Hybrid RESNET and Regional Convolution Neural Network Framework for Accident Estimation in Smart Roads

Youcef Djenouri, *Member, IEEE*, Gautam Srivastava, *Senior Member, IEEE*, Djamel Djenouri, Asma Belhadi, and Jerry Chun-Wei Lin, *Senior Member, IEEE*

***Abstract*— Road safety is tackled and an intelligent deep learning framework is proposed in this work, which includes outlier detection, vehicle detection, and accident estimation. The road state is first collected, while an intelligent filter, based on SIFT extractor and a Chinese restaurant process is used to remove noise. The extended region-based convolution neural network is then applied to identify the closest vehicles to the given driver. The residual network will benefit from the vehicle detection process to make a binary classification on whether the current road state might cause an accident or not. Finally, we propose a novel optimization model for optimizing hyper-parameters in deep learning methodologies by using evolutionary computation. The proposed solution has been tested using benchmark vehicle detection and accident estimation datasets. The results are very promising and show superiority over many current state-of-the-art solutions in terms of runtime and accuracy, where the proposed solution has more than 5% of improved accident estimation rate compared to the conventional methods.**

***Index Terms*— Deep learning, vehicle detection, accident estimation, region convolution neural network, outlier detection, smart roads.**

I. Introduction

MODERN technologies such as wireless sensing, the Internet of Things (IoT), and Machine Learning (ML) are revolutionizing traffic management policies and making our roads and cities smart [1]. Many applications benefited from this revolution, i.e., road safety [2]–[4]. The ultimate goal of such an application is to propose intelligent systems to assess accidents before they occur. Deep learning (DL) is a trend that could help achieve this goal. Some solutions based on DL are already proposed [5]–[7], but they are not mature for real-world deployment due to their low accuracy. Dealing with this problem is the subject herein, where we give an end-to-end hybrid DL methodology for accident estimation while targeting high accuracy and reasonable runtime.

A. Motivation

Hybrid deep learning and analytics [8], [9] is a hot topic in intelligent transportation applications such as group anomaly detection, object detection, and accident estimation. Vehicle detection is the task of retrieving the cars in a given urban road scene [10], [11]. Vehicle detection can be very useful for accident estimation, where the detected closer vehicles of the given car in the current road scene might be beneficial for predicting whether the current road scene caused the accident or not. Motivated by the success of object detection and accident estimation models in accurately detecting the various objects and estimating the accident, this paper presents an end-to-end framework for accident estimation based on the detected closer vehicles of the given car.

B. Contributions

We developed HR2CNN (Hybrid RESNET and Convolution Neural Network for Accident Estimation) in this work, an intelligent hybrid framework for accident estimation. The framework uses several tasks, including outlier detection, vehicle detection, and accident estimation. First, road states are collected, using an intelligent filter based on SIFT extractor and Chinese restaurant process to remove noise. The enhanced convolutional neural network is then used to identify the closer vehicles of each driver. The rest of the network benefits from vehicle detection to classify whether the current road condition could cause an accident or not. Finally, we implement a novel optimization model with hyperparameters using evolutionary computation that can be used for parameter tuning of an indicated deep learning methodology. The proposed framework, HR2CNN (Hybrid RESNET and Convolution Neural Network for Accident Estimation), makes the following contributions.

1) A novel filtering algorithm based on SIFT extractor and Chinese restaurant process used to remove noise from the image database.

Manuscript received 19 July 2021; revised 23 November 2021 and 10 January 2022; accepted 30 March 2022. Date of publication 14 April 2022; date of current version 5 December 2022. This work was supported in part by the National Centre for Research and Development through the Project Automated Guided Vehicles integrated with Collaborative Robots for Smart Industry Perspective under Contract NOR/POLNOR/CoBotAGV/0027/2019-00. The Associate Editor for this article was H. Lu. *(Corresponding author: Jerry Chun-Wei Lin.)*

Youcef Djenouri is with SINTEF Digital, 0314 Oslo, Norway.

Gautam Srivastava is with the Department of Mathematics and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada, and also with the Research Centre for Interneural Computing, China Medical University, Taichung 40402, Taiwan.

Djamel Djenouri is with CSRC, Department of Computer Science and Creative Technologies, University of the West of England, Bristol BS16 1QY, U.K.

Asma Belhadi is with the Department of Technology, Kristiania University College, 0107 Oslo, Norway.

Jerry Chun-Wei Lin is with the Department of Computing, Mathematics and Physics, Western Norway University of Applied Sciences, 5063 Bergen, Norway (e-mail: jerrylin@ieee.org).

Digital Object Identifier 10.1109/TITS.2022.3165156

This is the author accepted version of an article published in
IEEE Transactions on Intelligent Transportation Systems (volume 23, issue 12, December 2022)
https://doi.org/10.1109/TITS.2022.3165156

2) An extended vehicle detection algorithm based on region convolution neural network is presented. The algorithm employs hard negative exploration and multi-scale training. In addition, a residual blocking model is used to estimate the likelihood of road states causing accidents.
3) An intelligent evolutionary computation algorithm to accurately explore the hyper-parameters space for optimal intrinsic parameters is designed.
4) HR2CNN is evaluated through extensive experiments on well-known data for vehicle detection and accident estimation. The results show that our methodology performs better than baseline algorithms when looking at accuracy, as well as runtime performance.

Section II gives an in-depth review of the main works for vehicle detection and accident estimation, followed by details of HR2CNN in Section III. Next, Section IV evaluates our framework. Lastly, Section V gives some brief concluding remarks.

II. Related Work

Hybrid deep learning has recently been used in different topics related to applications of intelligent transportation such as group anomaly detection, object detection, and accident estimation [8]. Vehicle detection is a fundamental component essential in most applications. It can be defined as the task of retrieving the cars in a given urban road scene [10], [11]. Detecting close vehicles in a road scene might be beneficial for predicting if the current scene will cause an accident. Mostofa *et al.* [12] proposed an up-scaling generative adversarial network for identifying multi-scale vehicles by learning the hierarchical features. Wu *et al.* [13] suggested a transfer learning approach for multi-source vehicle detection while incorporating fine-tuning unsupervised learning for creating the ground-truth intelligent transportation data. Arabi *et al.* [14] presented an intelligent system for detecting construction vehicles. It adopted a pre-trained approach called MobileNet for running the model on mobile and embedded devices. Tran and Tsai [15] used convolution operators to extract the urban traffic features in a streaming way for vehicle detection. Chetouane *et al.* [16] studied different object detection architectures for identifying good and relevant bounding boxes for vehicles by exploring the Gaussian mixture analysis with Kalman filter and optical flow strategy.

Chen *et al.* [17] considered lightweight detection where they developed a solution based on a deep convolution neural network, which reduces the memory footprint. Three new guidelines were also developed to find the optimal number of group convolution operators. Yang *et al.* [18] developed a solution based on a faster region convolution neural network to simultaneously detect both 2D and 3D vehicles from a single scene. It is a multi-task solution that also integrates orientation estimation and key point detection into a generic deep convolution neural network. Hassaballah *et al.* [19] proposed a vehicle detection method that handles emergence restriction in-camera functionalities, e.g., weather conditions. It is a multi-scale based convolution Gaussian network that combines Gaussian mixture probability with the convolution neural network. Fan *et al.* [20] investigated the use of approximate joint training to learn the vehicle detection model using faster region convolution neural network architecture [21]. Wang *et al.* [22] developed a weighted ensemble learning method which combines the region convolution neural network [23] and the "you only look once" [24] algorithms. Chen *et al.* [25] developed the cascade pyramid region proposal convolution neural network, which learns from pseudo-images for vehicle detection. It also integrated hybrid learning using the sparse points and residual network.

Dinh *et al.* [26] transformed the vehicle bounding boxes to a binary map that is injected into the convolution neural network. The evolutionary algorithm is also integrated into the entire system to determine the correlation among the camera parameters of the different videos of vehicles. Kumar *et al.* [27] developed a deep learning network to identify vehicles captured in fisheye images. It explores different tasks such as depth estimation, visual odometry, semantic and motion segmentation, and vehicle lens soiling detection. In the same context, Rashed *et al.* [28] explored the oriented bounding box, the ellipse, and the generic polygon for determining vehicles in fisheye by designing a curvature adaptive perimeter sampling strategy for deriving the polygon vertices. Li *et al.* [29] proposed a convolution neural network based on light enhancement to increase the ability to identify cars at night. It also developed a generic system that is capable to convert daytime images to low-light images and then using the resulting images in the training phase as ground-truth data. Chen *et al.* [30] developed an intelligent agent solution based on an explainable reinforcement learning process for vehicle detection. The semantic bird-eye mask is first deduced using the sequential latent model. The trained model is combined with the reinforcement learning process to explain the learning outputs. Zhao *et al.* [31] adopted the single-shot multi-box detector to make the trade-off between the runtime and accuracy for detecting vehicles in real-time. Both the relevant features and receptive fields are merged to identify the candidate bounding boxes. The cascade detection strategy with the convolution neural network is deployed to strengthen the positioning capability of the model. Wang *et al.* [32] handled with 3D object detection and developed a voxel-based representation to process 3D shapes. It models the height and the number of points for each voxel as Gaussian distribution. The bounding box uncertainty is calculated with a multivariate Gaussian loss function. Li *et al.* [33] proposed the logistic regression solution to identify vehicles in the context of emergency lane congestion context. It also computes the impact of the convolution kernels in the convolution neural network for detecting different vehicles in the emergency lane.

The existing solutions for vehicle detection suffer from several drawbacks. The first one is the scalability in terms of accuracy and computational cost while handling real setting scenarios. The second one is that these solutions are incomplete, while only the detection model is learned, there is no end-to-end safety approach for an autonomous driving scenario. Motivated by the success of the recent object detection models in accurately capturing the different objects, this paper explores an intelligent and end-to-end framework for vehicle detection that can serve autonomous driving settings.

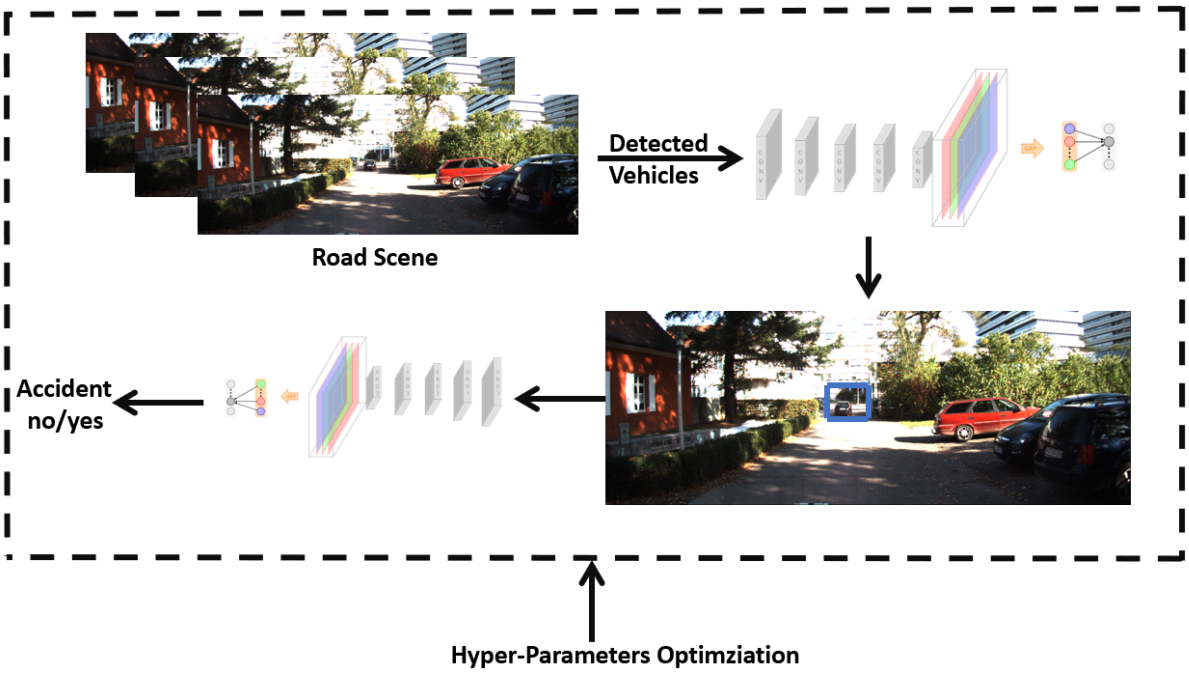

Fig. 1. HR2CNN framework.

III. HR2CNN

A. Framework Overview

A detailed explanation of the HR2CNN (Hybrid RESNET and Region Convolution Neural Network for Accident Estimation) framework is given in this part which is illustrated in Fig. 1. HR2CNN includes four stages: 1) Removing noise: This stage aims to remove noises before the image processing. In this context, the outlier detection algorithm is used to identify images that deviate from the whole input image. 2) Vehicle Detection: In this stage, an extension of the region convolution neural network architecture is made to accurately identify the closest vehicles of the given car. 3) Accident Estimation: After the vehicle detection process, the RESNET classifier is integrated into the entire system to estimate whether every configuration is likely to cause an accident or not. 4) Hyper-parameters Optimization: In this stage, we propose an intelligent mechanism to automatically identify the best hyper-parameters for all the previous stages of the HR2CNN framework. Notice the proposed framework requires a large number of tuned parameters.

B. Removing Noise

Images captured from sensors and cameras in the intelligent transportation environment have high resolution with an immense number of pixels. The number of pixels per image can be highly variables, from $250,000$ to more than $4,000,000$ pixels. Moreover, noise may be observed due to errors in data acquisition. Data analysis before processing is thus crucial, especially for mitigating the generation of a high number of regions proposals (which can exceed billion of regions). This makes the vehicle detection process costly in terms of time and memory. This stage has the goal of removing abnormal images from the whole image database.

We first consider the set of n images $I = \{I_1, I_2, \ldots, I_n\}$. The outlier detection task aims to remove images from I that deviate from the normal images. The SIFT (Scale-Invariant Feature Transform) extractor [34] is first used to figure out the relevant features of the images. The scale-space function, $S(I_i, \sigma)$, is calculated for each image I_i, and with a parameter threshold σ. This function is based on the Gaussian kernel, K, and defined as:

$$S(I_i, \sigma) = K(I_i, \sigma) \times I_i. \tag{1}$$

The spatial information of each candidate keypoint is then identified based on the interpolation procedure. The spatial interpolated information is computed, which allows the stability of the extracted features. The Taylor function $Y(I_i, \sigma)$ is defined by the interpolation function, which can be defined as:

$$Y(I_i, \sigma) = D + \frac{dY^T}{dI_i} I_i + 0.5 I^T \frac{d^2 Y}{dI_i^2} I_i, \tag{2}$$

where d_x and d^2 are the first and the second derivative functions on x.

The descriptor vector for the key points is then calculated by generating different orientation histograms of 4×4 pixel neighborhoods. After this procedure, the SIFT features for each and every I_i, say F_i, is determined. Furthermore, we propose a novel statistical analysis-based outlier detection algorithm for images. This algorithm adopts the Dirichlet process mixture model for identifying outliers from I. The set of all features $F = \{F_1, F_2, \ldots, F_{|F|}\}$ is transformed to k-dimensional space, where each point P_i is represented by $\{P_i, P_{i+1}, \ldots, P_{i+k}\}$. For instance, consider the features of ten different images, $F = \{F_1, F_2, \ldots, F_{10}\}$ with $k = 3$. Various points are generated, and each of which contains three different features. The first point contains the features $\{F_1, F_2, F_3\}$, the second

one is $\{F_2, F_3, F_4\}$, the third one is $\{F_3, F_4, F_5\}$, and so on until reaching the last point $\{F_8, F_9, F_{10}\}$. The features are then reduced to two dimensions using principal components analysis (PCA). The two derived dimensions will be entered into the Dirichlet process for finding noises. Highly correlated groups are determined using the Chinese restaurant process. The features are divided into groups, and every image is assigned to the previously created group if a distance between such image and the center of the groups exceeds a given threshold, otherwise, it is assigned to a new group. All images of the largest group are considered normal and used in the next process. The others are considered as noises and removed from the image database.

C. Vehicle Detection

The vehicle detection algorithm is executed to identify the closer vehicles from the driver. We adopt the granular region convolution neural network [35] for this purpose. It is appropriate for the uncertainty of the scenario of the vehicle, where vehicles can appear and disappear at any time. It incorporates a probabilistic model represented by the derivative knowledge. The transfer learning is also integrated by using the pre-trained models on the `ImageNet` data[1]. The main operations of the vehicle detection stage are:

1) **Bounding Boxes Creation:** The goal of this operation is to generate the bounding box candidates. The convolution neural network is explored to accurately find the bounding box candidates. The refinement process is then executed with the regression model.
2) **Hard Negative Exploration:** The purpose of this operation is to reduce the model error. The hard negative bounding boxes are re-processed using reinforcement learning. This operation enhances the detection ratio and eliminates the false-negative by learning unexpected behaviours of the drivers. Note that if the intersection over union between the generated bounding box, and the ground truth is 25% or less, then we can consider it as a hard negative.
3) **Multi Scale Training:** This operation aims to create bounding boxes with different sizes that simulate the real scenario of the vehicles. Thus, the vehicles and cars can have different lengths and widths. This could cause the regional convolutional neural network based vehicle detection models to be inaccurate. To address this, different types of bounding boxes are generated, and each type considers the bounding of boxes with the same height and the same width.

D. Accident Estimation

After detecting the closer car, the next step is to estimate whether the given state of the road could cause an accident. The RESNET classifier is used to perform binary classification, i.e., whether the current state may cause an accident or not. The input of the RESNET classifier is the current state of the road with the bounding box of each car, and the output is the class of this state, "0" for a non-accident, and "1" for an accident. The traditional deep neural architectures, e.g., AlexNet, and VGG perform the training in the entire layers. This reduces the performance of the model, particularly for deeper architecture such as VGG19. RESNET is developed to address this issue by integrating a new concept called "micro-architectures". We used 50 layers where each 2-layer block is connected by a bottleneck block which results in more than 3.8 billion parameters being optimized. The whole layers are organized in residual blocks that process different operators such as convolution, pooling, and batch normalization. Even though RESNET is much deeper than VGG16 and VGG19, this block partition-based allows to effectively learn the different weights of such model. The residual blocks also allow learning different patterns inside the data, which is missing in the state-of-the-art deep learning architecture.

E. Hyper-Parameters Optimization

Let $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{|P|}\}$ be the set of all parameters used by the HR2CNN framework, and let us denote by $D(\mathcal{P}_i)$, the domain space of the parameter $\mathcal{P}_i$, which contains the possible values of $\mathcal{P}_i$. The configuration space $\mathcal{C}$ is represented by the possible configurations, while each configuration is the set of values of all parameters in $\mathcal{P}$. Determining the optimal values of all parameters in $\mathcal{P}$ requires the exploration of all configurations in $\mathcal{C}$, which needs high computational and memory resources, particularly for those parameters having continuous values such as the error rate. Moreover, the number of all configurations is very immense as it is proportional to the number of all parameters and the domain values of each parameter. It is set to $\prod_{i=1}^{|P|} |D(\mathcal{P}_i)|$. For example, if we only consider $1,000$ different values for epoch parameter (varied epoch from 1 to $1,000$) 100 different value for error rate (varied error rate from 0.01 to 1.00), and $2,000$ different values for the number of bounding boxes (from 1 to $2,000$), the number of all configurations in $\mathcal{C}$ will be 20 million configurations. Therefore, the traditional enumeration-based methods including branch and bound [36], and A* [37] will be bluntly blocked when processing the above use case. To solve this problem, we propose an efficient evolutionary computation-based method for exploring the configurations in $\mathcal{C}$. The elementary operations of the suggested approach are defined in the following:

1) **Population Initialization:** Any evolutionary computation algorithm is based on the initial population represented by the set of individuals. Each individual is defined by the possible values of each parameter in $\mathcal{P}$. For instance, if we consider the previous example, $(20, 0.75, 740)$ is a solution that represents the configuration where epochs are set to 20, the error rate is set to 0.75, and the number of bounding boxes is set to 740. The population of the proposed evolutionary computation algorithm should have a fixed size, i.e., the same number of individuals. For a better exploration of the configuration space, the individuals in one population should be heterogeneous. To ensure such diversity, the

[1] http://www.image-net.org/

initial population is created by generating individuals while maximizing the distance among them. Thus, the distance between two solutions (individuals) S_1, and S_2, is defined as follows:

$$D(S_1, S_2) = \sum_{i=1}^{|\mathcal{P}|} |S_1^i - S_2^i| \quad (3)$$

Note that S_1^i and S_2^i are the i^{th} value of the solutions S_1, and S_2, respectively. $|\mathcal{P}|$ is the population size.

2) **Crossover:** Crossover operator allows to intensively explore one region in the configuration space. The following crossover operator is applied in pair of individuals of the current population: The crossover point is randomly selected from 1 to $|\mathcal{P}|$ which allows dividing every individual into two parts, *left side*, and *right side*. The first child takes the left side of the first individual and the right side of the second individual, while the second child takes the right side of the first individual and the left side of the second individual. For instance, if we consider two individuals, $(20, 0.75, 740)$ and $(10, 0.65, 820)$, and if the crossover point is set to 2, two other individuals are generated, the first one is $(20, 0.75, 820)$, and the second one is $(10, 0.65, 740)$.
3) **Mutation:** The mutation operator allows for the diversification process, i.e., it generates individuals far from the current region. A given parameter of each individual is randomly selected and updated. For instance, consider the following individual generated by the crossover operator $(20, 0.75, 820)$. The mutation operator generates the following individual $(15, 0.75, 125)$ by updating the first and the third elements and keeping the second element as it is.

First, the initial population is randomly generated and each individual is created based on the population initialization. The crossover and mutation operators are then applied for exploring the configuration space. To maintain consistent population size, each and every individual is evaluated making use of the object detection and classification accuracy. The best individual is kept and the others are removed. This process is repeated in multiple iterations until the max number of iterations is reached.

Algorithm 1 presents the pseudo-code of the designed HR2CNN. The input data is the set of n road scenes in the training and the set of k new road scenes in the inference. The process starts by removing outliers and noises using the outlier detection step explained in Section III.B. The cleaned road scenes are trained in the object detection model as explained in Section III.C. The bounding boxes of the second step are trained to learn the accident estimation process as explained in Section III.D. As a result of the training phase, the weights of the two models, noted M_{VO}, and M_{AE} are adjusted. In the inference step, the propagation of weights of M_{VO}, and M_{AE} is performed for each new road scene, to predict whether it causes an outlier. We remark that the training phase, performed only once independently from the number of road scenes in the inference, is a high time-consuming task that includes several training models, and processing. However, the inference step contains only one loop and needs simple propagation of the learned models in the training phase.

Algorithm 1 HR2CNN Algorithm

Input: $R = \{R_1, R_2, \ldots, R_n\}$: the set of n road scenes in the training. $R_{new} = \{R_{new}^1, R_{new}^2, \ldots, R_{new}^k\}$: the set of k new road scenes in the inference.
Output: $P(R_{new})$: prediction of the new road scenes whether they cause accident or not.
**********Training**********
$R \leftarrow OutlierDetection(R)$;
$(M_{VO}, BB) \leftarrow VehicleDetection(R)$;
$M_{AE} \leftarrow AccidentEstimation(BB)$;
**********Inference**********
$P(R_{new} \leftarrow \emptyset$;
for $R_{new}^i \in R_{new}$ **do**
  $BB_{new}^i \leftarrow M_{VO}(R_{new}^i)$;
  $P(R_{new}) \leftarrow P(R_{new}) \cup M_{AE}(BB_{new}^i)$;
end for
return $P(R_{new})$.

IV. Performance Evaluation

To validate the proposed HR2CNN framework, intensive experiments have been carried out for outlier detection, vehicle detection, and accident estimation. Several datasets were used in this research as described in the following:

1) Kitti[2]: It is collected from cameras of 15 drivers and 30 different pedestrians walking around the Karlsruhe city. It is used for vehicle detection, image retrieval, and place recognition tasks.
2) BOXY vehicle data [38]: It is a large vehicle detection dataset with almost two million annotated vehicles for training and evaluating object detection methods for self-driving cars on freeways.
3) Car object detection data[3]: It contains cars and vehicles in all views and rotation. It is designed for the object detection task.
4) Safe drive dataset[4]: It is related to the safe drive product which is used by the people who ride a bike or drive a car. It provides a warning to the user when an accident may take place based on a particular scenario.
5) Accident dataset[5]: It contains the road status of 3,210 different cases, which illustrates whether the road state might have caused an accident or not.
6) Self-Driving Cars[6]: This data contains different road statuses for a given car in a self-driving environment.

To evaluate the proposed framework, the mAP (mean Average Precision) is used as metric of comparison. mAP is largely used to test outlier detection, object detection, and accident

[2]http://www.cvlibs.net/datasets/kitti/
[3]https://www.kaggle.com/sshikamaru/car-object-detection
[4]https://www.kaggle.com/satishtilwani/safe-drive-dataset
[5]https://www.kaggle.com/jerrinbright/accident
[6]https://www.kaggle.com/alincijov/self-driving-cars

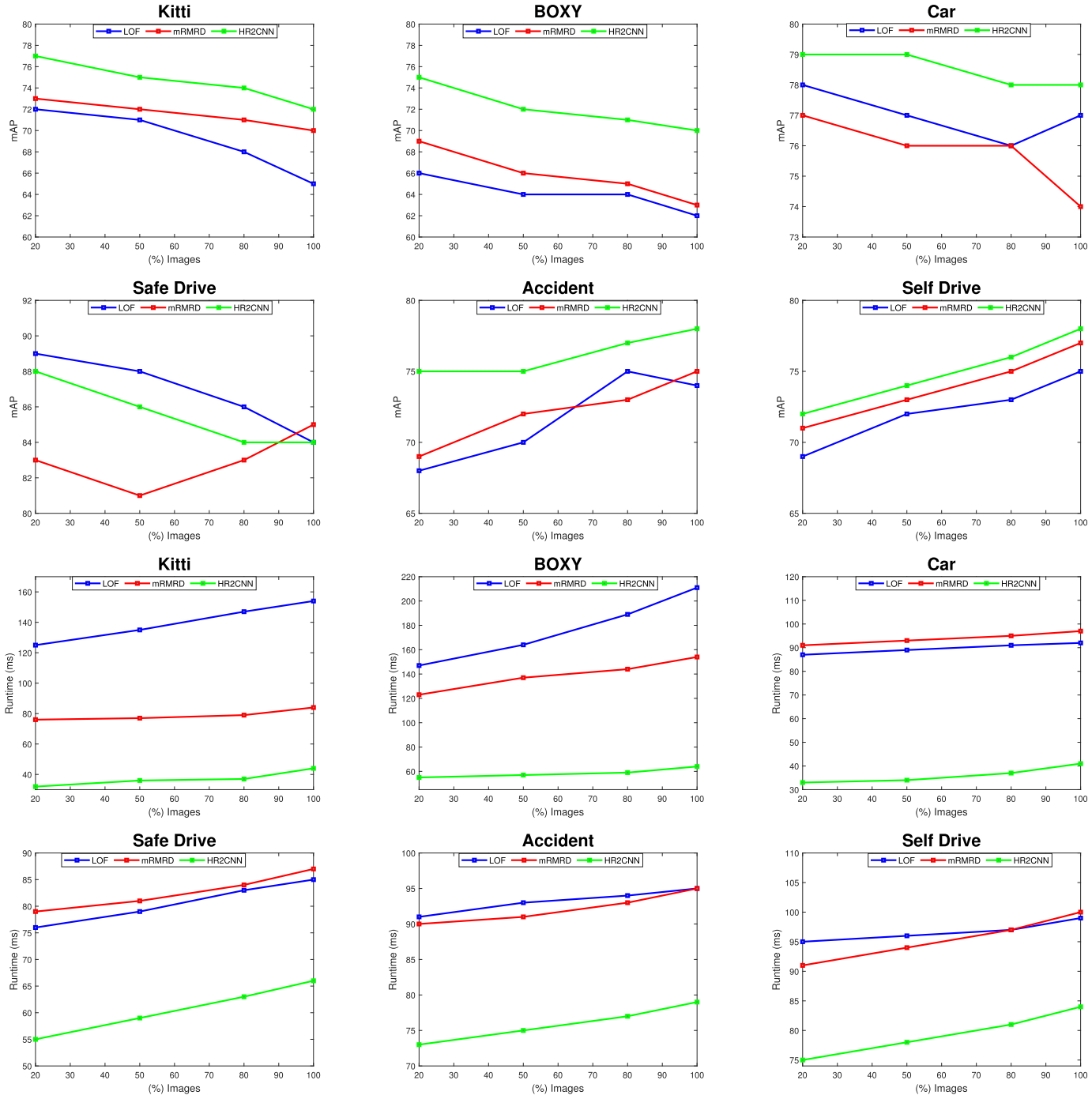

Fig. 2. Accuracy and runtime of outlier detection step of HR2CNN compared to the state-of-the-art outlier detection solutions.

estimation systems. It can be defined by:

$$mAP = \frac{\sum_{i=0}^{n} AvgP(i)}{n}, \tag{4}$$

where n is considered as the corrected objects handled among all objects, and $AvgP(i)$ is calculated as the precision results at i-rank. For outlier detection, the corrected objects handled are the corrected outliers. For vehicle detection, the corrected objects handled are the vehicle detected. For accident estimation, the corrected objects handled are the number of corrected accidents. The models are implemented on a machine fitted with an Intel-Core i7 processor and combined with NVIDIA GeForce GTX 1070 GPU. The HR2CNN is compared with recent outlier detection, vehicle detection, and accident estimation solutions under a varied number of images as input.

A. Outlier Detection Step

We compare the outlier detection step of the HR2CNN with the following baseline outlier detection solutions:

- LOF (Local Outlier Factor) [39]: It is a state-of-the-art outlier detection algorithm based on distance reachability.

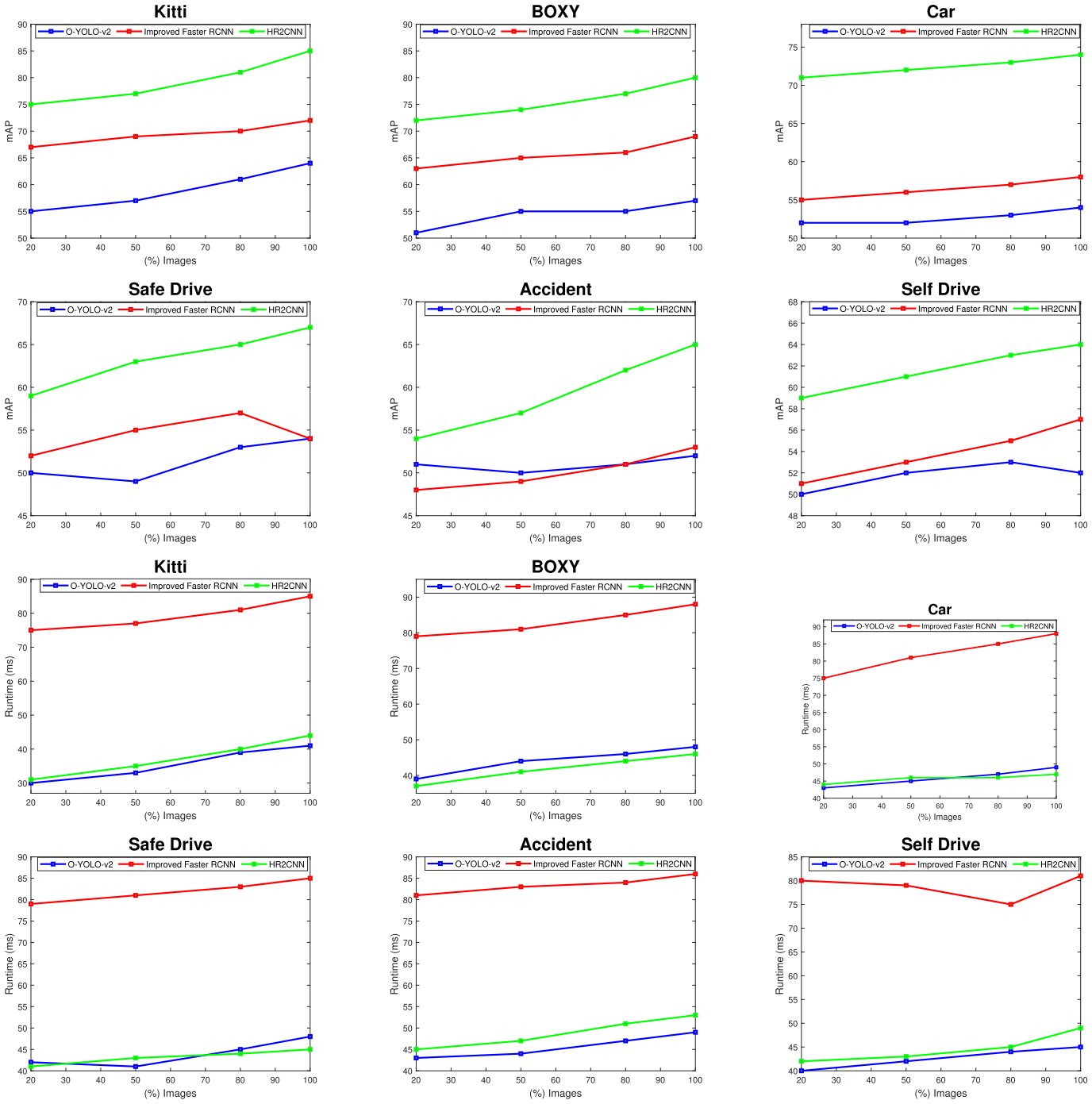

Fig. 3. Accuracy and runtime of vehicle detection step of HR2CNN compared to the state-of-the-art vehicle detection solutions.

It considers the set of images as point space, and the local outlier score is determined for each image. This score is based on neighbourhood computation. If the local outlier score is close to 1, then the image is considered as normal, and outlier otherwise.

- mRMRD (Minimum-Redundancy-Maximum-Relevance to-Density) [40]: This is a recent algorithm for outlier detection used for high dimensional data. It fits the image data when a high number of pixels is considered for each image. It is an unsupervised density-based subspace selection method, which first groups the set of features on several sub-spaces. Instead of calculating the outlier score on each data point, it is determined for each subspace.

Fig. 2 shows the accuracy and the runtime of the algorithms. The results reveal that the proposed solution outperforms the two baseline algorithms in terms of processing accuracy and processing runtime. The mAP of the HR2CNN reached 73% for handling 100% of the Kitti data, whereas the mAP of the mRMRD does not exceed 71%, and the mAP of the LOF algorithm is below 65% for dealing with the same configuration. Regarding the runtime, the HR2CNN does not

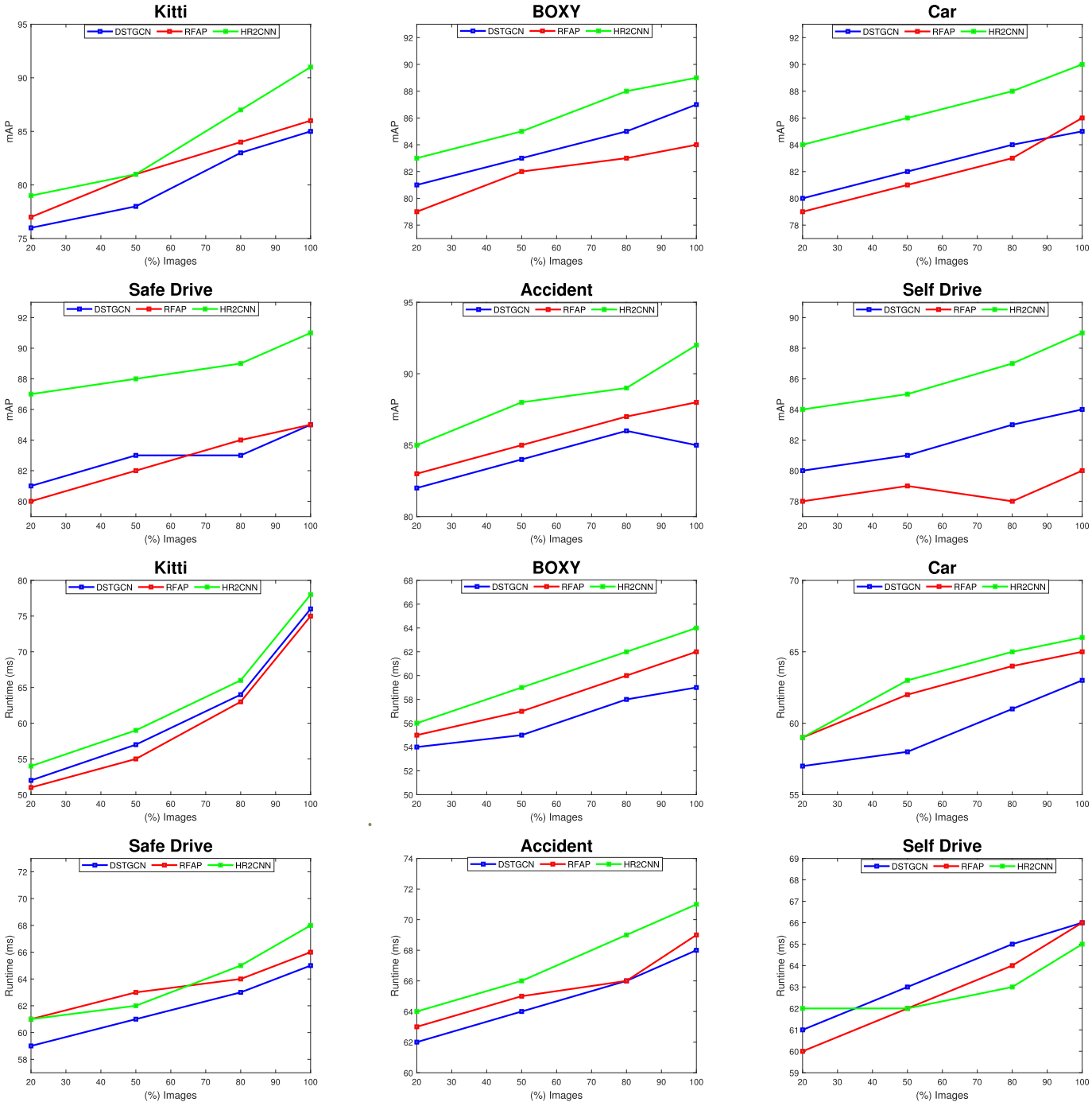

Fig. 4. Accuracy and runtime of accident estimation step of HR2CNN compared to the state-of-the-art accident estimation solutions.

exceed 50 ms for processing 100% of the Kitti data, whereas the runtime for the other algorithms reached 150 ms for handling the same configuration. These results are obtained thanks to the efficient strategy used in the outlier detection process which is based on an intelligent clustering process, whereas the other algorithms are based on traditional methods in determining the outliers.

B. Vehicle Detection Step

We compare the vehicle detection step of the HR2CNN framework, with the following baseline solutions:

- O-YOLO-v2 (Optimized You Only Look Once Version 2) [41]: It is based on the YOLO algorithm which is a state-of-the-art object detection algorithms. The residual blocks are added to the YOLO-v2 for solving the gradient dispersion issue. In addition, convolution layers are added at different locations for better feature extraction and addressing the accuracy of the YOLO-based methods.
- Improved Faster RCNN (Improved Faster Region Convolution Neural Network) [20]: It is an extended version of the Faster RCNN algorithm by proposing an approximate joint training strategy to learn the original images using

the Faster RCNN architecture. An improvement of the size and the proportion of anchors is also investigated.

Fig. 3 reveals that the proposed solution outperforms the two baseline algorithms in terms of accuracy and it is very competitive with O-YOLO-v2 in terms of processing runtime. The mAP of the HR2CNN reached 84% for handling 100% of the Kitti data, whereas the mAP for the other algorithms goes under 75% for dealing with the same configuration. Regarding the runtime, HR2CNN, and O-YOLO-v2 do not exceed 45 ms for processing 100% of the Kitti data, whereas the runtime for the improved RCNN algorithm reached 85 ms for handling the same configuration as input. These results are obtained thanks to the proposed operations during the vehicle detection process such as the bounding boxes creation, the hard negative exploration, and the multi-scale training. However, the other algorithms do not consider the size-varied of the objects which is common in intelligent transportation applications, and vehicle detection in particular.

C. Accident Estimation Step

We compare the accident estimation step of the HR2CNN framework, with the following baseline solutions:

- DSTGCN (Deep Spatio-Temporal Graph Convolutional Network) [2]: It is based on a graph convolution neural network and includes three steps. The first is the learning of the different correlations among the spatial information based on graph convolution operators. The second employs the standard convolution to learn both the spatial and temporal dimensions. The third step is the interpretation of the semantic representation of contextual knowledge by adding an embedded layer in the whole architecture.
- RFAP (Random Forest for Accident Prediction) [42]: It is based on the random forest which is composed of the set of the decision trees. Every tree gives a local decision on the current status of the road. An aggregation function is used to merge the local decisions of the trees and find the final prediction results on whether the current status of the road causes an accident or not.

The results reported in Fig. 4 reveal that the proposed solution outperforms the two baseline algorithms in terms of accuracy and that it is very competitive in terms of processing runtime. The mAP of the HR2CNN reached 91% for handling 100% of the Kitti data, whereas the mAP for the other algorithms goes under 86% for dealing the same configuration. The gap for the runtime between the HR2CNN, and the other algorithms does not exceed 3 ms for processing 100% of the Kitti data. The proposed framework needs more time compared to the two other algorithms because the HR2CNN is an end-to-end framework composed of three steps. However, the two other algorithms are based on a single pass, which has low accuracy, whereas HR2CNN integrates several steps, i.e, removing the noises, finding the vehicles closer to the given driver, and estimating the accidents from the detected cars.

The last experiments aim to compare the performance of the HR2CNN solutions versus advanced deep learning architectures. We adopted Xception and SqueezeNet for accident estimation. Table I shows both the runtime in mile seconds, and the accident estimation accuracy of the HR2CNN, Xception, and SqueezeNet using different sizes of the BOXY dataset. The results reveal the superiority of HR2CNN compared to the two other solutions in terms of accident estimation accuracy for all cases. However, there is a gap in terms of inference runtime which needs more investigation in the future to make the proposed framework suitable for deployment in mobile devices as the case of the SqueezeNet.

TABLE I
HR2CNN V.S. ADVANCED ACCIDENT ESTIMATION SOLUTIONS

BOXY X	HR2CNN		Xception		SqueezeNet	
	CPU(ms)	mAP	CPU(ms)	mAP	CPU(ms)	mAP
5	90	86	74	69	71	67
10	97	85	77	70	73	68
20	95	87	79	72	74	69

V. CONCLUSION

In this research, outlier detection is first performed to remove noise from the set of original urban traffic data images. A Hybrid SIFT extractor with a Chinese restaurant process is developed to efficiently filter the noise. The extended region convolution neural network is then launched to detect the closer vehicles of the given driver. Different improvements have been suggested on the region convolution neural network model including, bounding box creation, hard negative exploration, and multi-scale training. The RESNET algorithm was used for this purpose. Finally, and to better find the optimal parameter values of the proposed framework, an intelligent evolutionary computation algorithm was incorporated by developing diversification and intensification strategies for exploring the configuration space. Experimental evaluation was carried out to validate the applicability of the proposed framework using the well-known vehicle detection and accident estimation data. The results are strong and give reason to believe that the proposed model can outperform all baseline vehicle detection as given and the accident estimation solutions in many factors including detection, computational runtime, as well as estimation accuracy. In retrospect, we envision going through rigorous extensions of the proposed framework so that it can deal with large and big vehicle data in real-time by exploring high-performance computing tools, and it can also deal with technologies such as drone systems [43]. Handling 3D vehicle object data by exploring brain intelligence methods [44] is also on our future agenda.

REFERENCES

[1] M. A. Kafi, Y. Challal, D. Djenouri, M. Doudou, A. Bouabdallah, and N. Badache, "A study of wireless sensor networks for urban traffic monitoring: Applications and architectures," *Proc. Comput. Sci.*, vol. 19, pp. 617–626, Jan. 2013.

[2] L. Yu, B. Du, X. Hu, L. Sun, L. Han, and W. Lv, "Deep spatio-temporal graph convolutional network for traffic accident prediction," *Neurocomputing*, vol. 423, pp. 135–147, Jan. 2021.

[3] I. Kotseruba, A. Rasouli, and J. K. Tsotsos, "Benchmark for evaluating pedestrian action prediction," in *Proc. Winter Conf. Appl. Comput. Vis.*, 2021, pp. 1258–1268.

[4] S. Subramanian, D. Djenouri, G. Sindre, and I. Balasingham, "CoP4V: Context-based protocol for vehicle's safety in highways using wireless sensor networks," in *Proc. 6th Int. Conf. Inf. Technol., New Generat.*, 2009, pp. 613–618.

[5] B. Zhong, X. Pan, P. E. D. Love, J. Sun, and C. Tao, "Hazard analysis: A deep learning and text mining framework for accident prevention," *Adv. Eng. Informat.*, vol. 46, Oct. 2020, Art. no. 101152.

[6] B. Zhong, X. Pan, P. E. D. Love, L. Ding, and W. Fang, "Deep learning and network analysis: Classifying and visualizing accident narratives in construction," *Autom. Construct.*, vol. 113, May 2020, Art. no. 103089.

[7] J. Guerrero-Ibañez, J. Contreras-Castillo, and S. Zeadally, "Deep learning support for intelligent transportation systems," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 3, p. 4169, Mar. 2021.

[8] A. Belhadi, Y. Djenouri, G. Srivastava, D. Djenouri, A. Cano, and J. C.-W. Lin, "A two-phase anomaly detection model for secure intelligent transportation ride-hailing trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4496–4506, Jul. 2021.

[9] G. Srivastava, J. C.-W. Lin, A. Jolfaei, Y. Li, and Y. Djenouri, "Uncertain-driven analytics of sequence data in IoCV environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5403–5414, Aug. 2021.

[10] A. Bouguettaya, H. Zarzour, A. Kechida, and A. M. Taberkit, "Vehicle detection from UAV imagery with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 25, 2021, doi: 10.1109/TNNLS.2021.3080276.

[11] J. Zhang, X. Jia, J. Hu, and K. Tan, "Moving vehicle detection for remote sensing video surveillance with nonstationary satellite platform," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Mar. 17, 2021, doi: 10.1109/TPAMI.2021.3066696.

[12] M. Mostofa, S. N. Ferdous, B. S. Riggan, and N. M. Nasrabadi, "Joint-SRVDNet: Joint super resolution and vehicle detection network," *IEEE Access*, vol. 8, pp. 82306–82319, 2020.

[13] X. Wu, W. Li, D. Hong, J. Tian, R. Tao, and Q. Du, "Vehicle detection of multi-source remote sensing data using active fine-tuning network," *ISPRS J. Photogramm. Remote Sens.*, vol. 167, pp. 39–53, Sep. 2020.

[14] S. Arabi, A. Haghighat, and A. Sharma, "A deep-learning-based computer vision solution for construction vehicle detection," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 35, no. 7, pp. 753–767, Jul. 2020.

[15] V.-T. Tran and W.-H. Tsai, "Acoustic-based emergency vehicle detection using convolutional neural networks," *IEEE Access*, vol. 8, pp. 75702–75713, 2020.

[16] A. Chetouane, S. Mabrouk, I. Jemili, and M. Mosbah, "Vision-based vehicle detection for road traffic congestion classification," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 7, Mar. 2022.

[17] L. Chen, Q. Ding, Q. Zou, Z. Chen, and L. Li, "DenseLightNet: A lightweight vehicle detection network for autonomous driving," *IEEE Trans. Ind. Electron.*, vol. 67, no. 12, pp. 10600–10609, Dec. 2020.

[18] W. Yang, Z. Li, C. Wang, and J. Li, "A multi-task faster R-CNN method for 3D vehicle detection based on a single image," *Appl. Soft Comput.*, vol. 95, Oct. 2020, Art. no. 106533.

[19] M. Hassaballah, M. A. Kenk, K. Muhammad, and S. Minaee, "Vehicle detection and tracking in adverse weather using a deep learning framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4230–4242, Jul. 2021.

[20] J. Fan, T. Huo, X. Li, T. Qu, B. Gao, and H. Chen, "Covered vehicle detection in autonomous driving based on faster rcnn," in *Proc. Chin. Control Conf.*, 2020, pp. 7020–7025.

[21] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[22] H. Wang, Y. Yu, Y. Cai, X. Chen, L. Chen, and Y. Li, "Soft-weighted-average ensemble vehicle detection method based on single-stage and two-stage deep learning models," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 1, pp. 100–109, Mar. 2021.

[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. Conf. Comput. Vis. pattern Recognit.*, Dec. 2016, pp. 779–788.

[25] G. Chen *et al.*, "Pseudo-image and sparse points: Vehicle detection with 2D LiDAR revisited by deep learning-based methods," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 12, pp. 7699–7711, Dec. 2021.

[26] V. Quang Dinh, F. Munir, S. Azam, K.-C. Yow, and M. Jeon, "Transfer learning for vehicle detection using two cameras with different focal lengths," *Inf. Sci.*, vol. 514, pp. 71–87, Apr. 2020.

[27] V. Ravi Kumar *et al.*, "OmniDet: Surround view cameras based multi-task visual perception network for autonomous driving," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2830–2837, Apr. 2021.

[28] H. Rashed *et al.*, "Generalized object detection on fisheye cameras for autonomous driving: Dataset, representations and baseline," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 2272–2280.

[29] G. Li, Y. Yang, X. Qu, D. Cao, and K. Li, "A deep learning based image enhancement approach for autonomous driving at night," *Knowl.-Based Syst.*, vol. 213, Feb. 2021, Art. no. 106617.

[30] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, early access, Feb. 30, 2021, doi: 10.1109/TITS.2020.3046646.

[31] M. Zhao, Y. Zhong, D. Sun, and Y. Chen, "Accurate and efficient vehicle detection framework based on SSD algorithm," *IET Image Process.*, vol. 15, no. 13, pp. 3094–3104, Nov. 2021.

[32] G. Wang, J. Wu, T. Xu, and B. Tian, "3D vehicle detection with RSU LiDAR for autonomous mine," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 344–355, Jan. 2021.

[33] G. Li, Q. Wang, and C. Zuo, "Emergency lane vehicle detection and classification method based on logistic regression and a deep convolutional network," *Neural Comput. Appl.*, vol. 4, pp. 1–10, Sep. 2021.

[34] S. Gupta, K. Thakur, and M. Kumar, "2D-human face recognition using sift and surf descriptors of face's feature regions," *Vis. Comput.*, vol. 37, pp. 447–456, Mar. 2020.

[35] A. Pramanik, S. K. Pal, J. Maiti, and P. Mitra, "Granulated RCNN and multi-class deep SORT for multi-object detection and tracking," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 1, pp. 171–181, Feb. 2022.

[36] S. Coniglio, F. Furini, and P. San Segundo, "A new combinatorial branch-and-bound algorithm for the knapsack problem with conflicts," *Eur. J. Oper. Res.*, vol. 289, no. 2, pp. 435–455, Mar. 2021.

[37] Z. Bu and R. E. Korf, "A*+BFHS: A hybrid heuristic search algorithm," 2021, *arXiv:2103.12701*.

[38] K. Behrendt, "Boxy vehicle detection in large images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1–5.

[39] A. Tomlinson, J. Bryans, and S. A. Shaikh, "Using internal context to detect automotive controller area network attacks," *Comput. Electr. Eng.*, vol. 91, May 2021, Art. no. 107048.

[40] M. Riahi-Madvar, A. Akbari Azirani, B. Nasersharif, and B. Raahemi, "A new density-based subspace selection method using mutual information for high dimensional outlier detection," *Knowl.-Based Syst.*, vol. 216, Mar. 2021, Art. no. 106733.

[41] X. Han, J. Chang, and K. Wang, "Real-time object detection based on YOLO-V2 for tiny vehicle object," *Proc. Comput. Sci.*, vol. 183, pp. 61–72, Jan. 2021.

[42] X. Zhou, P. Lu, Z. Zheng, D. Tolliver, and A. Keramati, "Accident prediction accuracy assessment for highway-rail grade crossings using random forest algorithm compared with decision tree," *Rel. Eng. Syst. Saf.*, vol. 200, Aug. 2020, Art. no. 106931.

[43] H. Lu, Y. Li, S. Mu, D. Wang, H. Kim, and S. Serikawa, "Motor anomaly detection for unmanned aerial vehicles using reinforcement learning," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2315–2322, Aug. 2018.

[44] H. Lu, Y. Li, M. Chen, H. Kim, and S. Serikawa, "Brain intelligence: Go beyond artificial intelligence," *Mobile Netw. Appl.*, vol. 23, no. 2, pp. 368–375, Apr. 2018.