University of Baghdad
College of Engineering
JE
JOURNAL OF ENGINEERING

Journal of Engineering
journal homepage: www.jcoeng.edu.iq
Volume 30 Number 8 August 2024

Comparative Study of Different Classification Techniques for Pedestrian Detection Application

Tuqa Hani Abd-Alamir *, **Mohammed Sadoon Hathal**

Department of Computer Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq

ABSTRACT

Pedestrian detection is well known as one of the most important applications in computer vision. However, reliable pedestrian detection is difficult due to a variety of factors, including changing size of pedestrian characteristics and crowded backgrounds. This study aims to evaluate and compare the pedestrian detection performance of three different types of classifiers: Random-Forest (RF), Convolution-Neural-Network (CNN), and Support-Vector-Machine (SVM). The presented methodology involves using You_Only_Look_Once (YOLOv8) architecture for object segmentation and the Histogram of Oriented Gradients (HOG) for feature extraction. Then, RF, CNN and SVM classifiers are trained and tested using the extracted HOG features. Using the EPFL pedestrian dataset, the experiment showed that the CNN model returned the highest results which had a speed of 0.42s and an accuracy percentage of 93.34%. Compared to SVM and RF, CNN provides a high detection speed and accuracy. RF has the slowest detection speed, while SVM has the lowest detection accuracy. This study gives useful information regarding the efficacy of these classifiers in detecting pedestrians under various weather circumstances, and the findings show that CNNs can achieve high accuracy while maintaining remarkable detection efficiency.

Keywords: Histogram of oriented gradients, You only look once, Convolution neural network, Random forest, Support vector machine.

1. INTRODUCTION

The pedestrian detection application is one of the most significant applications in the field of image processing and computer vision because of its multiple applications, comprising self-driving cars, tracking, and counting applications **(Iftikhar et al., 2022)**. Consequently, it is needed to build an effective pedestrian detection system for pedestrian safety **(Zuo et**

*Corresponding author

Peer review under the responsibility of University of Baghdad.
https://doi.org/10.31026/j.eng.2024.08.10

Article received: 15/11/2023
Article revised: 18/02/2024
Article accepted: 18/02/2024
Article published: 01/08/2024

al., 2021). The main objective of designing pedestrian detection algorithms is to build a model that can automatically distinguish and determine the position of pedestrians, differentiating from the background elements or other things found in images or video **(Braik et al., 2020)**. However, there are several difficulties related to this task, for example, people's behavior and clothes changing moreover to several postures such as walking, running, or standing **(Chong and Tay, 2017)**. Additionally, the lighting changes, weather conditions, and viewing angle as well as occlusion, which happens when pedestrians hide behind each other or are hidden by surrounding elements of the environment like billboards or walls **(Ben Khalifa et al., 2020)**. Therefore, developing an algorithm that can address all these problems in real-time is not a simple task **(Chong and Tay, 2017)**.
Many researchers in recent years have studied pedestrian detection methods **(Iftikhar et al., 2022)**. Such methods exploit different detection algorithms, feature descriptors, classification methods, and computational complexity. Classical approaches have been used by some researchers, whereas deep learning methods have been preferred by others **(Wang, 2019)**. (**Seemanthini and Manjunath, 2018)** introduced a novel human detection approach that fuses the cluster segmentation method, Gabor feature extraction, and support vector machine (SVM) classifier. HOG descriptor was chosen as the feature for recognizing actions. Besides, the temporal information has been used to aid the tracking process. The suggested method is proved to be more effective than the most recent methods in terms of accuracy which equals 89.59%. (**Mateus et al., 2019)** propose a hybrid model for pedestrian detection and tracking in a robotic navigation example. The current model that was introduced comprises the Aggregate Channel Features and the Deep Convolution Neural Networks (D-CNN) for pedestrian detection and the Neighborhood Probabilistic Joint Data Association (NNJPDA) besides the Kalman_Filter for pedestrian tracking. The results indicated that the proposed model is both accurate and fast. **(Braik et al., 2020)** offer a pedestrian detection method using contour cues, edge-based features, color information, and cascaded RF classifier with CENTRIST visual descriptors. The algorithm is tested on three datasets of images with various resolutions. The experiment outcomes have indicated that the proposed pedestrian detector has outperformed the state-of-the-art method on many of the human datasets. **(Ben Khalifa et al., 2020)** suggested a moving camera-based approach for pedestrian detection. A block matching algorithm (BMA) obtains motion vectors from the area of interest (ROI). The rest blocks are divided into foregrounds and backgrounds depending on a threshold. Local_Binary_Pattern (LBP) is utilized for feature extraction and SVM is used for classification. Experimental results demonstrate that the accuracy of the current model is higher, but the time is slightly increased. (**Zhang et al., 2022)** introduced a two-stage pedestrian detection pipeline with Haar feature + AdaBoost classifier and HOG-LBP + SVM classifier. The results indicate that the proposed solution can effectively improve the performance of the detection pedestrians, achieving a detection rate of 89.73%.
The current work investigated three classification algorithms that are widely utilized for pedestrian detection. The proposed process incorporates object segmentation using the You_Only_Look_Once (YOLOv8) network, followed by feature extraction based on the HOG descriptor. The features were then used to train three different classifiers: CNN, SVM, and RF. Lastly, these classifiers were assessed to establish which of them is most suitable for the suggested pedestrian detection model. In regards to the previously published work on pedestrian detection, existing algorithms have mainly been aimed at improving their performance. However, designing an algorithm that produces high accuracy with low computing cost seems to be a challenge.

2. MATERIAL AND METHOD

The hardware and software components required to implement the proposed pedestrian detection system are listed below:
Hardware Component:
-Computer with sufficient processing power and memory (Dell, Cori 5,8th Gen).
Software Components:
-Google Colab Editor.
-Programing Language (Python).
-Image processing and ML algorithms (HOG, YOLOv8, CNN, SVM, RF).
-Libraries for image processing and ML techniques.
-Dataset Images.
The procedure of the proposed system is shown in **Fig. 1.** Initially, the image dataset is collected and processed by YOLOv8 for segmentation, resulting in segmented objects with specific class IDs. confidence score.

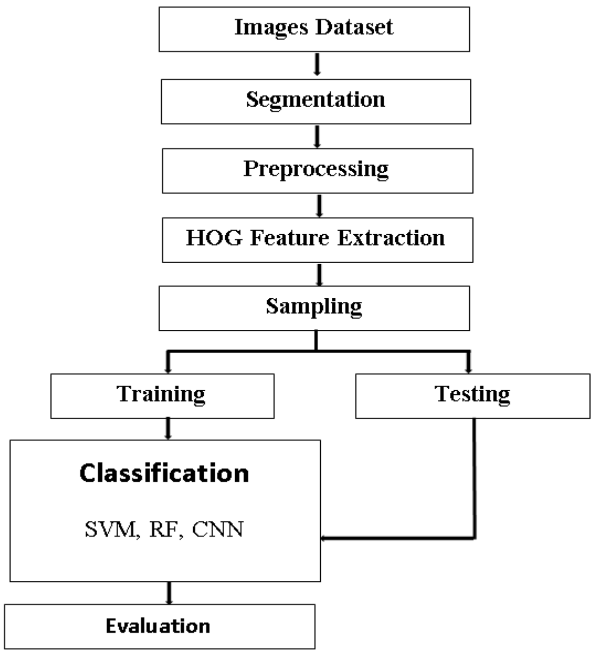


Figure 1. Block diagram of the proposed method.

Next, the segmented object undergoes pre-processing, which includes converting the RGB images to grayscale and resizing to adjust the height and width. Next, the HOG descriptor is used to extract features from the segmented regions. Features are then stored according to their class IDs; for example, pedestrian features are stored as positive features if the value of Class_id is 0. Otherwise; the features are stored as negative features. A new dataset is formed using positive and negative HOG features, which is subsequently utilized for training and testing classifiers (SVM, RF, and CNN). The system's output is a boundary box around the detected pedestrian with a label and confidence score.

2.1 Dataset

Experiments were conducted on the EPFL datasets comprising campus passengers and subway scenes, which exhibit various poses and illumination variations. The EPFL dataset consists of outdoor sequences with six videos for the campus scene and four videos for the passageway scene. Initially, videos are converted into frames, generating 50,000 images. These images are passed into YOLOv8 for object segmentation. Each segmented object is further processed using the HOG descriptor, resulting in the creation of a new feature dataset consisting of 87,260 object features. The dataset is divided into 56,710 positive and 30,550 negative features, each of size 64×128. The generated datasets are grouped for training and testing with 80% and 20%, respectively. The training dataset contained 69,808 data samples, while the test dataset contained 17,452 data samples.

2.2 HOG

The word feature descriptor refers to an algorithm that takes an image as input and generates a feature vector or array as output **(Mohsin and Sadoon, 2019)**. The primary function of a feature descriptor is to summarize the image by turning it into a numerical representation. This is achieved by extracting useful information and ignoring irrelevant data. The feature descriptor classifies different elements according to their particular characteristics. Some of the feature descriptors include SIFT BRIEF, FAST, ORB, HOG, and many others **(Rahman et al., 2021)**.
In 2005, the HOG descriptor **(Dalal and Triggs, 2005)**, which is one of the highly reliable methods for pedestrian detection in both images and videos, was introduced by Dalal and Triggs. HOG has since become the most commonly used method in object detection systems. The descriptor functions analyse the distribution of local gradients to extract information from object appearance and shape, making it highly effective in capturing texture and shape for object detection tasks hence this technique remains a popular and effective choice, especially when used on powerful computers or machines **(Aslan et al., 2020; Diwakar and Raj, 2022)**.
The detailed steps to implement the algorithm are as follows:

i. Gamma/Color normalization: The image pre-processing step aims to standardize color and gamma values to ensure uniformity. This step is not always necessary as it has minimal impact on performance **(Sancho, 2014)**.
ii. Gradient computation: the first step is to filter an image using horizontal and vertical operators in both the x and y directions. This result is obtained for image gradients of both x and y directions **(Amraee et al., 2022)**.

$$G_X = D_X * I \tag{1}$$

$$G_y = D_y * I \tag{2}$$

$$D_x = [-1 \ \ 0 \ \ 1] \, , \, D_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \tag{3}$$

Here, I represents the original image, while D_X and D_y are the filtering masks used for the x and y directions, respectively. These masks are represented as vectors in Eq. (3).G_X and G_y represent the image's gradients in the x and y directions, respectively. The symbol * indicates the convolution operation, which involves summing the neighbouring pixels of a central pixel with specific weights determined by a weight matrix or convolution mask, and assigning the result as the current pixel value. Subsequently calculating G_X and G_y, the magnitude and orientation of the gradient are obtained using Eq. (4) and (5), respectively **(Amraee et al., 2022)**:

$$|G| = \sqrt{\mathrm{G_X}^2 + \mathrm{G_y}^2} \tag{4}$$

$$\theta = tan^{-1}\frac{G_y}{G_X} \tag{5}$$

Here, the term |G| refers to the gradient's magnitude, while θ indicates the direction of the gradient. In cases of edges and corners, the gradient magnitude tends to be high, while in smooth regions, it is nearly zero. Hence, a significant amount of unnecessary information in the background of images is eliminated **(Rahman et al., 2021)**.

iii. Spatial / Orientation binning: to calculate the histogram, the image is first divided into small cells, typically 8 × 8 pixels. The horizontal and vertical gradients are then calculated for each pixel in each cell. Then, each cell is split into many smaller blocks, usually 2 × 2 cells as illustrated in **Fig. 2**.

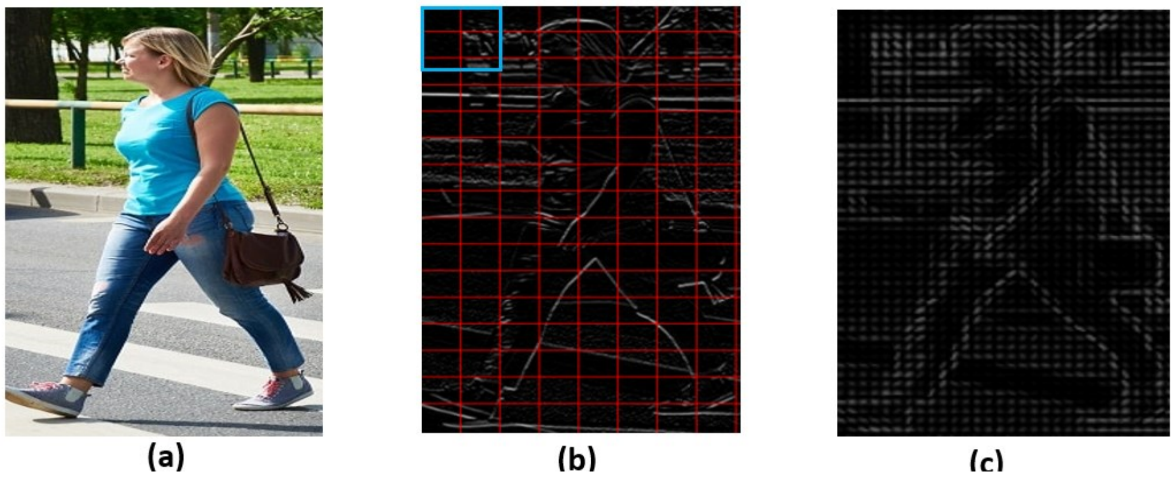


Figure 2. HOG Calculation, (a) Original image, (b) Divide the gradient image into cells (c) HOG image

iv. The gradient values are normalized using L2 normalization within each block, which increases the robustness of the feature descriptor. After that, histograms of gradient orientations are computed for each cell in the block. These histograms count the number of gradients that fall into specific orientation bins, such as (0-20) degrees, (20-40) degrees, etc. **(Sancho, 2014)**.

v. Concatenation: the normalized block histograms are concatenated to form the final HOG feature vector for the image **(NGO, 2022)**, as seen in **Fig. 3**.

The length of the feature vector depends on the number of cells and blocks. In this work, the image size was 64 × 128, and the images were divided into 8 × 16 cells. Each cell was 8x8 pixels with each block consisting of 2 × 2 cells. The histogram has 9 orientation bins, and each block of the histogram has a length of 9 × 4 = 36. Hence, the final HOG feature vector has a length of 7 × 15 × 36 = 3780, where 7 and 15 represent the number of cells in the height and width dimensions of the image, respectively. Afterwards, classifiers are trained using these characteristics in order to identify pedestrians **(Dalal and Triggs, 2005)**.

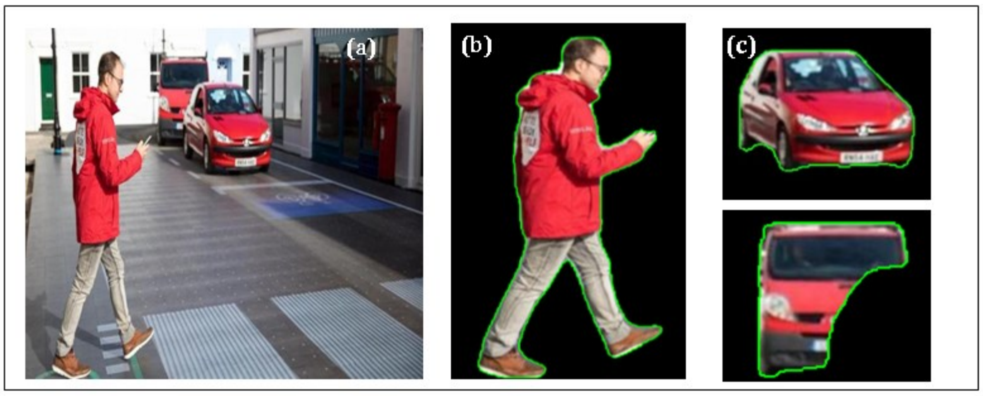


Figure 3. Computation of Histograms of Oriented Gradients.

2.3 YOLOv8

YOLO is object detection and image segmentation model that uses one neural network for detection and position estimation **(Alkentar et al., 2021)**. YOLO is considered to be a revolutionary model in computer vision because of its high speed and efficiency.
From its initial launch to today, the YOLO algorithm has been modified several times and every version further improved itself by implementing new features to enhance precision **(Diwakar and Raj, 2022)**.
YOLOv8 is a variant of YOLO with an integrated feature of object segmentation. The goal of object segmentation is to obtain the object's contour and generate more accurate information. To achieve this, YOLOv8 modifies the design by adding a segmentation branch **(Dumitriu et al., 2023)**. The segmentation branch has some extra auxiliary convolutional layers that interact with the detection branch. Applying this model it produces the masks of segmented objects. This approach describes the shape and gives accurate spatial data **(Patel et al., 2023)**. This framework has proved to perform efficiently in different datasets such as Cityscapes and Pascal VOC **(Kang et al., 2023). Fig. 4** presents the outputs after using YOLOv8 in the system.

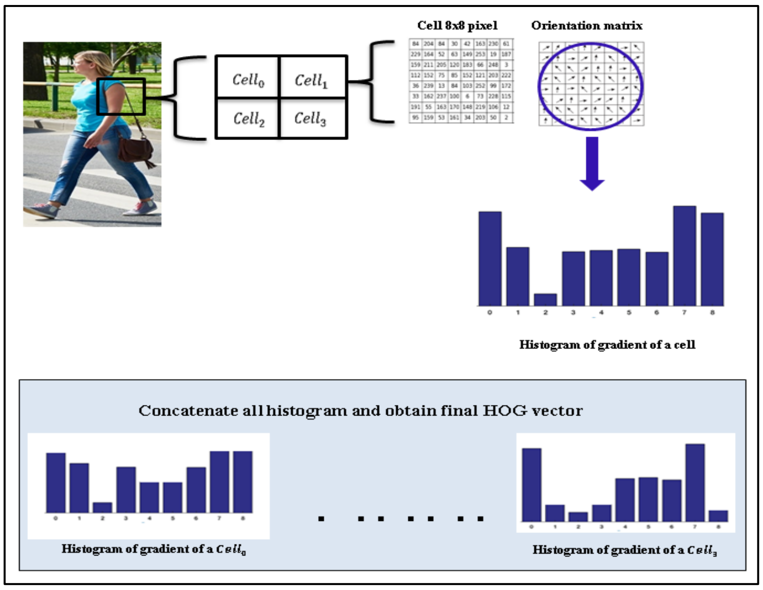

Cell 8x8 pixel
Orientation matrix
$Cell_0$
$Cell_1$
$Cell_2$
$Cell_3$
Histogram of gradient of a cell
Concatenate all histogram and obtain final HOG vector
Histogram of gradient of a $Cell_0$
Histogram of gradient of a $Cell_3$

Figure 4. Results of image segmentation (a) Input image (b) Positive object (Pedestrian) (c) Negative objects (Non-Pedestrian)

2.4 Classifications

Classification is a method to categorize objects into classes or categories according to similarity. It assigns specific labels to instances in a dataset. Therefore, data is categorized based on the attributes that they have in common **(Reddy and Babu, 2018)**. The classification process is typically separated into two stages **(Sancho, 2014)**:

- Training stage:

In this stage, the classification model learns from a training set consisting of samples from a number of categories. The model is developed to capture the concept or the patterns of the target category by using the features taken from the dataset. This process involves performing various mathematical operations and continues until the model reaches the desired level of accuracy or performance. The model can then classify specific data samples using the learned patterns.

- Testing stage:

The classification model obtained from the previous stage is utilized to classify new or unknown instances.
Image classification involves organizing all pixels or areas in a digital image into one of many potential classes and then determining whether or not the image contains a certain object. Pedestrian detection is a subset of binary image classification in which images represent samples, and class labels are either 1 (indicating the presence of a pedestrian, positives) or 0 or -1 (indicating the absence of a pedestrian, negatives) **(Amraee et al., 2022).** To the best of my knowledge, classification methods like SVM, RF, and CNN are frequently used in pedestrian identification. These classification techniques are briefly described in the following section.

2.4.1 Support Vector Machine

The SVM approach was first presented by Cortes and Vapnik **(Cortes and Vapnik, 1995)** as a supervised classifier. It is often used in pedestrian identification tasks due to its ability to successfully classify data by creating a hyperplane. The hyperplane is found by determining the line that is farthest from the nearest sample points, which are known as support vectors **(Aslan et al., 2020)**; **Fig. 5** displays an instance of SVM classifier. One of the main objectives of SVM algorithm is to improve the classification accuracy by maximizing the distance between support vectors and a hyperplane **(Wang, 2019).**
Linear and nonlinear SVM methods are two kinds of SVM. The first covers the instances where the data is linearly separable, whereas the second when the data cannot be linearly separated **(Mahdi, 2020)**.In the linear method, a straight line is employed for categorizing the features into two classes. To choose the best hyperplane, we will need first to identify support vector points. These support vectors play an important role in defining the position and orientation of that hyperplane **(Wang, 2019).** Eq. (6) can be used to find the best hyperplane for separable training data **(Abdulmunem and Hato, 2018)**.

$$Y_i(w.x_i + b) \geq 1,\ Y_i \in (1, -1) \tag{6}$$

Where x_i refers to the features vector, w is the weight vector, and b_i indicates the bias terms.

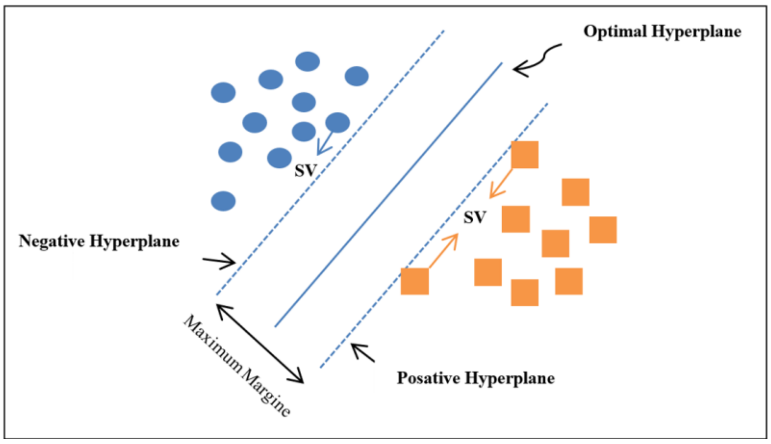


Figure 5. SVM classifier.

For nonlinear, separating the data using a straight line is difficult. Therefore, SVM utilized Kernel functions to address this problem. Kernel function enables separation in higher dimensions **(Mahdi, 2020).** The choice of kernel function greatly affects the performance of SVM; The most common kernel functions as **(Khanday et al., 2021):**

i. Linear: It is a kernel function used in ML algorithms to fit a linear or non-linear regression line or classifier. Suppose we have two vectors, p_i and p_j, the dot product between these two vectors produces the linear kernel **(Aiad et al., 2021)**.

$$K(p_i, p_j) = p_i . p_j \tag{7}$$

ii. Polynomial: It is a popular choice for kernel functionality in image processing and machine learning techniques; It is used to find a nonlinear polynomial classifier or regression line.

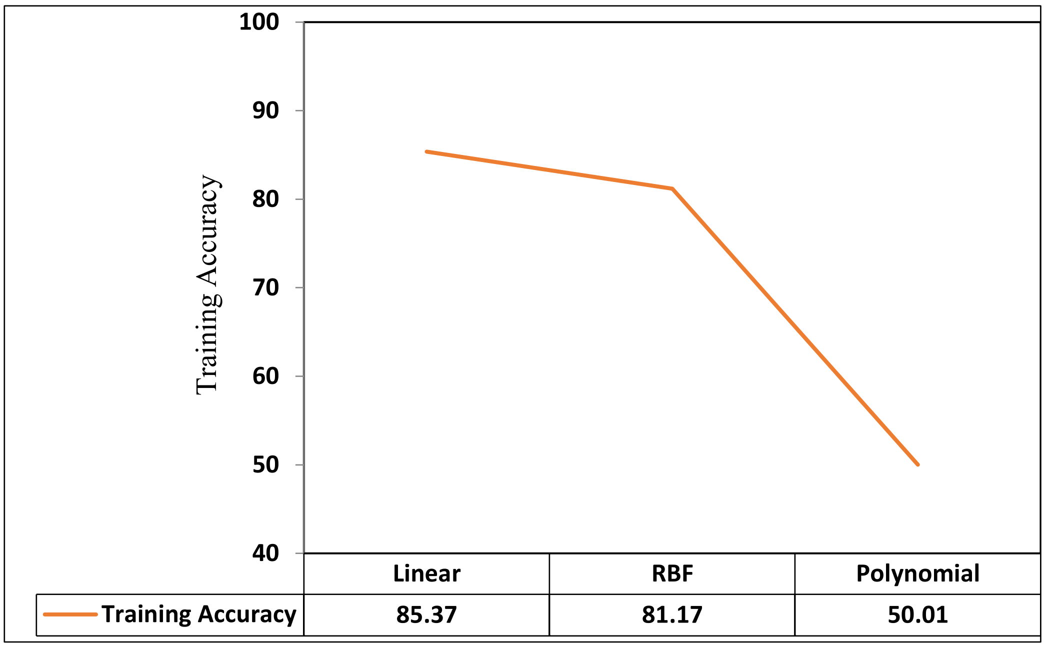
$$K(p_i, p_j) = (p_i^T p_j + c)^d;\ c \geq 0 \tag{8}$$

Here d is the degree of the polynomial, $c \geq 0$ is an independent variable that determines the balance of higher and lower-order polynomial terms, and T is a user-defined variable **(Aiad et al., 2021)**.

iii. Radial basis function (RBF): is an effective kernel for classification and regression tasks. Its non-parametric model performs effectively when dealing with high-dimensional, non-linear data.

$$K(p_i, p_j) = e^{\left(-\gamma \|p_i - p_j\|^2\right)} \tag{9}$$

Here, γ is a hyper-parameter that determines the kernel's width, and the Euclidean distance between the vectors (p_i, p_j) is represented by ||.|| **(Aiad *et al.*, 2021)**.
To determine the most suitable kernel function for this comparative study, the accuracy of the trained models is calculated using three different kernel functions: linear, RBF, and polynomial. The resulting accuracy values are presented in **Fig. 6.**



	Linear	RBF	Polynomial
Training Accuracy	85.37	81.17	50.01

Figure 6. Accuracy of training SVM using different kernels.

It is evident from the figure that Linear kernel achieved the highest accuracy compared to other functions for numerous reasons: When compared to RBF and Polynomial kernel functions, Linear kernel is the simplest type of SVM and is less influenced by overfitting, resulting in higher accuracy. Furthermore, Linear kernel is less impacted by noise and outliers in the data compared to RBF and Polynomial kernels. As a consequence, a Linear kernel was chosen for the comparison analysis.

The following is a description of utilizing SVM as a classifier in the suggested methodology:

i. Installing the libraries required to build and train the SVM classifier.
ii. The positive and negative features and the positive and negative labels (collected from the feature extraction stage) are concatenated to form database features ($DB_{Features}$) of size (87260,3780) and labels of size (87260), where 87260 refers to the total number of feature samples and 3780 refers to the size of each feature vector.
iii. Splitting $DB_{Features}$ with corresponding labels into training and testing sets with 80% and 20%, respectively. Produce training data of size (69808,3780) and testing data of size (17452,3780).
iv. Encoding the labels, meaning they are converted to numeric values, where 0 represents pedestrian labels and 1 represents non-pedestrian labels. This convention is used to make the model easier to understand.
v. Using the training data, train the SVM model with three types of kernels: Linear, RBF, and Polynomial.
vi. Once the training process is completed, the model is saved to be used later in the prediction process on testing or new data.

2.4.2 Random Forest

Random Forest is a classification approach that relies on several decision trees. It builds a huge number of decision trees, each one trained on a distinct portion of the training data using a randomly chosen set of characteristics **(Braik et al., 2020).** At the prediction stage, each tree in the forest independently predicts the class for the input, with the final prediction being the majority votes of the trees. RF is known to be capable of handling multidimensional data and missing values. This is one of the most popular methods applied in the classification domain because it deals with the problem of handling complicated datasets accurately and efficiently **(Hiranmai et al., 2018).** The algorithm can be summarized by Eq. (10) **(Hiranmai et al., 2018)**:

$$\text{Prediction} = \text{Majority Vote}(\text{Pred}_{\text{tree1}}, \text{Pred}_{\text{tree2}} \ldots, \text{Pred}_{\text{treen}}) \tag{10}$$

Here, $Pred_{tree_i}$ indicates the prediction of the i-th decision tree in the RF. RF can develop the precision and robustness of classification through the combination of predictions suggested by numerous trees **(Hiranmai et al., 2018)**.

The following steps describe the procedure for using RF as a classifier in the suggested model:

i. Installing the necessary libraries to build and train the RF classifier.
ii. Loading $DB_{Features}$ with their corresponding labels.
iii. Splitting $DB_{Features}$ with corresponding labels into training and testing sets with 80% and 20%, respectively. Produce training data of size (69808,3780) and testing data of size (17452, 3780).
iv. Encoding the labels.
v. Defining the number of decision trees in the ensemble (n_estimators=100), which is determined through experiments.
vi. Train the RF system using the training data.
vii. After the training process is completed, the model is stored to be utilized later in the prediction process on testing or new data.

2.4.3 Convolution Neural Network

In modern approaches to human perception, deep learning is a beneficial tool for object recognition, classification, segmentation, and natural language processing **(Aslan et al., 2020)**. A CNN, often known as a ConvNet, is a form of DL architecture utilized for visual data analysis. CNNs function the same even if they are 1D,2D or 3D, with the difference in the input data format and the movement of the feature detector or convolutional kernel over the data **(Haamied et al., 2021).** Deep CNN designs often comprise several convolutional blocks and a connected layer for end-to-end operation, where each convolutional block consists of a convolutional layer, filters, an activation unit, and a pooling layer. Filters play an important function in the convolution layer since they convolve the outputs of previous layers **(Kim et al., 2020).** In this context, convolution denotes a mathematical process that represents how the shape of one function is influenced by another **(Diwakar and Raj, 2022)**.

Convolution in a two-dimensional space is represented by the following mathematical equation **(Rahman et al., 2021)** :

$$Y[i,j] = \sum_m \sum_n k[m,n].x[i-m, j-n] \quad (11)$$
where x and k represent the input image and convolution kernel, respectively, and the resulting output image is denoted by y. The parameters (m, n) and (i, j) are associated with the height and width of the kernel and input image, respectively. To facilitate convolutions on the entire image, padding is often employed, and stride refers to the number of pixels which the kernel moves after each convolution. In DL, images are typically resized to have equal width and height; kernels are often designed with similar heights and widths, such as 3x3, 5x5, and so on **(Rahman et al., 2021)**.
Fig. 7 presents a proposed architecture of CNN. The input undergoes three pairs of convolutional operations with Rectified Linear Unit (ReLU) activation functions and max-pooling layers for accurate classification. A flattening operation is then applied to convert the 3D layer into a 1D vector.

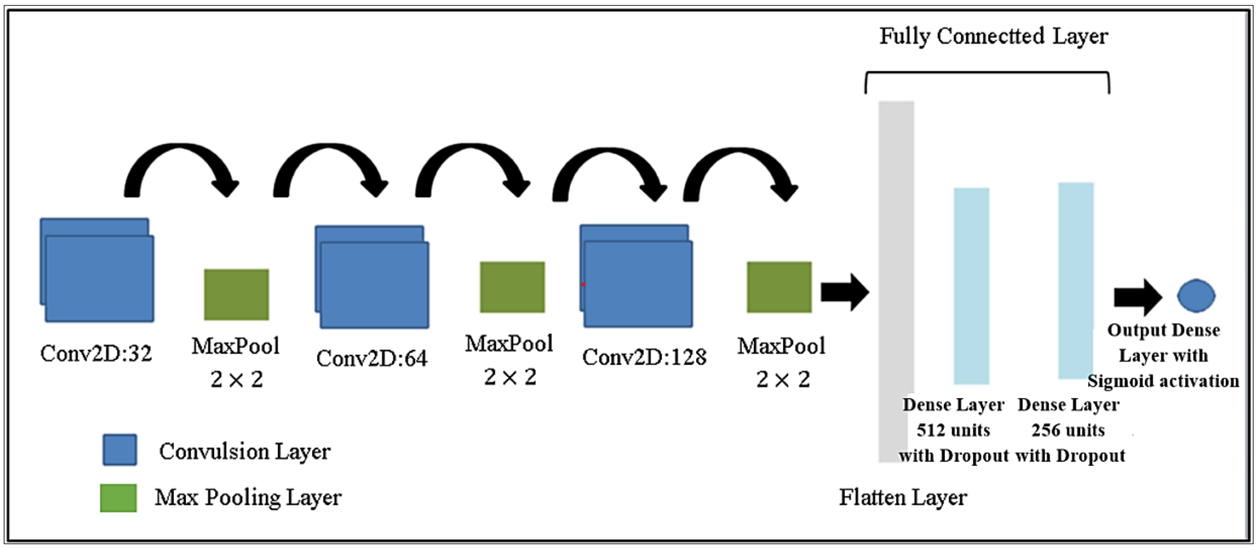


Figure 7. The design of proposed CNN architecture.

This vector is then passed to the dropout layers, where some neurons are dropped from the network to reduce overfitting and improve the network's generalization performance. Subsequently, the output of the CNN is passed through the Non-Maximum Suppression (NMS) algorithms, which select the most accurate boundary boxes. During CNN training, the error is computed from the output used to fine-tune the trainable parameters, including filter values and neuron weights.
The procedure for using CNN as a classifier in the proposed model is explained as follows:

i. Installing the libraries required to build and train the CNN model.
ii. Load $DB_{Features}$ and their corresponding labels.
iii. Dividing $DB_{Features}$ into subsets for training and testing using a ratio of 80% and 20%, respectively. Produce training data of size (69808, 3780) and testing data of size (17452, 3780).
iv. To ensure compatibility with the accepted input format for the CNN, the training and testing features were transformed into a 2D array, which produced training dataset dimensions of (69808, 60, 63) and testing dataset dimensions of (17452, 60, 63).

v. Encoding the labels.

vi. Before model training, key parameters such as the optimizer (Adam), learning rate (0.0001), loss function (binary cross-entropy), number of epochs (100), and batch size (32) were determined after performing several experiments. Next, the training process started, with a total of 2182 iterations.

vii. After completing the training process, the model was saved for subsequent use during the testing stage.

3. RESULTS AND DISCUSSIONS

The execution of the entire script of this work was performed in the environment of Google Colab. Google Colab is an online cloud platform designed for high-performance computations and training DL models **(Byju et al., 2021)**. This platform was chosen to overcome any constraints in computing resources and leverage its capabilities to satisfy the experimental requirements of this work.
Different performance measures are used to evaluate the detection model. The performance of the proposed model is reported using these metrics; namely, accuracy, precision recall, F1-score and detection speed. Accuracy can be stated as the percentage of data points correctly predicted over the total number of data points. It shows the precision with which the system correctly classifies input data. For pedestrian detection, the True Positives (TP) denotes the number of samples that were correctly identified as pedestrians. The number of samples that are classified as non-pedestrians but are not is False Positives (FP). True Negatives (TN) denote the number of samples that accurately predict non-pedestrians. Finally, False Negatives (FN), characterize the number of samples which misclassified pedestrians as non-pedestrians **(Mohanad et al., 2023)**. Accuracy can be computed utilizing the Eq. (12) **(Rahman et al., 2021)**:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{12}$$

Precision denotes the ratio of accurately predicted positive observations to all predicted positive observations. The precision of a system is determined by using the Eq. (13) **(Ali et al., 2022)**:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{13}$$

Recall is the ratio of correctly predicting positive observations to all observations in the actual class. The recall is computed by applying the Eq. (14) **(Lan et al., 2018)**:

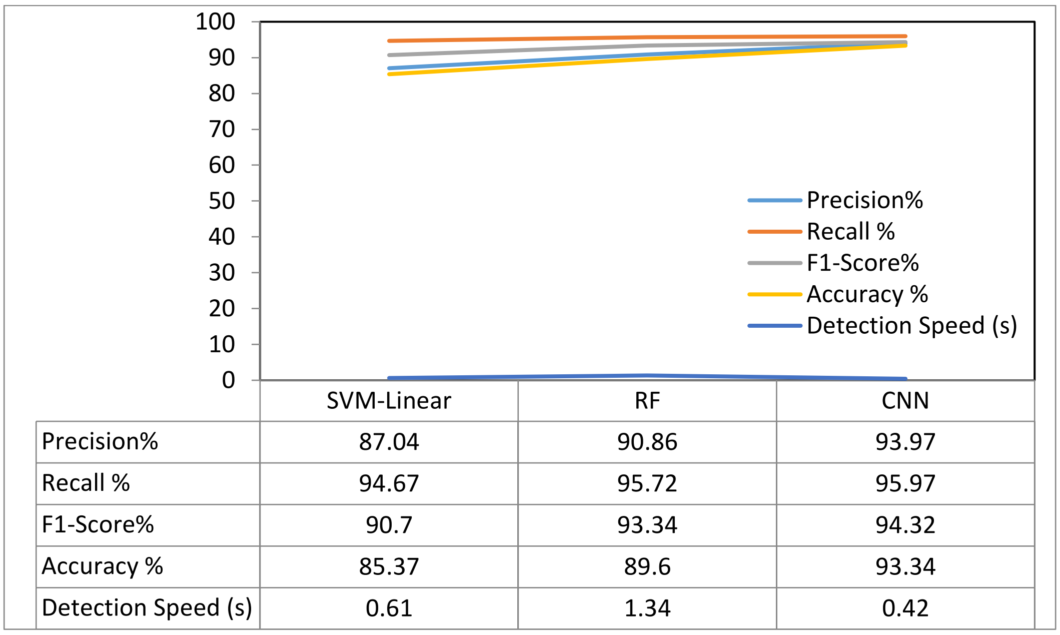
$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{14}$$

F1-score is defined as the harmonic mean of precision and recall, always within the range of 0 to 1. The calculation of the F1-score is done using the Eq. (15) **(Kim et al., 2020)**:

$$\text{F1} = 2 \text{ x } \frac{\text{Precision x Recall}}{\text{Precision+ Recall}} \tag{15}$$

As precision and recall have an inverse relationship, increasing precision often leads to a decrease in recall, and vice versa; the F1-score provides a more comprehensive measure of model performance **(Rahman et al., 2021).**
Detection speed refers to the time required for the model to detect pedestrians **(Raghavachari et al., 2015)**. **Fig. 8** summarizes the findings of the comparative analysis accomplished by this study.

These findings illustrate the superiority of the CNN classifier in terms of accuracy and detection time. This can be justified by the fact that CNN can effectively learn and extract complicated features via its several layers. On the contrary, the performance of the SVM linear classifier appeared less accurate compared to RF and CNN. The linearity of the SVM classifier is one of the causes of this worse performance. When dealing with complicated and nonlinear patterns in the data, linear SVM might perform poorly, producing lower accuracy than RF and CNN classifiers. In general, SVM has good robustness in particular scenarios, and the comparison demonstrates that RF and CNN surpass SVM in pedestrian detection tasks.
The RF classifier detects at a slower rate than SVM and CNN because it is made up of an ensemble of decision trees. The implementation of each tree in the ensemble is isolated and every prediction from all trees is used together to get a final class label. This ensemble technique increases the computational complexity of the algorithm since it visits several trees and pools their predictions. As a consequence, RF demands more computational resources, particularly when dealing with huge data sets, which might slow down detection speeds.
SVM and CNN classifiers, on the other hand, have distinct properties that increase detection speed. SVM, especially the linear type, has a simpler structure and depends on locating the best hyperplane to divide the classes. This method is computationally efficient and avoids traversing numerous trees. Similarly, CNNs extract features and produce predictions using the efficient operations provided by convolutional layers. The convolutional layers share weights over different input parts, lowering the computing load.



	SVM-Linear	RF	CNN
Precision%	87.04	90.86	93.97
Recall %	94.67	95.72	95.97
F1-Score%	90.7	93.34	94.32
Accuracy %	85.37	89.6	93.34
Detection Speed (s)	0.61	1.34	0.42

Figure 8. Results of SVM, RF, CNN for pedestrian detection.

When choosing between these classifiers, it is necessary to consider the specific requirements of the application to obtain a trade-off between accuracy and speed. While RF may have a slower detection speed, it offers the advantage of group learning and can achieve higher accuracy in certain scenarios. Instead, SVM outperforms in terms of computational efficiency and can produce the lowest time.

Fig.9 shows different cases of experimental results for the proposed model when evaluated using different classifiers across various weather conditions. As illustrated in **Fig.9,** the results can be summarized using the confusion matrix as explained in **Figs. 10 to 14**.

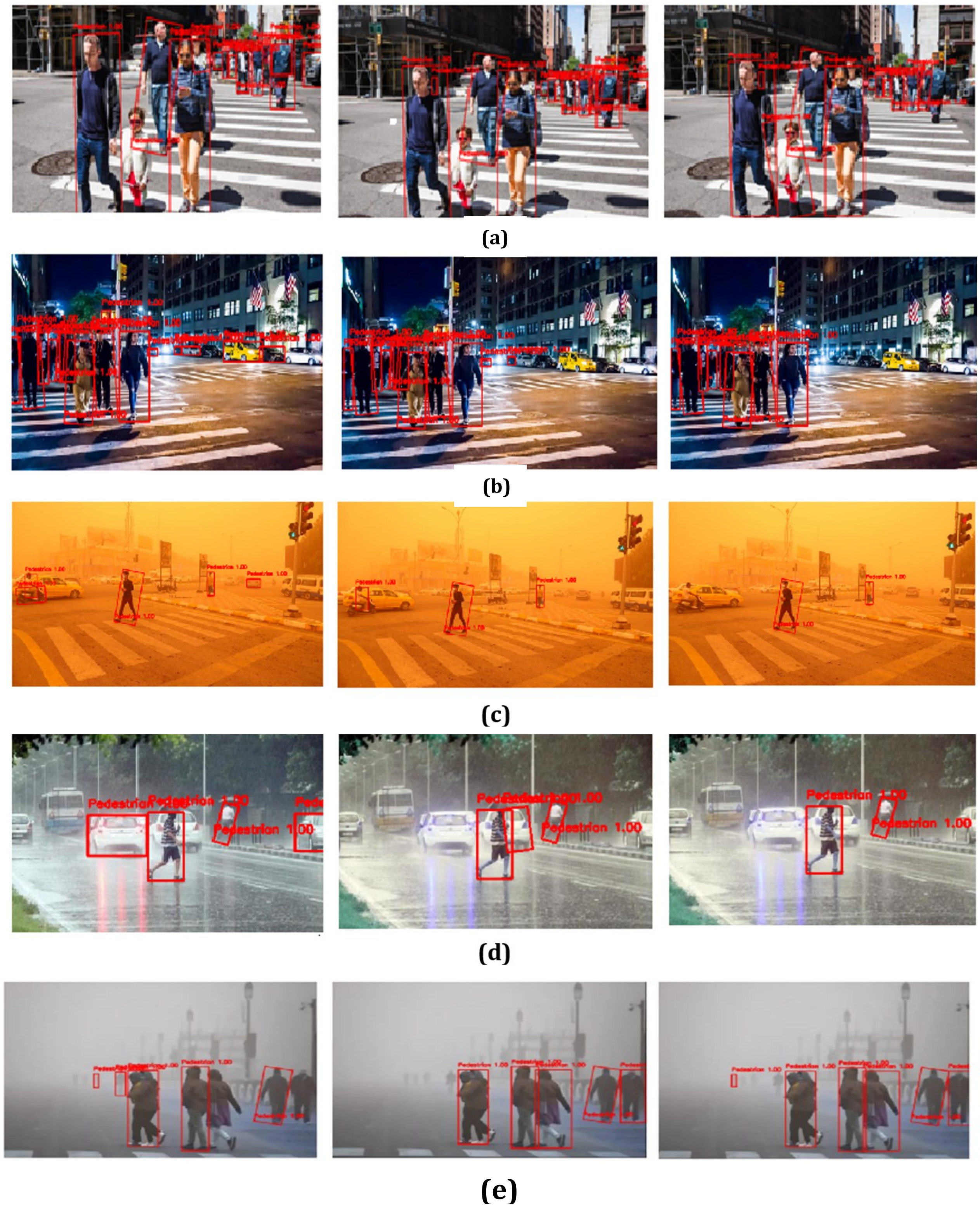


Figure 9. Comparison of detection results in weather conditions: SVM (left), RF (middle), and CNN (right) for (a) During Day, (b) Night, (c) Dusty, (d) Rainy Day, and (e) Foggy

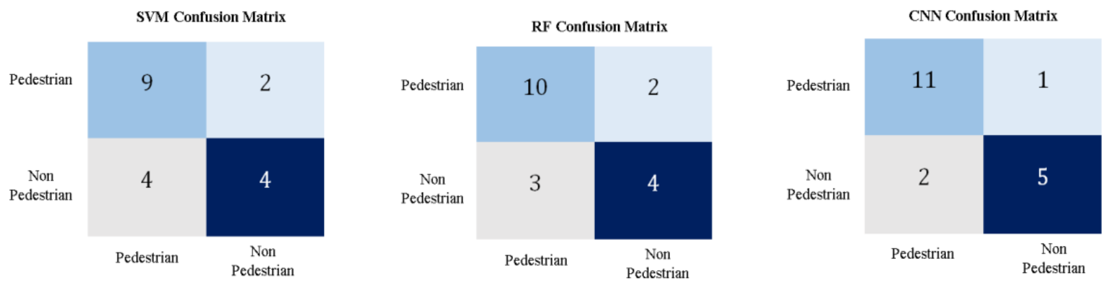


Figure 10. Confusion matrix of detection results during day scene

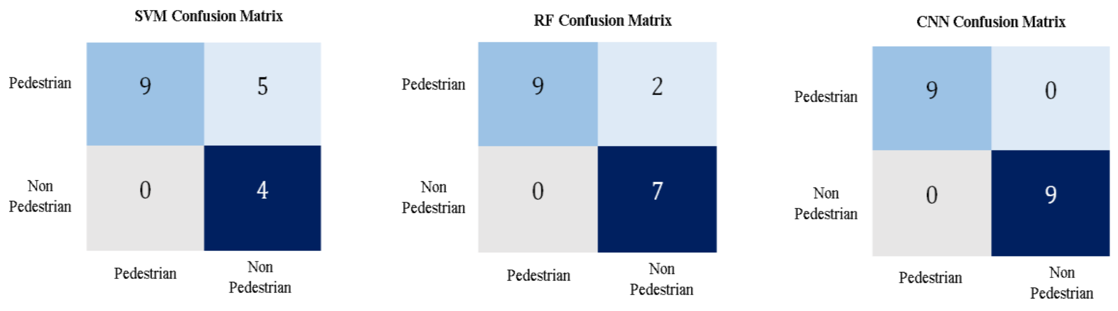


Figure 11. Confusion matrix of detection results in night scene

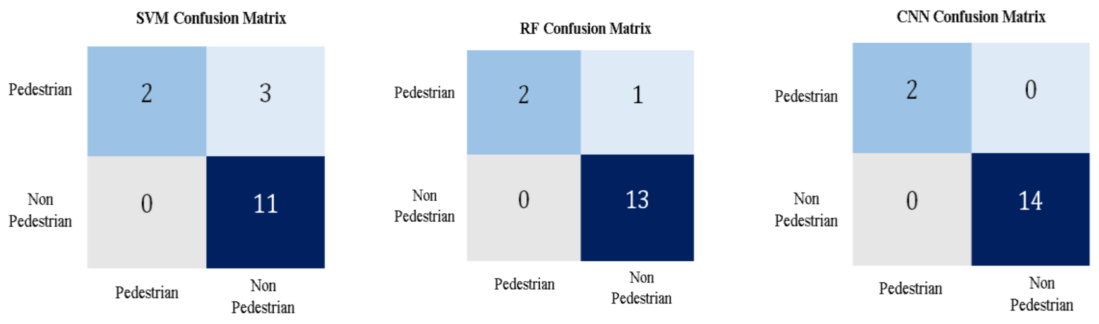


Figure 12. Confusion matrix of detection results in dusty scene

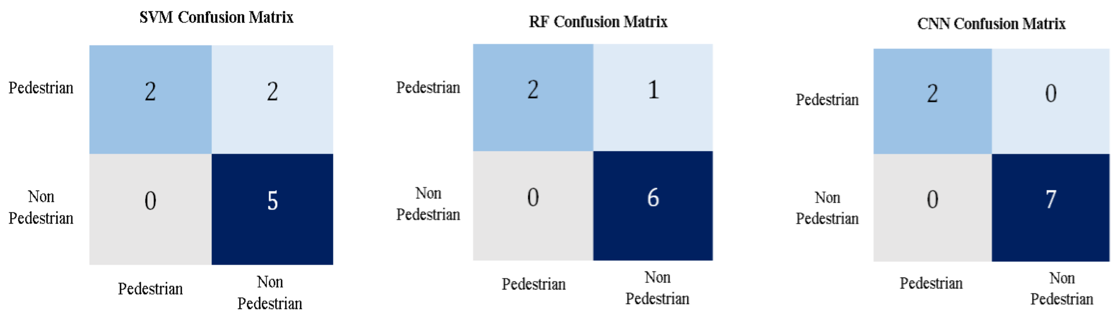


Figure 13. Confusion matrix of detection results in rainy scene

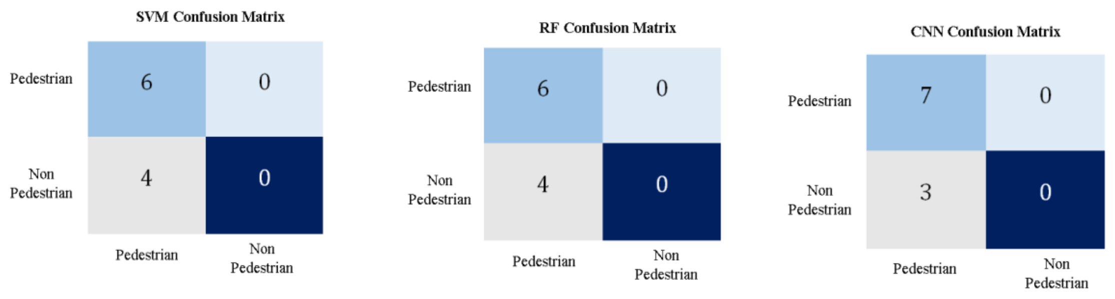


Figure 14. The confusion matrix of detection results in a foggy scene

As observed in the confusion matrixes above, CNN performed well in all-weather situations, recording the largest number of TP, lowest number of FP, and lowest missed detection. This superiority comes from the fact that CNN relies on deep and complex neural networks, which allows it to extract visual information from images and represent it efficiently. This means that CNN can successfully detect patterns and fine-grained features in pedestrian images containing colour contrasts, shapes, structures, and different weather conditions, allowing for accurate pedestrian identification.

4. CONCLUSIONS

Pedestrian detection plays a vital role in many computer vision applications. Therefore, the need to develop an accurate pedestrian detection system has become essential. In this work, a hybrid model for pedestrian detection is proposed by integrating YOLOv8 for object segmentation and HOG for feature extraction. For classification, three types of classifiers are compared, namely RF, CNN, and SVM. For the SVM classifier, three types of kernels are compared, namely Linear, RBF, and polynomial. Experiments were conducted using the EPFL dataset and all simulation parts of the building model were done using Python language in the Google Colab environment. The results indicate that SVM with linear kernels achieved better performance compared to other kernels. Moreover, CNN showed superior results with an accuracy rate of 93.34% and detection speeds of 0.42s compared to 85.37%, 0.61s, and 89.60%, 1.34s of SVM and RF, respectively.

NOMENCLATURE

Symbol	Description	Symbol	Description
b	Bias.	$Pred_{tree_i}$	Prediction of the i-th decision tree.
C	Independent variable that determines the balance of higher and lower-order polynomial	p	Vector represents the data point.
D_X	Filtering mask in X direction.	T	User-defined variable.
D_y	Filtering mask in Y direction.	TP	True positive.
d	Degree of the polynomial.	TN	True negative.
FP	False positive.	w	Weight vector.
FN	False negative.	x_i	Features vector.
G_X	Gradient in X direction.	Y[i, j]	Output image.
G_y	Gradient in Y direction.	(i, j)	Height and width of the input image.
|G|	Gradient's magnitude.	(m, n)	Height and width of the kernel.
I	The original image	θ	The direction of the gradient.
K	Kernel function.	γ	Hyper-parameter that determines the kernel's width.

Acknowledgment

The authors are grateful for the academic support and resources provided by the Department of Computer Engineering, College of Engineering, University of Baghdad.

Credit Authorship Contribution Statement

Tuqa Hani Abd-Alamir: Writing –original draft. Mohammed Sadoon Hathel: Writing –review & editing, Validation, Conceptualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

Aiad, B. A. E., Zarif, K. B., Gadallah, Z. M., and Abd EL-kareem, H., 2021. Support Vector Machine Kernel Functions Comparison. *International Undergraduate Research Conference,*5(5), pp. 84-88.

Aslan, M. F., Durdu, A., Sabanci, K., and Mutluer, M. A., 2020. CNN and HOG based comparison study for complete occlusion handling in human tracking. *Measurement: Journal of the International Measurement Confederation*, 158, P. 107704. https://doi.org/10.1016/j.measurement.2020.107704

Amraee, S., Chinipardaz, M. and Charoosaei, M., 2022. Analytical study of two feature extraction methods in comparison with deep learning methods for classification of small metal objects. *Visual Computing for Industry, Biomedicine, and Art*, 5(1), P.13. https://doi.org/10.1186/s42492-022-00111-6.

Alkentar, S. M., Alsahwa, B., Assalem, A., and Karakolla, D., 2021. Practical comparation of the accuracy and speed of YOLO, SSD and Faster RCNN for drone detection. *Journal of Engineering,* 27(8), pp. 19–31. https://doi.org/10.31026/j.eng.2021.08.02

Abdulmunem, M. E., and Hato, E., 2018. Outdoor Scene Classification Using Multiple SVM. *Iraqi Journal of Science,* pp. 2323-2335. https://doi.org/10.24996/ijs.2018.59.4C.20

Ali, M. A., Hussain, A. J., and Sadiq, A. T., 2022. Detection And Count of Human Bodies In a Crowd Scene Based on Enhancement Features By Using The YOLO v5 Algorithm. *Iraqi Journal of Computers, Communications, Control and Systems Engineering*, 22(2), pp.125-134. https://doi.org/10.33103/uot.ijccce.22.2.11

Braik, M., Al-Zoubi, H. and Al-Hiary, H., 2020. Pedestrian detection using multiple feature channels and contour cues with census transform histogram and random forest classifier. *Pattern Analysis and Applications*, 23(2), pp. 751–769. https://doi.org/10.1007/s10044-019-00835-x.

Byju, J., Chitra, R., Pranesh, P. E., Pavan, R. S., and Aravinth, J., 2021. Pedestrian Detection and Tracking in Challenging Conditions. *In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS),* Vol. 1, pp. 399-403. https://ieeexplore.ieee.org/document/9441934

Chong, P., and Tay, Y. H., 2017. A novel pedestrian detection and tracking with boosted HOG classifiers and Kalman filter. *Proceedings - 14th IEEE Student Conference on Research and Development: Advancing Technology for Humanity, SCOReD 2016, pp.1-5.* https://ieeexplore.ieee.org/document/7810052

Cortes, C., and Vapnik, V., 1995. Support-vector networks. Machine learning, 20, pp.273-297. https://doi.org/10.1007/BF00994018

Dalal, N., and Triggs, B., 2005. Histograms of oriented gradients for human detection. *Proc 2005 IEEE Comput Soc Conf Comput Vis Pattern Recognit 1063-6919/05 $2000 © 2005 IEEE*, 1, pp. 886–893. https://doi.org/10.1109/CVPR.2005.177

Diwakar and Raj, D., 2022. Recent Object Detection Techniques: A Survey. *International Journal of Image, Graphics and Signal Processing*, 14(2), pp. 47–60. https://doi.org/10.5815/ijigsp.2022.02.05

Dumitriu,A., Tatui,F., Miron,F., Ionescu,R.,Timofte,R., 2023. Rip Current Segmentation: A Novel Benchmark and YOLOv8 Baseline Results. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops,* 2023-June, pp. 1261–1271. https://doi.org/10.1109/CVPRW59228.2023.00133

Hiranmai, M., Krupa, N. B. and Nagaraj, H. K., 2018. Comparative Study of Various Feature Extraction Techniques for Pedestrian Detection. *Procedia Computer Science*, 154, pp. 622–628. https://doi.org/10.1016/j.procs.2019.06.098

Haamied, R. D., Al-Abudi, B. Q. and Hassan, R. N., 2021. Automatic Object Detection, Labelling, and Localization by Cameras Drone System. *Iraqi Journal of Science*, 62(12), pp. 5008–5023. https://doi.org/10.24996/ijs.2021.62.12.37

Iftikhar, S., Zhang, Z., Asim, M., Muthanna, A., Koucheryavy, A., and Abd El-Latif, A. A., 2022. Deep Learning-Based Pedestrian Detection in Autonomous Vehicles: Substantial Issues and Challenges. *Electronics*, 11(21), P. 3551. https://doi.org/10.3390/electronics11213551

Joodi, M. A., Saleh, M. H., and Khadhim, D. J., 2023. Proposed Face Detection Classification Model Based on Amazon Web Services Cloud (AWS). *Journal of Engineering,* 29(4), pp. 176-206. https://doi.org/10.31026/j.eng.2023.04.12

Khalifa, A. B., Alouani, I., Mahjoub, M. A., and Amara, N. E. B., 2020. Pedestrian detection using a moving camera: A novel framework for foreground detection. *Cognitive Systems Research*, 60, pp. 77–96. https://doi.org/10.1016/j.cogsys.2019.12.003

Kang,J., Zhao,L.,Wang,K.,Shandong,K., 2023. Research on an Improved YOLOV8 Image Segmentation Model for Crop Pests. *Advances in Computer, Signals and Systems*,7(3), pp. 1–8. https://dx.doi.org/10.23977/acss.2023.070301.

Khan, Akib Mohi Ud Din Khanday, Q. R. and Rabani, S. T., 2021. Detecting Textual Propaganda Using Machine Learning Techniques. *Baghdad Science Journal*, 18, pp. 0199--0199. https://doi.org/10.21123/bsj.2020.17.1(Suppl.).0385.

Kim, B., Yuvaraj, N., Sri Preethaa, K. R., Santhosh, R., and Sabari, A., 2020. Enhanced pedestrian detection using optimized deep convolution neural network for smart building surveillance. *Soft Computing*, 24(22), pp. 17081-17092. https://doi.org/10.1007/s00500-020-04999-1

Lan, W., Dang, J., Wang, Y., and Wang, S., 2018, August. Pedestrian detection based on YOLO network model. *In 2018 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1547-1551. https://doi.org/10.1109/ICMA.2018.8484698.

Mateus, A., Ribeiro, D., Miraldo, P., and Nascimento, J. C., 2019. Efficient and robust Pedestrian Detection using Deep Learning for Human-Aware Navigation. *Robotics and Autonomous Systems*, 113, pp. 23–37. https://doi.org/10.1016/j.robot.2018.12.007.

Mohsin, A., and Sadoon, M., 2019. Developing an Arabic Handwritten Recognition System by Means of Artificial Neural Network. *Journal of Engineering and Applied Sciences*, 15(1), pp. 1–3.

Mahdi, G. J. M., 2020. A modified support vector machine classifiers using stochastic gradient descent with application to leukemia cancer type dataset. *Baghdad Science Journal*, 17(4), pp. 1255–1266. https://doi.org/10.21123/bsj.2020.17.4.1255

Patel, N., Dabhi, V. and Adhvaryu, R., 2023. Identify Road Potholes Using Image Semanticsegmentation. *Journal of Data Acquisition and Processing*, 38(2), pp. 2307–2315. https://doi.org/10.5281/zenodo.776921.

Raghavachari, C., Aparna, V., Chithira, S., and Balasubramanian, V., 2015. A comparative study of vision based human detection techniques in people counting applications. *Procedia Computer Science*, 58, pp. 461-469. https://doi.org/10.1016/j.procs.2015.08.064

Reddy, R. V. K. and Babu, U. R., 2018. A Review on Classification Techniques in Machine Learning. *International Journal of Advance Research in Science and Engineering*, 7(30), pp. 40–47.

Rahman, M. M., Nooruddin, S., Hasan, K. A., and Dey, N. K., 2021. HOG + CNN Net: Diagnosing COVID-19 and Pneumonia by Deep Neural Network from Chest X-Ray Images. *SN Computer Science*, 2(5), pp. 1–15. https://doi.org/10.1007/s42979-021-00762-x

Seemanthini, K. and Manjunath, S. S., 2018. Human Detection and Tracking using HOG for Action Recognition. *Procedia Computer Science*, 132(Iccids), pp. 1317–1326. https://doi.org/10.1016/j.procs.2018.05.048

Sancho, C., 2014. Pedestrian Detection using a boosted cascade of Histogram of Oriented Gradients. MSc. thesis, Department of Electrical Engineering, Linköping University.

Wang, B., 2019. Research on Pedestrian Detection Algorithm Based on Image. *Journal of Physics: Conference Series*, 1345(6), p. 062023. https://doi.org/10.1088/1742-6596/1345/6/062023.

Zuo, X., Li, J., Huang, J., Yang, F., Qiu, T., and Jiang, Y., 2021. Pedestrian detection based on one-stage YOLO algorithm. *Journal of Physics: Conference Series*, 1871(1), p. 012131. https://doi.org/10.1088/1742-6596/1871/1/012131.

Zhang, Y., Xu, L., and Zhang, Y., 2022. Research on hierarchical pedestrian detection based on SVM classifier with improved kernel function. *Measurement and Control (United Kingdom)*, 55(9–10), pp. 1088–1096. https://doi.org/10.1177/00202940221110164

دراسة مقارنة تقنيات التصنيف المختلفة في تطبيقات كشف المشاة

تقى هاني الأمير*، محمد سعدون حثيل

قسم هندسة الحاسبات، كلية الهندسة، جامعة بغداد، بغداد، العراق

الخلاصة

يعد اكتشاف المشاة أحد أهم تطبيقات الرؤية الحاسوبية. ومع ذلك، فإن الاكتشاف الدقيق للمشاة يعد مهمة صعبة بسبب مجموعة متنوعة من المشكلات مثل الأحجام المختلفة لميزات المشاة والخلفيات المزدحمة. تهدف هذه الدراسة إلى تقييم ومقارنة أداء الكشف عن المشاة لثلاثة أنواع مختلفة من المصنفات: الغابة العشوائية (RF)، والشبكة العصبية التلافيفية (CNN)، والمتجه الداعم (SVM) في ظل ظروف الطقس المختلفة. تجمع المنهجية المقترحة بين الرسم البياني للتدرجات الموجهة (HOG) لاستخراج الميزات وخوارزمية (YOLOv8) للتجزئة. يتم بعد ذلك تدريب واختبار مصنفات RF وCNN وSVM باستخدام ميزات HOG المستخرجة. تم إجراء التجربة باستخدام مجموعة بيانات EPFL. واظهرت النتائج ان مصنف CNN حقق أفضل النتائج بسرعة 0.42 ثانية ودقة 93.34%.وعند المقارنة ، تفوقت (CNN) على مصنفات (SVM) و (RF) ،حيث اظهر مصنف (RF) ابطأ سرعة كشف بينما كان لمصنف (SVM) اقل دقة. يوفر هذا البحث رؤيا قيمة حول فعالية هذه المصنفات في مهام الكشف عن المشاة في ظل ظروف جوية مختلفة، ويسلط الضوء على امكانية مصنف (CNN) لتحقيق الدقة العالية والكفاءة في الكشف.

الكلمات المفتاحية: خورازمية الرسم البياني للتدرجات الموجهة, خوارزمية (YOLOv8) , مصنف الشبكة العصبية الالتفافية, مصنف الغابة العشوائية , مصنف المتجه الداعم.