A Vehicle Classification System for Intelligent Transport System using Machine Learning in Constrained Environment

Ahmed S. Alghamdi[1], Talha Imran[2], Khalid T. Mursi[3], Atika Ejaz[4], Muhammad Kamran[5]*, Abdullah Alamri[6]
Department of Cyber Security, College of Computer Science and Engineering,
University of Jeddah, Jeddah, 21959, Saudi Arabia[1, 3, 5]
Department of Computer Science, COMSATS University Islamabad, Wah Cantt, Wah Cantt, 47010, Pakistan[2, 4]
College of Computer Science and Engineering, University of Jeddah, Jeddah, 21959, Saudi Arabia[6]

***Abstract*—Vehicle type classification has an extensive variety of applications which include intelligent parking systems, traffic flow-statistics, toll collecting system, vehicle access control, congestion management, security system and many more. These applications are designed for reliable and secure transportation. Vehicle classification is one of the major challenges of these applications particularly in a constrained environment. The constrained environment in the real world put a limit on data quality due to noise, poor lightning condition, low resolution images and bad weather conditions. In this research, we build a more practical and more robust vehicle type classification system for real world constrained environment with promising results and got a validation accuracy of 90.85 and a testing accuracy of 87%. To this end, we design a framework of vehicle type classification from vehicle images by using machine learning. We investigate the deep learning method Convolutional neural network (CNN), a specific type of neural networks. CNN is biologically inspired with multi-layer feed forward neural networks. It can learn automatically at several stages of invariant features for the particular chore. For evaluation, we also compared the performance of our model with the performance of other machine learning algorithms like Naïve Bayes, SVM and Decision Trees.**

Keywords—Vehicle classification; intelligent transport system; deep learning; machine learning; CNN; digital image processing

I. INTRODUCTION

Vehicle classification and detection has emerged as an integral part of the Intelligent Transportation System (ITS). The aim of ITS is to provide public safety, improve travel and transit information, minimize blocking, produce cost funds to tragedies operators. To cope with the rising demand of surface transportation system, the ITS technologies assist cities, states and towns nationwide. The effectiveness of an IT system is generally based on the performance of vehicle detection and classification technology. It collects all or part of the information which is used in an efficient ITS. In the development of ITS, traffic monitoring is one of the significant applications encompassing the categorization and detection of vehicles. Vehicle type classification has an extensive variety of applications which include intelligent parking systems, traffic flow statistics, toll collecting system, vehicle access control, congestion management, security system and many more. These applications are designed for reliable and secure transportation. Vehicle classification is one of the major challenges of these applications. Many researchers from different domains have been involved in this area to overwhelm the primary transportation issues.

In recent past, the research in vehicle type classification and detection from static images has achieved huge attention because of its essential role in an enormous variety of applications [1], [2]. Vehicle type classification and detection from still images is a demanding job. Because of their great intraclass distinctions, numerous vehicle types fitting into similar classes have features of several shapes and sizes. Furthermore, shadow, lighting, noise, and obstruction make the classification job more demanding [1]. Moreover, there can be some constraints on image quality while working with limited computing resources. These types of complications can be determined by using efficient and reliable vehicle classification systems for such constrained environment.

Several image-based techniques have been presented until now and the majority of them comes under into two groups: appearance-based methods and model-based methods. To classify vehicle type, appearance-based method uses appearance features (for example SIFT [3], sobel edges [4]) from either vehicle side view or front view image for classifying vehicle type. Model-based methods [5], [6], [7], [8] retrieve the 3D parameter of the vehicles just like height, length, and width for classification. To define the vehicle shape and structure 3D models are used. In 3D models, the main features to deal with are faces, point, lines and topological structures. The vehicle classification is generally performed using a combination of hardware and software-based approaches. The hardware used for this purpose mainly includes radars, magnetic detectors, infrared detectors, ultrasonic detectors, acceleration detectors and many more. But this capable hardware cannot be used without having automated vehicle classification software embedded in their backend. The focus of all the previous works as well as this work is also to develop such a capable software that when embedded in any capable hardware, can provide accurate vehicle classification results. The core Idea behind the development of automated vehicle classification software's is automated intelligence. This also goes the other way around. If a capable software gets developed that can accurately classify

*Corresponding Author, email id- mkkamran@uj.edu.sa vehicle classes, it will not be able to do anything until it is embedded into a capable hardware to provide real time monitoring, surveillance, and other activities.

Nowadays, many real-world applications like auto controlling traffic are totally dependent on vehicle classification and detection. Several methods have been proposed for vehicle classification. Vehicle classification is very complicated area in computer vision because it requires high accuracy which may lead towards complex algorithm. Until now various works have considered for this classification, some of the work has been done on the limited scenarios. There are some challenges which must overcome for the accomplishment of this task like image quality, size, image scale and color [9], [10]. Moreover, existing research works on vehicle image classification using deep learning have made significant progress in achieving high accuracy and robustness. However, they also exhibit certain limitations that need to be addressed. One limitation is the lack of diverse and comprehensive datasets, which may lead to biased or overfitting models. Additionally, the focus on specific regions or vehicle types in some studies limits the generalizability of the proposed methods. Another limitation is the heavy reliance on manual annotation and labeling, which can be time-consuming, subjective, and prone to errors. Furthermore, the computational complexity and resource requirements of deep learning models pose challenges for real-time applications and resource-constrained environments. The interpretability and explain ability of deep learning models in the context of vehicle image classification also remain areas of concern. Addressing these limitations will contribute to the advancement of vehicle image classification research and enable more accurate and efficient solutions in real-world scenarios.

The main motivation behind our proposed work is that in real world scenarios, due to bad weather conditions and moving of the camera and/or vehicles, the images are sometimes not clear enough to distinguish between types of vehicles automatically. We believe that the need to address this problem is very important; particularly, in case of lack of high-end cameras that could provide a good quality image. Also, there can also be other factors like more than one vehicle in a single image and various angles of the vehicle when the image was captured. Furthermore, image rotation also presents a challenge for accurate classification of vehicles.

The vehicle images captured by camera are blurry (or not clear) if the shutter speed while capturing the images is not according to the vehicle speed (or stationary vehicle and moving camera). The image clearness is usually measured by its sharpness. A clear image has a high degree of sharpness; while on the other hand, a blurred image is less sharp. In our work we are concerned with motion-blurriness caused by the motion of vehicle and/or camera. So, a robust method would yield accurate results on blurred images as well. Deep learning based CNNs have revolutionized image processing and computer vision. They excel in extracting meaningful features directly from raw data, eliminating the need for manual feature engineering. CNNs leverage their hierarchical and localized approach to capture complex patterns and structures in images. They are adept at tasks like object recognition, image classification, and segmentation. Deep learning models handle large-scale datasets, ensuring scalability and generalization across different domains. With their ability to automatically adapt and learn from image data, deep learning and CNNs have significantly advanced applications such as autonomous driving, medical imaging, and video analysis. In this context, we propose a more practical and robust vehicle classification system with a promising accuracy rate. We recommend a structure of vehicle classification from vehicle images by using a machine learning system. Convolutional neural networks are the machine learning methods. They are specific neural networks. CNN is biologically inspired by that multi-layer feed forward neural network. It is capable to automatically learn several phases of invariant features for the chore. CNN has good ability of learning features in face detection [11], image classification [12], video classification [13], facial point detection [14], and human attribute inference [15]. CNN have verified great success in these kinds of tasks. For large scale image recognition task it has become the most popular architecture. Convolutional neural networks have a layered structure and each layer could learn the demonstration of the input. This capability is called feature learning. Feature learning is a method through which machine learns from raw data. In image classification, the raw data of the image is represented by an array of pixels. These arrays of pixels fed to the CNN and then CNN learns the useful features of the image to resolve the machine learning problem [16]. The major contributions of the work presented in this paper are as follows:

- Development of the algorithm proposed for automatic vehicle type classification in constrained environment.
- Modeling the vehicle classification system as a problem to be solved by using CNN algorithm.
- Performance analysis of the proposed technique using experiments.

The rest of this paper is organized as follows. Section II presents the current relevant techniques with their limitations. A detailed explanation of the proposed technique is presented in Section III. Details about experimental setup and results has been given in Section IV. Future directions have been provided in Section V and finally, this paper has been concluded in Section VI.

II. Related Work

Zhen dong et al. in [17] worked on the vehicle classification by using a semi supervised CNN. They used the front view of vehicle images. As an input, the network is given the vehicle image and then it outputs the vehicle class to which the vehicle belongs. Their technique achieves overall classification accuracy of 96.1% in the daylight images and 89.4% accuracy on the nighttime images. Similarly, in [18], a multi type vehicle classification system is established. For this purpose, they used Bag of Visual Words (BOVWs) and Random Neural Networks (RNNs). Their BOVW-RNN classification system achieves 95.63% accuracy which is higher than the LIVCS who achieve 91.21% accuracy. Another approach proposed in Yiren zhou et al. [19] uses Deep Neural Network (DNN) approach for the vehicle detection and classification problem. Their approach has the capability to be used for training on restricted size datasets and can be extended to diverse lighting condition cases. However, as opposed to the proposed work these techniques do not consider the images taken under the constraint environment. For vehicle classification, the authors in [20] used an immovable camera to detect and classify the vehicles. They located the camera on the height of the road surface. They used an algorithm which has two phases: In the first stage they obtained mobile vehicles in the traffic location, removed background from the image and performed morphology operations and edge detection. In the second stage, they captured the vehicle through camera and processed the feature extraction method. The classification rate for their proposed method is 90% of the whole data. In [21], Wei Wu et al. used a novel method by using neural networks and parameterized models for vehicle classification. They captured their own real vehicle images through CCD camera installed on toll collecting station. They took the vertices and their topological structure as the main feature. To recognize vehicle, they used a classifier which is based on the multi-layer perception network (MLPN). In this classifier they used a learning technique which is based on the gradient descent for the least exponential function error (LEFE). They achieved approximately 91% accuracy. The authors in [22]] proposed the appearance-based model for vehicle type classification. They used front view of the vehicle and SoftMax regression for classification. They used convolutional neural network and by using this network their method automatically learned good features for classification. Their method is relatively effective for the classification of vehicle types. For automatic vehicle classification, the method in [23] is based on the fusion classical neural network (CNN) and of fast neural network (FNN). They used FNN as the main classifier and CNN as a final classifier. Their proposed method produced 95.83% accuracy. However, all these techniques do not account for images taken under the constrained environment. The author addresses the limitations of single vehicle detection methods due to a lack of high-quality labeled training samples. They propose using the YOLO-v5 architecture for vehicle detection and classification, employing transfer learning by fine-tuning pre-trained weights. Extensive datasets collected by the authors, including various traffic patterns, occlusions, and weather conditions, are manually annotated to enhance training. The proposed YOLO-v5 model surpasses traditional methods in terms of accuracy and execution time, demonstrated through simulations on PKU, COCO, and DAWN datasets. The findings highlight the effectiveness of the proposed method in challenging scenarios [24].

The author aims to solve the problem of traffic density estimation for effective traffic management by utilizing deep learning techniques. They collect data from open-source libraries and label vehicles in images into six different classes. To address the imbalanced dataset, data augmentation techniques are applied. The proposed model is based on an ensemble of the Faster R-CNN and Single-shot detector (SSD) models, trained on processed datasets. Experimental results demonstrate that the proposed ensemble outperforms base estimators on FLIR thermal dataset, achieving a higher mean average precision (mAP) of 94%. The ensemble model shows promising results compared to other estimators, with better performance on thermal images. The proposed model also exhibits significant potential for traffic density estimation, offering valuable insights for traffic management [25]. The author aims to develop a method for in-vehicle passenger detection using MIMO radar. They propose a CNN-LSTM model that accurately detects, counts, and classifies passengers in five-seater vehicles. The model combines CNN for feature extraction and LSTM for temporal prediction. Reliable datasets collected from different car models are used to evaluate the model, which achieves high accuracy in detecting unattended infants/children (over 95%), counting passengers and identifying occupied seats (89% accuracy), and classifying passengers (over 74% accuracy). The results demonstrate the generality of the proposed method for deployment in any five-seater vehicle [26]. A vehicle classification system based on low cost triaxle anisotropic magneto resistive sensor was proposed in [27]. To detect the vehicle within a segment and single lane the vehicle signals effectively, a fixed threshold state machine procedure is presented. Through experimentation it is concluded that the detection accuracy of the presented technique can reach up to 99.05% and the overall average classification accuracy is almost 93.66. These results show the efficiency of presented technique for low-speed congested traffic, but authors did not test their technique in the constrained environment. In [28], author presented a vehicle classification problem based on laser scanner profiles, which is found as a part of electronic tolling systems. Vehicle shapes are extracted from laser scans and then explore them as multi-dimensional shapes. The presented technique explores identical and non-identical shapes for discovering typical shapes then classified using global alignment of shapes. This technique shows state of art results by overcoming per class error by 4 to 17%. In military operations, nighttime multi-vehicle detection over a long distance is essential. The visual characteristics of automobiles are hard to identify at night due to inadequate lighting, which leads to numerous missed detections. Based on Gm-APD LiDAR intensity images and point cloud data, this work [29] suggests a two-level detection strategy for long-distance nighttime multi-vehicles. There are two layers to the approach. The first level is 2D detection, which raises the brightness of weak and small objects and improves the local contrast of the intensity image. The detection result of more than the threshold is reserved as a dependable item when the confidence threshold is set, while the detection result of less than the threshold is a suspicious object. The suspicious object region from the first level of 3D recognition is translated into the appropriate point cloud classification judgment in the second level, and the object detection score is achieved by thorough judgment. The method outperforms current state-of-the-art detection techniques, according to experimental data, achieving a detection accuracy of 96.38% and effectively enhancing the detection accuracy of multiple cars at night.

LiDAR point cloud semantic analysis accuracy is necessary for autonomous cars to interface with the actual world. By fusing 3D global Hough features and 2D local Hough features with a classification deep learning network, this work [30] suggests a hybrid 2D and 3D Hough Net. First, the global Hough features are extracted from the 3D object point clouds by mapping them into the 3D Hough space. For training global features, the 3D convolutional neural network receives the generated global Hough features as input. To limit the calculation of duplicate points, a multi-scale critical point sampling approach is created to extract crucial points in the 2D images projected from the point clouds. A grid-based dynamic nearest neighbors' method is created by exploring the important locations' neighbors to extract local information. To classify objects, completely connected layers that are input into the two networks are then connected to the full connection layer. The experiments show that the classification accuracy achieved 97.6% by allocating the 25 × 25 × 25 cells for the 3D Hough spaces and specifying 0.25 as the unit size of the local Hough spaces. The results of the studies demonstrate that by assigning the 25 25 25 cells for the 3D Hough spaces and choosing 0.25 as the local Hough spaces' unit size, the classification accuracy was increased to 97.6%.

As per the research carried out [31], numerous automated traffic monitoring systems have been created because of the growing number of cars circulating in various urban cities. Roadside camera traffic monitoring systems are being widely used because they provide crucial technological benefits over other traffic monitoring systems. The performance of the entire system is significantly impacted by the methods used for vehicle identification and traffic congestion categorization, which are the two primary processes for video-based traffic congestion detection systems. Investigate four chosen vehicle recognition techniques in two contexts: urban and highway, including the Gaussian Mixture Model (GMM), GMM-Kalman filter, optical flow, and ACF object detector. Additionally, three classification approaches for traffic congestion are examined. The comparison of the various approaches enables us to select the ones that will function best when incorporated into the framework suggested for addressing the traffic problems on the Bizerte Bridge. For mobile robots in industrial sectors with dusty environments, the author [32] seeks to develop noise-filtering algorithms that can remove dust from LiDAR sensory data. It was created as an intensity-based filter (LIDROR) based on a thorough examination of the characteristics of dust point clouds detected by a LiDAR sensor in order to accomplish the desired result. To the best of our knowledge, the suggested approach and the previously created LIOR are the first initiatives in this industry to design a de-dust filter utilizing non-AI method.

Utilizing 3D Light Detection and ranging, this work focuses on the issue of vehicle detection and tracking for an autonomous vehicle (LiDAR). A new clustering approach is suggested [33] to get vehicle candidates from preprocessed point cloud data gathered by the LiDAR to improve the accuracy of vehicle detection and tracking. To distinguish automobiles from vehicle candidates, a support vector machine (SVM) trained classifier is used. Vehicles are tracked using the Kalman filter and the global nearest neighbor (GNN) method, and the tracking data is used to further increase the accuracy of the vehicle detection results. A testing platform has been used to validate the suggested technique. In this study [34], a hierarchical approach based on RCNN was presented for the detection and recognition of vehicles and vehicle license plates. The tasks were related to RCNN's complexity. Multiple complicated level RCNNs can be used in the same system in this fashion. For intelligent traffic monitoring, a sample vehicle detection system was created as an example. The license plate recognition RCNN was then taken into consideration for vehicle identification. Authors in [35] suggested a simple, cooperative object classification framework for CAV that protects user privacy and focuses on the object classification problem. The framework's main novelty is the inclusion of the P-CNN model, which analyses encrypted images sent from cars and, when combined, yields accurate classification results. To protect the image during storage and transmission, extra image distortion is used. In P-CNN, our safe protocols based on the additive secret sharing technique execute the operations in the original CNN model (VGG16). The outcomes of the experiments show that P-CNN provides precisely the same classification outcomes as the VGG16 model without requiring vehicles to provide raw image data directly, which might contain private information. Additionally, the communication overhead and processing cost on two edge servers are reasonable.

The study [36] utilizes data from naturalistic driving to calculate the driver-vehicle volatilities. This work intends to forecast the occurrence of safety-critical events and deliver appropriate feedback to drivers and nearby vehicles by integrating and fusing several real-time streams of data, such as driver distraction, vehicle movements and kinematics, and instability in driving. The naturalistic driving data, which was gathered from more than 3500 drivers, includes 7566 normal driving events, 1315 severe events (such as crashes and near-crash situations), car kinematics, and driver behavior. Long Short-Term Memory (LSTM), 1DCNN-LSTM, and 1D-Convolutional Neural Network (1D-CNN) are used to capture the local dependency and volatility in time-series data. The input parameters include driving volatility, vehicle kinematics, and distracted driving impairment. The findings show that the 1DCNN-LSTM model performs best, with an accuracy of 95.45% and a precision of 95.67% for predicting crashes that would occur in 73.4% of cases. The CNN layers extract additional information and consider the temporal dependency between observations, assisting the network in learning driving patterns and volatile behavior. A recent trend in research using deep learning [37], [38], [39], [40], [41] has shown its tremendous potential for vehicle classification. However, these techniques mostly did not consider the constrained environment where their restrictions of data quality or hardware constraints. Similarly, some of the techniques presented in [42] does not consider constrained environments as considered in the proposed work in this paper.

According to the literature review, the research that have been conducted for vehicle classification have some limitations particularly in case if constrained environment where either only low-quality images are present (for instance, due to bad weather conditions) or there are some hardware constraints. Some researchers stated their future work as to work with poor training datasets. So, in this work, we created custom dataset with poor quality images with multiples classes such as Car, Bike, Truck, Rickshaw and Bus. Every class used in our project is totally different from each other in terms of model. We take every view of the vehicle like front, back, side, front side and back side to train our network. We conduct our vehicle classification work based on two types of datasets: noisy dataset and a relatively clean dataset. Both these datasets are custom datasets created and collected from real-world images using the image search feature of Google. In this work, the dataset is collected by keeping in view the limitations of the previous work. The most used methodologies of vehicle classification are implemented and evaluated using datasets to analyze their performance in real world environment.

III. Proposed Methodology

In our work, we apply deep learning techniques for the vehicle classification task. The initial step of methods is the data collection technique. The images used in this project are taken from Google images we contribute of ownership protection to prove the ownership of the data. We provide ownership protection systems through watermarking technique. After that for classification purposes the initial step is preprocessing step in which noise is removed from images. After preprocessing is done on images, they can be used to train or to test the network which is also known as recognition step. In the training phase, the preprocessed images are passed to the network with their specified labels so that, for the classification the best network weights can be found. In the testing phase, the network is configured with the set of weights found during the training phase and then the recognition of images is performed. The recognition also outputs the confidence level against each vehicle image. The highest confidence level is used to depict the vehicle class. For increasing the number of training samples, in short for more generalization, the method of synthetic image generation is used during the preprocessing phase (the synthetic images are not being used in testing phase). Fig. 1. presents the general architecture of our proposed framework. A brief description of the proposed technique is mentioned in the following section. It begins with the data collection technique. After collecting the data, ownership protection of that data through watermarking is explained. For classification, preprocessing and the detailed description of convolutional neural network are also presented.

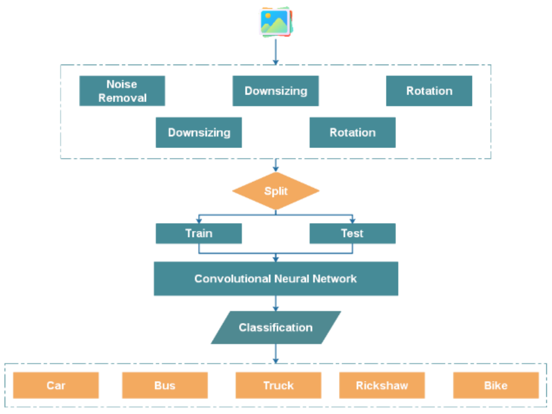

Fig. 1. Proposed framework.

A. Data Collection

In this work, we used the images obtained from Google images. We used "get them all" downloader in chrome to download bunch of images at once. The entire project is executed on a personal computer with restricted computational resources. Our project is also limited to the simple classification of five classes: Car, Bike, Rickshaw, Bus and Truck. Every class is visually different from one another. But our dataset is boundless. Fig. 2 shows the representation of our dataset.

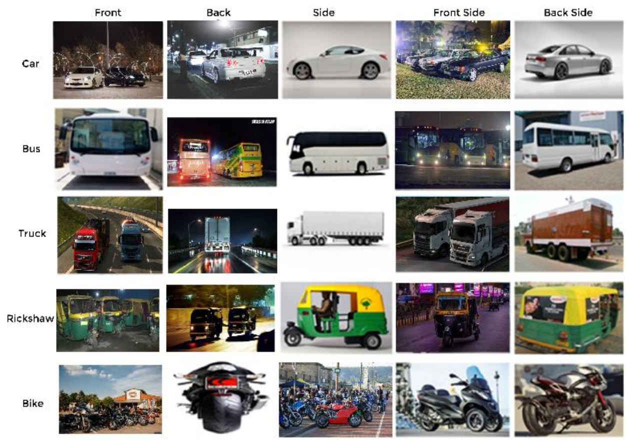

Fig. 2. Dataset representation.

We performed different experiments to achieve maximum accuracy on two types of datasets based: noisy dataset and clean dataset. If one used an immovable camera, mainly CCTV or any other camera that is not generally configured once installed, in any application to detect and classify vehicles, there is a huge chance of noise involvement. This also leads to the foundation of our reason for using images from Google images there is a massive possibility of the involvement of noisy images in them. Moreover, these noisy images contain more than a single desired object in the image, for example, many bikes in a single image and the same issue for remaining classes as well. Therefore, we cleaned our dataset by removing such noisy images. As a result, our finalized dataset contains 4000 images of cars, 3213 images of buses, 4000 images of bikes, 4000 images of trucks, and 1159 images of rickshaws; that is, a total of 16372 images.

These evaluation configuration aims to measure the performance of the system with the vehicle's image database. The database is separated into five groups of non-overlapping classes. To perform experiments, the databases used in each experiment were divided into three sets: The training set, which is used for training the system, the validation set, which is used for dynamically tuning the Meta parameters of the system and then the test set, which is used to measure the accuracy of the system. In all experiments, it is ensured that there is no subject overlap among the three sets.

B. Preprocessing

The images should be preprocessed before it fed into the convolutional neural network. Our preprocessing pipeline consists of various steps as follows:

1) Noise removal: In a bid to work with real-world images with low quality and noise, we used the images obtained from Google images to train the neural network. As images are taken from Google images, there is a massive possibility of the involvement of noisy images and - low-resolution images with lots of variation. In this step, those images contain more than a single desired object in the image --for example, many bikes in a single image and the same issue for the remaining classes as well. At this point, we need to refine whole the dataset. We used certain filters including the Gaussian filter and median filter to refine the images containing multiple objects and those captured in low-light conditions. The unsharp filter is also applied to refine edges and make the objects in ambiguous images clearer thus making it easier for the model to classify such images.

2) Downsizing: The training set contains a diversity of image dimensions as are downloaded from Google images. We resized the images into 32x32 and 64x64 dimensions as required by the model to work with low resolutions. Because images with large dimensions like 256x256 are too time-consuming and sometimes it is very difficult to process them due to hardware limitations or due to the availability of - low-resolution images only in the real-world environment. By using downsize images, there is a chance of information being lost which s to poor results. As in our experiments, the accuracy stuck at 87% by using 32x32 dimension images. When we move towards 64x64 dimension images we get better results approximately 90%. So, if we use high dimension images for training there might be a chance to get better results. We used a light image resizer to resize these all images at once.

3) Rotation: Convolutional neural networks are more suitable to work with huge amounts of data. The network trained well with maximum samples for training. Maximum training allows the network to better learn the variation which is present in the data. To increase the data, data augmentation technique is used. By using this technique, images can be transformed and produced a new image. Rotation is one of the simple transformations through which we can increase our data by just angular rotating the original image. In our approach, we applied 10 different angular rotations. In our dataset, with concern of the different classes, the Bus and Rickshaw chavings has low representation meanwhile the dataset in this phase is unbalanced; therefore, the rotation phase helps in generating additional data through 10 angular rotations. We perform rotations with the angles [-2, 2, -4, 4, -6, 6, -8, 8, 10, -10].

4) Flipping: Several data augmentation techniques can be used to increase the size of training datasets. One of the widely employed techniques is to horizontally flip photographs. Different flipping images can be used to train a model. To double the size of the data, we treat the image that has been flipped as a genuine image. the in this project, we use random flipping to train our model. In our dataset, with the concern of the different classes, the Bus and Rickshaw class is underrepresented meanwhile the dataset in this phase is unbalanced therefore generating additional data through flipping. Color space.

5) Transformation: The source image is used if it has already been colored and is in the RGB color space. Any source images found in the dataset that are not in RGB color space and exist in black and white, grayscale, or any other non-RGB color system are transformed into RGB color space using industry-standard procedures. The convolutional neural network (CNN) only accepts and processes RGB images, hence we have included colored images in the suggested work as well. Red, green, and blue hues are defined by RGB's three channels.

C. Convolutional Neural Network

CNN are the specified neural network. In recent past, much research work regarding image processing and classification has been done by using CNN. The CNN architecture is made up of various kinds of layers. These layers comprise of convolutional layer, pooling layer, and fully connected layers. These layers may contain some supplementary hyper parameters such as size of filter, padding, stride for the convolutional layers and the quantity of neurons used in fully connected layer. The all number of layer types with their supplementary hyper parameters and their consequence on training quality and speed, all these circumstances make it hard to choose architecture. The capacity of learning makes this classifier much superior to other classifiers. It not only learns the weight of features but the features themselves. On universal image classification, CNNs have accomplished revolutionary accuracy. Similar concepts are adjustable for audio and videos as well. In this segment, we define CNN for the processing of images.

Fig. 3 represents the generic architecture of CNN used in the proposed scheme. CNN comprises of numerous convolution layers, pooling layers, and a fully connected layer. After one or several convolution layers, a pooling layer is applied. In this architecture, the convolution layers extract the useful features from the input that produce feature maps. The pooling layers then decrease the spatial size of these feature maps. There are multiple filters used in this architecture with different kernel sizes. It displays the edges of the input image as a result and then this idea of filters for feature extraction is interpreted to CNN. CNN learns these filters as a replacement of hard coded filters.

1) Convolutional layer: As a normal neural network, the convolution layer comprises of units as well, however with a rare arrangement and an unusual connectionism of the units. The main differences between these two networks are based on weight sharing, local connectivity, and three-dimensional arrangement of the unit.

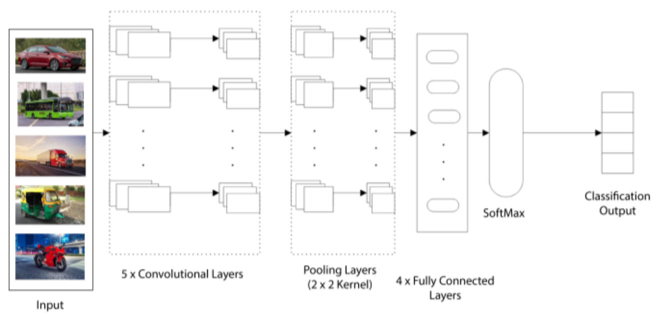

Fig. 3. Architectural flow of CNN on input data.

In weight sharing, the same weights are used for several output units. Basically, the weights are the filters and these filters are used on the entire input to generate the output. In local connectivity, the size of this connectivity is defined by the kernel size. In this connection, all the units of input are not connected with the output unit. The three-dimensional arrangement of the unit arises from the image. Colored image normally comprises of three channels (red, green, and blue). Each channel is defined by a two-dimensional matrix. Consequently, the input and output of the CNN is a three-dimensional matrix. The output comprises of three-dimensional matrix with feature maps of two dimensions and this x times for the filter numbers used in this layer. Each filter creates a feature map. In convolutional layer, the feature map is obtained by convolving a filter across the input volume, adding a term bias. If we represent the k^{th} input feature map of the convolutional layer ash^k and its filter is denoted by w^k and term bias isb^k, then the output which is feature map h^{k+1} is obtained as:

$$h^{k+1} = W^k \times h^k + b^k \tag{1}$$

Where × shows a convolutional function

2) Pooling layer: The pooling layer is applied after convolution layers. This layer is used to minimize the spatial size of the feature maps. It actually downs samples the feature maps. The spatial dimension of the feature maps is reduced of 75% by one pooling with the stride size of 2*2. Size of sliding over the input channel is called stride. The pooling layer has no activation function or weight to learn because it works independently on each feature map. The max-pooling and average-pooling are the common pooling layers. The max-pooling layer calculates the maximum, which lies on the receptive side of filter and the average-pooling layer calculates the average.

3) Fully connected layer: These layers work identically to hidden layers of a regular neural network. To determine the output scores and arbitrary features, these fully connected layers can be used at the termination of the CNN after numerous stages. As the name shows, in this section all layers of the units relate to the input layer.

4) Training: In this phase, we have a trained set of input vectors with a parallel set of output vectors. The goal is to use training data to train the network's biases and weights so that the network performs good in classifying input. The further training data given to a network, the additional iterations and weight updates can possible and then it came out with improved network architecture which gives better results. Through a training process, the way a computer is capable to adjust its filter values or weights is called back propagation. Backpropagation is an efficient method to propagate back the error to calculate the gradient from the loss function through the network. It computes the gradient referring to the weights. The backpropagation has the following recursively equation to calculate the gradient in respect to the weights.

$$\frac{\partial L(\theta)}{\partial w_{l,k}^{(i)}} = \delta_l^{(i)} . Z_K^{(i-1)} \tag{2}$$

Backpropagation can be divided into four different segments. Forward pass, loss function, backward pass and weight update. In forward pass, the training images pass through the whole network. In the first training image all the filter values or weights are casually initialized thus output does not give any sensible conclusion about what the classification could be. This leads towards the loss function of backpropagation. Loss function is used to reduce the error. Modest loss function is the mean squared error (MSE) which computes the squared difference between half of the difference between actual and predicted as follows:

$$L(\theta) = \frac{1}{2N} \sum_{n=1}^{N} \| f(\check{Y}_n ; \theta) - \check{Z}_n \|^2 \tag{3}$$

The backward pass in the propagation decides which weights donates maximum loss and it adjusts the weights to decrease the loss. After this in the weight update process, all the weights of the filters got updated.

$$W = W_t - \eta \frac{dL}{dW} \tag{4}$$

5) Gradient descent: To minimize the loss function, gradient descent method is mostly used. It needs the gradient of the loss function in terms of α. Backpropagation method is used to calculate the gradient. Gradient descent is an iterative optimization algorithm. In this algorithm weights can be updated with the help of gradient.

$$\theta \leftarrow \theta - \eta . \frac{\partial L(\theta)}{\partial \theta} \tag{5}$$

Where η defines the learning rate. Learning rate is a parameter that is selected by the programmer. High learning rate meant to take higher steps to update the weight.

6) Dropout layer: Dropout layers have a very definite function in neural networks. The idea of dropouts is basic in nature. It is a technique used for decreasing over fitting in neural networks. The units will be arbitrarily dropped for each iteration of the training. Dropout layer is only used in training time. Units are dropped with the probability of p and this p can be different for each layer. It is a method that prevents overfitting. It drops out an arbitrary set of activations in that layer by setting them to zero in the forward pass.

By using dropout, the network should be able to deliver right classification if some of the activations are dropout. In supervised learning, it improves the performance of neural networks. Except loss layer, dropout can be used in every layer and applied during supervised training alongside with end to end back propagation. The option of dropping the unit is random. In dropout, the forward process is defined as:

$$h^{k+1} = M * (w^k \times h^k + b^k) \tag{6}$$

7) Rectified linear unit: Rectified Linear Unit (ReLu) is the efficient activation function because the network is capable to train a lot faster while not making any significant difference to the accuracy. The value for ReLu can be calculated as:

$$\sigma(y) = \max(0, y) = \begin{cases} 0, & if y < 0 \\ y, & if y \geq 0 \end{cases} \tag{7}$$

8) Exponential linear unit: Exponential linear unit (ELu) is for negative values. Over the negative activation, the mean activation is near zero and this led towards the faster merging of the network because the input for the next layer has zero mean. The formula for ELu is:

$$\sigma(x) = f(x) = \begin{cases} \alpha.(e^x - 1), & if x < 0 \\ x, & if x \geq 0 \end{cases} \quad (8)$$

Where α>0 and is generally set to 1.

9) SoftMax: SoftMax activation function is used for the encoding of the output. It represents a probability distribution over k classes and hence it is only used in the output layer. The formula of SoftMax is:

$$\sigma(y)_j = \frac{e_j^y}{\sum_{k=1}^{k} e^{z_k}}, for\ j = 1, \ldots.. k. (9)$$

IV. Results and Discussion

For experimentation, we used libraries like Tensorflow, Keras, Sklearn, Matplotlib, Scipy, and OpenCV in python language running on a Core i3 processor with 8GB of RAM.

A. Convolutional Neural Network for Vehicle Classification

The proposed architecture of our CNN has been presented in Fig. 3. Various convolutional neural network architectures were used in our implementation. The reason behind this is to pinpoint the architecture that achieved a reasonable accuracy with under restricted environment with respect to computational resources and quality of the data. The network takes an input of 64x64 RGB image and then classifies the given image assigning it a vehicle type. Our architecture of CNN consists of 5 Convolutional layers, 4 fully connected layers and then a softmax output layer. We used ReLUs as activation function. The first CNN layer is a convolutional layer on which we apply Convolutional kernel of 5x5 and results an image of 32x32 pixels. This convolutional layer is then followed by another sub sampling layer that uses max pooling (with 2x2 kernel size) to shrink the image to half of its original size. Then drop out layer is applied with drop out 0.2. The second CNN layer is a convolutional layer on which we apply a Convolutional kernel of 3x3 and results an image of 16x16 pixels. This convolutional layer is then followed by another sub sampling layer that uses max pooling (with 2x2 kernel size) to shrink the image into half of its original size. Then drop out layer is applied with drop out 0.2. The third CNN layer is a convolutional layer on which we apply a Convolutional kernel of 3x3 and results an image of 8x8 pixels. This convolutional layer is then followed by another sub sampling layer that uses max pooling (with 2x2 kernel size) to shrink the image into half of its original size. Then drop out layer is applied with drop out 0.2.The fourth CNN layer is a convolutional layer on which we apply a Convolutional kernel of 3x3 and results an image of 4x4 pixels. This convolutional layer is then followed by another sub sampling layer that uses max pooling (with 2x2 kernel size) to shrink the image into half of its original size. Then drop out layer is applied with drop out 0.2.The fifth CNN layer is a convolutional layer on which we apply a Convolutional kernel of 3x3 and results an image of 2x2 pixels. This convolutional layer is then followed by another sub sampling layer that uses max pooling (with 2x2 kernel size) to shrink the image into half of its original size. Then drop out layer is applied with drop out 0.2.

B. Experiment 1: Noisy Dataset

Since the real-world images are often noisy and blur; therefore, we also tested the performance of our framework on noisy data. Table I shows the detailed dataset of noisy images. As the rickshaw class contains few images so we generated additional data for the rickshaw class to improve the imbalance ratio. Now our new data contains 96273 images.

TABLE I. NOISY DATASET

Class	Number of Images
Car	19333
Bus	18844
Truck	21142
Bike	19374
Rickshaw	15822

1) Model 1: In the first set of experiments, we took 20% of data for training to investigate the results when many images are not available for the training. So, our first split contains 19254 images in the training set. Next, we flipped our training data of 19254 images and generated new 38504 (two times the size of training data) samples through flipping because the training set was quite small as compared to the testing set that contained 80% of the images. Then we again apply the split of 34656 training images and 3848 images for validation; we got a validation accuracy of 74% and validation split accuracy of 73% and testing accuracy of 66% in this set of experiments. Since this accuracy is not good enough, we proceed to the next set of experiments.

2) Model 2: In the experiment set no 2, data is the same as used in the experiment set no 1. But this time we applied the split of 50/50. So, our split A becomes 48136/48137. Now we do not y apply any flipping and the network remains the same we use a split B of 43322 training images and 4814 images for validation. We got a validation accuracy of 75%, validation split accuracy of 76%, and testing accuracy of 62%. Through the results of the experiment set 1 and 2, we conclude that flipping yields higher accuracy on test data. This accuracy can be improved by working with higher resolution images and using a machine with higher computational power, such as a GPU (graphical processing unit) as there are too many samples for normalizing but in this paper, we are examining the results in a constrained environment with hardware constraints and low data quality. This is because the real-world images usually do not have good quality and are at times noisy. Moreover, we are also advocating the case for the importance of increasing the computational resources for better results. In our experimental study, we observed that training samples greater than 80,000 become too computationally intensive for the CPU to normalize data. For 80,000 images with 32x32 image resolution, the normalization function must perform 32x32x3*80,000 computations.

3) Model 3: In the experiment set no 3, the data used was the same as in the previous experiment. We applied a split of 30/70 which is 28881/67392 for the training/testing dataset. Next, we applied -left-right flipping on the training dataset obtaining 57762 images. We split this data again for applying validation during the classification process with a split of 51985 images for training and 5777 images for validation. In this experiment, we got a validation accuracy of 74.8%, a validation split accuracy of 74%, and testing accuracy of 63.4%. In this experiment, we notice that test accuracy is not increasing so we decided to tweak the network architecture further.

4) Model 4: In this set of experiments, data remained same as in experiment set 3 and we only changed the network architecture by adding one convolutional layer and got increased validation accuracy of 77%, validation split accuracy of 75% and testing accuracy of 67%. This motivated us to further investigate the effect of network architecture on overall classification results.

5) Model 5: In experiment set no 5, we again changed the network architecture by adding one more convolutional layer and got validation accuracy of 78%, validation split accuracy of 77% and testing accuracy of 67%. In this experiment, we noticed that adding an additional layer did not help in increasing accuracy.

Table II summarizes the experimental results obtained while running the experiments on the noisy dataset. The highest testing accuracy we obtained in these experiments is 67 percent. We can get higher accuracy through this dataset by using higher image dimensions but in this paper, we are concerned with low-resolution images and experiments with hardware constraints. The images used in this dataset were taken from Google images to cater for real world scenarios under noisy environment. Some images contain more than a single desired object in the image. For example, multiple bikes in a single image. Same for the other classes as well. At this point, we need to refine all dataset because training with such samples require higher image dimensions such as training with 96x96 images for 40,000 samples took around 2 hours.

C. Experiment 2: Clean Dataset (32-Bit Images)

Our data for the previous set of experiments was noisy and we cleaned the data for further experiments. We kept those images which contain a single desired object but again with a low resolution of 32x32. Table III presents the statistics of this dataset.

TABLE II. RESULTS OF CNN ON NOISY DATA

Exp. No.	Split	Flip	Training Data	Validation Acc	Testing Accuracy	F1-Score
1	20/80	Yes	96273	74%	66%	74%
2	50/50	No	96273	75%	62%	75%
3	30/70	No	96273	74.8%	63.4%	74%
4	30/70	No	96273	77%	67%	77.4%
5	30/70	No	96273	78%	67%	76.8%

TABLE III. OVERVIEW OF CLEAN DATASET (32-BIT IMAGES)

Class	Number of images for training	Number of images for testing
Car	4000	2000
Bus	3213	1999
Truck	4000	2000
Bike	4000	2000
Rickshaw	1159	400
Total images	16372	8399

1) Model 1: In experiment set no 1, we applied -left-right flipping on the training data and our data size becomes 32744 images. We trained on 31106 samples and validated on 1368 samples and got a validation accuracy of 84% and a testing accuracy of 80%. From a similar experiment on the noisy dataset, we noticed that test accuracy improves with cleaner data but the bus and rickshaw class is underrepresented. Therefore, we applied rotations for generating additional data for the next set of experiments.

2) Model 2: In experiment no. 2 on the clean dataset, our dataset contains 18690 samples when we applied flipping. We trained on 35511 and validated on 1869 samples and got a validation accuracy of 92.83% and a testing accuracy of 83%. In this experiment, we notice that accuracy further improves by generating additional data. Now in the next set of experiments, we investigate the change in results due to changes in network architecture.

3) Model 3: In this set of experiments, the data is the same as in the previous experiment. We trained on 35511 and validated 1869 samples. However, in this experiment, we changed the network architecture by increasing the filter depth and got a validation accuracy of 89% and a testing accuracy of 86.4%. Increasing filter depth improved testing accuracy. Next, we examine the effect of adding more layers on the overall classification results.

4) Model 4: In this experiment, data is again the same as in the previous experiment. We added one more convolutional layer and got a validation accuracy of 90% and a testing accuracy of 87% which is not significant. So next we add a fully connected layer to see its effect on the results.

5) Model 5: In experiment no. 5, we only added one fully connected layer and got a validation accuracy of 90.85 and a testing accuracy of 87%. The results of these experimental studies have been tabulated in Table IV.

TABLE IV. RESULTS OF CNN EXPERIMENTS ON CLEAN DATASET (32-BIT IMAGES)

Exp. No.	Rotation	Flip	Training Data	Validation Accuracy	Testing Accuracy	F1-Score
1	No	Yes	32744	84%	80%	83.3%
2	Yes	Yes	37380	92.83%	83%	92%
3	No	No	37380	89%	86.4%	89.5%
4	No	No	37380	90%	87%	90.2%

Table IV summarizes the experiments which we performed on a clean dataset of images with a resolution of 32x32. This dataset produces higher accuracy than noisy datasets because in this dataset only desired object is involved. Experiments seem to be giving fine results while using 5 convolutional layers and 4 fully connected layers. But accuracy is stuck here because the images have been downsized to 32x32 and due to this downsizing information has been lost. This motivated us for our next set of experiments with 64x64 images.

D. Experiment 4: Clean Dataset (64-Bit Images)

We performed previous experiments on 32x32 bit images that was a very low image resolution making it challenging to obtain good classification results. To further investigate the performance of CNN on images with slightly better resolution, we performed experiments with 64x64 images. Table V shows the statistics dataset of clean 64x64 images.

TABLE V. OVERVIEW OF CLEAN DATASET (964-BIT IMAGES)

Class	Number of images for training	Number of images for testing
Car	4000	2000
Bus	3213	1999
Truck	4000	2000
Bike	4000	2000
Rickshaw	1159	400
Total images	16372	8399

1) *Model 1:* In the first experiment set for 64-bit images, to improve the imbalance ratio, we generated some additional data for rickshaw and bus classes and now have 18690 samples. We trained on 17755 samples and validated on 935 samples and got validation accuracy 0f 88% and test accuracy of 86%. We notice that even without flipping 64x64 bit-sized training yielded high accuracy.

2) *Model 2:* In the experiment set 2, we took the sample size of 16372 and applied rotation; as a result, our data has now images 18690. Next, we apply after flipping and the data now has 37380 images. We trained on 35511 and validated in 1869 and we got a validation accuracy of 92% and a test accuracy of 88%. After this experiment, we observed that flipping the training data improved the accuracy by 2%.

3) *Model 3:* In experiment no. 3, we changed the convolutional architecture by just adding one more convolutional layer. By changing the architecture, we got a validation accuracy of 93% and a test accuracy of 89% and observed a slight improvement in the results. Next, we investigate the effect of modifying the network architecture a bit further.

4) *Model 4:* In this experiment set, we changed the convolutional architecture by adding a fully connected layer and got a validation accuracy of 93% and a test accuracy of 90%.

Table VI summarizes the results of experiments that we performed on the clean dataset with the dimension of 64x64. By using 64x64 dimension images, we get 90% accuracy which is higher than for those 32-bit images and noisy images as evident from Tab. 2, 4 and 6. We can safely claim that we can get higher efficiency with our architecture if we work with higher resolution images and better computer power like GPU. However, in this paper, we are investigating the case of real-world noisy images and low-resolution images with hardware constraint; therefore, we did not perform experiments with better resolution, say images with 256x256 resolution.

TABLE VI. RESULTS OF CNN EXPERIMENTS ON CLEAN DATASET (64-BIT IMAGES)

Exp. No.	Rotation	Flip	Training Data	Validation Accuracy	Testing Accuracy	F1-Score
1	Yes	No	18690	88%	86%	88.5%
2	Yes	Yes	37380	92%	88%	91.5%
3	No	No	37380	93%	89%	93.7%

E. Discussion

In the proposed work, different sets are experiments are performed for both noisy and clean images dataset. The reason for doing number of experiments for both the cases is to go through different combinations, data splits and see the impact on performance. The experiments conducted on noisy images dataset explored the impact of different settings and conditions on the accuracy of classification in image processing. The findings revealed that using a smaller portion of the data for training resulted in lower accuracy, emphasizing the need for sufficient training data. Flipping techniques were applied to augment the training data, leading to improved accuracy in some cases. The importance of factors such as image resolution, computational resources, and network architecture were highlighted, with suggestions for further optimization. Overall, the experiments demonstrated the significance of data augmentation, hardware capabilities, and network design in achieving higher accuracy in image classification tasks. Similarly, the findings deducted from experiments performed on clean dataset with 32-bit images concluded that data augmentation techniques, such as flipping and rotations, led to improved accuracy in image classification tasks. Experiment set 1 demonstrated that left-right flipping resulted in a validation accuracy of 84% and testing accuracy of 80%. Further experiments on clean datasets (experiment set 2) showed that additional data generation through flipping enhanced accuracy, with a validation accuracy of 92.83% and testing accuracy of 83%. Changes in network architecture (experiment set 3) by increasing the filter depth led to improved testing accuracy of 86.4%. Subsequent experiments involving the addition of more layers (experiment set 4) and a fully connected layer (experiment set 5) yielded validation accuracies of 90% and 90.85%, respectively, contributing to marginal improvements in testing accuracy. These findings highlight the significance of data augmentation and network architecture modifications in enhancing the classification results of image processing models.

In case of experiment performed on clean dataset with 64-bit images, the conducted experiments revealed interesting findings in image classification tasks. In experiment set 1, the training of 64-bit images without flipping resulted in high accuracy, with a validation accuracy of 88% and test accuracy of 86%. Experiment set 2 demonstrated the benefits of data augmentation through flipping and rotation, leading to an increased sample size and improved accuracy of 92% in validation and 88% in testing. Further modifications in the convolutional architecture, specifically adding one more convolutional layer (experiment set 3), contributed to a slight improvement in accuracy, with a validation accuracy of 93% and test accuracy of 89%. Finally, in experiment set 4, the addition of a fully connected layer resulted in a validation accuracy of 93% and test accuracy of 90%. These findings emphasize the importance of data augmentation and architectural adjustments in enhancing the performance of image classification models.

For comparison purposes, we also ran the experiments using other classifiers such as SVM, Naïve Bayes, and Decision trees on 64-bit images. Table VII lists the results of these experiments. Along with the results represent the dataset which we use for comparative analysis. In this comparison, we compare our result with other machine learning algorithms. For the dataset which gives higher accuracy in CNN, we take that dataset and compute result in SVM, Naïve Bayes and Decision tree.

TABLE VII. PERFORMANCE COMPARISON WITH OTHER CLASSIFIERS

Experiment	**Cell per Block**	**Pixel per cell**	**Feature Vector size**	**Validation Accuracy**	**Test Accuracy**
SVM	2	8	1764	82.9%	83.39%
Naïve Bayes	2	8	1764	71.39%	69.6%
Decision Tree	2	8	1764	50%	50.67%
VGG-16	-	-	1764	56%	55.97%
ResNet-50	-	-	1764	64%	68.52%

We also present the comparison of the performance of CNN with other classifiers: SVM, Decision Tree, and Naïve Bayes in Fig. 4. It is evident from this figure that our convolutional neural network achieves highest accuracy in comparison to other machine learning algorithms. CNN attains 90% accuracy while using 5 convolutional layers and 4 fully connected layers. The results are very conclusive, and CNN shows the highest accuracy rate in comparison to other classifiers.

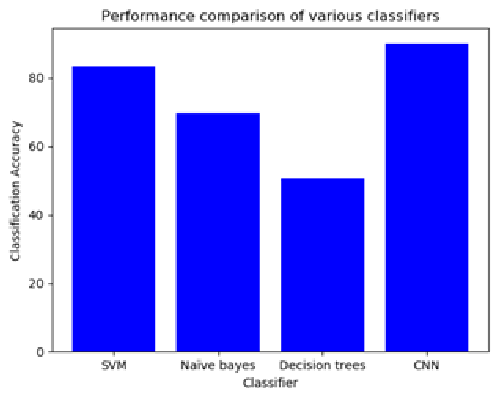

Fig. 4. Comparison of proposed model with other classifiers.

Although our technique has a unique characteristic of working under constrained environment, comparing it with deep learning-based vehicle-type classification techniques like [34] (proposed by Wang et al.) shows that our technique gives better results as compared to 65.55% accuracy of their work. However, the advantage of [34] over our proposed technique is that the technique of Wang et al. can be used to classify similar vehicles (for instance, cars only) as well. While on the other hand, we considered different types of vehicles like cars, bikes, buses etc.

Table VIII shows the comparison of the proposed with some of the latest works on same domain using similar approaches. The proposed models outperform the existing works in terms of accuracy.

TABLE VIII. COMPARISON OF PROPOSED MODEL WITH PREVIOUS WORKS

References	**Years**	**Accuracy**
[44]	2020	89.50%
[34]	2021	65.55%
[43]	2021	77.55%
[45]	2023	77%
Proposed	2022	**90.85%**

V. LIMITATIONS AND FUTURE WORK

Several limitations are there that effect the streamline vehicle image classification using deep learning mechanisms. One limitation is that the current work was trained and tested on 32-bit and 64-bit datasets, which may restrict the model's efficiency. To address this, the researchers suggest training the model using a higher resolution dataset, such as 256x256 bits, which could lead to improved results. However, they acknowledge hardware constraints and propose the use of GPU clusters for training on larger datasets and more complex neural networks. Gathering more samples from different vehicle classes is also identified as a way to enhance the results and expand the scope of the study. In the future, the researchers aim to overcome these hardware and software limitations to achieve higher accuracy. Additionally, they suggest extending the work by considering vehicle images captured under poor lighting conditions or adverse weather situations. By addressing these limitations, such as utilizing higher-resolution datasets, overcoming hardware constraints, and expanding the dataset size, the researchers anticipate achieving improved accuracy and broader applicability of the classification technique in the future.

VI. CONCLUSION

Data is the most important asset. To explore and extract useful information from this data, the data is also shared with several stake holders via diverse channels. Hence ownership protection of such datasets is very essential. We provide ownership protection systems through watermarking technique. After ownership protection of the data, we have presented an image-based vehicle classification technique from vehicle images by using a supervised learning convolutional neural network. The vehicle image is taken to the network as an input and the network outputs the vehicle class to which the vehicle belongs. For the vehicle classification, we have established our own CNN architecture and compared these CNN results with furthermore machine learning algorithms like Naïve Bayes, SVM and Decision Tree. As the images are taken from Google, we contribute ownership protection to prove the ownership of the data. For this we insert a watermark in our dataset. Binary bits of URL of the images are used as the secret information of ownership. Every image used in our project may lead towards the path where it has been taken from. One limitation of current work is that we trained and tested on 32 bit and 64-bit datasets. In our work we have some hardware constraints. If we trained our model by using 256x256 bit dataset we may lead towards higher efficiency. For much better results GPU clusters are needed to train on large datasets and higher dimensions and deeper neural networks. If we gather more samples from different classes, we can improve our results and increase our scope. In future we are considering coping with all these hardware and software limitations for much better accuracy. Further work can be done by using vehicle images on poor lightning condition or bad weather situations.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number MoE-IF-20-07.

REFERENCES

[1] H.-J. Choand M.-T. Tseng, "A Support Vector Machine Approach to Cmos-Based Radar Signal Processing for Vehicle Classification and Speed Estimation," *Mathematical and Computer Modelling*, vol. 58, no.1-2, pp. 438–448 , 2013.

[2] W.-H. Lin, J.Dahlgren,and H. Huo, "An enhancement to speed estimationusing single loop detectors," in *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*, vol. 1. IEEE, 10 2003,pp. 417–422.

[3] D. G. Lowe, "Distinctive image features from scale-invariant key points", *International journal of computer vision*,` vol.60,no.2,pp. 91–110 , 2004.

[4] Sobel, "Camera models and machine perception," Computer Science Department, Technion, Tech. Rep., 1972.

[5] H. Lai, G. S. Fung and N. H. Yung, "Vehicle type classification from visual-based dimension estimation," in ITSC 2001. 2001 *IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585)*. IEEE, 2001, pp. 16 201–206.

[6] S. Gupte,O. Masoud, R.F. Martin and N.P. Papanikolopoulos,"Detection and Classification of Vehicles," *IEEE Transactions on intelligent transportation systems*, vol.3,no.1,pp. 37–47 , 2002.

[7] J.-W.Hsieh, S.-H.Yu, Y.-S.Chen and W.-F. Hu, "Automatictraffic surveillancesystem forvehicletrackingand 20 classification,"*IEEE Transactions on Intelligent Transportation Systems*, vol.7,no.2,pp. 175–187 , 2006.

[8] Z. Zhang, T. Tan, K. Huang and Y. Wang, "Three-dimensional Deformable-model-based Localization and Recognition of Road Vehicles," *IEEE transactions on image processing*,vol.21,no.1,pp. 1–13 , 2011.

[9] Hussain, M. Hannan, A. Mohamed, H. Sanusi and A. Ariffin, "Vehicle crash analysis for airbag deployment decision," *International journal of automotive technology*, vol. 7, no. 2, pp. 179–185, 2006.

[10] Liu, H. Huo, T. Fang and D. Li, "Fault tolerant spatio-temporal fusion for moving vehicle classification in wireless sensor networks," *IET communications*, vol. 5, no. 4, pp. 434–442, 2011.

[11] Garcia and M. Delakis, "A neural architecture for fast and robust face detection," in *Object recognition supported by user interaction for service robots*, vol. 2. IEEE, 2002, pp. 44–47.

[12] Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[13] Karpathy,G.Toderici,S.Shetty,T.Leung, R.Sukthankar,and L.Fei-Fei, "Large-scalevideoclassificationwith convolutionalneural networks,"in*Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014,pp. 1725–1732.

[14] Y.Sun, X.Wang and X.Tang, "Deep Convolutional Network Cascade for Facial Point Detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3476–3483, 2013.

[15] N.Zhang, M.Paluri,M.Ranzato,T. Darrell,and L.Bourdev, "Panda: Pose aligned networksfordeep attribute modeling,"in*Proceedings of the IEEE conference on computer vision and pattern recognition* , 2014,pp.1637–1644.

[16] Y. Le Cun, Y. Bengio and G.Hinton, "Deep Learning," *Nature*, vol.521, no.7553, pp. 436–444, 2015.

[17] 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,no.4,pp. 2247–2256 , 2015.

[18] K. F. Hussain and G. S. Moussa, "On-road vehicle classification based on random neural network and bag-of-visual words," *Probability in the Engineering and Informational Sciences*, vol. 30, no. 3, pp. 403–412, 2016.

[19] Y. Zhou, H. Nejati,T.-T.Do, N.-M. Cheung,and L. Cheah, "Image-basedvehicle analysis using deep neuralnetwork:Asystematicstudy,"in*2016 IEEE International Conference on Digital Signal Processing (DSP)*. IEEE, 2016,pp. 276–280.

[20] S. Fazli, S.Mohammadi and M. Rahmani, "Neural Network Based Vehicle Classification for Intelligent Traffic Control," *International Journal of Software Engineering & Applications*,vol.3,no.3,p.17 , 2012.

[21] W. Wu, Z.Qi Sen and W. Mingjun, "A Method of Vehicle Classification Using Models and Neural Networks," in *IEEE VTS 53rd Vehicular Technology Conference, Spring 2001*, pp. 3022–3026, 2001.

[22] Z.Dong, M.Pei, Y.He,T.Liu,Y.Dong and Y.Jia, "Vehicle Type Classification Using Unsupervised Convolutional Neural Network," in*2014 22nd International Conference on Pattern Recognition*, pp. 172–177, 2014.M. A. Hannan, C. T. Gee and M. S. Javadi, "Automatic Vehicle Classification Using Fast Neural Network and Classical Neural Network for Traffic Monitoring," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol.23, no. Sup. 1, pp. 2031–2042, 2015.

[23] S. Wang, Z. Li, H. Zhang, Y. Ji and Y. Li,"Classifying Vehicles with Convolutional Neural Network and Feature Encoding," in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pp.18 784–787, 2016.

[24] Farid, A., Hussain, F., Khan, K., Shahzad, M., Khan, U. and Mahmood, Z., 2023. A Fast and Accurate Real-Time Vehicle Detection Method Using Deep Learning for Unconstrained Environments. *Applied Sciences*, 13(5), p.3059.

[25] Mittal, U. and Chawla, P., 2023. Vehicle detection and traffic density estimation using ensemble of deep learning models. *Multimedia Tools and Applications*, 82(7), pp.10397-10419.

[26] Abedi, H., Ma, M., He, J., Yu, J., Ansariyan, A. and Shaker, G., 2023. Deep Learning-Based In-Cabin Monitoring and Vehicle Safety System Using a 4D Imaging Radar Sensor. *IEEE Sensors Journal.*

[27] Ding Y, Qu Y, Sun J, Du D, Jiang Y, Zhang H. Long-Distance Multi-Vehicle Detection at Night Based on Gm-APD Lidar. Remote Sensing. 2022 Jul 24;14(15):3553.

[28] Song W, Li D, Sun S, Zhang L, Xin Y, Sung Y, Choi R. 2D&3DHNet for 3D object classification in LiDAR point cloud. Remote Sensing.

2022 Jun 30;14(13):3146.

[29] Chetouane A, Mabrouk S, Jemili I, Mosbah M. Vision-based vehicle detection for road traffic congestion classification. Concurrency and Computation: Practice and Experience. 2022 Mar 25;34(7):e5983.

[30] Afzalaghaeinaeini A, Seo J, Lee D, Lee H. Design of Dust-Filtering Algorithms for LiDAR Sensors Using Intensity and Range Information in Off-Road Vehicles. Sensors. 2022 May 27;22(11):4051.

[31] Wang H, Zhang X. Real-time vehicle detection and tracking using 3D LiDAR. Asian Journal of Control. 2022 May;24(3):1459-69.

[32] Tu C, Du S. A hierarchical RCNN for vehicle and vehicle license plate detection and recognition. International Journal of Electrical and Computer Engineering. 2022 Feb 1;12(1):731.

[33] Xiong, J., Bi, R., Tian, Y., Liu, X. and Wu, D., 2021. Toward lightweight, privacy-preserving cooperative object classification for connected autonomous vehicles. *IEEE Internet of Things Journal*, *9*(4), pp.2787-2801.

[34] M. Won, "Intelligent Traffic Monitoring Systems for Vehicle Classification: A Survey," *IEEEAccess*, vol. 8, no. 1, pp. 73340–73358 , 2020.

[35] S. Yu, Y. Wu, W. Li, Z. Song and W. Zeng, "A Model for Fine-Grained Vehicle Classification Based on Deep Learning," *Neurocomputing*, vol.257, no. 1, pp. 97–103 , 2017.

[36] Zhao, Y. Chen and L. Lv, "Deep Reinforcement Learning with Visual Attention for Vehicle Classification, " *IEEE Transactions on Cognitive and Developmenta lSystems*, vol.9, no.4, pp. 356–367 , 2016.

[37] M. Simoncini, L. Taccari, F. Sambo, L. Bravi, S. Salti and A. Lori, "Vehicle Classification from Low-Frequency GPS Data with Recurrent Neural Networks," *Transportation Research Part C: Emerging Technologies*, vol. 91, pp.176–191, 2018.

[38] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z.Si and J.Guo, "Fine-Grained Vehicle Classification with Channel Max Pooling Modified CNNs," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3224–3233, 2019.

[39] H. Ammar, A., Koubaa, A., Boulila, W., Benjdira, B. and Alhabashi, Y., 2023. A multi-stage deep-learning-based vehicle and license plate recognition system with real-time edge inference. *Sensors*, 23(4), p.2120.

[40] Javed, Abdul Rajawat, A.S., Goyal, S.B., Bhaladhare, P., Bedi, P., Verma, C., Florin-Emilian, Ț. and Candin, M.T., 2023, May. Real-Time Driver Sleepiness Detection and Classification Using Fusion Deep Learning Algorithm. *In Proceedings of International Conference on Recent Innovations in Computing: ICRIC 2022*, Volume 1 (pp. 447-457). Singapore: Springer Nature Singapore.

[41] Javed, Abdul Rehman, Muhammad Usman, Saif Ur Rehman, Mohib Ullah Khan, and Mohammad Sayad Haghighi. "Anomaly Detection in Automated Vehicles Using Multistage Attention-Based Convolutional Neural Network." IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 7, pp. 4291-4300, 2020.

[42] Joshi GP, Alenezi F, Thirumoorthy G, Dutta AK, You J. Ensemble of deep learning-based multimodal remote sensing image classification model on unmanned aerial vehicle networks. Mathematics. 2021 Nov 22;9(22):2984.

[43] Kyrkou C, Theocharides T. Deep-Learning-Based Aerial Image Classification for Emergency Response Applications Using Unmanned Aerial Vehicles. InCVPR Workshops 2019 Jun 20 (pp. 517-525).

[44] Asgarian Dehkordi, R.; Khosravi, H. Vehicle type recognition based on dimension estimation and bag of word classification. *J. AI Data Min*. 2020, 8, 427–438.

[45] Mazhar, T., Asif, R.N., Malik, M.A., Nadeem, M.A., Haq, I., Iqbal, M., Kamran, M. and Ashraf, S., 2023. Electric Vehicle Charging System in the Smart Grid Using Different Machine Learning Methods. *Sustainability*, 15(3), p.2603.