as follows:

2.1.1. Haar Wavelets

The Haar Wavelet method applies object detection based on Wavelet transformation, which is similar to Fourier transforms [29]. The Haar wavelet-based cascade framework [39] provides an efficient extension to the sliding window approach by introducing a degenerate decision tree of increasingly complex detector layers in which each layer employs a set of non-adaptive Haar wavelet features [27]. These features are expressed in different scales and locations, comprising horizontal and vertical descriptors, corresponding tilted features, as well as point detectors. Figure 2 shows the basic overview of the Haar-Wavelet operation. The most important advantage of the Haar-Wavelet algorithm is its simple mathematical representation and fast evaluation [40]. However, it is not suitable for detection in cases where a complex background exists [47]. Because autonomous perception samples usually consist of complex backgrounds, this approach is not selected in the paper.

Figure 2. Overview of the Haar wavelets [9]

2.1.2. SIFT

Scale Invariant Feature Transform (SIFT) descriptor [25] selects a 16x16 image and then divides it into 4×4 windows. Over each of these 4 windows, a histogram of gradients is computed. While computing this histogram, it also performs an interpolation between neighboring angles. Once all the 4x4 windows are obtained, it uses a Gaussian of half the window size, centered at the 16x16 block to weigh the values in the whole 16x16 descriptor. Considering the formulation, the SIFT algorithm gives separate importance to each point of the image. Although successful in many cases, [34, 35, 48], it imposes large computational time and is unsuitable for autonomous perception. If any algorithm can convert the point importance to overall image importance, it can outperform SIFT, in terms of test time.

2.1.3. HOG

The Histogram Oriented Gradient (HOG) algorithm is a simpler representation of SIFT introduced by [7], which is used as a feature extraction method for Human and pedestrian detection. This method aims to describe an image by a set of local histograms. These histograms count occurrences of gradient orientation in a local part of the image. The difference between HOG and SIFT is that contrary to point gradient evaluation, HOG uses horizontal and vertical gradients similar to edge detection filters. This results in a better outline of the object, meaning that it captures the gradient structure that is characteristic of the human shape, and it proposes much less computational complexity.

HOG also has several other advantages as well, and it is highly generalizable and its features are invariant to illumination and local geometrical changes, which makes it a suitable person detection algorithm [38, 46, 51]. A study in [1, 5] showed that HOG has better performance in pedestrian detection problems, in terms of both detection and accuracy rate when using a linear SVM classification. Also, the original paper [7] on HOG compared its performance to other feature extractors and illustrated that it outperforms all of them in terms of miss rate (Figure 3). Due to its formulation, HOG can solve specific challenges in autonomous perception. This formulation and the way it helps to solve those challenges will be discussed in the methodology section (see section 3 for more details).

Classical machine learning approaches in autonomous vehicle (AV) perception and pedestrian detection have several limitations. They rely on hand-engineered features. These approaches often struggle to generalize to new environments and scenarios, leading to reduced accuracy and robustness. Moreover, they may not effectively handle occlusions, variations in lighting and weather conditions, and multiple pedestrians with diverse poses and orientations. These drawbacks underscore the need for more advanced and robust techniques, such as deep learning-based methods, to enhance performance and reliability.

2.2. Deep Learning Models

Since the advent of GPUs and computationally advanced processors, deep learning models have received huge attention in the machine learning field.

2.2.1 Convolutional Neural Networks (CNNs)

A special computer vision-related subcategory of these models is Convolutional Neural Networks (CNNs). Convolutional Neural Networks (CNNs), introduced in [51], are one of the special kinds of deep NNs highly suitable for applications related to images and videos. Generally, a CNN takes the input and calculates the output by optimizing the weights of an equation expressed in multiple layers. The problem with conventional NNs is that when the input size is big, the number of weights increases exponentially and the optimization problem takes a lot of time to be solved.

Since the inputs of a computer vision problem are image pixels, the input size is usually considerably big, and CNNs take a lot of time to converge. However, CNN reduces the number of weights by applying convolutional layers to the input, and it does not differentiate between the image pixels and other inputs. In other words, it cannot capture the spatial and temporal characteristics of an image or a video. On the other hand, CNNs use image-related filters that can extract the image-related features. In each convolutional layer, specific kernels/filters are passed through the image, and new input is extracted from the original inputs. These kernels are chosen based on the problem and usually change from simple edge extractors to more advanced filters as the input goes through the deep learning model. The advanced filters extract more complicated features. Different deep learning models that are suitable for object detection are studied in this paper and eventually, the YOLOv3 is selected. The advantages and disadvantages of each method are mentioned and the rationale behind choosing YOLOv3 is discussed.

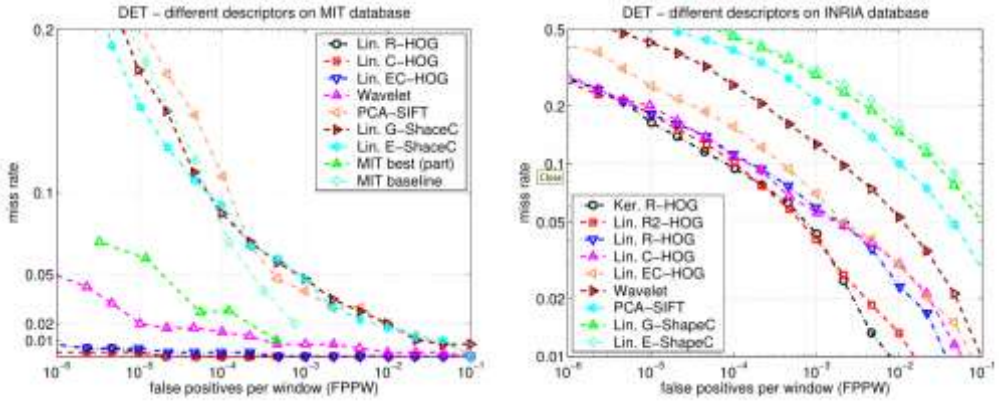

Figure 3. Performance comparison between hog and other classic feature extractors [7]. Best Viewed when zoomed in.

2.2.2 R-CNN

One of the important issues associated with object detection with classic methods is the concept of sliding a window through the whole image and using the classifier for each of those windows. This issue makes the test process very time-consuming and not practical for real-time implementation. To address this issue, the paper in [13] proposed the R-CNN method. In this approach, instead of considering all the windows, 2000 regions of the image are chosen and fed as the input vector to the CNN; these regions are called "Region Proposals". The output of the feature extraction is then put into an SVM classifier (same as classic methods). The classifier predicts the width and height of the bounding box and the offset value for the center of the object concerning the region proposal [14]. However, R-CNN object detection has slow inference times due to its multi-stage pipeline. This makes it unsuitable for real-time applications like pedestrian detection and may result in missed detections or inaccuracies.

2.2.3 Fast R-CNN

Several updates have been applied to R-CNN to make it a better choice than R-CNN. Instead of feeding the 2000 region proposals to the CNN, [12] fed the whole image into the convolution part to get the whole feature map and then selected the sections associated with the region proposals from the feature map, after the convolution part. The region of interest pooling, which is somehow similar to applying the max pooling layer to the region proposals, is applied to the dataset and the reshaped vector is then fed to the fully connected layer. This algorithm is much faster than R-CNN because the 2000 convolution operations are changed to only one.

Nevertheless, Fast R-CNN suffers from slow inference speed due to its multi-stage pipeline such as proposal generation, RoI pooling, and classification. This makes it less suitable for real-time pedestrian detection, where speed is one of the important parameters. Additionally, the RoI pooling module might struggle with small or crowded pedestrians, impacting accuracy.

2.2.4. Faster R-CNN

Faster R-CNN is a popular object detection technique used in many applications. However, the selective search algorithm used in this method is time-consuming, and the overall test time can be reduced by applying another method for finding the region proposals. To address this issue, Faster R-CNN utilized a technique that involves applying convolution operation on the whole image instead of using a search algorithm for finding the region proposals. Another network is also used to give the regions as its prediction.

Although Faster R-CNN outperforms R-CNN in terms of inference time and accuracy, it has a trade-off. The two-stage architecture, involving sequential region proposal generation and object classification, introduces computational complexity. This complexity, coupled with the slower performance compared to single-stage detectors, particularly hampers its suitability for real-time pedestrian detection tasks.

To achieve a better inference time and higher frames per second (FPS), alternative object detection approaches such as Single Shot MultiBox Detector (SSD) or You Only Look Once (YOLO) are proposed and followed by pedestrian detection researchers, as they offer faster inference speeds and improved performance under challenging conditions.

2.2.5 Single Shot MultiBox Detector (SSD)

The Single Shot MultiBox Detector (SSD) [26] is a faster and more efficient object detection compared to the Faster R-CNN. Unlike Faster R-

CNN, SSD detects objects in a single pass and eliminates the need for region proposal generation and secondary processing. SSD's architecture consists of a single stage that predicts object locations and classes directly from full images, making it highly efficient for real-time applications. However, SSD struggles with accuracy performance in terms of mAP, particularly for occluded and small objects or those with complex backgrounds, and requires an optimized anchor box.

2.2.6. You Only Look Once (YOLOv1)

Contrary to previous object detection deep learning algorithms, In the YOLO algorithm, all the image is passed through a CNN, not part of an image [32]. On top of that, similar to Faster R-CNN, CNN is used only once for determining the bounding boxes and their probability.

The YOLO algorithm redefines the detection problem as a single regression problem which maps image pixels to bounding boxes and class probabilities. It has the following advantages over the other well-known detection algorithms: (1) Due to the one-time implementation of the classifier, the YOLO algorithm is extremely fast and can process images at the rate of 45 frames per second. This characteristic makes it perfectly suitable for real-time implementation and online autonomous vehicle perception. (2) The YOLO algorithm uses the global representation of the image since it takes the whole image as the input to the CNN.

This makes it less sensitive to parts of the image that are not important for detection. One of these parts is the background. As mentioned in the R-CNN section, this algorithm sometimes takes background patches as the objects. This is because only the selected regions are trained in CNN. YOLO, on the other hand, is not susceptible to trivial parts of the image. Therefore, the challenge of complex backgrounds can be solved with this method. (3) The YOLO algorithm is highly generalizable. Despite other algorithms that use the same distribution for training and testing, YOLO learns the general representation of each object, not along with the data, but contextually. As a result, it performs well on unexpected inputs and different test environments. Despite the many advantages that YOLOv1 has, it has some disadvantages as well. Generally, YOLOv1 makes more localization errors than Faster R-CNN. Also, each grid cell is constrained to predict two bounding boxes and only one object. This spatial constraint results in the problem that YOLOv1 cannot localize small objects.

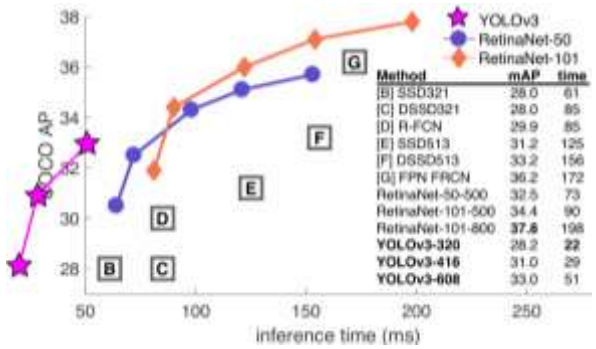

Figure 4. Performance comparison of YOLOv3 with other detectors on the benchmark COCO dataset [23].

Moreover, if the objects are very close to each other, this algorithm faces difficulties in dealing with them. Additionally, YOLOv1 works well with single-scale, meaning that if the training is on small objects, it cannot perform well on larger objects and vice versa. The mentioned issues may make YOLO an inappropriate candidate for this paper but these problems were addressed and completely solved in the next generations of YOLO algorithm.

2.2.6. YOLOv2

YOLOv2, also famous as YOLO 9000, uses tricks and modifications to solve the issues of the previous generation [30].

First, batch normalization is applied to the convolution layers of YOLOv1. This trick increases the mean accuracy precision and removes the need for dropouts. It also improves the convergence, which then decreases the probability of unstable gradients [16]. Second, YOLOv2 uses Anchor Boxes for predicting the bounding boxes.

The difference between this method and the one used in YOLOv1 is that, in the previous generation, the coordinates of the bounding boxes were predicted directly. However, in YOLOv2 the offsets of the anchor boxes are predicted. This change removes the need for spatial locations and improves the accuracy. By the use of anchor boxes, the number of predicted boxes changes from 98 to 1000. Interestingly, instead of hand-picking the anchor boxes, different boxes are cross-validated using Dimension Clusters. The k-means clustering algorithm is used for determining the best set of anchor boxes. Third, YOLO 9000 trains with high-resolution inputs as well, which has the benefit of adjusting to the new input resolution and increasing the accuracy. Fourth, YOLOv2 uses fined-grained features to account for smaller objects. The final difference that YOLO 9000 imposes is that it exploits multi-scaling training to solve the problem of single-scale sensitivity and the different scaling challenges in autonomous perception.

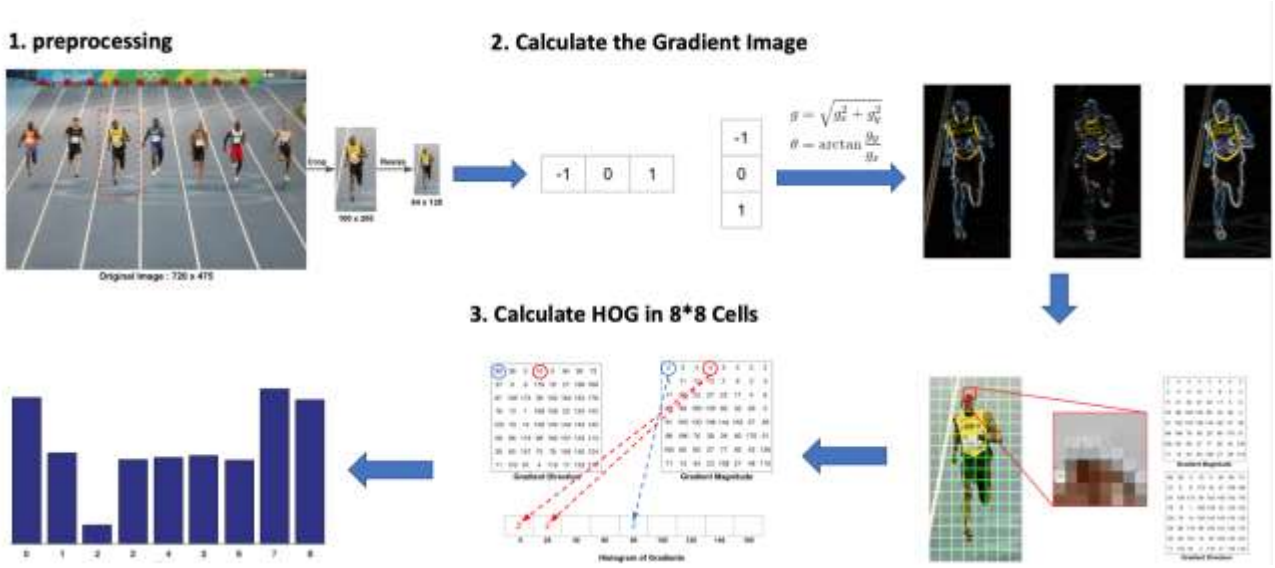

Figure 5. HOG process.

2.2.7. YOLOv3

The most recent generation of YOLOv3 uses approximately the same formulation as YOLOv2 [31] (see section 3 for more details). YOLOv3 is faster than the previous generations. Performance comparison of the well-known object detection algorithms has been published in the original paper of YOLOv3 [31] and it is shown that YOLOv3 has better speed and one of the highest mAPs (Figure 4). Moreover, understanding how it works and the problems it can solve showed that, like the HOG+SVM method in classic algorithms, YOLOv3 can resolve many of the mentioned issues in autonomous vehicle perception. Consequently, this algorithm is chosen for the deep learning implementation.

It is worth mentioning that there are also other recent versions of YOLOS as described in [31, 41-43]. For a fair comparison with the classical approaches, we only compared our results with YOLOv3. However, we conducted some experiments with other YOLO version models and compared their performance with YOLOv3 (See Table 1).

3. Methodology

In this section, we present the classical and deep learning algorithms utilized for comparison in the evaluation, as discussed in Section 4. For the classical method, the proposed method is described in Figure 5 and contains multiple sub-modules for feature extraction and classification that are described as follows:

3.1. The proposed Classical Machine Learning Techniques

In this section, we first provide the process of utilizing the classical machine learning approach including feature extraction and classification.

3.1.1 HOG

To obtain a complete descriptor of an infrared image, a local histogram of gradients is computed according to the following steps: 1) computing gradients of the image, 2) building a histogram of orientation for each cell and 3) normalizing histograms within each block of cells.

In the HOG descriptor, the input image is divided into small spatial regions (cells), and for each cell, a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell is accumulated. After preprocessing and obtaining the 64×128 patch of images, the gradient of each image is calculated by passing two one-dimensional kernels through the image.

3.1.2. Calculating Histogram of Gradients in 8x8 cells

Each gradient includes a value and a phase. In this step, the image is divided into 8×8 cells and for each cell, the histogram of gradients is calculated by accumulating votes into bins for each orientation. The phases are divided into 9 bins (0 to 180) and the values for each bin are assembled. If the phase of a gradient is between two bins, its value is distributed between those bins based on a weighted approach.

3.1.3. Block Normalization

Due to the variability in the images, it is necessary to normalize cell histograms. They are locally normalized, according to the values of the neighborhood cell histograms. The normalization is done among a group of cells, which is called a block. The figure shows the process of HOG calculation.

3.1.4. Calculating the HOG feature vector

According to the following formulation: HOG feature vector = 7(horizontal position) * 15 (vertical position) * 4 (cell) * 9 (histogram bin) = 3780 feature descriptors are obtained for each image in the HOG algorithm. Each 64 × 128 detection window is represented by 7 × 15 blocks, giving a total of 3780 features per detection window. These features are then used to train a linear SVM classifier [37].

HOG formulation makes it one of the best classic methods for autonomous perception challenges: HOG discretizes the phase space into 9 bins. By doing so, slight changes in angle will not lead to a different histogram. Therefore, HOG is robust to different poses and angle changes. Batch normalization makes HOG robust to multiplication. In other words, if all the pixel values are multiplied by a number, the gradient of those values remains the same. This is very useful because contrast differences in an image act exactly like multiplying some of the values by a number. As a result, by being robust to multiplication, the HOG is robust to contrast change and can solve the illumination challenge in autonomous perception. To solve the problem of different scales, the image is downsized multiple times and the HOG is applied with a constant-size sliding window. This way, smaller and bigger (closer and further) objects can be detected. A variable called HOG-threshold is assigned to determine the number of downsizing steps.

3.1.5. SVM Classifier

In this paper, a set of training images with positive and negative samples, are first described by their HOG, for learning a decision function. A Support Vector Machine classifier (SVM) has been used, which is a binary classification algorithm that looks for an optimal hyperplane as a decision function in a high-dimensional space. Thus, considering one has a training data set: $x_k, y_k \in X \times \{-1,1\}$, where x_k is the training examples HOG feature vector and y_k is the class label. At first, the method maps x_k in a high dimensional space owing to a function Φ. Then, it looks for a decision function of the form: $f(x) = w\Phi(x) + b$ and $f(x)$ is optimal in the sense that it maximizes the distance between the nearest point $\Phi(x_i)$ and the hyperplane. The class label of x is then obtained by considering the sign of $f(x)$. This optimization problem can be turned, in the case of the L1 soft-margin SVM classifier (misclassified examples are linearly penalized), in the following way:

$$\min_{w,\zeta} \frac{1}{2} \| w \|^2 + C \sum_{k=1}^{m} \zeta_k \quad (1)$$

under the constraint $\forall k, y_k f(x_k) \geq 1 - \zeta_k$ The solution to this problem is obtained using the Lagrangian theory and it is possible to show that the vector w is of the form:

$$w = \sum_{k=1}^{m} \alpha_k^* y_k \Phi(x_k) \quad (2)$$

Where α_i^* is the solution to the following quadratic optimization problem:

$$\max_{\alpha} W(\alpha) = \sum_{k=1}^{m} \alpha_k^* - \frac{1}{2} \sum_{k,l}^{m} \alpha_k \alpha_l y_k y_l K(x_k, x_l) \quad (3)$$

subject to $\sum_{k=1}^{m} y_k \alpha_k = 0$ and $\forall k, 0 \leq \alpha_k \leq C$ where $K(x_k, x_l) = \prec \Phi(x_k), \Phi(x_l) \succ$. According to (2) and (3), the solution of the SVM problem depends only on the Gram Matrix K. Data sets are not always linearly separable. In the case of nonlinear datasets, we can use kernel functions to paper the dataset into a higher dimensional space in which data is linearly separable. However, in the case of pedestrian detection problems, typical linear SVM is sufficient to get a high detection rate. Using the kernel function results in a higher detection rate and decreased false positives, but it takes more computational processing time.

3.2. The Leveraged Deep Learning Techniques

In this section, we present the utilized deep learning techniques for evaluation. We used several well-known and recent models to compare their performance with classical machine learning techniques. As we mentioned in Section 2, two-stage detectors like Faster R-CNN are not suitable for real-time object detection without high computation machine equipped with high memory GPU. Although the SSD architecture is fast, but it does not provide a high accuracy. Different versions of YOLO models are available as open source and easy to modify, extend, and provide better efficiency and good performance.

Among different versions of YOLO models, the recent models (YOLOv8, YOLOv9) performed

better in terms of accuracy and efficiency but have complex architecture compared to the first YOLO generations (YOLOv1 and YOLOv2). In this section, we provide a detailed description of YOLOv3 and select it as the main pipeline for comparison due to its trade-off role in the straightforward pipeline and efficient performance for a fair comparison with classical methods.

3.2.1 YOLOv3

The YOLOv3 [31] algorithm is exploited in the deep learning section. Since this algorithm works in real-time and performs well for multi-class detection, other objects are added to the pedestrian category. One of the important changes in the YOLOv3 is that instead of using SoftMax in the fully connected layer, logistic regression is used. The advantage of this modification is that it produces good results when the classes are not mutually exclusive, e.g. when the class labels include "pedestrian" and "man". Additionally, YOLOv3 uses a similar feature extraction strategy as Feature Pyramid Networks [24]. Three-scale predictions are made for each location of the input image in grid cells and features are extracted from those predictions. This helps even further for different-scale classification.

Each image is divided into 13×13 grid cells and each grid cell is responsible for detecting the objects whose center points are located in it. The output for each object should be the width and height of the box, the center of the box, and the probability of the class in the box. Although used in classic detection methods, predicting the (x, y) location of the box makes the formulation unconstrained and unstable [30]. This is because the anchor box can be in any section of the image regardless of the predicted box location. Instead of this formulation, the ground truth is normalized by the image width and length, and the top left corner of the image and the bounding box prior are employed to keep the value between 0 and 1. Additionally, a logistic activation function is operated on the dimensions to keep the output in the range (See Figure 6). These predicted values can be found in (4):

$$b_x = \sigma(t_x) + c_x, b_y = \sigma(t_y) + c_y, b_w = p_w e^{t_w}, b_h = p_h e^{t_h} \quad (4)$$

In (4), p_w and p_h are the dimensions of the anchor, c_x and c_y are the top left corner of the dimension of the anchor. The grid cell also predicts the objectness score for each object, which determines the probability of being the selected object (5). To avoid multiple detections of an object (a problem faced in classic methods), a method called *Non-Max Suppression* [28] is applied.

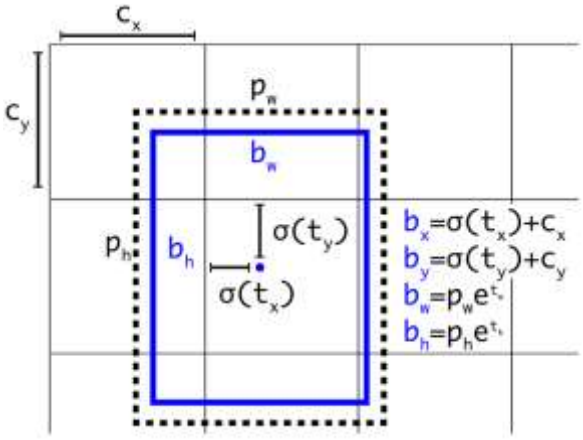

Figure 6. The dimensions and the center of the bounding box

In this approach, the prediction with the highest objective score is picked and the bounding boxes with more than 50% intersection with it is discarded. Considering three scales for each location and a total of 80 class categories, the prediction output for each image is $13\times13\times\left[3\times(5+80)\right]$.

$$\text{Objectness} = \Pr(Obj) * IoU \quad (5)$$

To account for mutually dependent classes, a hierarchical approach is exploited. The conditional probability of each class is computed given that an object is selected. The global probability of the class is calculated using hierarchical propagation. For example, given that a pedestrian can be a man, woman, or child, the global probability of a pedestrian is as follows:

$$\begin{aligned}\Pr(Ped) &= \Pr(Ped \mid Man)\times\Pr(Man)\times IoU_{t_l} \\ &+ \Pr(Ped \mid Woman)\times\Pr(Child)\times IoU_{t_l} \\ &+ \Pr(Ped \mid Child)\times\Pr(Child)\times IoU_{t_l}\end{aligned} \quad (6)$$

Where IoU denotes the intersection over union, and t_l is true to label in the ground truth. YOLOv3 uses the Darknet-53 architecture with 53 convolutional layers in its network. With the use of batch normalization, overfitting is avoided, and dropout methods, in which some of the neurons in the fully connected layers are turned off randomly [15], can be discarded. Binary class entropy is utilized as the loss function, instead of the sum of squared error that was previously utilized in YOLOv1.

4. Experimental Results

In this section, we first introduce the dataset used for evaluation. Then, we provide the implementation details of the training and inference settings of the models. Finally, we provide some quantitative and qualitative results to compare the classical and deep-learning methods.

Table 1. The mean average precision and the test time for classic methods.

Method	mAP	Average test time (s)
PCA+SVM	75.23	90.32
HOG + Linear SVM	79.95	96.12
HOG + Gaussian SVM	82.0	104.58
Faster-RCNN	86.23	203.9
SSD	84.11	31.23
YOLOv3	86.31	15.61
YOLOv5	89.78	10.21
YOLOv8	92.35	9.27
YOLOv9	**96.51**	**8.96**

4.1. INRIA Dataset

The INRIA Person dataset [7] comprises images utilized for pedestrian detection tasks, encompassing 614 instances for training and 288 for testing purposes. This dataset serves as a foundational resource in the domain of pedestrian detection, facilitating the development and evaluation of detection algorithms across varied scenarios and conditions.

4.2. Implementation Details

The training and test procedure is done on an Intel(R) Core™ i7-6500U CPU 2.5 GHz. HOG Threshold=0.2 has been chosen for feature extraction. Cross-validation has been applied to find the best C for the SVM and $C=1$ has been selected. Both linear and Gaussian kernels have been tested on the dataset. The linear SVM is written in MATLAB without the SVM toolbox and for Gaussian SVM, the FITCSVM toolbox is utilized. It is worth mentioning that the main goal of the manuscript is to evaluate and compare different classical machine and deep learning methods. The main contribution of the proposed method is the presentation of HOG + Gaussian SVM. For other methods, we utilize, with minor modifications, the models of the researchers available from their GitHub pages.

For YOLOV3 deep learning, a pre-trained model of YOLOv3 is used in the DarkNet-53 structure to evaluate its performance on the INRIA test set. Since YOLOv3 has a good performance on multi-class label detection, 80 class categories are set for the inference. Hence, other objects are added to the single-class pedestrian category. For a fair comparison, similar to the classic method, the inference section is done on the CPU using a Ubuntu 18.0 OS. For other YOLO models used in the experiments, a pre-trained model on the MS-COCO dataset is used for evaluation with the same setting of YOLOv3 using the statistical setting proposed in [53].

Figure 7. HOG+SVM classic method on INRIA dataset.

4.3. Evaluation Metrics

Mean Average Precision (mAP) is a widely used evaluation metric in object detection tasks, including pedestrian detection. It provides a comprehensive assessment of the precision-recall trade-off across different confidence thresholds. In this paper, we adopted the same criteria as used in [7, 19, 20] for evaluating the performance of the models.

4.4. Classical method's results

The HOG feature extraction and SVM classifier are applied to the INRIA dataset [7]. The choice of this dataset is because it contains all the intended challenges in autonomous perception. Also, PCA feature extraction with SVM classification has been utilized to compare the results. A POV method determines the number of features to be extracted and it selects 83 best pixel values. The mean average precision and the average test time is shown in Table 1. Also, some of the test samples with challenging problems are depicted in Figure 7. These problems include illumination, clothing, far objects, crowded backgrounds, and different scales. Red squares represent the bounding boxes. Each object is detected multiple times as the image is downsized. This is because non-max suppression is not used in the classic method and the issue is resolved in deep learning results. As shown, the classic method can solve most of the problems but struggles to detect far objects, partially occluded objects, and objects with complex clothing. These missed objects are shown with yellow squares in Figure 7. The PCA+SVM has the lowest test time since it does not calculate the gradients for each grid cell. However, its mAP is considerably lower on the INRIA dataset. HOG + Gaussian SVM improves the mAP of linear SVM by 4%, but it

suffers from very high computational time. Generally, HOG + Linear SVM has the best trade-off between accuracy and computation time and therefore, is chosen as the classic method, even though the structure of the problem is nonlinear. The mean average precision and the test time for classic methods.

4.5. Deep Learning Method's Results

To evaluate deep-learning techniques, we utilized different two-stage and one-stage object detectors. As seen from Table 1, the two-stage Faster R-CNN detector provides good performance in terms of mAP by achieving (86.23). It outperformed the classical methods and surpassed the SSD model. However, taking about 204 seconds in the inference time shows that it is not efficient for real-time applications. This is because Faster R-CNN's architecture has multiple stages of processing and several detection modules that make it computationally expensive and it is not suitable for real-time applications.

On the other hand, the one-stage detector SSD model performed better in inference time (~31.23 seconds) than Faster-RCNN while falling behind other YOLO-based deep-learning detectors by achieving 84.11% in terms of mAP performance.

The YOLOv3 algorithm achieved 86.31% mAP. The mean test time is 15.61 seconds. Although the model is still far from the order of milliseconds and real-time implementation, the test time has been reduced by a factor of 6.15 compared to classic methods. Real-time implementation would be achieved if GPU was used.

More importantly, YOLOv3 can solve partial occlusion, far objects, and clothing problems. The images from Figure 7 are also represented in Figure 8 (pedestrians with mentioned challenges are shown with yellow squares). From Table 1, it is evident that the recent model of YOLOs perform better than the previous generations. However, these recent models have complicated architecture and require extensive computations to achieve better performance than the YOLOv3.

5. Conclusions

In conclusion, our investigation into pedestrian detection techniques highlights the effectiveness of both classical methods, such as HOG feature extraction with SVM classification, and modern deep learning approaches like YOLO models. By evaluating these methods on the INRIA benchmark dataset, we contribute insights into their respective strengths and potential applications in real-world scenarios.

Figure 8. Object Detection with YOLOv3.

HOG+ linear SVM achieved a mean average accuracy of 79.95% with a mean average test time of 96.12 seconds, effectively addressing various challenges in autonomous perception such as illumination and scale variations. However, it encountered difficulties with samples featuring partial occlusion, distant objects, and complex clothing, and was unsuitable for real-time implementation. Conversely, YOLOv3 demonstrated a ~ 7% increase in mean average precision and a 6.15-fold reduction in test time, effectively resolving different challenges. We also conducted some experiments on the utilized INRIA dataset on recent YOLO models, which outperformed the YOLOv3 with a high margin of mAP performance.

References

[1] T. Barbu, "Pedestrian detection and tracking using temporal differencing and HOG features," *Computers & Electrical Engineering*, Vol. 40, No. 4, pp. 1072–1079, 2014, doi:10.1016/j.compeleceng.2013.12.004.

[2] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020, doi: https://doi.org/10.48550/arXiv.2004.10934.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, pp. 213–229. Springer, 2020, doi: 10.1007/978-3-030-58452-8_13.

[4] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1907–1915, Honolulu, HI, USA, 2017, doi: 10.1109/CVPR.2017.691.

[5] Z. Chen, K. Chen, and J. Chen, "Vehicle and pedestrian detection using support vector machine and

histogram of oriented gradient features," in *2013 International Conference on Computer Sciences and Applications*, pp. 365–368, Wuhan, China, IEEE, 2013, doi: 10.1109/CSA.2013.92.

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, Vol. 20, pp. 273–297, 1995.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 886–893, San Diego, CA, USA, IEEE, 2005, doi: 10.1109/CVPR.2005.177.

[8] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 304–311, Miami, FL, USA, IEEE, 2009, doi: 10.1109/CVPR.2009.5206631.

[9] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 12, pp. 2179–2195, Dec. 2009, doi: 10.1109/TPAMI.2008.260.

[10] A. Ess, B. Leibe, and L. Van Gool, "Depth and appearance for mobile scene analysis," in *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, Rio de Janeiro, Brazil, IEEE, 2007, doi: 10.1109/ICCV.2007.4409092.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) challenge," *International Journal of Computer Vision*, Vol. 88, pp. 303–338, 2010, doi: 10.1007/s11263-009-0275-4.

[12] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, Santiago, Chile, IEEE, 2015, doi: 10.1109/ICCV.2015.169.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, IEEE, 2014, doi:10.1109/CVPR.2014.81.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, No. 9, pp. 1904–1916, Sept. 2015, doi: 10.1109/TPAMI.2015.2389824.

[15] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012. Available: https://arxiv.org/pdf/1207.0580.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, pp. 448–456, PMLR, 2015.

[17] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, Vol. 31, No. 3, pp. 685–695, 2021, doi: 10.1007/s12525-021-00475-2.

[18] R. Kaur and S. Singh, "A comprehensive review of object detection with deep learning," *Digital Signal Processing*, Vol. 132, p. 103812, 2023, doi: 10.1016/j.dsp.2022.103812.

[19] T. Kobayashi, A. Hidaka, and T. Kurita, "Selection of histograms of oriented gradient features for pedestrian detection," in *Neural Information Processing: 14th International Conference, ICONIP 2007, Kitakyushu, Japan, November 13-16, 2007, Revised Selected Papers, Part II 14*, pp. 598–607, Springer, 2008, doi: 10.1007/978-3-540-69162-4_62.

[20] K. Lei and Y. Luo, "A new pedestrian detection method based on histogram of oriented gradients and support vector data description," in *Electronics, Communications and Networks*, IOS Press, 2024, pp. 333–342.

[21] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, et al., "YOLOv6: A single-stage object detection framework for industrial applications," *arXiv preprint arXiv:2209.02976*, 2022, doi: https://doi.org/10.48550/arXiv.2209.02976.

[22] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware Fast R-CNN for pedestrian detection," *IEEE Transactions on Multimedia*, Vol. 20, No. 4, pp. 985–996, April 2018, doi: 10.1109/TMM.2017.2759508.

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755, Springer, 2014.

[24] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944, Honolulu, HI, USA, IEEE, 2017, doi: 10.1109/CVPR.2017.106.

[25] T. Lindeberg, "Scale-invariant feature transform," 2012. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2:480321.

[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37, Springer, 2016.

[27] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *IEEE Transactions on Pattern Analysis*

and Machine Intelligence, Vol. 23, No. 4, pp. 349–361, April 2001, doi: 10.1109/34.917571.

[28] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, pp. 850–855, Hong Kong, China, IEEE, 2006, doi: 10.1109/ICPR.2006.479.

[29] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 555–562, Bombay, India, IEEE, 1998, doi: 10.1109/ICCV.1998.710772.

[30] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525, Honolulu, HI, USA, IEEE, 2017, doi: 10.1109/CVPR.2017.690.

[31] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018. Available: https://doi.org/10.48550/arXiv.1804.02767.

[32] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, NV, USA, IEEE, 2016, doi: 10.1109/CVPR.2016.91.

[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137–1149, June 2017, doi: 10.1109/TPAMI.2016.2577031.

[34] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, pp. 2564–2571, Barcelona, Spain, IEEE, 2011, doi: 10.1109/ICCV.2011.6126544.

[35] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional SIFT descriptor and its application to action recognition," in *Proceedings of the 15th ACM International Conference on Multimedia (MM '07)*, pp. 357–360, Association for Computing Machinery, New York, NY, USA, 2007, doi: 10.1145/1291233.1291311.

[36] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, Vol. 6, No. 1, pp. 1–48, 2019, doi: 10.1186/s40537-019-0197-0.

[37] F. Suard, A. Rakotomamonjy, A. Bensrhair, and A. Broggi, "Pedestrian detection using infrared images and histograms of oriented gradients," in *2006 IEEE Intelligent Vehicles Symposium*, pp. 206–212, Meguro-Ku, Japan, IEEE, 2006, doi: 10.1109/IVS.2006.1689629.

[38] F. Suard, A. Rakotomamonjy, A. Bensrhair, and A. Broggi, "Pedestrian detection using infrared images and histograms of oriented gradients," in *2006 IEEE Intelligent Vehicles Symposium*, pp. 206–212, Meguro-Ku, Japan, IEEE, 2006, doi: 10.1109/IVS.2006.1689629.

[39] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 734–741, vol. 2, Nice, France, 2003, doi: 10.1109/ICCV.2003.1238422.

[40] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 734–741, vol. 2, Nice, France, 2003, doi: 10.1109/ICCV.2003.1238422.

[41] J. Reis, D. Dillon, J. Kupec, J. Hong, and A. Daoudi, "Real-time flying object detection with YOLOv8," *arXiv preprint arXiv:2305.09972*, 2023. Available: https://doi.org/10.48550/arXiv.2305.09972.

[42] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You only learn one representation: A unified network for multiple tasks," *arXiv preprint arXiv:2105.04206*, 2021. Available: https://doi.org/10.48550/arXiv.2105.04206.

[43] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning what you want to learn using programmable gradient information," *arXiv preprint arXiv:2402.13616*, 2024. Available: https://doi.org/10.48550/arXiv.2402.13616.

[44] T. Watanabe, S. Ito, and K. Yokoi, "Cooccurrence histograms of oriented gradients for pedestrian detection," in *Advances in Image and Video Technology: Third Pacific Rim Symposium, PSIVT 2009, Tokyo, Japan, January 13-16, 2009. Proceedings 3*, pp. 37–47, Springer, 2009, doi: 10.1007/978-3-540-92957-4_4.

[45] G. Xu, X. Wu, L. Liu, and Z. Wu, "Real-time pedestrian detection based on edge factor and histogram of oriented gradient," in *2011 IEEE International Conference on Information and Automation*, pp. 384–389, Shenzhen, China, IEEE, 2011, doi: 10.1109/ICINFA.2011.5949022.

[46] Y. Yamauchi, H. Fujiyoshi, B.-W. Hwang, and T. Kanade, "People detection based on co-occurrence of appearance and spatiotemporal features," in *2008 19th International Conference on Pattern Recognition*, pp. 1–4, Tampa, FL, USA, IEEE, 2008, doi: 10.1109/ICPR.2008.4761809.

[47] S. Yao, S. Pan, T. Wang, C. Zheng, W. Shen, and Y. Chong, "A new pedestrian detection method based on combined HOG and LSS features," *Neurocomputing*, Vol. 151, pp. 1006–1014, 2015, doi: 10.1016/j.neucom.2014.08.080.

[48] W.-L. Zhao and C.-W. Ngo, "Flip-invariant SIFT for copy and object detection," *IEEE Transactions on Image Processing*, Vol. 22, No. 3, pp. 980–991, March 2013, doi: 10.1109/TIP.2012.2226043.

[49] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 30, No. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[50] C. Zhou and J. Yuan, "Multi-label learning of part detectors for heavily occluded pedestrian detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3506–3515, Venice, Italy, IEEE, 2017, doi: 10.1109/ICCV.2017.377.

[51] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pp. 1491–1498, New York, NY, USA, IEEE, 2006, doi: 10.1109/CVPR.2006.119.

[52] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020. Available: https://doi.org/10.48550/arXiv.2010.04159.

[53] M.-W. Li, R.-Z. Xu, Z.-Y. Yang, W.-C. Hong, X. An, and Y.-H. Yeh, "Optimization approach of berth-quay crane-truck allocation by the tide, environment and uncertainty factors based on chaos quantum adaptive seagull optimization algorithm," *Applied Soft Computing*, Vol. 152, p. 111197, 2024, doi: 10.1016/j.asoc.2023.111197.

[54] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, Vol. 37, No. 3, pp. 362–386, 2020.

[55] V. Bharilya and N. Kumar, "Machine learning for autonomous vehicle's trajectory prediction: A comprehensive survey, challenges, and future research directions," *Vehicular Communications*, 2024, p. 100733, doi: 10.1016/j.vehcom.2024.100733.

[56] S. M. Ghazali and Y. Baleghi, "Pedestrian detection in infrared outdoor images based on atmospheric situation estimation," *Journal of AI and Data Mining*, Vol. 7, No. 1, pp. 1–16, 2019, doi: 10.22044/jadm.2018.5742.1696.

[57] M. Nasehi, M. Ashourian, and H. Emami, "Vehicle type, color and speed detection implementation by integrating VGG neural network and YOLO algorithm utilizing Raspberry Pi hardware," *Journal of AI and Data Mining*, Vol. 10, No. 4, pp. 579–588, 2022, doi: 10.22044/jadm.2022.11915.2338.

بررسی روش‌های تشخیص اشیاء برای استنباط وسایل نقلیه خودران: مطالعه‌ی مقایسه‌ای بین رویکردهای کلاسیک و یادگیری عمیق

زبیر رئیسی*، ولی محمد نظرزهی، رسول دامنی و اسماعیل سارانی

گروه مهندسی الکترونیک و مخابرات دریایی، دانشگاه دریانوردی و علوم دریایی چابهار، چابهار، ایران.

ارسال ۲۰۲۴/۰۲/۲۵؛ بازنگری ۲۰۲۴/۰۴/۰۸؛ پذیرش ۲۰۲۴/۰۶/۰۵

چکیده:

این مقاله به بررسی عملکرد تکنیک‌های مختلف تشخیص اشیا برای استنباط خودروهای خودران از طریق تحلیل مدل‌های یادگیری ماشین کلاسیک و مدل‌های یادگیری عمیق اخیر می‌پردازد. ما سه روش کلاسیک شامل SIFT، PCA و HOG به همراه نسخه‌های مختلفی از طبقه‌بندی‌کننده‌ی SVM و پنج مدل یادگیری عمیق شامل مدل‌هایFaster-RCNN ، SSD، YOLOv3، YOLOv5 و YOLOv9 را با استفاده از مجموعه داده‌ی استاندارد INRIA ارزیابی کرده‌ایم. نتایج آزمایش‌ها نشان می‌دهد که اگرچه روش‌های کلاسیک مانند HOG + Gaussian SVM از سایر روش‌های کلاسیک عملکرد بهتری دارند، اما توسط تکنیک‌های یادگیری عمیق پشت سر گذاشته می‌شوند. علاوه بر این، روش‌های کلاسیک در تشخیص اشیای نیمه‌پنهان، اشیای دور و چالش‌های مربوط به پوشش پیچیده محدودیت دارند، در حالی که مدل‌های یادگیری عمیق اخیر (YOLOv9) در این چالش‌ها کارآمدتر بوده و عملکرد بهتری ارائه می‌دهند.

کلمات کلیدی: وسیله نقلیه خودران، تشخیص عابر پیاده، یادگیری کلاسیک و عمیق، هیستوگرام شیب های جهت دار، گرادیان.