International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Aug-2022)

https://dx.doi.org/10.12785/ijcds/120146

A Review of Machine Learning Approaches for Human Detection through Feature Based Classification

Mohd. Aquib Ansari[1] **and Dushyant Kumar Singh**[2]

[1, 2]*Department of Computer Science & Engineering, MNNIT Allahabad, Prayagraj, Uttar Pradesh, INDIA*

Received 2 Dec.2021, Revised 17 Jul. 2022, Accepted 31 Jul. 2022, Published 6 Aug. 2022

Abstract: Human detection has always been a task of sincere importance for many automation activities under computer vision. The problem concentrates on identifying regions of human presence in an image or frames of running video. The application areas may have the varied role of human detection objectives ranging from normal to serious and very critical/sensitive. Vision based attendance, traffic flow analysis, driver assistance, etc., are examples of applications with a normal role. Simultaneously, it plays a serious role in vision based theft identification system and a critical role in intruder detection in the border or sensitive places. This paper presents and explores various machine learning based human detection techniques. These techniques include feature learning based and deep learning based human detection paradigms. Through experiments, it has been found that Deep Neural Network based human detection techniques provide more efficient results than feature learning based techniques in terms of detection accuracy.

Keywords: Surveillance, Human Detection, Low-Level features, Classifiers, Deep Neural Network

1. Introduction

Human detection in automated video surveillance [1], [2], [3], [4], [5], [6], [7], [8] has been one of the major and fascinating research topics in image processing and computer vision. It is a process of automatically localizing all instances of the person in the image or video sequences. For an expert video surveillance system, detecting a human being is vital for person identification, person counting, fall detection for older adults, activity detection, gait recognition, gender classification etc.

Human detection involves several techniques to identify a human body's appearance in the region, ranging from modest image processing to learning paradigm based techniques. The image processing based human detection techniques involve simple pixel-wise operations to spot the human in the region space. In contrast, learning paradigm based techniques involve two essential tasks: object detection and classification. The general human detection framework based on the learning paradigm is illustrated in Figure 1. Here, CCTV cameras are used to capture video sequences or frames. Next, the sequences are passed to the object detection module, where vital information or features are extracted from each frame. These features are then passed to an object classification module to build a classifier that decides the probability of the object being a person.

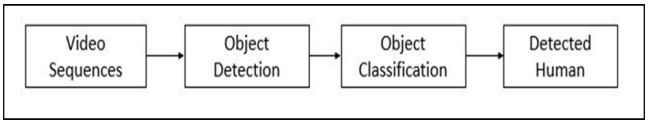


Figure 1. Basic human detection pipelining

This paper presents and experimentally explores different learning based human detection techniques. These techniques require features to characterize the image uniquely. The features can be visual, textural, and shape features. The visual features consist of color information of an image, whereas textural and shape features consist of texture patterns and shape information present in an image. These features put great importance on object detection. Features contain a piece of information about the content of an image that is suitable for solving computational work related to a certain application. Determining independent, discriminating, and informative features is a critical measure for building effective algorithms in the field of regression, classification, and pattern recognition. Different learning based techniques use these features to create decision-making modalities for identifying different objects in an image. In machine learning, techniques involve two major functions: feature extraction and classification. Feature extraction is a task to mine the features or essential information from the object. Further, the mined features are passed to the classification module to assign the label to the object. This paper covers the Color Histogram Descriptor [9], Histogram of

E-mail address: mansari@mnnit.ac.in, dushyant@mnnit.ac.in

Gradient [10], Local Binary Pattern [11], Discrete Wavelet Transform [12], and Gray Level Co-occurrence Matrix [13] descriptors in detail. The behavior of classifiers, along with different feature extraction techniques, are also explored in the experimental section.

On the other hand, deep learning [14], [15], [16], [17] works on the principle of biological neurons and extracts the features from an image implicitly. A deep neural network consists of the convolutional layer to extract the features, pooling layer to reduce the spatial representation, fully connected layer to assign the weight and bias to the network automatically, and an output layer. Deep learning has the advantage of faster computation and higher accuracy than feature learning based techniques. This paper also covers deep learning based techniques to identify the individual's presence in an image in brief.

The rest of the paper is arranged as follows. Section 2 presents the related works done on human detection. Section 3 includes a brief discussion on various learning-based human detection techniques. Section 4 involves the experimental exploration of different human detection techniques and their brief analyses. Finally, conclusions are presented in the section 5.

2. RELATED WORKS

Skin color modeling is a traditional approach that can detect humans based on the human's face [7]. This approach aims to find out the skin pixels within an image. There is a need to choose a suitable color space before skin color modeling [18]. Therefore, its performance depends on the appropriate color space. Although skin color modeling is fast and computationally efficient, it is sensitive to deal with lighting changes, cluttered backgrounds, ethnicity, etc. Background subtraction [19] and frame differencing [20] algorithms are another fast and computationally efficient algorithms for detecting moving objects. Both algorithms' performance may be degraded due to noise, reflection, different lighting conditions, and moving objects in the background. These methods have low complexity and are easy to use in real-time problems. This makes them more popular among the research community. As a result, several competent variants have been proposed for these algorithms. Jiajia Guo et al. [21] suggested a three-way frame differencing approach to detect moving objects more accurately. It is an enhanced variant of the frame differencing algorithm. Stauffer et al. [22] proposed an adaptive background fusion model for tracking objects in real-time. This method overcomes the various limitations where background subtraction becomes unstable. But, it is still unfit to deal with object overlapping and change in illumination problems. Li and Xu [23] proposed a framework for moving object detection using the Gaussian Mixture Model (GMM). It is found that the GMM method can't deal with dynamically varying backgrounds proficiently. Karpagavalli and Ramprasad [24] suggested a human detection framework to deal with the complex environment and illumination changes. They used a hybrid adaptive Gaussian mixture model and improved the human detection rate by up to 95%. This algorithm performs better only if the camera is stationary. Horn and Schunck [25] proposed an Optical flow algorithm for object detection by estimating the object position from consecutive video sequences or frames. This algorithm can't deal with cases where objects are fast in motion. Yugui et al. [26] suggested an effective motion based object detection method using optical flow estimation under a moving camera.

As traditional approaches still need improvement, different machine learning and deep learning paradigms have come into existence for object detection and recognition. Object detection & recognition through machine learning approaches require object features to build a classifier. These features can be color, texture, and shape features. Various popular feature representation techniques help to extract meaningful information (color, texture, and shape) from the images. These techniques include Color Moments [27], Local Binary Pattern (LBP) [11], Histogram of Gradients (HOG) [10], Edgelet Features [28], Histogram of Oriented Optical Flow (HOOF) [29], Fourier Transform [30] etc.

In 2001, Viola and Jones [31] suggested a robust framework to identify real-time objects. This research systematized the object detection framework using the Haar feature selection, Integral image concept, Adaboost training, and Cascade classifier. This algorithm sets a foundation in the field of facial detection by running at fifteen frames per second in real-time. Another robust visual object detection framework was proposed by Dalal and B. Triggs [32]. This framework adopted the Histogram of Gradient descriptor (HOG) for feature extraction and Support Vector Machine (SVM) for object classification. In addition to the HOG descriptor, linear binary pattern (LBP) can be an invaluable alternative for obtaining an image's necessary visual characteristics. However, LBP cannot deal with illumination variation and occlusion efficiently. Aftab Ahmed et al. [11] suggested that Local Binary Patterns can be another good option to extract the image's texture features. This research adopted the Cascade classifier for object classification. As a limitation, this method can't deal with occlusion, pose variation, and illumination changes efficiently. Vijay and Shashikant [33] used edgelet features with the cascade structure of K-Means clustering for pedestrian detection. It is found that this proposed method is suitable for real-time detection due to its minimum computational complexity. Irfan, Jingchun, and Hyunchul [34] proposed a human detection framework for the thermal image with the help of the CENTRIST visual descriptor. CENTRIST descriptor used object contours for representing meaningful information within an image. Further, these extracted features are used to build an SVM classifier. In this research, the authors discovered that the CENTRIST transform's performance is better than HOG in detection accuracy and execution time. Bahri et al. [35] configured the GPU environment using CUDA for moving human detection. They used

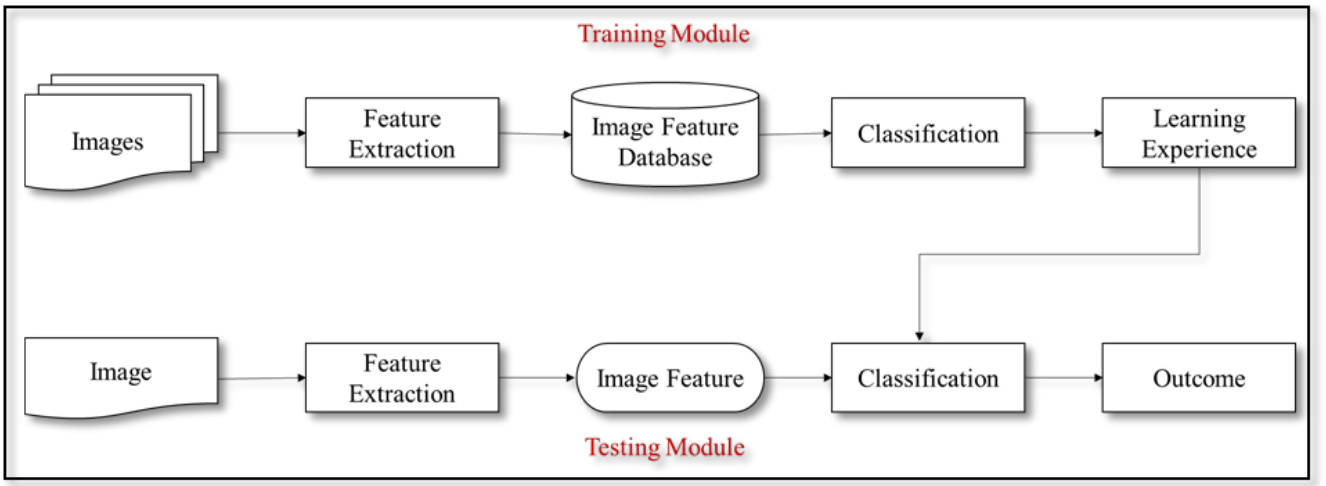


Figure 2. Complete human detection pipelining

contour-based and region-based descriptors to extract the essential features from the image. Also, both descriptors are effectively integrated with SVM classifiers for classification. The proposed framework performs 19 times faster with the GPU than the CPU. P. P. Gangal [36] deployed 2-D Discrete Wavelet Transform (DWT) for object detection and tracking. DWT is a multiresolution approach to obtain the required information from the image in more detail.

Recently, impressive advances in object detection have been discerned due to the robust feature representation capability employing the deep neural network [37], [38]. Deep learning models have outperformed other classical models on image classification, and they are now state of the art in object detection. Advances in deep learning algorithms perform increasingly well at handling vast amounts of image data. Tome et al. [39] studied various region proposal algorithms for object localization and adapted a general-purpose CNN framework for proposing an object detection module. Bubryur Kim [16] et al. proposed an advanced surveillance system to detect pedestrians using an optimized VGG16 network. They found that optimized VGG16 is a better option for building a smart building block to detect pedestrians for a surveillance system, which outperforms the VGG-16 and hybrid approaches of machine learning models. Ansari and Singh [40] used CNN architecture to detect humans for monitoring social distancing in real-time scenarios.

The existing feature learning based approaches for object detection have moderate accuracy and sometimes produce large false positive instances during the detection process. Therefore, in order to increase accuracy and reduce false-positive examples, different deep learning based models have been introduced to detect and track humans efficiently.

3. HUMAN DETECTION TECHNIQUES

Figure 2 depicts the complete framework of practicing humans to detect. This framework works in two modules: the training and testing modules. In the training module, the features are first extracted from each image of the dataset and stored in disk space. These stored features are called image feature database. Next, the classification task is performed for training purposes, which takes the image feature database as input and produces the learning experience with trained parameters. Further, this experience is used by the testing module to produce an outcome. In contrast to the testing module, it takes an image as input and decides an output. First, the test module extracts the input image features and passes the image feature to the classification task. By adopting trained learning experiences and norms, the classifier makes an outcome/decision as to whether the input image is of a person.

As shown in Figure 2, feature extraction and classification are the two main functions for structuring a practical human detection framework. Therefore, this section explores various learning based feature extraction techniques as well as classifiers in detail, which are as follows:

A. *Features*

The feature extraction task takes an image as input and produces features of interest as output. This subsection is focused on different techniques involved in feature extraction. These techniques include Color Histogram Descriptor (CHD), Histogram of Gradient (HOG), Local Binary Pattern (LBP), Discrete Wavelet Transform (DWT), and Gray Level Co-occurrence Matrix (GLCM). Apart from these, a more advanced machine learning technique, named Deep Convolutional Neural Network (CNN), is hyper-tuned with different parameters like epochs, optimizer, activation function etc., and produces various variations in human detection techniques, shown in section 3-C. These techniques are briefly described as follows:

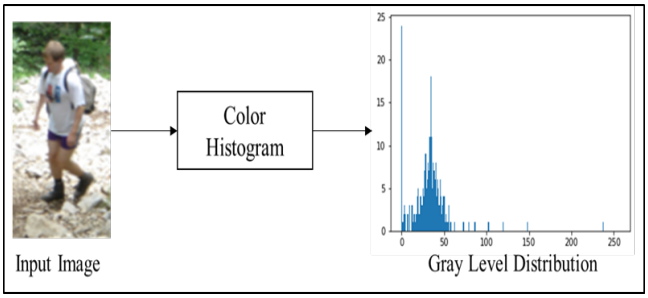


Figure 3. Color histogram

1) Color Histogram Descriptor (CHD)

Color is the dominant descriptor that often categorizes the object and extraction scene. Color Histogram Descriptor (CHD) [9] is a color based descriptor that extracts the color features from an image. Color histogram is a graphical representation of the distribution of colors within an image. CHD is assessed by counting each gray level's occurrence using the respective color space in an image and forming a representing histogram. It shows the appearance of the color with their pixel counts in an image. Figure 3 shows the color histogram of the inputted image. The horizontal axis shows the gray-level values between 0 and 255, and the vertical axis indicates a particular gray level occurrence value within an image. Its simplicity and rapid feature generation capability are the main characteristics of the CHD descriptor. Apart from these, CHD is relatively invariant to the rotation and translation of an image that makes the CHD a robust color descriptor.

2) Histogram of Gradients (HOG)

In 1986, Robert K. McConnell [41] of Wayland Research Inc. illustrated the idea behind HOG. However, HOG's usage became pervasive in 2005 when Navneet and Bill [32] offered supplemental work on the HOG descriptor for human detection. Precisely, it can be styled as evaluating a non-linear function of individual edge alignments within an image and inserting them into miniature spatial regions to reduce sensitivity for accurate localization of edges. The technique counts the gradient orientation events in local parts of an image that take contours and silhouette statistics of greyscale images. HOG takes an image as input and provides 1D histogram vectors as output. Further, these vectors are used to build a classifier. Figure 4 visualizes the extracted HOG features extracted from the image.

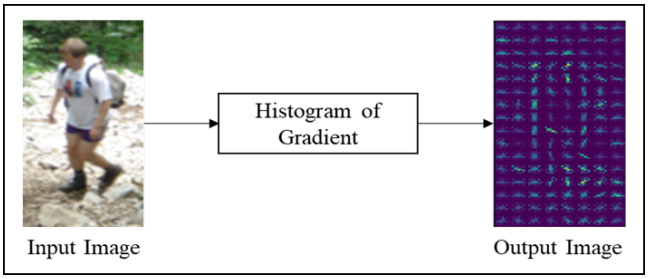


Figure 4. HOG features visualization

The idea of HOG is to group the pixels into cells of fixed size like 8×8 or 16×16, etc. All the gradient's directions are calculated for each cell, and then they are arranged in several orientation bins like 0^o, 20^o, 45^o, etc. Gradients are the significant variation in intensity in the successive directions that are evaluated using equations 1 and 2.

$$Mag_{(i,j)} = \sqrt[2]{(G_i^2 + G_j^2)} \tag{1}$$

$$\tan\theta_{(i,j)} = \left(\frac{G_i}{G_j}\right) \tag{2}$$

Henceforth, the value of θ is shown as follows:

$$\theta_{i,j} = \tan^{-1}\left(\frac{G_i}{G_j}\right) \tag{3}$$

Where $Mag_{i,j)}$ is the magnitude of the gradient that represents the strength of the gradient. G_i and G_j are the gradients in i and j directions. θ is the orientation of gradients. Further, these evaluated gradients are placed into the bins according to their magnitude and direction and create a histogram. This process helps to create a histogram-based gradient on the image. Because gradients are sensitive to lightning variations, normalization is used to minimize the effect of lightning. The block normalization can be possible by choosing one of the norms. At last, histograms for each cell are concatenated into a large vector, called feature vector, that uniquely represents the image. Furthermore, this large feature vector can be tied with different machine learning paradigm for object detection and recognition.

3) Local Binary Pattern (LBP)

LBP is a visual texture descriptor that is used to extract local spatial patterns in computer vision. LBP was first described in 1994 by T. Ojala, M. Pietikäinen, and D. Harwood [42]. It was found to be a very powerful feature descriptor for texture classification due to its discriminative power and simplicity. LBP labels the pixels by thresholding the value of neighborhood pixels based on the current pixel's value. It is a unified methodology for the statistical and structural model in texture classification. The operation of LBP for 3x3 pixel size is shown in equation 4. Each pixel in the 3x3 neighborhood is compared to its eight neighborhoods by subtracting the central pixel value. Where resulting positive values are set with one and the others with zero.

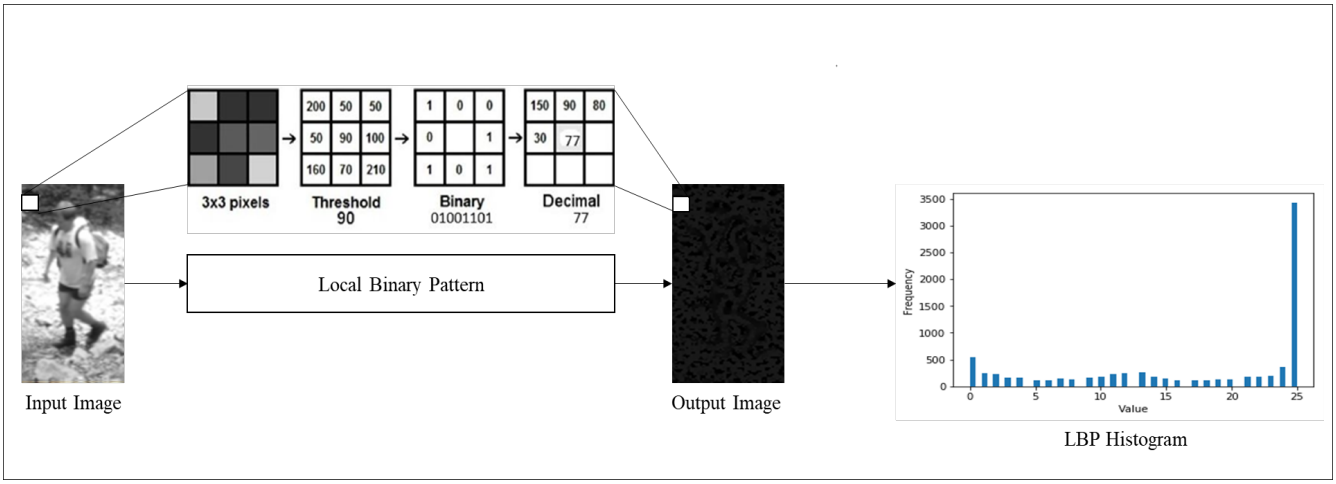


Figure 5. LBP operation

$$S(x) = \begin{cases} 1 & x \geq 0 \\ 0 & otherwise \end{cases} \quad (4)$$

Where x is an anticipated pixel for which the LBP operation is evaluated. Figure 5 illustrates the LBP operation, which takes a grey image as input and produces an LBP image as output. The histogram of the LBP output image is calculated for 26 bins. Further, these 26 bins are considered as feature sequences of LBP. The LBP operation first evaluates the binary codes from the central pixel difference. These binary codes are flattened in a clockwise direction, and their decimal value is calculated. Finally, it replaces the evaluated decimal value with its corresponding pixel. This phenomenon is performed for the entire image that produces an LBP output image.

4) Discrete Wavelet Transform (DWT)

Discrete Wavelet Transform [12] is another influential and well-known texture descriptor. It deals with the structural information of an image. The term wavelet denotes short waves of varying frequencies with finite timestamps. The idea behind the wavelet transform is that a large-sized object at high resolution can be represented by a small-sized object with a low contrast image. DWT mainly includes two sets of functions: scaling function and wavelet function. The low pass filter is used for the scaling function, and the high pass filter is used for the wavelet function. Different frequency bands are obtained while passing a signal in the time domain to successive low and high pass filters. These frequency bands consist of essential details of the signal. This constitute is called one level DWT. This process can be repeated to get the information more in-depth by passing a signal into the successive low pass and high pass filter.

Here, 2 level DWT is used to extract the structural information from an image's surface, shown in Figure 6. The wavelet transforms decomposed the signal into the LL, HL, LH, and HH bands, including approximation,

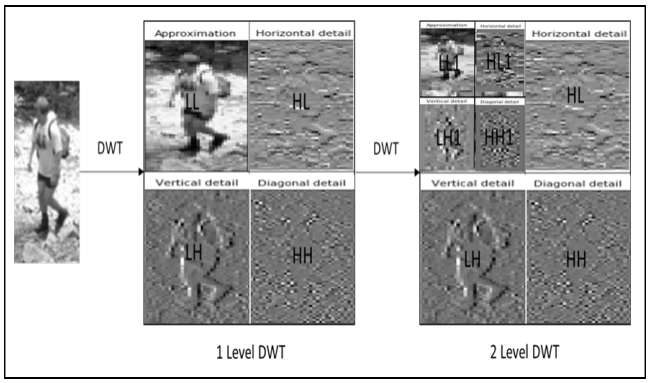


Figure 6. Two-level discrete wavelet decomposition

horizontal, vertical, and diagonal details of an image. It is found that only the LL band contains complete information of an image, while HL, LH, and HH bands are contaminated by noises. So, we left these bands and only processed them to the LL band. Further, the LL band is again decomposed in LL1, HL1, LH1, and HH1 bands. At last, features are evaluated for the LL1 band by calculating the row-wise mean and standard deviation value.

5) Gray Level Co-Occurrence Matrix (GLCM)

GLCM [13], [9] is a textural feature representative descriptor that examines texture by considering the spatial relationship between pixels relative to the high order statistics from an image. It epitomizes pixel frequency formation by calculating how frequently a pixel with brightness value i occur in a particular spatial connection to a pixel with the brightness value j. Elements of the resulting matrix follow a second-order statistical probability value based on gray-level values. Each element of the GLCM 2D feature matrix is the sum of the frequently occurred relationship of each pixel pair (i, j) in the specified direction. GLCM takes an image as input and produces a 2D feature matrix as output. This 2D feature matrix consists of statistical

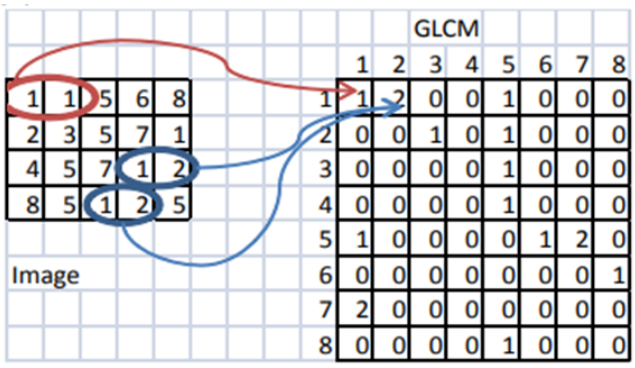


Figure 7. GLCM Operation [13]

values by evaluating the co-occurrence of gray levels for the specified direction. Further, spatial measures are evaluated to form the essential features from the 2D feature matrix. The widely used spatial measures of GLCM are as follows: contrast, entropy, energy, homogeneity, difference variance, difference entropy, autocorrelation, inverse difference normalized, etc. The basic operation of GLCM is depicted in Figure 7 for *distance* = 1, and *angle* = 0^o. Where D is any fixed integer value between 1 to the size of the image, and angle represents the direction (such as $0^o, 45^o, 90^o and 135^o$) in which spatial relations are required to be maintained.

B. Classification

Classification [6], [12], [43], [44] is a task in which objects are understood, differentiated, and recognized. The ultimate goal of classification is to find the class under which new data falls. It takes features as input and maps them to a specific category to produce output. The classification is based on supervised learning, where the target vectors are also provided along with the input data. Many classification algorithms are available nowadays, but it is still under research as to which one is better. Some of the well-known classifiers are discussed as follows.

Logistic Regression (LR) is a predictive analysis algorithm that is based on the concept of probability. It has a logistic function that creates possibilities to describe the possible results of a single test. LR works fine only when the predicted variable is categorical in nature. It is the most straightforward machine learning algorithm which follows the assumption that independent variables must be independent of each other. LR uses the sigmoid function (shown in equation 5) in data modeling and maps the values between 0 to 1.

$$g(z) = \frac{1}{\sqrt{1 + e^{-z}}} \tag{5}$$

Where z is the data's numeric value, and e is the logarithmic base. The LR involves the concept of thresholding that expresses the probability of either 0 or 1. Such as a value below the threshold value tends to be 1; otherwise, it tends to 0.

Naïve Bayes (NB) classification algorithm is exceedingly fast compared to other classification algorithms, and it can work on non-linear problems. It is based on Bayes' theorem with the hypothesis of independence between the characteristics of each pair. It follows the assumption of predictor's liberation in which features in one class would not depend on another class. The Naïve Bayes classification algorithm is easy to implement and effectively handles enormous datasets. Gaussian NB is a special case of the NB algorithm in which all features are assumed to be followed the Gaussian distribution. Equation 6 shows the Gaussian distribution function used for Gaussian NB.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{6}$$

Where σ_y and μ_y are mean and standard deviation that are assessed using maximum likelihood. Gaussian NB can deal with features with continuous values.

Support Vector Classifier (SVC) is proved to be a defacto standard algorithm for classification problems. The SVC is a discriminative classifier that uses a hyperplane for categorizing the data. It mapped the data points into separate categories by introducing a margin between them as wide as possible. SVC's primary goal is to form the best-suited decision boundary/hyperplane with the maximum possible margin to separate the data space of n-dimensional into their respective classes. It consists of different kernel functions to determine the decision boundaries. The kernel function can be linear, non-linear, polynomial, radial basis function, and sigmoid. Nu-SVC [45] is one of the support vector machine variants that control the number of support vectors. The support vectors are the closest points to the hyperplane that decides the position of the hyperplane. Equation 7 shows the way in which any hyperplane can be written as:

$$\vec{w}.\vec{x} - b = 0 \tag{7}$$

Where $\vec{x}$, $\vec{w}$ and b are set of points, weight vector, and bias, respectively. The parameter $\frac{b}{\|\vec{w}\|}$ controls the offset of the hyperplane. In D dimensional space, the hyperplane would always be D-1.

K Nearest Neighbor (KNN) is the most basic and popular classification algorithm of machine learning. KNN carries the assumption that related things fall in close proximity, which makes it a more sophisticated classifier. It is also known as a lazy learner algorithm that does not learn immediately from the training dataset. It first stores the dataset and then performs an action at the time of classification. It allots a test pattern to its nearest neighbors. Assume there are n training patterns like $(X_1, \theta_1), (X_2, \theta_2), , , (X_n, \theta_n)$,

where X_i and θ_i are the dimension (d) and class label of i^{th} pattern, respectively. If the test pattern is P, then

$$d(P, X_k) = \min\{d(P, X_i)\} \quad (8)$$

Where $1 \leq i \leq n$ and P is allotted to the class θ_k accompanying with X_k. KNN stores all existing data and categorizes the new data point based on similarity. It can also handle arbitrary distribution.

Decision Tree (DT) is a powerful tool for classification. It consists of a tree-like structure in which internal nodes represent the test on features, braches specifies the test's outcome, and leaf nodes entail class label. RF takes image features together with input vectors as inputs and gives a sequence of rules that can be used to classify the data. It utilizes IF-THEN rule sets, which make the classifier mutually exclusive and exhaustive. DT algorithm uses attribute selection measures to select the finest attributes. It breaks the dataset into smaller subsets and builds a decision tree based on some measurements. DT can handle both numerical and categorical data.

Another classification algorithm, named ***Random Forest (RF)***, is used for both regression and classification purposes. RF algorithm builds decision trees on data samples and then drives predictions from each of them and chooses the best solution from voting. One of the critical features of RF is that it works well for data items on a larger scale than a decision tree and has a lower degree of the variance than a single decision tree. RF also surpasses the overfitting problem by averaging the various decision tree's results and produces good accuracy. RF algorithm is much complex and time-consuming than a decision tree.

Apart from these, some other classifiers such as ***Gradient Boosting Classifier (GBC)***, ***Adaboost Classifier (AC)***, ***Linear Discriminant Analysis (LDA)*** and ***Quadratic Discriminant Analysis (QDA)*** have also gained popularity among classifiers. GBC creates a robust predictive model by combining many weak learning models. It provides more effective results for classifying more complex datasets. Adaboost classifier is a meta-estimator that combines multiple poorly performing classifiers to get a robust classifier that produces better accuracy. The LDA classifier uses Bayes' rule to fit the conditional class densities to the data with a linear decision boundary. It fits the Gaussian density to each class. LDA follows the assumption that all classes share the same covariance matrix. In contrast, QDA behaves like LDA, but the only difference is that QDA follows the belief that each class consists of its own covariance matrix, while LDA does not.

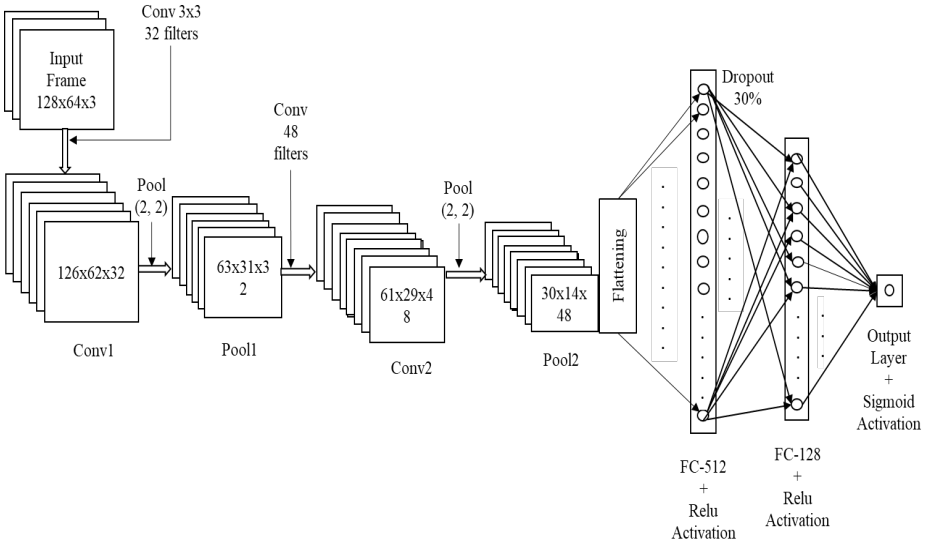
C. Deep Learning

Deep Learning [15], [46] is an advanced machine learning method based on artificial intelligence networks with representation learning. It mimics the functionality of the human brain in processing the data for making decisions. Deep learning has a broader range of applications, including bioinformatics, machine translation, audio recognition, image classification, computer vision, etc. One of the advantages of deep learning is that it doesn't extract the features externally as the previous machine learning algorithm does. It extracts features implicitly with the help of convolution operation, and then learning is performed based on these implicit extracted features.

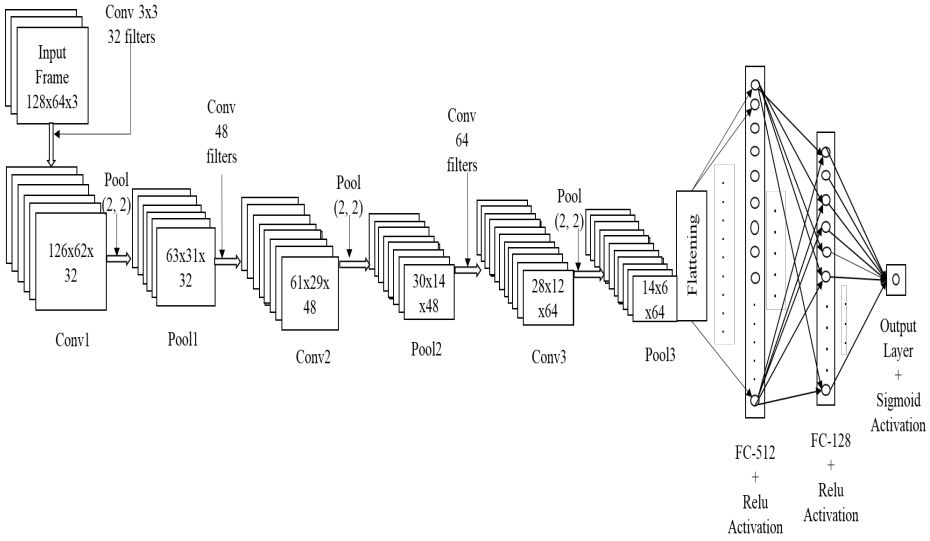
Convolutional Neural Network (CNN) or ConvNet [2], [3], [47], [48], [38] is a deep learning network used to analyze visual imageries. ConvNet is quiet in the emerging phase and used extensively to solve various real-time computer vision problems. CNN has a proficient capability to discover and learn good representations using automatic feature learning. It inputs an image scene as input, assigns significance to different objects based on weight and bias, and distinguishes each object proficiently. CNN consists of a convolution layer, pooling layer, fully connected layer, and output layer. In this context, the convolutional layer incorporates a specific kind of linear operation, called convolution operation, which extracts some meaningful information from the image. The pooling layer is another building block of CNN. It operates on each feature map independently and reduces the spatial representation of the input data. In contrast, fully connected layers consist of sets of neurons, take input from the pooling layer, and apply weights to predict the correct label. The output layer is the last layer of the CNN building block in which final probabilities for each label are evaluated.

This paper suggested two deep learning models to spot a person's existence within an image capably. The models contain a convolutional layer, pooling layer, fully connected layer, and output layer. The output shapes corresponding to evaluated parameters are also illustrated for each of them. The network architecture of both Models is quite similar, except Model 2 has one additional combination of convolutional and pooling layers. In this context, Model 1 contains two consecutive sets of convolutional and pooling layers, followed by a fully connected layer and output layer. In contrast, Model 2 includes three consecutive sets of convolutional layers and pooling layer, fully connected layer, and output layer. Due to this variation, Model 1 yields around $10,402,993$ training parameters, while Model 2 yields $2,861,297$.

Figure 8 shows the pictorial structure of both Models. Both inputs color image and provide the predicted value as an outcome. Model 1 incorporates 32 filters of convolution with a pooling layer of size 2×2 in one set and 48 filters of convolution with a pooling layer of 2×2 in the second set. In contrast, the architecture of Model 2 is the same as Model 1, except it consists additional layer of a convolutional layer (incorporates 64 filters) and a pooling layer of size 2×2 in the last set. The stride value and kernel size used for convolution operation are $(1, 1)$, and $(3, 3)$, respectively.



(a)



(b)

Figure 8. Architecture of (a) Model 1, (b) Model 2

Both of the Models use two fully connected (FC) layers and one output layer. FC layers are responsible for assigning weight and bias to neural networks, while the output layer produces the results for the given input.

4. EXPERIMENTAL OUTCOMES

This paper presents multiple techniques to notice the human presence. All experiments have been done on the machine running on the Window 10 operating environment. The machine comprises of Intel i3 (5^{th} Generation) CPU@2.00 GHz processor and Google Colab in Python. INRIA image dataset [49] is used for training and testing purposes. It involves 6562 images, of which 4146 images are negative images and 2416 images are positive. Finally, we spitted our image dataset into an 80:20 ratio for training and testing purposes. The experimental outcomes of these proposed techniques for human detection are discussed as follows:

The ***first technique*** uses a color histogram descriptor for feature extraction. It is tuned with 256 bins for storing histogram outcomes. It takes 151.695 seconds to extract the features from 6562 images, while it takes 1.99 seconds to load these extracted features into memory for training purposes. The behavior of color histogram descriptor with different classifiers is depicted in Table I.

The ***second technique*** uses the HOG descriptor to

TABLE I. Performance of CHD for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	442	327	185	176	0.715	0.361	68.053	0.591	0.454
KNN	489	388	138	115	0.809	0.262	77.610	1.974	0.116
SVC Linear	559	377	94	100	0.848	0.999	82.832	0.385	5.211
SVC RBF	669	369	0	90	0.881	0.000	92.021	0.277	6.384
Nu SVC	629	389	0	112	0.848	0.000	90.088	0.281	1.411
DT	548	471	79	32	0.945	0.143	90.177	3.292	0.509
RF	590	471	37	32	0.948	0.072	**93.893**	0.273	0.240
AC	552	429	75	74	0.882	0.148	86.814	0.642	2.439
GBC	574	450	53	53	0.915	0.105	90.619	0.256	5.325
Gaussian NB	306	443	321	60	0.836	0.420	66.283	10.682	0.025
LDA	462	266	165	237	0.661	0.383	64.425	0.618	0.226
QDA	285	444	342	59	0.828	0.435	64.513	11.769	0.306

TABLE II. Performance of HOG for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	616	436	32	46	0.930	0.068	**93.097**	0.186	4.293
KNN	648	37	0	445	0.592	0.000	60.619	12.158	2.187
SVC Linear	602	428	46	54	0.917	0.097	91.150	0.220	321.679
SVC RBF	648	0	0	482	0.573	0.000	57.345	0.384	975.230
Nu SVC	620	410	28	72	0.895	0.063	91.150	0.213	754.722
DT	519	335	129	147	0.779	0.278	75.575	8.436	46.134
RF	593	359	55	123	0.828	0.133	84.247	0.389	3.8429
AC	590	418	58	64	0.902	0.121	89.203	0.646	226.396
GBC	617	434	31	48	0.928	0.066	93.008	0.204	421.600
Gaussian NB	517	416	131	66	0.886	0.239	82.566	5.529	1.202
LDA	533	370	115	112	0.826	0.237	79.911	3.753	157.062
QDA	9	482	639	0	1.000	0.570	43.451	19.531	51.762

extract the features. The HOG descriptor is hyper tuned for nine orientations, (8, 8) pixels_per_cell, (3, 3) cells_per_block, and 'L1' block norm. It yields features of length 6479. It takes 227.226 seconds to extract HOG features from 6562 images, while it takes 17.685 seconds to load these extracted HOG features into memory for training purposes. Finally, the classification process uses HOG features to label the object in its corresponding class. The behavior of HOG descriptors in the presence of different classifiers is presented in Table II.

The ***third technique*** employs the LBP feature descriptor for feature extraction. It is hyper tuned for '3' radius value and 24 points. It produces 26 lengths of feature vectors. LBP operation takes 190.042 seconds for feature extraction of 6562 images, while it takes 2.259 seconds to load the features into memory for training purposes. Finally, Table III presents the performance of LBP for different classifiers.

The ***fourth technique*** uses the DWT feature descriptor to extract the features. Further, these features are used to build different classifiers. The DWT operation is hyper tuned for two-level transformation. The features are formed by evaluating column-wise mean and variance values of the resultant approximation band of DWT. It yields 104 vectors as features. DWT operation takes 248.887 seconds to extract the features from 6562 images, while it takes 7.215 seconds to load the feature into memory for training purposes. The performance of the DWT descriptor for different classifiers is presented in Table IV.

The ***fifth technique*** uses the GLCM descriptor to extract the features from the image. The features are evaluated based on different statistical measures: contrast, dissimilarity, homogeneity, ASM, energy, and correlation. GLCM is hyper tuned with *distance* = 2 and *angle* = 0 degrees. It takes 172.661 seconds for feature extraction, while it takes 2.954 seconds to load the features into memory for training purposes. Table V presents the performance of GLCM for different classifiers.

The ***sixth technique*** uses the fusion-based approach by combining CHD and HOG descriptors for feature extraction. This fusion-based method provides 6735 vectors as

TABLE III. Performance of LBP for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	509	309	134	178	0.740	0.302	72.389	0.575	0.031
KNN	539	445	104	42	0.927	0.189	87.079	1.518	0.016
SVC Linear	480	385	163	102	0.824	0.313	76.548	0.559	4.413
SVC RBF	567	41	76	446	0.559	0.649	53.805	0.649	9.269
Nu SVC	540	423	103	64	0.894	0.196	85.221	0.371	8.217
DT	580	458	63	29	0.952	0.121	91.858	2.812	0.233
RF	602	452	41	35	0.945	0.083	**93.274**	0.337	0.453
AC	568	419	75	68	0.893	0.151	87.345	0.584	0.891
GBC	586	431	57	56	0.912	0.117	90.000	0.244	1.344
Gaussian NB	501	449	142	38	0.929	0.240	84.070	1.635	0.046
LDA	542	401	101	86	0.863	0.201	83.451	0.386	0.265
QDA	387	481	256	6	0.984	0.347	76.814	1.087	0.015

TABLE IV. Performance of DWT for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	567	385	92	8	6 0.868	0.193	84.247	0.359	0.408
KNN	553	410	106	61	0.900	0.205	85.221	1.367	0.094
SVC Linear	557	382	82	89	0.862	0.176	84.867	0.354	6.362
SVC RBF	580	373	79	98	0.855	0.174	84.336	0.351	0.351
Nu SVC	580	385	79	86	0.870	0.170	85.398	0.331	14.446
DT	563	449	96	22	0.962	0.176	89.557	3.607	0.618
RF	607	437	52	34	0.947	0.106	**92.389**	0.300	0.484
AC	570	397	89	74	0.885	0.183	85.575	0.644	3.365
GBC	593	433	66	38	0.940	0.132	90.796	0.250	4.166
Gaussian NB	449	417	210	54	0.892	0.335	76.637	2.279	0.016
LDA	590	378	89	93	0.864	0.190	83.894	0.363	1.608
QDA	597	160	62	311	0.657	0.279	66.991	7.726	0.078

TABLE V. Performance of GLCM for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	365	329	271	165	0.688	0.451	61.452	0.612	0.379
KNN	442	370	194	124	0.781	0.344	71.858	2.606	0.039
SVC Linear	254	428	382	66	0.794	0.471	60.354	0.612	2.841
SVC RBF	250	425	386	69	0.783	0.475	59.734	0.636	5.240
Nu SVC	474	274	162	220	0.683	0.371	66.194	0.615	5.221
DT	529	446	107	48	0.917	0.193	86.283	4.737	0.065
RF	522	455	114	39	0.930	0.200	**86.460**	0.699	0.321
AC	433	425	203	69	0.862	0.323	75.929	0.615	0.453
GBC	489	418	147	76	0.865	0.260	80.265	0.439	0.485
Gaussian NB	211	468	425	26	0.890	0.476	60.088	3.350	0.01
LDA	418	242	218	252	0.624	0.474	58.407	0.617	1.528
QDA	274	462	362	32	0.895	0.439	65.133	1.813	0.309

features. It takes 330.121 seconds to extract the features, while it takes 112.954 seconds to load the features in the memory for training. Table VI presents the performance of CHD + HOG descriptors for different classifiers.

The ***seventh technique*** is practiced with a fusion-based approach by combining HOG and LBP feature descriptors to extract the features. It provides 6506 vectors as features. Table VII presents the experimental results for this fused approach. The fusion operation of HOG and LBP takes 335.903 seconds for feature extraction of 5650 images, while it takes 121.256 seconds to load the features in the memory for training purposes.

The ***eighth technique*** also uses a fusion-based approach by combining LBP and two-level DWT feature descriptors to extract the image's meaningful features. This combined approach produces 130 vectors as features. It takes 286.617 seconds for feature extraction of 6562 images, while it takes 2.398 seconds to load the features in the memory for training purposes. Table VIII presents the experimental results for LBP + 2-L DWT with different classifiers.

The ***ninth technique*** continues a fusion-based approach involving two-level DWT and GLCM feature descriptors to extract the image's features. It produces overall 110 feature vectors. It takes 602.649 seconds for feature extraction of 6562 images, while it takes 6.324 seconds to load the features in the memory for training purposes. Table IX presents the performance of 2-DWT and GLCM descriptors for different classifiers.

The ***tenth technique*** uses deep CNN architecture to detect the human's presence. This technique proposed two different Models, namely ***Model 1*** and ***Model 2***. A brief description of these models is already described in section 3-C. Here, the Models are modeled with various parameters like Epochs, Optimizer, Dropout, Batch size and Activation function. Table X presents the hyperparameter tuning for different variants of the proposed Models. The experimentation cost for CNN based Models is quite expensive in our existing platform (i3 Processor, 4GB Ram). In contrast, it runs smoothly in the Google Colab (GC) platform with the GPU environment and provides low computation costs.

These proposed models are trained and tested over different hyper-parameters and deliver proficient results, as shown in Table XI. From ***Model 1***, ***Model 1.1*** and ***Model 1.3*** provide most accurate results in terms of accuracy with little variation. ***Model 1.1*** is hyper tuned with '16' batch size, 'Relu' activation function for convolutional layer and FC layer, 'Sigmoid' activation for Output layer, '120' epochs, 'Adam' optimizer and '30%' dropout rate. The structure of Model 1.3 is hyper-tuned in the same way as ***Model 1.1***, except ***Model 1.3*** is run on the Google Colab environment while ***Model 1.1*** is run on our system. After experimentations, it is observed that our system process the whole database for ***Model 1.1*** in 8.51 Hrs while the same variant finishes their computation in 43.37 minutes using Google Colab. Therefore, we used Google Colab to experiment with most of the variants of Models. Another variant of ***Model 1***, namely ***Model 1.5***, also achieves 98% testing accuracy with relatively shorter validation loss.

From ***Model 2***, ***Model 2.1*** has explored its range adequately in terms of accuracy. This model comprises batch size '8', dropout '30%', activation function 'Relu' for convolutional layer and FC layer, output layer activation 'Sigmoid' and optimizer 'Adam'. It yields 99% testing accuracy and takes about 36 minutes to train in Google Colab Platform with the GPU environment. Another variant of ***Model 2***, namely ***Model 2.3***, is similar to ***Model 2.1***, except it uses 'Tanh' activation in the FC layer instead of 'Relu' activation. ***Model 2.3*** produces the second-highest result in terms of accuracy and takes 37 minutes in training.

Upon analysis, it has been found that ***Model 2.1*** offers more encouraging outcomes compared to others. ***Model 2.3*** is also providing quite promising results after ***Model 2.1***. Figure 9 illustrates the accuracy and loss plots over 120 epochs for ***Model 2.1***.

A. Experimental Analysis

CHD is a color descriptor that mines color information from the image and forms features. It yields 94.89% accuracy with the RF classifier. But, in CHD, the representation is dependent on colors only. However, two different images with different object content may have a similar color histogram. Therefore, along with color information, some texture and shape information is required in order to build a proficient object detector.

HOG is a shape descriptor that extracts shape information in terms of gradients' magnitude and direction. It provides 93.09% and 93.00% accuracy with the LR classifier and GBC classifier, respectively. HOG provides the most prolonged feature sequences for object representation. Therefore, it takes more time to perform feature extraction and training tasks than other descriptors. Another shape-based descriptor, namely DWT, follows a multiresolution approach and provides 92.38% accuracy with the RF classifier. DWT descriptors perform proficiently, but their outputted bands are more immune to noise. Therefore, it affects the resultant images.

LBP extracts textural patterns from the image. It has high discriminative power and provides 93.274% accuracy with the RF classifier. It is invariant to grayscale images and computationally efficient. However, it is sensitive to affine transform. In contrast, GLCM uses higher-level statistics for features extraction and provides 86.460% accuracy with the RF classifier. In study [12], it is found that the single feature descriptor can't reflect the whole image dataset accurately. The fusion-based approaches deliver more accurate outcomes than single descriptor-based methods. Therefore, combining these descriptors is necessary to build a more accurate approach. Hence, this paper also tried to explore some fusion based techniques.

TABLE VI. Performance of CHD + HOG for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	451	313	208	158	0.740	0.399	67.610	0.661	6.774
KNN	519	342	140	129	0.801	0.290	76.194	2.107	1.774
SVC Linear	611	411	57	51	0.923	0.121	90.442	0.211	700.008
SVC RBF	659	367	0	104	0.863	0.000	90.796	0.267	682.750
Nu SVC	659	365	2	104	0.864	0.005	90.620	0.266	643.574
DT	540	361	119	110	0.831	0.248	79.734	6.999	29.062
RF	618	367	41	104	0.856	0.100	87.168	0.387	2.166
AC	613	413	46	58	0.913	0.100	90.796	0.636	136.281
GBC	636	441	23	30	0.955	0.049	**95.309**	0.158	304.026
Gaussian NB	471	445	188	26	0.947	0.297	81.062	6.057	0.723
LDA	538	377	121	94	0.851	0.243	80.973	3.610	156.486
QDA	12	471	647	0	1.000	0.578	42.743	19.775	41.246

TABLE VII. Performance of HOG + LBP for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	604	453	35	38	0.941	0.071	93.539	0.170	5.875
KNN	638	33	1	458	0.582	0.029	59.380	12.676	2.265
SVC Linear	583	455	56	36	0.942	0.109	91.858	0.195	307.306
SVC RBF	639	0	0	491	0.565	0.000	56.54	0.375	958.845
Nu SVC	606	428	33	63	0.958	0.071	91.504	0.209	819.344
DT	552	399	87	92	0.857	0.179	84.159	5.471	150.836
RF	602	386	37	105	0.851	0.087	87.433	0.0355	7.249
AC	591	458	48	33	0.947	0.094	92.831	0.6205	257.644
GBC	618	463	21	28	0.956	0.043	**95.663**	0.130	483.820
Gaussian NB	519	437	54	120	0.812	0.109	84.601	4.930	2.25
LDA	533	412	106	79	0.871	0.309	83.628	3.257	173.43
QDA	13	491	626	0	1.00	0.560	44.601	19.113	82.626

TABLE VIII. Performance of LBP + 2-DWT for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	525	277	115	213	0.711	0.293	70.973	0.567	0.108
KNN	529	461	111	29	0.948	0.194	87.610	1.571	0.046
SVC Linear	489	390	151	100	0.830	0.279	77.787	0.536	13.246
SVC RBF	640	0	0	490	0.566	0.000	56.637	0.618	19.954
Nu SVC	509	439	131	51	0.909	0.229	83.893	0.381	15.961
DT	557	468	83	22	0.962	0.150	90.708	3.209	0.125
RF	595	474	45	16	0.974	0.086	**94.601**	0.281	0.125
AC	549	429	91	61	0.900	0.175	86.548	0.640	0.811
GBC	573	440	67	50	0.919	0.126	89.646	0.250	1.203
Gaussian NB	481	456	159	34	0.934	0.258	82.920	1.816	0.032
LDA	554	375	86	115	0.828	0.186	82.212	0.383	0.171
QDA	160	490	480	0	1.000	0.495	57.522	12.925	0.046

TABLE IX. Performance of 2-DWT + GLCM for Different Classifiers

Classifiers	TP	TN	FP	FN	TPR	FPR	Accuracy	Log Loss	Training Time (*In Sec.*)
LR	582	362	79	107	0.845	0.179	83.539	0.359	0.377
KNN	559	416	102	53	0.913	0.197	86.283	1.125	0.039
SVC Linear	577	376	84	93	0.861	0.182	84.336	0.358	11.698
SVC RBF	584	364	77	105	0.847	0.174	83.894	0.365	17.570
Nu SVC	580	365	81	104	0.848	0.181	83.628	0.358	25.666
DT	575	437	86	32	0.947	0.164	89.557	3.606	0.798
RF	628	436	33	33	0.950	0.070	**94.159**	0.347	0.719
AC	588	389	73	80	0.880	0.158	86.460	0.619	4.732
GBC	599	425	62	44	0.931	0.127	90.619	0.245	7.182
Gaussian NB	412	420	249	49	0.894	0.372	73.628	3.653	0.024
LDA	565	366	96	103	0.846	0.208	82.389	0.376	1.253
QDA	483	422	178	47	0.911	0.296	80.088	1.534	0.175

TABLE X. Parameter Hyper Tuning for Proposed Models

Name	Variant	Batch Size	Dropout	Activation			Optimizer	Epochs	Environment
				Convolutional Layer	FC Layer	Output Layer			
Model 1	Model 1.1	16	0.30	Relu	Relu	Sigmoid	Adam	120	GC
	Model 1.2	8	0.30	Relu	Relu	Sigmoid	Adam	120	Our System
	Model 1.3	16	0.30	Relu	Relu	Sigmoid	Adam	120	Our System
	Model 1.4	8	0.30	Relu	Relu	Sigmoid	Adam	120	GC
	Model 1.5	8	0.40	Relu	Tanh	Sigmoid	Ada Delta	100	GC
	Model 1.6	16	0.40	Tanh	Tanh	Sigmoid	Rmsprop	100	GC
Model 2	Model 2.1	8	0.30	Relu	Relu	Sigmoid	Adam	120	GC
	Model 2.2	16	0.30	Relu	Relu	Sigmoid	Adam	120	GC
	Model 2.3	8	0.50	Relu	Tanh	Sigmoid	Ada Delta	120	GC
	Model 2.4	16	0.50	Tanh	Tanh	Sigmoid	Rmsprop	100	GC
	Model 2.5	8	0.20	LeakyRelu	Tanh	Sigmoid	Ada Delta	100	GC
	Model 2.6	16	0.35	LeakyRelu	Tanh	Sigmoid	Adagrad	110	GC

TABLE XI. Outcomes of Different Variants for Model 1 and Model 2

Model Variants	TP	TN	FP	FN	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Training Time
Model 1.1	100	96	0	4	99.92	0.0022	**98.00**	1.6323e-05	43.37 *Min.*
Model 1.2	100	94	0	6	99.57	0.0213	97.00	0.00178	7.56 *Hrs.*
Model 1.3	100	95	0	5	99.81	0.0132	97.50	2.3809e-10	8:51 *Hrs.*
Model 1.4	100	83	0	17	99.65	0.0143	91.50	5.4591	41.921 *Min.*
Model 1.5	98	98	2	2	99.75	0.0098	**98.00**	5.8867e-05	44.863 *Min.*
Model 1.6	100	81	0	19	97.89	0.0692	90.52	0.6356	31.695 *Min.*
Model 2.1	100	98	0	2	99.54	0.0180	**99.00**	0.0142	35.52 *Min.*
Model 2.2	100	90	0	10	99.73	0.0097	95.02	0.6425	35.84 Min.
Model 2.3	100	97	0	3	99.81	0.0057	**98.50**	9.6017e-05	36.99 *Min.*
Model 2.4	100	93	0	7	99.18	0.0322	96.43	0.3866	36.333 *Min.*
Model 2.5	99	96	1	4	99.87	0.0048	97.50	1.7378e-07	32.059 *Min.*
Model 2.6	100	90	0	10	99.97	0.0011	94.99	0.7467	39.931 *Min.*

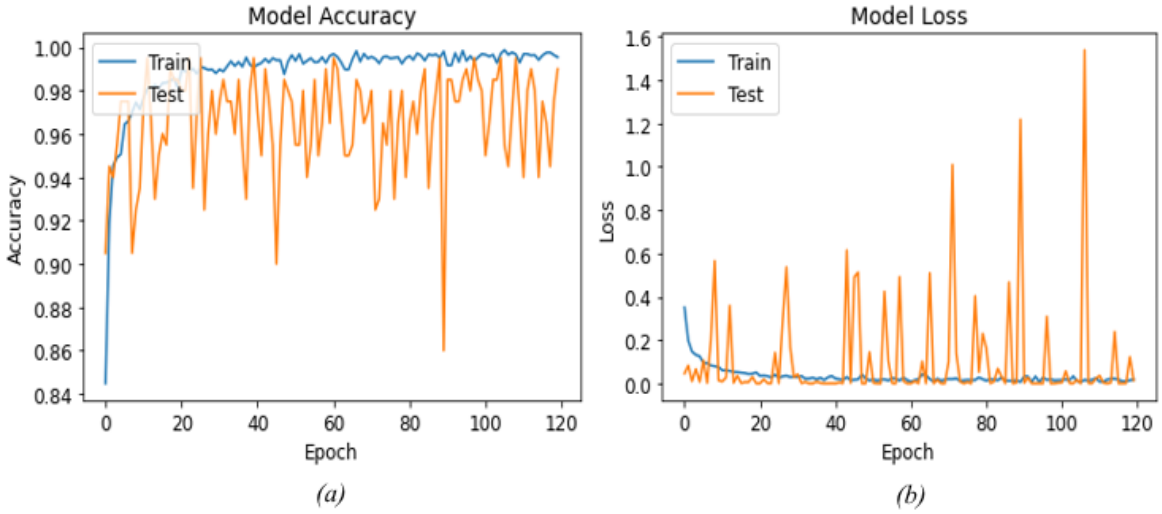


Figure 9. Accuracy and loss plot for ***Model 2.1***

Figure 10. (a) TPR curve, (b) FPR curve, (c) Accuracy curve, and (d) Log loss curve

In the context of fusion based techniques, CHD + HOG produces 95.30% accuracy with the GB classifier, and HOG + LBP descriptor delivers 95.66% accuracy with the same classifier. On the other hand, LBP + 2DWT with RF classifier and 2DWT + GLCM with RF classifier yield 94.60% and 94.15% accuracy, respectively. The practical consequences have shown that fusion-based techniques produce outstanding results than single descriptor-based methods.

Figure 10 illustrates the various curves built over these machine learning based techniques. Figure 10(a) shows the TPR curve that provides the truthiness or probability of truth alarm for these techniques. In contrast, Figure 10(b) exemplifies the FPR curve that offers the possibility of false alarm for these techniques. A high TPR value is desirable, whereas a low value of FPR is quite preferable for building a proficient model. In the second technique, the HOG with Quadratic Discriminant Analysis (QDA) classifier achieves a higher TPR value and moderate value of FPR, which costs to compromise with accuracy. In the same techniques, HOG with KNN classifier and HOG with SVC-RBF (Radial Basis Function) classifier provide the lower value of FPR and moderate TPR value, which also cost to compromise with accuracy. Therefore, the intention is quite clear that there should be a relation between TPR and FPR to get higher accuracy.

Finally, in the ***first technique***, the CHD with RF classifier produces higher TPR and lower FPR values, which improves the model's accuracy. On the other side, HOG with Logistic Regression classifier from second technique, LBP with Random Forest classifier from the third technique, 2-DWT with Random Forest classifier from fourth technique, GLCM with Random Forest classifier from fifth technique, CHD + HOG with Gradient Boosting classifier from sixth technique, HOG + LBP with Gradient Boosting classifier from seventh technique, LBP + 2-DWT with Random Forest classifier from eight technique, and 2-DWT + GLCM with Random Forest classifier from ninth technique has achieved preferable TPR and FPR values and produced higher accuracy. Overall, in all machine learning based object detection techniques, HOG + LBP with Gradient Boosting Classifier provides higher accuracy than others, as shown in Figure 10(c). Figure 10(d) displays the Log loss curve for these four techniques, which helps measure the uncertainty of prediction build on how much it differs from the actual one. The value of log loss increases when approaching the higher prediction value and decreases when approaching the lower prediction value.

The ***tenth technique*** uses deep learning based approach and introduces two models with different variants, namely ***Model 1*** and ***Model 2***. Both models produce outstanding results in terms of accuracy, such as 98% for ***Model 1.5*** and 99% for ***Model 2.1***.

The intrinsic comparisons among deep learning Models are illustrated in Figure 11. The first part of the same figure shows the trade-off between training accuracy and validation accuracy, and another part shows the trade-off between training loss and validation loss. The training accuracy for all of these models is quite well, but these models' performance is evaluated from validation accuracy. Here, ***Model 2.1*** secures its first position in terms of accuracy compared to others, while ***Model 2.3***, ***Model 1.1***, and ***Model 1.5*** also provide pleasing training and validation accuracy results. On observing the Model loss from Figure 9, it is found that all the Models except ***Model 1.4*** are trained very well because their training loss is approximately near to validation loss. However, ***Model 1.4*** and ***Model 1.6*** might be overfitted because their validation loss is much lesser than training loss.

Table XII presents the comparative analysis of different techniques to spot the presence of humans. Among all methods, the best frameworks are chosen out and compared with each other. In comparison, it is found that CNN based technique ***(Model 2.1)*** outperforms others and provides promising accuracy of up to 99%. Deep Neural architectures spend much time on training and require extensive computational resources, while machine learning-based approaches take less time in training and require less computational resources. These computational resources might be CPU, RAM, Disk, etc. On analyzing all the presented techniques, it is observed that although the CNN based approaches took a longer time to train, they produced accurate results compared to single or combined machine learning-based approaches.

5. CONCLUSION

This article incorporates the review of various learning based techniques for human detection. These learning based techniques include different machine learning and deep learning based object detectors, such as CHD + Classifier (technique 1), HOG + Classifier (technique 2), LBP + Classifier (technique 3), 2-DWT + Classifier (technique 4), GLCM + Classifier (technique 5), CHD + HOG + Classifier (technique 6), HOG + LBP + Classifier (technique 7), LBP + 2-DWT + Classifier (technique 8), 2-DWT + GLCM + Classifier (technique 9) and CNN based object detector (technique 10). From machine learning based human detectors, HOG + LBP with Gradient Boosting classifier has outperformed the others with an accuracy of up to 95.663%. In experiments, we have also found that machine learning based human detectors require less time in training than deep learning based detectors. The obtained results prove that CNN based Model 2.1, thanks to its efficiency, has achieved the highest results up to 99% accuracy among all described techniques, with improved adaptability and detection effects.

In the end, this article addresses a discussion to brief the future work required to enhance the human detection process in surveillance video. These involve adopting a multi-view approach and utilizing an advanced model based

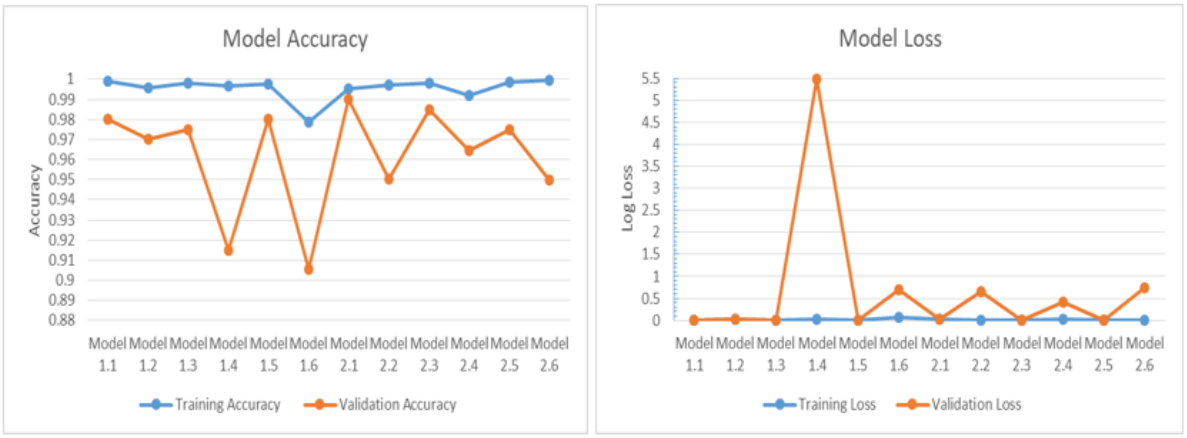


Figure 11. (a) Model accuracy curve and (b) Model loss curve comparison among proposed models

TABLE XII. Performance Comparisons for Human Detection Methods

Methods	Accuracy	Feature Extraction Time (*In Sec.*)	Feature Loading Time (*In Sec.*)	Training Time
CHD + Random Forest	94.425	151.695	1.99	0.240 Sec.
HOG + Logistic Regression Classifier	93.097	227.226	17. 685	4.293 *Sec.*
LBP + Random Forest Classifier	93.274	190.042	2.259	0.453 *Sec.*
DWT + Random Forest Classifier	92.389	248.887	7.215	0.484 *Sec.*
GLCM + Random Forest Classifier	86.460	172.661	2.954	0.321 *Sec.*
CHD + HOG + GBC	95.309	321.853	98.156	304.026 *Sec.*
HOG + LBP + Gradient Boosting Classifier	95.663	335.902	121.256	483.820 *Sec.*
LBP + 2DWT + Random Forest Classifier	94.601	289.769	27.981	0.358 *Sec.*
2DWT + GLCM + Random Forest Classifier	94.159	602.649	6.324	0.719 *Sec.*
CNN Based (Model 2.1)	99.000		-	35.52 *Min.*

on localized portions of an image. Besides, different modern object detectors like Faster RCNN, YOLO, RFCN, SSD, etc. may be incorporated with the synthesized dataset to achieve even better recognition effects.

References

[1] Y. Zhang, D. Zhu, P. Wang, G. Zhang, and H. Leung, "Vision-based vehicle detection for videosar surveillance using low-rank plus sparse three-term decomposition," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 4711–4726, 2020.

[2] D. Chahyati, M. I. Fanany, and A. M. Arymurthy, "Tracking people by detection using cnn features," *Procedia Computer Science*, vol. 124, pp. 167–172, 2017.

[3] E. S. Marquez, J. S. Hare, and M. Niranjan, "Deep cascade learning," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 11, pp. 5475–5485, 2018.

[4] D. K. Singh and D. S. Kushwaha, "Tracking movements of humans in a real-time surveillance scene," in *Proceedings of fifth international conference on soft computing for problem solving*. Springer, 2016, pp. 491–500.

[5] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.

[6] D. K. Singh, S. Paroothi, M. K. Rusia, and M. A. Ansari, "Human crowd detection for city wide surveillance," *Procedia Computer Science*, vol. 171, pp. 350–359, 2020.

[7] S. N. Endah, R. Kusumaningrum, and H. A. Wibawa, "Color space to detect skin image: the procedure and implication," *Scientific Journal of Informatics*, vol. 4, no. 2, pp. 143–149, 2017.

[8] V. Gajjar, A. Gurnani, and Y. Khandhediya, "Human detection and tracking for video surveillance: A cognitive science approach," in *Proceedings of the IEEE international conference on computer vision workshops*, 2017, pp. 2805–2809.

[9] M. A. Ansari and M. Dixit, "An image retrieval framework: A review." *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[10] R. Matsumura and A. Hanazawa, "Human detection using color contrast-based histograms of oriented gradients," *International Journal of Innovative Computing, Information and Control*, vol. 15, no. 4, pp. 1211–1222, 2019.

[11] A. Ahmed, J. Guo, F. Ali, F. Deeba, and A. Ahmed, "Lbph based improved face recognition at low resolution," in *2018 international conference on Artificial Intelligence and big data (ICAIBD)*. IEEE, 2018, pp. 144–147.

[12] M. A. Ansari and M. Dixit, "An enhanced cbir using hsv quantization, discrete wavelet transform and edge histogram descriptor," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 2017, pp. 1136–1141.

[13] Ş. Öztürk and B. Akdemir, "Application of feature extraction and classification methods for histopathological image using glcm, lbp, lbglcm, glrlm and sfta," *Procedia computer science*, vol. 132, pp. 40–46, 2018.

[14] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE access*, vol. 7, pp. 128 837–128 868, 2019.

[15] D. Wang, C. Li, S. Wen, Q.-L. Han, S. Nepal, X. Zhang, and Y. Xiang, "Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples," *IEEE Transactions on Cybernetics*, 2021.

[16] B. Kim, N. Yuvaraj, K. S. Preethaa, R. Santhosh, and A. Sabari, "Enhanced pedestrian detection using optimized deep convolution neural network for smart building surveillance," *Soft Computing*, vol. 24, no. 22, pp. 17 081–17 092, 2020.

[17] K. Abughalieh and S. Alawneh, "Pedestrian orientation estimation using cnn and depth camera," Tech. Rep., 2020.

[18] N. A. Ibraheem, M. M. Hasan, R. Z. Khan, and P. K. Mishra, "Understanding color models: a review," *ARPN Journal of science and technology*, vol. 2, no. 3, pp. 265–275, 2012.

[19] N. Dwivedi, D. K. Singh, and D. S. Kushwaha, "An approach for unattended object detection through contour formation using background subtraction," *Procedia Computer Science*, vol. 171, pp. 1979–1988, 2020.

[20] S. S. Sengar and S. Mukhopadhyay, "Foreground detection via background subtraction and improved three-frame differencing," *Arabian Journal for Science and Engineering*, vol. 42, no. 8, pp. 3621–3633, 2017.

[21] J. Guo, J. Wang, R. Bai, Y. Zhang, and Y. Li, "A new moving object detection method based on frame-difference and background subtraction," in *IOP conference series: materials science and engineering*, vol. 242, no. 1. IOP Publishing, 2017, p. 012115.

[22] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149)*, vol. 2. IEEE, 1999, pp. 246–252.

[23] L. Li and J. Xu, "Moving human detection algorithm based on gaussian mixture model," in *Proceedings of the 29th Chinese Control Conference*. IEEE, 2010, pp. 2853–2856.

[24] P. Karpagavalli and A. Ramprasad, "An adaptive hybrid gmm for multiple human detection in crowd scenario," *Multimedia Tools and Applications*, vol. 76, no. 12, pp. 14 129–14 149, 2017.

[25] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[26] Y. Zhang, J. Zheng, C. Zhang, and B. Li, "An effective motion object detection method using optical flow estimation under a moving camera," *Journal of Visual Communication and Image Representation*, vol. 55, pp. 215–228, 2018.

[27] R. Jayaswal, J. Jha, M. Dixit, and M. MITS, "Mining of images by k-medoid clustering using content based descriptors."

[28] K. Bhuvaneswari and H. A. Rauf, "Edgelet based human detection and tracking by combined segmentation and soft decision," in *2009 International Conference on Control, Automation, Communication and Energy Conservation*. IEEE, 2009, pp. 1–6.

[29] F.-C. Hsu, J. Gubbi, and M. Palaniswami, "Human head detection using histograms of oriented optical flow in low quality videos with occlusion," in *2013, 7th International Conference on Signal Processing and Communication Systems (ICSPCS)*. IEEE, 2013, pp. 1–6.

[30] V. Seib, G. Schmidt, M. Kusenbach, and D. Paulus, "Fourier features for person detection in depth data," in *International Conference on Computer Analysis of Images and Patterns*. Springer, 2015, pp. 824–836.

[31] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1. Ieee, 2001, pp. I–I.

[32] 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)*, vol. 1. Ieee, 2005, pp. 886–893.

[33] V. Gaikwad and S. Lokhande, "Vision based pedestrian detection for advanced driver assistance," *Procedia Computer Science*, vol. 46, pp. 321–328, 2015.

[34] I. Riaz, J. Piao, and H. Shin, "Human detection by using centrist features for thermal images," in *International Conference Computer Graphics, Visualization, Computer Vision and Image Processing*. Citeseer, 2013.

[35] H. Bahri, M. Chouchene, F. E. Sayadi, and M. Atri, "Real-time moving human detection using hog and fourier descriptor based on cuda implementation," *Journal of Real-Time Image Processing*, vol. 17, no. 6, pp. 1841–1856, 2020.

[36] P. Gangal, V. Satpute, K. Kulat, and A. Keskar, "Object detection and tracking using 2d—dwt and variance method," in *2014 Students Conference on Engineering and Systems*. IEEE, 2014, pp. 1–6.

[37] N. Najva and K. E. Bijoy, "Sift and tensor based object detection

and classification in videos using deep neural networks," *Procedia Computer Science*, vol. 93, pp. 351–358, 2016.

[38] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International journal of computer vision*, vol. 128, no. 2, pp. 261–318, 2020.

[39] D. Tome, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, and S. Tubaro, "Deep convoluted neural netwoks for pedestrian detection," *Preprint. Elsevier Journal of Signal Processing: Image Communication*, 2015.

[40] M. A. Ansari and D. K. Singh, "Monitoring social distancing through human detection for preventing/reducing covid spread," *International Journal of Information Technology*, vol. 13, no. 3, pp. 1255–1264, 2021.

[41] R. K. McConnell, "United states patent," Patent 4,567,610, 1986.

[42] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Proceedings of 12th international conference on pattern recognition*, vol. 1. IEEE, 1994, pp. 582–585.

[43] M. P. Yadav, N. Pal, and D. K. Yadav, "Workload prediction over cloud server using time series data," in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. IEEE, 2021, pp. 267–272.

[44] J. Kumari, R. Venkatesan, T. J. Jebaseeli, V. A. Felsit, K. S. Selvanayaki, and T. J. Sarah, "A comparison of machine learning techniques for the prediction of the student's academic performance," in *International Conference on Emerging Current Trends in Computing and Expert Technology*. Springer, 2019, pp. 1052–1062.

[45] A. Aggarwal, T. Sahay, A. Bansal, and M. Chandra, "Grid search analysis of nu-svc for text-dependent speaker-identification," in *2015 Annual IEEE India Conference (INDICON)*. IEEE, 2015, pp. 1–5.

[46] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, and R. Yang, "Salient object detection in the deep learning era: An in-depth survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[47] U. Ojha, U. Adhikari, and D. K. Singh, "Image annotation using deep learning: A review," in *2017 International Conference on Intelligent Computing and Control (I2C2)*. IEEE, 2017, pp. 1–5.

[48] M. A. Ansari and D. K. Singh, "Review of deep learning techniques for object detection and classification," in *International Conference on Communication, Networks and Computing*. Springer, 2018, pp. 422–431.

[49] I. P. Dataset, "http://pascal.inrialpes.fr/data/human/," Tech. Rep., 2018.

Mr. Mohd. Aquib Ansari received the B.E. degree in Information Technology from SATI, Vidisha, in 2014. He has done his M.Tech. in Information Technology from MITS, Gwalior, in 2017. He is currently pursuing his Ph.D. from MNNIT Allahabad, Prayagraj. He is an IEEE student member. He has published more than 17 research papers in various International Journals and Conferences. His research interests include Computer Vision & Image Processing, AI and Machine Learning. He can be contacted at mansari@mnnit.ac.in.

Dr. Dushyant Kumar Singh received the B.Tech. and M.Tech. degree in computer science and engineering from AMU Aligarh, India in 2007 and 2010. He received the Ph.D. degree from MNNIT Allahabad, Prayagraj, India in 2017. Since 2012, he is working as Assistant Professor in CSED, MNNIT Allahabad, Prayagraj, India. He has published more than 50 research papers in various International Journals and Conferences. His research interests include Computer Vision, Image Processing, AI and Machine Learning for Vision, Embedded System Design. He can be contacted at dushyant@mnnit.ac.in.