International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

An Efficient Deep Learning Based Hyperbolic Back Propagate Boltzmann Neural Network for Automated Vehicular Surveillance

N.Meenakshi[1], I.Juvanna*[2], P.Anbhazhagan[3],T.Shanmuganathan[4]

Submitted: 11/11/2022 **Accepted:** 13/02/2023

Abstract: In recent years, there has been a significant expansion in the infrastructure for video surveillance, which has resulted in an increase in the number of intelligent surveillance systems that use computer vision and pattern recognition algorithms. In this article, we offer a unique intelligent surveillance system that is based on deep learning and is utilized for the management of road vehicles. In other words, the system is able to label any vehicle via the use of computer vision, and as a result, it is able to quickly distinguish cars that have visual tags. This capability allows the system to extract the vehicle visual tags that are present on urban roadways. The visual tags that are discussed in this article include the license plate number, the color of the car, and the kind of vehicle. These visual tags also contain a variety of other attributes, such as passing location and passing time. In this work the particular area video footage was retrieved. Then the formation of clustered hub was done for the fusion of the several footage data . The violation based vehicle details are grouped by using the Gaussian mac clustering approach. Then from the grouped features the specialized features according to the required event was isolated using the Metaheuristic chaos vortex optimization. Finally the proposed hyperbolic back propagate boltzmann neural network (HBPBNN) architecture detects the vehicle and its related violations in a precise manner. The whole experimentation was carried out in a real time database. On-road experimental findings show that the algorithm outperforms the most cutting-edge vehicle recognition algorithm in testing data sets. The findings of the comparative assessment indicated that the recommended model performed better than the other models in use.

Keywords: *Automated Vehicular Surveillance , Deep Learning, Gaussian mac clustering approach, Metaheuristic chaos vortex optimization, Hyperbolic back propagate Boltzmann neural network*

1. Introduction

Congestion, accidents, and violations all present serious problems for traffic management systems in most major and medium-sized cities due to the increasing urban population. As a result, there has been a lot of interest in studies of active traffic surveillance, which seek to keep tabs on and control vehicular traffic. The development of computer vision has made the video camera a cost-effective and promising sensor for monitoring traffic. In the last 30 years, video surveillance systems have become an integral aspect of ITS (ITSs). Vehicle detection, tracking, identification, behavioral analysis, and so on are just some of the ways these systems gather visual data about moving vehicles. For the most part, current surveillance systems gather data on traffic flows, which consists mostly of traffic metrics and incident detection.

[1] *SRM Institute of Science and Technology, Kattankulathur - 603203*
ORCID ID : https://orcid.org/0000-0003-1994-5461
[2] *Hindustan Institute of Technology and Science, chennai*
ORCID ID : https://orcid.org/0000-0003-0289-9454
[3] *Kebri Dehar University ,Ethiopia*
ORCID ID : https://orcid.org/0000-0002-9672-6396
[4] *Hindustan Institute of Technology and Science, chennai*
ORCID ID : https://orcid.org/0000-0001-5896-7255
** Corresponding Author Email: ijuvanna@hindustanuniv.ac.in*

It is more difficult to discover traffic incidents, but this area offers a lot of room for exploration. Video-based traffic surveillance has made significant strides, but there are still a number of issues that need to be overcome before it can be used in an ITS setting. Here is a rundown of some of the problems that have been encountered so far with video surveillance systems.

• Daytime and evening illumination conditions vary significantly. Although nighttime operations are possible with the use of supplementary illumination equipment, these devices often have restricted viewing ranges. In high-traffic situations, pedestrians, bicyclists, trees, and buildings may readily obscure a car's view, and the same is true for other cars. When cars change lanes or turn, their poses might be quite different from one another.

• Vehicles come in all sizes, colors, and forms imaginable. The picture size in pixels varies when a vehicle moves within the camera's field of vision (FOV). As a result, certain finer points of visual information are lost, and the reliability of detection models is put to the test.

To better comprehend the roadway behavior of vehicles and drivers: When following a vehicle as it moves through a road system, cameras must work together to get an understanding of the whole traffic state and the status of the driver via global behavior analysis. In order to better

understand how vehicles behave, we describe a generic system architecture of hierarchical and networked vehicle surveillance in ITSs based on an examination of current surveillance systems. There are four distinct levels that make up the hierarchy. Obtaining Data at the First Network Layer and the Emergence of Hub Clusters . This layer's job is to use visual sensors to collect information about traffic situations in a given region, organize that information, and finally obtain pictures. The Layer 2 Attributes use a Gaussian mac clustering technique to group up the attributes in the pictures based on their dynamic and static features. Dynamic qualities are those that pertain to the features of a moving vehicle, such as its speed, direction of travel, path taken along the road system, etc. License plate number, vehicle type, color, logo, driver status, and other such fixed qualities are used to describe the outward look of a vehicle. Metaheuristic chaos vortex optimization is used in the third layer to extract unique details about the desired event based on the attributes already retrieved in the previous layers. The Interpretation of Behavior is done at the Fourth Layer. Here, we use a Hyperbolic back propagate Boltzmann neural network to do some attribute analysis, learn about the aggressive driving habits of cars and drivers, and get a sense of how the transportation system is now congested. This rest of the paper is structured as follows. In the second section, we take a look at the current level of research in the fields of vehicle object identification and vehicle surveillance. In Part 3, we see an example of the stated issue. Layers of implementation for the recommended technique are explored in Section 4, and Section 5 presents the results of the experimental assessment. At last, this paper's last section provides closure.

2. Related Works

In [1], deep learning is used to solve the problems of vehicle identification and fine-grained categorization. THS-10 is a locally produced dataset with strong intraclass and low interclass variance, making it ideal for performing fine-grained classification and the associated complexity. There are a total of 4,250 car photos in the collection, including 10 different makes and models (Honda City, Honda Civic, Suzuki Alto, Suzuki Bolan, Suzuki Cultus, Suzuki Mehran, Suzuki Ravi, Suzuki Swift, Suzuki Wagon R, and Toyota Corolla). Using the AU-AIR dataset, the authors of [2] describe innovative methods for traffic monitoring and surveillance based on state-of-the-art and widely used DL object identification models (Faster-RCNN, SSD, YOLOv3, and YOLOv4). Due to the extreme inequity present in this data set, an additional 500 photos were harvested by web-mining methods. Twofold is the originality of this work's contribution. Firstly, this article provides a rigorous scholarly explanation for why ground-view photos are not suitable for aerial object recognition. Furthermore, the efficacy of these algorithms has been examined by conducting a regress comparison. On the Analytics Vidhya Emergency Vehicle dataset, [3] examined the results of eight different CNN architectures. The goal of [4] is to provide a Decision-Tree enabled technique driven by Deep Learning for extracting anomalies from traffic cameras while precisely predicting the beginning and end times of the anomalous occurrence. Anomaly detection and analysis were part of their methodology when a detection model was developed. Our detection model was built on top of YOLOv5. [5] In their study, they offer a novel method for categorizing vehicles in order to shed insight on the difficulties associated with sorting unbalanced data. The datasets are first compiled from two sources: the MIOvision Traffic Camera Dataset and the Beijing Institute of Technology Vehicle Dataset. In order to improve the quality of the gathered vehicle photos and to recognize cars from the denoised images, adaptive histogram equalization and the Gaussian mixture model are employed. The feature vectors are then extracted from the identified cars using the Steerable Pyramid Transform and the Weber Local Descriptor. In the end, an ensemble deep learning approach is used to classify vehicles based on the retrieved data. To identify automobiles in traffic cam footage, [6] used Generative Adversarial Nets (GANs). The three-stage process for vehicle categorization that was created is as follows. In order to create adversarial samples for the uncommon classes, first GAN was trained on a gathered traffic dataset. After training an ensemble-based Convolutional Neural Network (CNN) on the skewed dataset, the second step is to perform sample selection to get rid of the adversarial examples of poorer quality. Finally, the enhanced dataset was used to fine-tune the ensemble model using the chosen adversarial samples. Extensive trials revealed that the proposed GAN method performed well in vehicle classification on MIO-TCD as measured by the Cohen kappa score, mean recall, precision, and mean precision. However, as the deeper networks are ready to converge, degradation difficulties would arise in the created GAN technique. A novel method for classifying vehicles based on a hierarchical multi-SVM (multi-Support Vector Machine) classifier was proposed in [7]. First, they built a hierarchical multi-SVM method for vehicle classification after extracting foreground objects from surveillance movies. In addition, the categorized cars' performance was monitored using a voting-based rectification strategy. This research aimed to construct a viable system for robust vehicle classification in a high-volume traffic scene using the hierarchical multi-SVM approach. Since varied perspectives, shadows, and significant occlusion make the created approach useless in real-world busy traffic situations, the development was abandoned. In addition, [8] used the SVM classifier with the small YOLO to identify and classify vehicles. In the experimental phase, the created model's accuracy and recall were verified using the BIT Vehicle Dataset. The experimental results show that the built model accurately categorizes the vehicles in traffic videos that are being streamed in real-time. A significant drawback of this research was that SVM was a binary classifier, which only allows for binary categorization. In [9], the quicker R-CNN technology was used to create a system for classifying different kinds of vehicles. The created approach was tested using photographs of the actual traffic intersection taken in real time. To better recognize a vehicle that is partially obscured by changing lighting, camera orientation, and picture size, a revolutionary approach will be required in the future. A convolutional neural network (CNN) model for vehicle categorization was created in [10], and it features precise tuning and pretraining. To get the base model with connection weights, we pretrained it using GoogLeNet on the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) dataset. An initial model was constructed, and then fine-tuned on the vehicle dataset to get an accurate classification. Van, minibus, truck, bus, automobile, and motorbike are only some of the six vehicle types

included in the gathered highway surveillance footage used in this research review. It was during the experimental phase that the vehicle dataset was analyzed for performance in terms of accuracy. Overfitting is a key issue with the established CNN model, and it also requires a lot of computing power. There are six essential steps in [11]'s novel framework for vehicle type classification: data preparation, vehicle detection, vehicle tracking, structure matching, feature extraction, and vehicle classification. We used noise reduction and color conversion to prepare the collected traffic surveillance footage for analysis. Moreover, the cars were picked out thanks to the employment of the Otsu thresholding technique and background reduction. The Kalman filter was then used to keep track of the moving cars. The feature vectors were extracted using the log Gabor filter and the Harrish corner detector, and then the information was input into an ANFIS for vehicle classification. Extensive experimental evidence demonstrated that the established framework significantly improved upon previous attempts at vehicle categorization in terms of mistake rate and accuracy. Dimensionality, a cause of the complexity of the model, is made worse by the established framework. As of [12], a novel semisupervised CNN architecture has been introduced for vehicle type categorization. Sparse Laplacian filtering is used in the designed architecture to extract detailed and unique vehicle information. The multitask learning input layer softmax classifier was trained to distinguish between different kinds of vehicles. The characteristics learnt by the semisupervised CNN architecture were shown to be sufficiently discriminatory in this literature review's challenging scene environments. To test the effectiveness of the designed architecture in terms of classification accuracy, extensive experiments were performed on the BIT Vehicle Dataset and a publicly available dataset. Since the semisupervised CNN architecture has many more layers, training takes more time. On the BIT Vehicle Dataset and the MIO-TCD, [13] proposed a novel densely linked single-split super learner and used versions for vehicle type categorization. The established model was straightforward; it did not need the use of complex reasoning or specially constructed features to improve on the existing state of the art in vehicle type categorization. The suggested model brings the vanishing gradient issue, which is a prominent worry in this literature review, to the complex datasets. Using a Principal Component Analysis Convolutional Network, [14] created a novel semisupervised model for vehicle type classification (PCN). Convolutional filters were used in the created model to extract the discriminative and hierarchical features. The proposed model outperformed the baseline in simulations because it is resistant to noise contaminations, lighting conditions, rotation, and translation, all of which might affect performance in real-world settings. A bigger number of training parameters are included in the created PCN model, which causes overfitting. Sparse-Filtered Convolutional Neural Networks with Layer Skipping (SF-CNNLS) was designed for automobile classifiers by [15]. Using the SF-CNNLS methodology, which has three channels, this research aimed to extract discriminant and rich vehicle information. Also, information about the cars' color, brightness, and shape was collected from the three channels of each image to provide a complete picture of the vehicles' appearance. The created SF-CNNLS approach's performance was verified using a reference dataset, as detailed in the Experimental Results and Discussion. Truck, minivan, bus, passenger, taxi, automobile, and SUV classes were ultimately categorized using a softmax regression classifier. However, there may be a loss of vehicle type information due to the embedding of lower-resolution vehicle pictures in the created softmax regression classifier's deeper layers. For the purpose of classifying small, lightweight vehicles, [16] created a deep CNN method. In this paper, the authors use specificity, precision, and -score to verify the generated model's performance on a real-world dataset. Despite its potential, the constructed model fell short in some conditions, including those with a known baseline, camera jitter classes, and adverse weather. Each published study describes in detail the approaches used, the datasets used, the benefits and drawbacks of applying the established algorithms to the task of vehicle type classification, and so on. This study proposes a novel ensemble deep learning method to enhance vehicle type categorization, which should help with the aforementioned problems.

3. Problem Statement

According to the findings of a search of the relevant literature, there are two primary issues connected with the currently available deep learning-based algorithms for vehicle surveillance. Specifically, these issues are as follows: The first problem is the amount of time and computer processing that is required to cope with their enormous quantity of matrix products (due to their multilayer structures), as well as their sampling and second order optimization approaches, such as conjugate gradient. This is the primary worry. The second issue concerns the selection of the most suitable network topology, that is, choosing the optimum number of layers for the network as well as the number of neurons that should be placed in each of those levels. In this study, a unique approach to deep learning was devised in order to solve the problems that have previously been identified.

4. Proposed Work

Vehicle surveillance has recently been an important study topic in the field of intelligent traffic systems. This is owing to the diverse range of applications that vehicle surveillance might have, including the monitoring of traffic flow statistics and individual vehicles. There have been many different ways established that use vehicle type categorization. These systems are often based on cameras, magnetic induction, or optic fibers. Because of the widespread implementation of traffic surveillance cameras, the image-based techniques have garnered a lot of interest in the computer vision field. Figure 1 presents the method of optimization and deep learning associated flow diagram.

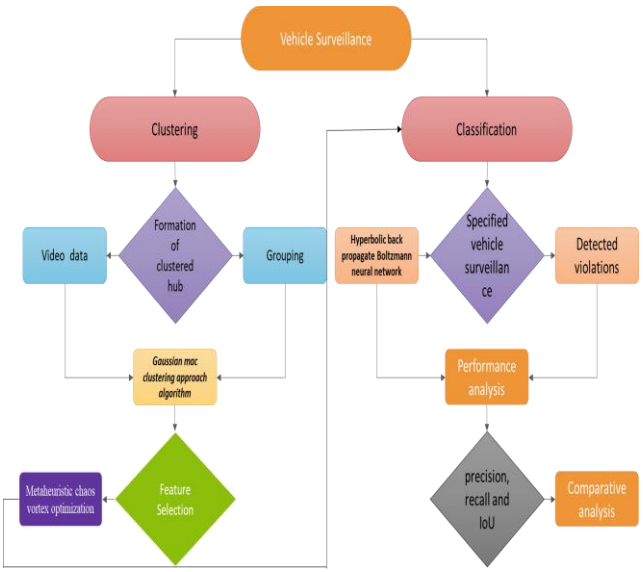

Fig. 1 Schematic representation of the suggested methodology

4.1 Dataset

The footage that is obtained by a surveillance camera that has been mounted on a xxxx is what is gathered for the purpose of data gathering. There are three movies total, and each one was filmed on a different day, resulting in different amounts of traffic and illumination. The road has a total of five lanes, and the traffic density is exceptionally high, with each car traveling at a speed that is somewhere in the middle. As may be seen in Figure 1, the front and somewhat side views of the vehicles that are present in the scenario.

4.2 Data clustering and feature grouping

Most variables have limited range, therefore the finite moments assumption seems sense. Take into account the possibility that each individual video is a random occurrence. The of the independent variables in a model can be predicted with full confidence in most cases, with the exception of seasonal and temporal trend data. This is the true regardless of whether or not the many, unrelated video data sources pertain to events that are fundamentally asynchronous. This means that the various video sources must be linked in order to provide efficient monitoring. So, let's go back to our generic model and pretend we don't have any data on P_{it} and Q_{it}. Only data from the $aggregates^1$.

$$Q_t = \sum_{i=1}^{M} Q_{it}, P_t = \sum_{i=1}^{M} P_{it}, \quad (1)$$

To solve this issue, we need to provide a meaningful connection between P_{it} and Q_{it}. The joint distribution of $P_{1t}, \ldots., P_{Mt}, B_{1t}, \ldots., B_{Mt}, V_{1t}, \ldots., V_{Mt}{}^5$ implies a conditional density ofP_{it}, B_{it} ,andV_{it}given P_t , denoted by$e_{it}(P_{it}, B_{it}, V_{it}|P_t)$. As a result, the definition of the macro-function is,

$$F(P_t|Q_t) = \sum_{i=1}^{M} F[L_i(P_{it}, B_{it}, V_{it})|P_t]$$
$$= \sum_{i=1}^{M} \int_{S_{it}} L_i(P_{it}, B_{it}, V_{it}) e_{it}(P_{it}, B_{it}, V_{it}|P_t) cP_{it} cB_{it} cV_{it}$$
$$= \sum_{i=1}^{M} \Gamma_{it}(P_t) = \Gamma_t(P_t), \quad (2)$$

where S_{it}, represents the appropriate integrated area

$$\Gamma_{it}(P_t) = \int_{S_{it}} L_i(P_{it}, B_{it}, V_{it}) e_{it}(P_{it}, B_{it}, V_{it}|P_t) cP_{it} cB_{it} cV_{it}$$

After the data has been integrated, the GMCA algorithm is used to categorize the characteristics into groups. The goal is to divide a dataset of training samples, $\{y_m\}_{m=1}^{M}$, into L categories, where m is the dimension of the dataset. We learn a latent feature $w \in \mathbb{S}^{N\times 1}$ for each training sample y. Our working hypothesis is that the latent characteristics have a Gaussian mixed distribution. To specify which Gaussian component contains the latent feature w, we add the binary vector $d\epsilon\{0,1\}^{l\times 1}$. Specifically, we assume a Gaussian mixture distribution for our data in this model. In particular, we model the generation process for a sample y as follows.

$$t(d;\pi) = \prod_{l=1}^{L} \pi_l^{d_l} \quad (3)$$
$$t(w|d_l = 1) = \mathcal{M}(\mu_l, diag(\sigma_l^2))$$
$$t\theta(y|w) = \begin{cases} Ber(\mu_y) & if\ y\ is\ binary \\ \mathcal{M}(\mu_y, \lambda J) & y\ is\ real-valued \end{cases} \quad (4)$$

If we designate the lth element of d_l and π_l and the lth entry of as d and π, then we can say that π_l satisfies $\sum_{l=1}^{L} \pi_l = 1, \mu_l$and σ_l^2 . Here, $\mu_y = h(w_m;\theta)$, J, J is the identity matrix, is a fixed parameter, and h is a network whose parameters may be adjusted by training. Maximum a posterior (MAP) of latent variables and maximum likelihood estimate (MLE) of parameters are challenging to discover by directly solving the generative model. To solve this problem, we introduce a new distribution $t\theta(w,d|y)$, selected from a narrow class and parametrized by a learnable parameter φ, that approximates the posterior distribution $v\phi(w,d|y)$. In particular, we suppose that $v\phi(w,d|y)$may be written as $v\phi(w,d|y) = v_{\phi 1}(w,d|y)v_{\phi 2}(d|w)$. We then provide a definition for,

$$v_{\phi 1}(w|y) = \mathcal{M}\left(\tilde{\mu}, \mathrm{diag}(\tilde{\sigma}^2)\right) \quad (5)$$
$$v_{\phi 2}(d|w) = \mathrm{Multinominal}\ (\tilde{\pi}) \quad (6)$$

Where

$$[\tilde{\mu}, log(\tilde{\sigma}^2)] = e_1(y;\phi_1)$$

$$\tilde{\pi} = e_2(w;\phi_2)$$

Note that e_1 and e_2 here represent neural networks with parameters ϕ_1 and ϕ_2, respectively. This model utilizes a pair of neural networks. The first, denoted e_1, uses the training samples to infer the latent distributions, and the second, denoted e_2, determines the likelihood that the latent features are distributed along a given Gaussian component. The parameters can be estimated within the generative model and inference model framework by means of maximizing the log-likelihood function, i.e.

$$\max_{\phi,\theta} \sum_{i=1}^{M} in\, t\theta(y_i) \quad (7)$$

After this we will see that the log-likelihood function can accommodate the function $H(\phi,\theta,y_i,y_j)$. A training sample y_i has a log-likelihood $\ln t\theta(y_i)$that may be broken down into its component parts.

$$In\, t\theta(y_i) = LK(v\phi(w,d|y_i)||t\theta(w,d|y_i)) + F_{q\phi(w,d|y_i)}\left[In\frac{t\theta(y_i,w,d)}{v_\phi(w,d|y_i)}\right] \quad (8)$$

Remarkably, if we substitute,

$v_\phi(w,d|y_i)$ with $v_\phi(w,d|y_j)$

$$In\, t\theta(y_i) = LK(v\phi(w,d|y_i)||t\theta(w,d|y_i)) + F_{q\phi(w,d|y_i)}\left[In\frac{t\theta(y_i,w,d)}{v_\phi(w,d|y_j)}\right] \quad (9)$$

"Averaging the equation (10) and (11), we have"

$$In\, t\theta(y_i) = H(\phi,\theta,y_i,y_j) + \frac{1}{2}(K(\theta,\phi;y_i) + KK(\theta,\phi;y_i,y_j)) \quad (10)$$

Where

$$K(\theta,\phi;y_i) = F_{v\phi(w,d|y_i)}\left[In\frac{t\theta(y_i,w,d)}{v_\phi(w,d|y_j)}\right]$$

$$K(\theta,\phi;y_i,y_j) = F_{v\phi(w,d|y_i)}\left[In\frac{t\theta(y_i,w,d)}{v_\phi(w,d|y_j)}\right]$$

Remember the restriction that $\sum_j z_{ij} = 1$, we can rewrite the objective function as

$$\max_{\phi,\theta} \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M} z_{ij}\,(K(\theta,\phi;y_i) + K(\theta,\phi;y_i,y_j)) \quad (11)$$

Similarly, $K(\theta,\phi;y_i,y_j)$ can be rewritten as

$$K(\theta,\phi;y_i,y_j) = F_{v\phi(w|y_i)v\phi(d|w)}\left[In\frac{t\theta(y_i|w)t(w|d)t(d)}{v_\phi(w|y_j)v_\phi(d|w)}\right] \quad (12)$$

and ultimately assessed by, $K(\theta,\phi;y_i,y_j) \approx$

$$\sum_{c=1}^{C} y_c^i log\mu_{y_j}|c + (1-y_c^i)\log\left(1-\mu_{y_j}\middle|c\right) - \sum_{l=1}^{L}\gamma il\sum_{n=1}^{N}(log\,\sigma_l^2|n + \frac{\tilde{\sigma}_j^2|n}{\sigma_l^2|n} + \frac{(\tilde{\mu}_j|n-\mu_l|n)^2}{\sigma_l^2|n}) + \sum_{l=1}^{L}\gamma jl\, log\frac{\pi_{il}}{\gamma_{jl}} + \frac{1}{2}\sum_{n=1}^{N}(1 + log\,\tilde{\sigma}_j^2|n) \quad (13)$$

Where

$$\mu_{y_j} = h(w_j;\theta)$$

$$w_j = \tilde{\mu}_j + \tilde{\sigma}_j o\epsilon$$

For instance, the element of the affinity matrix with the Gaussian kernel is defined as

$$z_{ij} = \begin{cases} \frac{1}{b_i}\exp\left(-\frac{\|y_i - y_j\|_2^2}{2r_i^2}\right) & if\, y_j \epsilon M(y_i) \\ 0, & otherwise \end{cases} \quad (14)$$

where b_i is a normalizer that sets $\sum_z z_{ij} = 1$ and $r_i$is a predefined scalar. $M(y_i)$ denotes the set consisting of the nearest associated features of y_i.

The pixel characteristics may finally be categorized.

4.3 Feature selection

All the formulae from chaos theory and the vortex search optimization technique are brought together in the suggested model. The objective is to identify the most advantageous aspects of the automobile occurrence. The following phrases describe these locations:

- Firstly, let us declare the generation of potential answers inside the search region.
- For the second state, if the answer is too far beyond the acceptable range, these transformations may be applied to bring it inside the bounds.
- Third, the MCVO method uses the retrieved characteristics from the previous step to narrow the search area using the reverse gamma function and other specialized features. The chaos map is used to brainstorm potential answers in this approach. Using a Gaussian chaotic distribution and a vortex search method, a number of neighbor solutions D p (r), where p specifies the iteration index and is p=0 in the beginning stages, were constructed arbitrarily close to the initial center _0 in C-dimensional space.. Here, D_0 (s) = { $r_1, r_2, .. r_n$} n = 1, 2, 3, …, the solutions are denoted by m, and the total number of possible solutions is also denoted by m. As shown in Eq. (15), the suggested technique is expressed as a formula.

$$t(y|\mu) = \frac{1}{\sqrt{2\pi^c}}\exp\left\{-\frac{1}{2}(dn-\mu)^T \sum^{-1}(dn-\mu)\right\} \quad (15)$$

where c stands for the number of dimensions, dn is the c × 1 vector of a specialized variable, μ is the c1 vector of the sample mean (center), and is the covariance matrix. These mappings are used to bring the solution within the required range if it is found to be outside. The present circle's center, μ_0, is swapped out with a solution (the best one), rϵD_0(r), which is chosen and remembered during the selection step. It is important to verify that the candidate solutions fall inside the search parameters before moving on to the selection step. This is achieved by relocating the solutions outside the limits within them, as in Eq (15). Eq. describes the MCVO when chaotic sequences are included (15). Eq. (15) represents the value of the chaotic map at the i-th iteration, denoted by Dn(i).

$$r_l^i = \begin{cases} Dn_i * \left(upperlimit^i_{lowerlimit^i}\right) + lowerlimit^i, r_l^i \\ < lowerlimit^i \\ 1 \\ lowerlimit^i \le r_l^i < upperlimit^i \\ Dn_i * \left(upperlimit^i_{lowerlimit^i}\right) + lowerlimit^i, r_l^i \\ < lowerlimit^i \end{cases} \quad (16)$$

The inverse gamma function and DNs were used to narrow the search area. With each iteration of the MCVO, the radius

is shrunk using the inverse incomplete gamma function. Eq. gives the incomplete gamma function (17).

$$\gamma(y.a) = \int_0^{dn} f^{-1}p^{b-1}cpb > 0 \quad (17)$$

"where b>0 is known as the shape parameter and dn ≥ 0 is a specialized variable".

. In this work, we use the chaos function to evaluate the appropriateness of the chosen characteristics. It was via trial and error that the L and distance strategy was chosen. The number of features, K_m, is the sum total, and the weighting factor takes on values between zero and one. The value of may be adjusted to modify the relative weights of characteristics used for categorization and the features actually chosen. The weight factor β is often set to values close to 1 since increasing accuracy is the main aim of any classifier. The value 0.8 was used in this study. The optimal strategy is one that simultaneously optimizes for classification accuracy and feature selection complexity.

$$\text{Fit} = \text{maximize}\left(a + \beta \times \left(1 - \frac{K_m}{K_r}\right)\right) \quad (18)$$

Atlast specialized features are isolated.

4.4 Classification

In this case, the recommended model was utilized to determine whether or not the car was under surveillance. The proposed HBPBNN has nodes with three levels, including an input layer, hidden layer(s), and an output layer.

"Where,

$Y = Y_i]_{i=1}^{M}$, i^{th}= no. of input vector, where $Y_i = \{Y_1, Y_2 \ldots Y_m\}$ is i^{th} the input with m no. of feature Repeat that for each epoch"

The activation function is applied to the computed link weights between the input layer nodes and the hidden layer nodes, which take the input data and represent the connection between them as weights. The activation function uses the weighted sum plus a bias to determine whether or not a certain node should be activated.

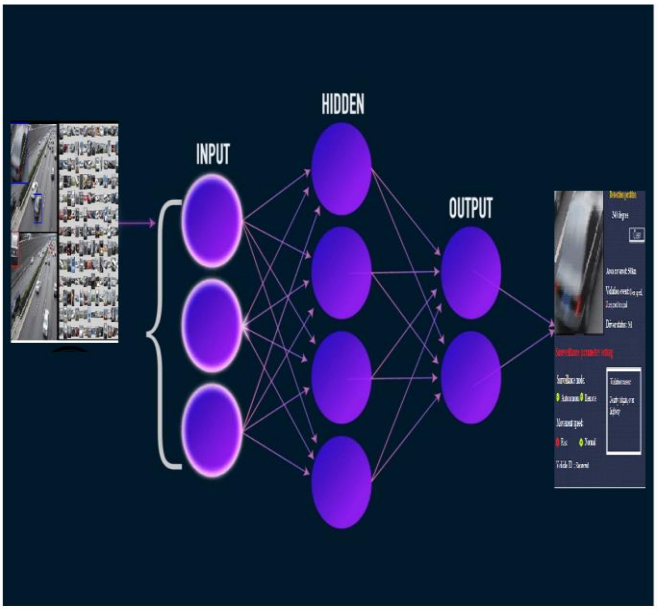

Fig.2 Architecture of the suggested network

"The net input to the neuron w^1 at layer k = 1 is computed as in Equation (19) by using weight Z^1, input y^1 and bias a^1. The net output B^1 has been computed by using Relu activation function on w^1".

$$W^1 = Z^1 \times Y + a^1 \quad (19)$$

$$B^1 = relu(W^1) \quad (20)$$

Forward propagation in HBPBNN involves layer 1's net output B^1 being computed and then passed on to layer k in the form of Equation (19) and Equation (20).

$$B^{k-1} = relu(W^{k-1}) \quad (21)$$

$$W^k = Z^k \times B^{k-1} + a^1 \quad (22)$$

The status of output was demonstrated as follows,

$$o^k = B^k = \sigma(W^k) \quad (23)$$

The HBPBNN learns by minimizing the loss gained (Equation (24)) from the network at the output$o^k = B^k = \sigma(W^k$. Figure 2 shows how the HBPBNN's parameters are tuned by backpropagation learning, where the gradient of the loss function κ with respect to the parameters (Equation (24)) is determined. Equations (21), (22), and (23) may be used to calculate the gradient of with regard to the values of Z^k, a^k,and B^{k-1}, respectively (where P is transpose matrix).

$$\kappa = -\frac{1}{n}\sum_{i=1}^{n}(x^i \log(o^i) + (1 - x^i)\log(1 - o^i)) \quad (24)$$

$$\Delta Z^k = \frac{\partial \kappa}{\partial Z^k} = \frac{1}{n}\partial W^k B^{k-1P} \quad (25)$$

$$\Delta a^k = \frac{\partial \kappa}{\partial a^k} = \frac{1}{n}\sum_{i=1}^{n}\partial W^{k,i} \quad (26)$$

$$\Delta B^{k-1} = \frac{\partial \kappa}{\partial B^{k-1}} = Z^{kp}\partial W^{k,i}$$

"Then the parameters W^k , a^k , and B^{k-1} are updated from the obtained gradients ΔZ^k , Δa^k , and

ΔB^{k-1} as presented in Equations (24), (25), and (26), respectively"

$$W^k = W^k - \eta \times \Delta Z^k \quad (27)$$

$$a^k = a^k - \eta \times \Delta a^k \quad (28)$$

$$B^{k-1} = B^{k-1} - \eta \times \Delta B^{k-1} \quad (29)$$

The suggested networks use a nonlinear activation function (Relu) to calculate the next layer's output from the weighted inputs based on the output and bias of the layer below it. In deep learning models, loss functions represent the cost function to be minimized. For multilabel classification, the traditional loss function is the categorical cross-entropy. The neural network's approximation of a yes or no answer is determined using activation functions. Depending on the range of the activation function, it then maps the succeeding characteristics. The sigmoid activation function is used to make the model's final prediction. Dropout of nodes is used to prevent overfitting. Within the context of dropout, randomly selected neurons are ignored during a training phase. The person is "dropped out" of the program without any explanation. That's because the weights update isn't applied to the neuron during reversible propagation, and it also represents the fact that the forward propagation is temporally passive with respect to the activation function of downstream neurons. This allows the network to train more rapidly, have less overfitting, and provide better predictions using deep learning. The normality of the nodes in the network structure makes them more resistant to the inputs, and the dropout of nodes has an effect on the simulation of many networks with diverse network structures.

Algorithm: HBPBNN

"**Input:** Size of feature space, training set, size of feature subspace, feature set, number of feature subspace, one test sample, and number of classes.

Output: Classification of vehicle surveillance data. Process:

For y = $\{Y_1, Y_2 \ldots Y_m\}$: classes Label the samples of i th class.

Train the feature subsets using HBPBNN .

Train {70%) Test {30%)

Neuron {code input}

Neural size{Number of nodes}

Ranking feature set utilizing

$W^1 = Z^1 \times Y + a^1 \leftarrow w^1$'s last ranked featur

Begin

Relu{

$$B^1 = relu(W^1)$$

}

For

output B^1

forward propagation$B^{k-1} = relu(W^{k-1})$

Else

$$W^k = Z^k \times B^{k-1} + a^1$$

Out {

$$o^k = B^k = \sigma(W^k)$$

}

Gradient comput{Z^k, a^k,and B^{k-1}}

$$\Delta Z^k = \frac{\partial \kappa}{\partial Z^k} = \frac{1}{n} \partial W^k B^{k-1P}$$

$$\Delta a^k = \frac{\partial \kappa}{\partial a^k} = \frac{1}{n} \sum_{i=1}^{n} \partial W^{k,i}$$

$$\Delta B^{k-1} = \frac{\partial \kappa}{\partial B^{k-1}} = Z^{kp} \partial W^{k,i}$$

For

Updation;

$$W^k = W^k - \eta \times \Delta Z^k$$

$$a^k = a^k - \eta \times \Delta a^k$$
$$B^{k-1} = B^{k-1} - \eta \times \Delta B^{k-1}$$

End for
End

Calculate the value of counter Output.
Return
End
End"

---

5. Performance Analysis

An Intel Core i3-3370 computer with 8 GB of RAM and a 3.60 GHz clock speed was used for the test. Python, which is free and widely used, was chosen to develop the method because of its extensive collection of pre-existing modules. The suggested work is implemented in Python 3.7 with the help of the CV2 packages for computer vision. Images and movies may be prepared for viewing and edited afterwards with the help of the tools available in the CV2 library. The little resources required to complete the process contribute to its low cost.

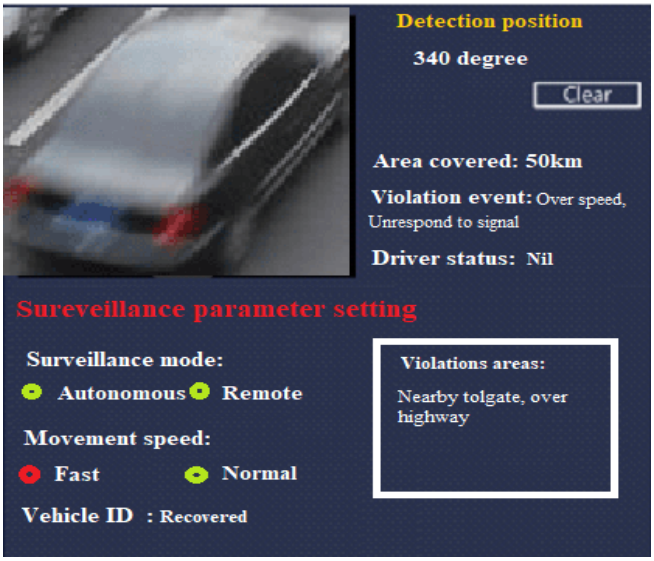

Fig.3. Clustered input and vehicle selection

The sample input and the process of the video data grouping along with the vehicle selection was done as depicted in figure 3.Here from the retrieved video the single hub was developed for fusing the data as a whole. Then the vehicle shows abnormal behavior was selected for further steps of surveillance.

Fig.4 . Simulated output

The simulated output and the status of the vehicle violence behavior was demonstrated in figure 4.Here the selected vehicle seems to be violating the traffic rules.

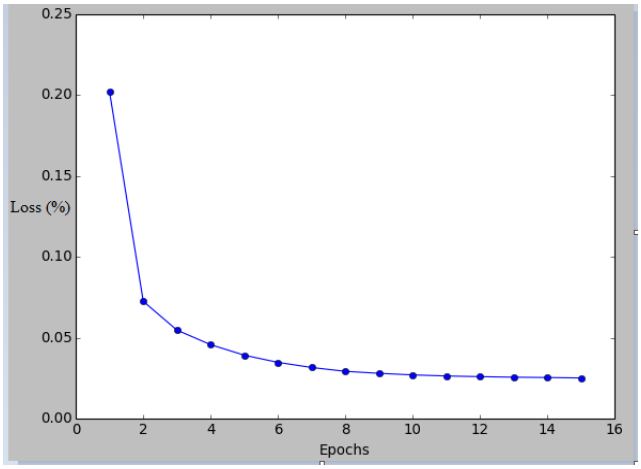

Fig.5. Epochs Vs. Loss

The loss function, which is created from all of the data in an epoch, is a quantitative measure of loss for the whole epoch. When building an iterative curve, some information must always be lost. The resulting curve demonstrates that, in comparison to other approaches, classifier training, validation, and testing losses are relatively low. If there is a little difference between the training loss and the validation loss, our model is probably underfitted. By expanding the size, the training loss might be minimized (either the number of layers or the raw number of neurons in each layer). The information utilized to estimate the loss is shown in Figure 5. The suggested technique outperforms previous methods because it experiences far less severe levels of level loss.

.

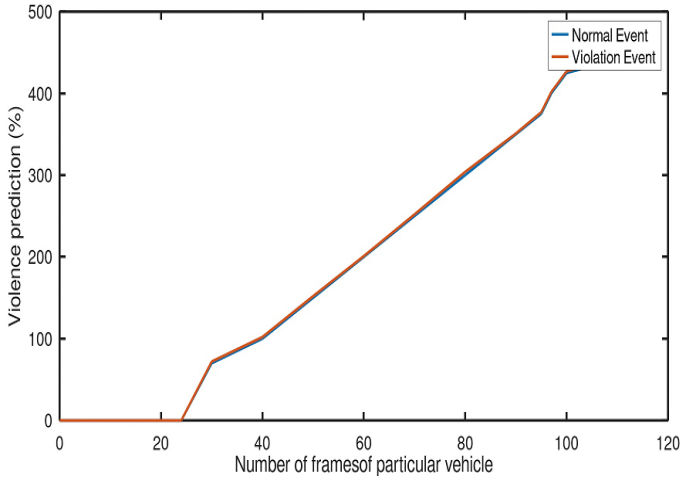

Fig.6. Violence rate prediction

According to the data shown in figure 6, the vehicle that was chosen exhibits a marginal breach of the regulations governing traffic.

In order to demonstrate that the proposed method is effective, it may be contrasted with the many mechanisms that are already in place [5]. The suggested system's accuracy, precision, and recall are evaluated in order to determine how well it functions.

Accuracy

It calculates the percentage of vehicles with an accurate violent status classification. It determines the degree to which the findings correspond to the real ground truth results.

ACCU =(TP+TN)/(TP+TN+FP+FN) (30)

Precision

By removing aberrant vehicle status from the dataset, it assesses how accurate the behavior of the proposed approach is and uses that information.

Precision= TP/(TP+FP) (31)

Recall

The ratio of correctly predicted instances and all instances.

Recall= TP/(TP+FN) (32)

It establishes the degree to which the actual results coincide with the predictions that were developed. The total number of anticipated positives and projected negatives should be multiplied by the sum of all the actual positives and negatives, and then that number should be divided by the total. In contrast, the accuracy of the strategy that was recommended is 99.5%, which is higher than the accuracy of the processes that are being employed at the moment (see Figure 7)

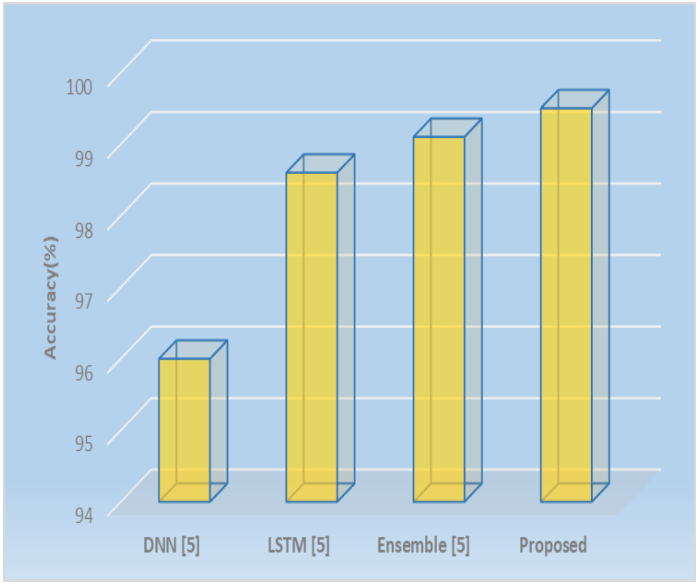

Fig.7. Accuracy rate analysis

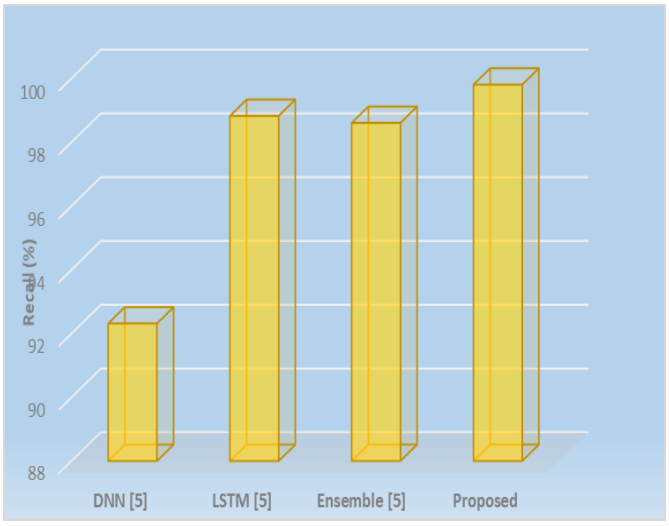

Fig.8. Recall rate analysis

The extent to which a classifier is able to properly identify all aberrant surveillance data that does not have a problem with their condition is referred to as their recall. In this particular scenario, as seen in figure 8, the proposed processes had a vast range of recall, which reached an impressive 99.8 percent. This percentage was fairly high in compared to other mechanisms that were previously in place

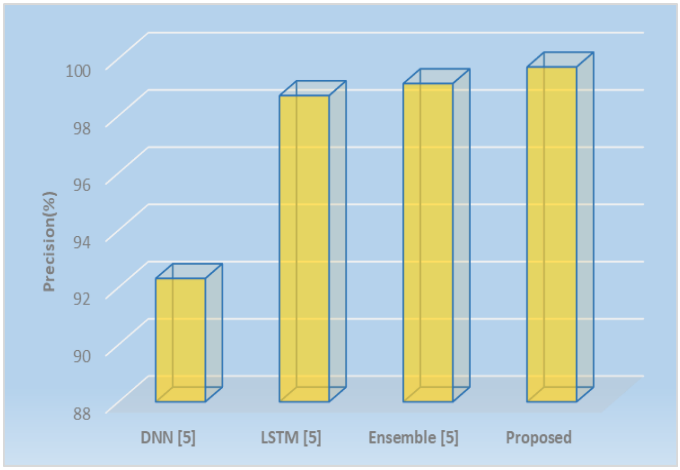

Fig.9. Precision rate analysis

As of from the figure 9 the suggested RSDDTC technique have the high rate of precision (99.7%) over the vehicle surveillance is very high that that of the other existing mechanisms.

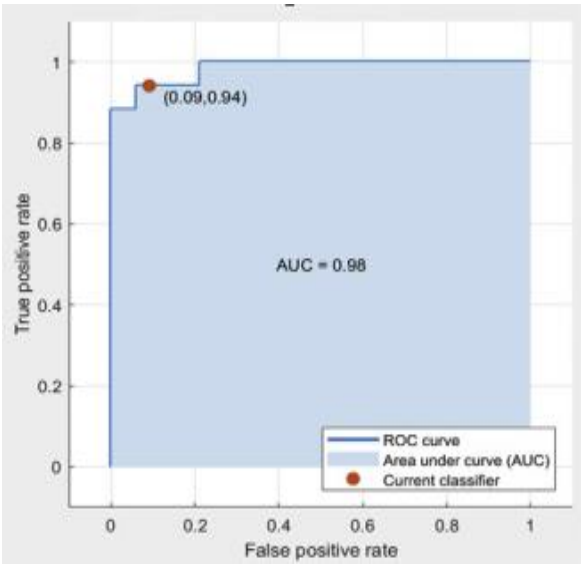

Fig.10. AUROC analysis

The plan outlined in that section takes into account the global performance measures shown in Figure 10. A value of 0.98 is shown for AUROC in Figure 10. The area under the curve (AUROC) score may be used to demonstrate that the classifier did a good job of correctly identifying the violation event. Each point on the ROC curve represents a pair of sensitivity and specificity values, which may be used to define cutoffs for decision-making purposes. ROC curves are often employed in medical diagnostics. The area under the ROC curve is a measurement that may be used to determine how well a parameter can differentiate between different types of activity..

Table1 Comparative analysis for False detection rate (FDR) with false acceptance rate (FOR)

"Feature extraction"	**"Classifier"**	**"FDR(%)"**	**"FOR (%)"**
	"MSVM"	22	18.03
	"KNN"	12	17
"SPT"	"DNN"	9.72	10.74
	"LSTM"	8	3.20
	"Ensemble"	6	1.29
	"MSVM"	11.91	9.5
	"KNN"	6.03	5.93
"WLD"	"DNN"	3	3.07
	"LSTM"	2.09	1.02
	"Ensemble"	1.72	0.86
	"MSVM"	4	3.29
	"KNN"	1.20	1.95
"Hybrid (SPT + WLD)"	"DNN"	0.98	0.83
	"LSTM"	0.79	0.35
	"Ensemble"	0.44	0.32
"MCVO"(Proposed)	"HBPBNN (proposed)"	0.2	0.2

Table 2 Performance metrics comparative analysis

	"Dataset"	"Precision (%)"	"Recall (%)"	"Accuracy (%)"
"GAN-based deep ensemble technique"	"MIO-TCD"	96.41	—	—
"Tiny YOLO with SVM"	"BIT Vehicle"	97.90	99.60	—
"Semisupervised CNN model"	"BIT Vehicle"	—	—	88.11
"PCN with softmax classifier	"BIT Vehicle"	—	—	88.52
"TC-SF-CNNLS	"BIT Vehicle"	90.52	90.41	93.80
	"MIO-TCD"	99.12	99.69	99.13
"Ensemble deep learning technique"	"BIT Vehicle"	98.24	99.72	99.28
	"Combined"	99.27	99.77	99.32
"HBPBNN (proposed)"	"Real time"	99.7	99.8	99.5

According to an analysis of Table 1 and 2, the optimization and deep learning methodology combined to provide a mini- mum FDR of 0.2% and a FOR of 0.2%, which are efficient in comparison to other classification methodologies. 1000 vehicle images are used for testing, and 7000 vehicle images are converted from 5 security cameras and used in the real-time vehicle dataset. On the BIT Vehicle Dataset, the proposed ensemble deep learning approach is compared graphically in terms of FDR and FOR. The suggested ensemble deep learning approach on the Vehicle Dataset also requires 1.2 seconds to execute for each frame. Additionally, the simulation results showed that the created model had a classification accuracy of 99.5%. When compared to other currently used methods, the created methodology performs very well throughout the testing phase in terms of recall, precision, and accuracy.

6. Conclusion

An approach based on deep learning, which has previously been used largely in traffic surveillance systems, is suggested to be applied for vehicle surveillance in this article. At this point in time, video monitoring is being used for a variety of extra purposes all over the globe during the COVID-19 pandemic. Our software makes use of a deep learning technique, which can be broken down into a number of distinct stages, including feature extraction and categorization. Within the scope of this study, MCVO feature descriptors are used to extract active feature vectors in order to cut down on training time, enhance classification accuracy, and lessen the likelihood of overfitting issues within the context of the deep learning methodology. In this investigation, the deep learning methodology is used to categorize various cars according to their features. When compared to other classification methods, the performance of the deep learning methodology in terms of precision, recall, accuracy, FDR, and FOR was superior to that of the other classification methods. This was shown in the Experimental Results and Discussion section. The suggested method exhibited an increase in classification accuracy of no more than 10 percentage points when compared to the benchmark approaches that are already in use. The suggested method for enhancing vehicle type identification and classification will involve, in further work, the use of a segmentation algorithm that is based on clustering. In addition to this, a good intelligent transportation system places a strong focus on three-dimensional modeling, vehicle tracking, and occlusion handling.

Appendix

Appendixes, if needed, appear before the acknowledgment.

Acknowledgment

.

References

[1] S. A. Najeeb, R. H. Raza, A. Yusuf, and Z. Sultan, "Fine-grained vehicle classification in urban traffic scenes using deep learning," in *Proceedings of the 11th International Conference on Robotics, Vision, Signal Processing and Power Applications*, 2022, pp. 902-908.

[2] H. Gupta and O. P. Verma, "Monitoring and surveillance of urban road traffic using low altitude drone images: a deep learning approach," *Multimedia Tools and Applications*, vol. 81, pp. 19683-19703, 2022.

[3] A. Kherraki and R. El Ouazzani, "Deep convolutional neural networks architecture for an efficient emergency vehicle classification in real-time traffic monitoring," *IAES International Journal of Artificial Intelligence*, vol. 11, p. 110, 2022.

[4] A. Aboah, "A vision-based system for traffic anomaly detection using deep learning and decision trees," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4207-4212.

[5] P. Jagannathan, S. Rajkumar, J. Frnda, P. B. Divakarachari, and P. Subramani, "Moving vehicle detection and classification using gaussian mixture model and ensemble deep learning technique," *Wireless Communications and Mobile Computing*, vol. 2021, 2021.

[6] W. Liu, Z. Luo, and S. Li, "Improving deep ensemble vehicle classification by using selected adversarial samples," *Knowledge-Based Systems*, vol. 160, pp. 167-175, 2018.

[7] H. Fu, H. Ma, Y. Liu, and D. Lu, "A vehicle classification system based on hierarchical multi-SVMs in crowded traffic scenes," *Neurocomputing*, vol. 211, pp. 182-190, 2016.

[8] A. Şentaş, İ. Tashiev, F. Küçükayvaz, S. Kul, S. Eken, A. Sayar, *et al.*, "Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification," *Evolutionary Intelligence*, vol. 13, pp. 83-91, 2020.

[9] X. Wang, W. Zhang, X. Wu, L. Xiao, Y. Qian, and Z. Fang, "Real-time vehicle type classification with deep convolutional neural networks," *Journal of Real-Time Image Processing*, vol. 16, pp. 5-14, 2019.

[10] L. Zhuo, L. Jiang, Z. Zhu, J. Li, J. Zhang, and H. Long, "Vehicle classification for large-scale traffic surveillance videos using convolutional neural networks," *Machine Vision and Applications*, vol. 28, pp. 793-802, 2017.

[11] V. Murugan and V. Vijaykumar, "Automatic moving vehicle detection and classification based on artificial neural fuzzy inference system," *Wireless Personal Communications*, vol. 100, pp. 745-766, 2018.

[12] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE transactions on intelligent transportation systems*, vol. 16, pp. 2247-2256, 2015.

[13] M. A. Hedeya, A. H. Eid, and R. F. Abdel-Kader, "A super-learner ensemble of deep networks for vehicle-type classification," *IEEE Access*, vol. 8, pp. 98266-98280, 2020.

[14] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, "Semisupervised PCA convolutional network for vehicle type classification," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 8267-8277, 2020.

[15] S. Awang, N. M. A. N. Azmi, and M. A. Rahman, "Vehicle type classification using an enhanced sparse-filtered convolutional neural network with layer-skipping strategy," *IEEE Access*, vol. 8, pp. 14265-14277, 2020.

[16] N. Nasaruddin, K. Muchtar, and A. Afdhal, "A lightweight moving vehicle classification system through attention-based method and deep learning," *IEEE Access*, vol. 7, pp. 157564-157573, 2019.