Ojala *et al. Journal of Big Data* (2022) 9:28
https://doi.org/10.1186/s40537-022-00581-8

Journal of Big Data

RESEARCH Open Access

Motion detection and classification: ultra-fast road user detection

Risto Ojala*, Jari Vepsäläinen and Kari Tammi

*Correspondence:
risto.j.ojala@aalto.fi
Department of Mechanical Engineering, Aalto University, Espoo, Finland

Abstract

With the emerge of intelligent and connected transportation systems, driver perception and on-board safety systems could be extended with roadside camera units. Computer vision can be utilised to detect road users, conveying their presence to vehicles that cannot perceive them. However, accurate object detection algorithms are typically computationally heavy, depending on delay-prone cloud computation or expensive local hardware. Similar problems are faced in many intelligent transportation applications, in which road users are detected with a roadside camera. We propose utilising Motion Detection and Classification (MoDeCla) for road user detection. The approach is computationally lightweight and capable of running in real-time on an inexpensive single-board computer. To validate the applicability of MoDeCla in intelligent transportation applications, a detection benchmark was carried out on manually labelled data gathered from surveillance cameras overseeing urban areas in Espoo, Finland. Separate datasets were gathered during winter and summer, enabling comparison of the detectors in significantly different weather conditions. Compared to state-of-the-art object detectors, MoDeCla performed detection an order of magnitude faster, yet achieved similar accuracy. The most impactful deficiency of MoDeCla was errors in bounding box placement. Car headlights and long dark shadows were found especially difficult for the motion detection, which caused incorrect bounding boxes. Future improvements are also required for separately detecting overlapping road users.

Keywords: Background subtraction, Convolutional neural networks, Intelligent transportation systems, Motion detection, Object detection, Winter conditions

Introduction

Motivation and background

Transportation is becoming increasingly connected, with revolutionary developments in road infrastructure and vehicle technology. Shared information about the locations and intentions of road users has the potential to improve traffic safety, efficiency and convenience. As road injuries are one of the most common causes of death [1], tremendous efforts are made worldwide to reduce the number of hazardous situations and collisions in traffic. Traditionally, vehicle safety systems have been limited to vehicle-mounted sensors and drivers have been constrained to perceive the environment from their perspective. Limited perception may lead to accident prone situations as other road users might be obscured by difficult weather conditions or obstacles. Driver perception can

be enhanced with sensors that are mounted to the road infrastructure, allowing them to receive information of road users behind corners and other obstacles. This type of *vehicle-to-infrastructure* (V2I) communication assists drivers and autonomous vehicles in traffic intersections, where the risk of collision is high and unexpected events may occur.

According to statistics from the US and EU, traffic intersections are some of the most hazardous areas of the road network. One fifth of all fatal vehicle accidents in the US occur in intersections [2]. EU has reported a similar figure for the junctions of its member countries [3]. Urban intersections are particularly hazardous due to occlusion caused by buildings, bus stops, advertisements, and flora. Limited perception is dangerous since pedestrians, cyclists or other vehicles may appear seemingly out of nowhere. We have recently developed a V2I system on Aalto University campus that transmits the locations of occluded road users to a connected research vehicle when approaching an intersection [4]. The system is based on a roadside camera unit, computer vision, and 4G LTE communication. The roadside camera unit utilises computer vision algorithms to detect road users in the video feed. Localisation of the detected road users is carried out with a monovision measurement approach, and a stereovision camera system is therefore not required [4]. The road user type and location is then transmitted to the research vehicle via 4G LTE.

Similar intelligent transportation system (ITS) applications which rely on road user detection from roadside cameras are abundant in the literature [5]. Real-time operation and low cost of roadside computer vision implementations is essential for adoption of the technology, as noted in the review of Buch et al. [5] and in the study of Atev et al. [6]. However, accurate real-time object detection typically requires extensive computing power. This translates into expensive roadside units, or reliance on cloud computing. Cloud computation can introduce notable delay into an ITS system, rendering real-time operation impossible. Thus, local computation and minimal computational load are necessary to achieve real-time operation and low unit cost. The requirements and price of the vision-based roadside units can be reduced by introducing a computationally light and reliable detection algorithm. We demonstrate the feasibility of applying an ultra-fast road user detector which consists of a motion detection algorithm and a *convolutional neural network* (CNN) classifier. Detecting road users with this approach is extremely lightweight, and can be run in real-time, over 35 *frames per second* (fps), on a 100 $ Jetson Nano [7] *single-board computer* (SBC) which is ideal for embedded systems. The accuracy of the detector is validated by comparing it to well-known state-of-the-art object detectors.

Scientific contributions

We propose a novel approach *Motion Detection and Classification* (MoDeCla) for road user detection. MoDeCla utilises motion detection to acquire bounding box proposals, which are classified with a CNN. The classification CNN is easily scalable for detecting practically any types of road users, whereas many traditional methods have been specifically designed for detecting instances of a certain type of road user, such as cars or pedestrians. We seek here to build on the work of Kim et al. [8], who have utilised similar methodology to detect people in surveillance footage. MoDeCla extends the detection framework with a compact and efficient state-of-the-art CNN classifier, instead

of the basic CNN classifier used in [8]. We show that MoDeCla can run in real-time at 35 fps on a single-board computer, whereas the previous study [8] reports a notably slower detection speed of 15 fps on a full desktop PC. Furthermore, MoDeCla is rigorously validated with extensive experimentation in varying conditions and comparison to other state-of-the-art detectors. The results show that our approach achieves similar accuracy to the existing state-of-the-art detectors with approximately 20 times faster computation. This is due to MoDeCla applying a notably smaller CNN than state-of-the-art detectors, which is possible due to the motion detection preprocessing. We demonstrate that the algorithm can perform detection in real-time on low-cost hardware, enabling a multitude of different ITS applications to be implemented on SBCs. Low-cost hardware translates into low unit costs which is critical for roadside installations, and as a lightweight algorithm MoDeCla can be run locally, eliminating delays caused by data transmission. This makes MoDeCla ideal for numerous ITS applications utilising roadside cameras, including the initially presented V2I intersection safety usecase.

State of the art

Reviewing the existing literature, camera-based technologies are an active area of research for different ITS applications [5]. Surveillance and traffic cameras have been applied in a multitude of tasks, ranging from safety usecases to traffic monitoring. Vehicle collision prediction has been studied by Atev et al. [6], who utilised roadside cameras to detect vehicles and estimate their trajectories. Lately, safety of vulnerable road users such as pedestrians has drawn a great deal of attention as well. Zhang et al. [9] have created trajectory based models for mixed traffic scenarios, which were capable of detecting hazardous occurrences in video feed. Similar studies regarding detection of dangerous or anomalous scenarios involving pedestrians in intersections and walkways have been conducted by Zhou et al. [10] and Pustokhina et al. [11], respectively. Many works focusing on traffic monitoring for advanced traffic control typically apply cameras as well. Neural network-based approaches have proven accurate for automatically extracting traffic information from camera views. Zhou et al. [12] and Zhang et al. [13] have applied neural network-based algorithms to extract a wide range of information of passing vehicles, such as as pose, size, speed, colour, and vehicle type. Neural networks have also been utilised to further process the traffic information acquired from images, as demonstrated by Sharma et al. [14] who applied neural networks for congestion prediction from highway camera data. However, there is still room for improvement when it comes to ensuring reliable operation of computer vision methods in traffic applications, as highlighted in the review of Buch et al. [5]. For maximal reliability of the technology, classification and detection capabilities of computer vision algorithms should be expanded to different road users and varying weather conditions.

Recently, traditional *image classification* methods such as haar-wavelets [15] and histograms of oriented gradients [16] have been surpassed by CNNs in terms of accuracy. Alexnet by Krizhevsky et al. [17] started the current trend by outperforming all previous records on the widely known ImageNet challenge. Building on top of the success, He et al. [18] developed the ResNet architecture which improved the classification accuracy greatly by introducing skip connections in the network. CNNs have traditionally been computationally expensive, yet modern GPU computation methods such as CUDA [19]

have enabled real-time inference. Previous research has also focused on lightweight classification CNN models. MobileNetV2 [20] and SqueezeNet [21] have reached impressive accuracies with minimal computational power.

In the field of *object detection*, CNNs are generally the state of the art as well. The previously presented CNN classifiers can be utilised as a base network that is combined with a detection back-end, such as a single shot multibox detector by Liu et al. [22], to perform object detection. Extensive research has also been on entire deep CNN architectures specifically developed for object detection. These can be divided to single-stage and multi-stage methods. Single-stage methods typically aim for faster speeds, where as multi-stage methods for greater accuracy. Well-known CNN object detection models representing these methods include single-stage YOLO by Redmon et al. [23–25] as well as Bochkovskiy et al. [26], and multi-stage R-CNN by Girshick et al. [27, 28]. Both detectors have accomplished state-of-the-art accuracy on the COCO object detection dataset [29]. All versions of YOLO are capable of running in real-time with modern high performance hardware, which is impressive considering typical inference times of deep CNN detectors. YOLOv3 also introduced multi-scale detection, which notably improved detection of small objects. Small object detection has been traditionally difficult for CNNs, and many recent detectors have commonly included multi-scale capabilities. Examples of such detectors are M2Det by Zhao et al. [30] and EfficientDet by Tan et al. [31], which have also scored state-of-the-art results on the COCO object detection benchmark. The presented CNN detectors are computationally demanding general object detectors applicable to any object detection task, including traffic applications.

With traffic being a common area to apply computer vision, less general methods have been developed to specifically answer the needs of ITSs. Typically this has involved detection of pedestrians or cars. CNNs specifically developed for these applications have recently delivered promising results. Du et al. [32] proposed a CNN architecture which fused detection and classification networks together for pedestrian detection, scoring state-of-the-art results on the Caltech pedestrian dataset [33]. Car detection has been studied in a similar manner by Wang et al. [34], who applied three CNNs back-to-back in their detector. Their approach was validated on the UA-DETRAC [35] car detection dataset, reaching excellent results. However, the accuracy of these detectors comes at the cost of computation, and both detectors achieved speeds of approximately 10 fps on a modern Nvidia Titan X graphics card. CNNs have also been applied for detecting pedestrians and cars by fusing pixel information with stereovision depth information in the study of Ferraz et al. [36]. Utilising depth information was noticed to positively impact the detection accuracy. The detection framework was not focused on real-time operation like the previous approaches, as the reported processing time was nearly half a second per image. The computation cost of CNNs typically remains greater than that of many traditional computer vision methods.

Traditional computer vision methods have been widely applied in traffic-specific computer vision. Extensive surveys on detection of pedestrians and vehicles have been provided by Benenson et al. [37] and Sivaraman et al. [38], respectively. Execution times of these detection algorithms have also been researched, with the goal of achieving real time operation with minimal resources. Benenson et al. [39] achieved the impressive results of 50 and 135 frames per second (fps) on their monovision and stereovision

pedestrian detectors, respectively. Benchmarking was performed on a desktop PC equipped with an Intel Core i7-870 CPU and an Nvidia GeForce GTX 470 GPU. The implementations were based on integral channel features and boosting decision trees, which were shown by Dollár et al. [40] to provide accurate results for pedestrian detection. Presented monovision modifications to the original algorithm included a single-scale detector and soft-cascading. Depth information from the stereovision camera was utilised to further improve detection speed by estimating the ground plane and applying a simplistic geometrical model for the visible environment.

Environment specific assumptions are often effective tools in computer vision, especially in road environments. When the assumption of a stationary camera and static background can be made, motion detection has been widely utilised for traffic-related object detection. Viola et al. [41] utilised motion information together with appearance features for pedestrian detection. In their approach, they obtained motion information by subtracting two consecutive images after applying different shifts to the latter image. A number of different rectangular filters were applied on the original image as well as images containing motion information, and the resulting features were classified with cascaded ADABoost [42]. Pedestrian detection is commonly utilised in ITS applications for analysing urban traffic, and thus the findings of Viola et al. provided an important resource for traffic authorities. For car detection in a highway environment, motion detection was applied in the study of Gupte et al. [43]. They accomplished motion detection by taking the difference image between the current and the previous frames. In their application, the camera view contained only the road, and therefore it was plausible to detect all moving objects passing certain dimensional thresholds as cars. Bai et al. [44] applied similar methods in their work to also detect cars on a highway. Their approach for motion detection was similar, yet they added to previous research by classifying acquired motion regions with haar-like features and cascaded ADABoost. Car detection in highway environments is commonly used for traffic counting and monitoring, offering a crucial source of data for traffic control applications. A motion detection-based multiclass detector for traffic environments has been proposed by Zhang et al. [45], who detected pedestrians, cyclists and cars from a traffic camera view. Their unsupervised learning based approach utilised clustering as well as Gaussian distributions of velocity and size of detected moving objects. Motion detection in their work was achieved with a Gaussian mixture model. Kim et al. [8] proposed applying Gaussian mixture model based motion detection in series with a CNN classifier for person detection in surveillance footage. The approach was tested on a self-gathered dataset of surveillance video, which yielded promising results for the accuracy and detection speed. However, the acquired accuracy results were not compared to other detectors evaluated on the same data. Furthermore, their approach was not significantly faster than the current state-of-the-art object detectors, reaching 15 fps on their Tesla M40 GPU based PC, performing similarly to YOLOv2.

Our approach, MoDeCla, extends the work of Kim et al. [8] by utilising a more advanced CNN classifier in the detection framework. The classification CNN is trained with data from public datasets on the internet, demonstrating that the approach can be generalised to any roadside camera. MoDeCla is meticulously benchmarked on video data from urban environments and the performance is

compared to state-of-the-art object detectors. Acquired results indicate that the proposed approach is nearly as accurate as state-of-the-art detectors, while performing detection tens of times faster on minimal embedded system hardware. Performed benchmark highlights that MoDeCla is capable of running at 35 fps on an inexpensive single-board computer, whereas the compared detectors would require remarkably bulkier and more expensive hardware to reach such processing speeds. Consequently, the proposed detector can be conveniently applied in many ITS applications, ranging from research to actual roadside infrastructure. Furthermore, the detector is scalable for detecting any types of road users, as the classification CNN can be trained to distinguish multiple classes of objects.

Methods

The road user detection algorithm proposed here consists of motion detection and classification, which are applied in series. The approach has been specifically tuned for ITS applications, with low computational cost and scalability in mind. In addition, multiple well-established image processing methods such as blur, erosion, and dilation are utilised to filter the contents of the images. Workflow of the algorithm is presented in Fig. 1, displaying the steps taken to reach detections for an image.

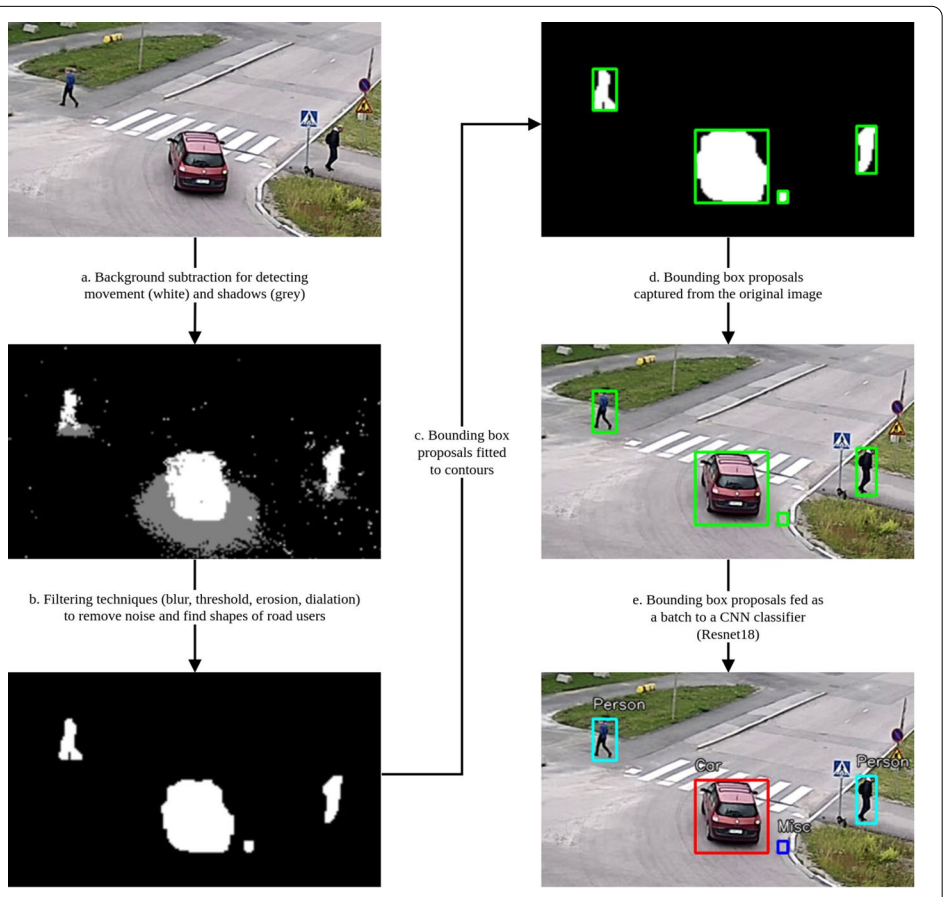

Fig. 1 MoDeCla workflow summarised step-by-step

When an 8-bit three channel RGB image is fed into the algorithm, the input image is first resized and processed with motion detection. The image is transformed into a grayscale image where each moving pixel is presented in white and shadows are presented in gray (a). Since the acquired grayscale image is typically noisy, the image is blurred, thresholded into a binary image, and finally eroded and dilated (b). Remaining areas after the filtering are fitted with bounding boxes (c). Using these bounding box coordinates, bounding box proposals are captured from the original RGB image (d). The acquired bounding box proposals are fed into a classification CNN, which gives the class of each bounding box proposal as an output (e). Consequently, the output of the MoDeCla algorithm is the image coordinates and types of the visible moving objects. Further details of the operation are provided in the following sections.

Bounding box proposals

Motion detection in MoDeCla is achieved by applying well-established Gaussian mixture model background subtraction [46]. This technique applies probabilistic methods to map each pixel into either background, foreground or shadow. Essentially, pixels that differ significantly from their previous values are categorised in the foreground, and can be interpreted to contain movement. The Gaussian mixture model is constructed utilising a defined number of the latest captured images. As new images are acquired, the model is continuously updated. The parameter values used in the model in this paper are presented in Table 1. These parameters affect the sensitivity of the background subtraction algorithm. The parameters were manually adjusted to such values that based on visual evaluation appeared to result in low noise as well as accurate representations of the shapes of moving road users. The history parameter, which determines how many previous frames are utilised for computing the Gaussian mixtures, was kept at the default value. However, the maximum number of mixtures was raised, as the increased complexity of the distributions allowed the algorithm to capture the background more accurately. Consequently, the threshold parameter which determines how similar a sample must be to belong to the background, was reduced as the background distributions were more accurate. Furthermore, as the scene was experiencing fairly significant changes caused by varying lighting and weather, the background ratio parameter was decreased. This parameter defines how rapidly changes in the scenery are incorporated into the background model. To improve detection speed, images were resized to 640x360 pixels prior to performing background subtraction.

As an output, the background subtraction provides an 8-bit grayscale image, in which foreground is provided in white with a value of 255, shadows with a value of 127 and background as 0. These probability values are smoothed by applying Gaussian blur with a 5x5 kernel and standard deviation of 1.1. All pixels with a value less than or equal to 127 are then thresholded, resulting in a binary image portraying movement. However, this binary image is typically noisy. The noise consists of a large number of detected areas of motion with only a few pixels in size, caused by slight changes in illumination or video compression. In addition, moving road users in the images are commonly only partially captured, consisting of multiple detected regions of movement. To remove the pixel sized areas of noise, the binary images are eroded once with a kernel of size 3x3. Afterwards, the images are dilated once with a 3x3 kernel to regain the pixels lost from

the larger areas during eroding, as well as uniting areas of motion in proximity of one another. Following these image processing steps, unified areas of motion are acquired by finding the remaining contours [47]. Areas smaller than 15 pixels are filtered away, and the rest are extracted as rectangular samples from the original RGB-image for classification.

Implementation of the methods presented above was performed with the OpenCV-Python library [48]. This library acts as an interface that executes C++ implementations of the algorithms, allowing reasonable benchmarking of the detection speed of MoDeCla.

Classification

The samples extracted from the original image act as bounding box proposals, which are classified in parallel as a single batch with a CNN. Before feeding the proposals to the CNN, each proposal is reshaped into a square by adding black bars to the sides. Subsequently, each proposal is resized to 48x48 pixels and pixel values are scaled between [0,1]. Finally, the RGB channels of the proposals are normalised with Z-score normalisation, after which the proposals are fully prepared for classification. The classification CNN evaluates each proposal, providing the probability of a proposal belonging to each class with a softmax layer. Theoretically any number of classes can be considered given the appropriate training data. In the presented experiments, only person, car and background classes were considered for benchmarking convenience.

Here, the classification was performed with the well-established CNN model ResNet18 [18]. This was due to the computationally light architecture, as well as the excellent classification accuracy of the network. The implementation of ResNet18 was acquired from the PyTorch library [49]. In order to optimise the inference time of the network, TensorRT [50] was utilised during benchmarking.

Training and testing

The ResNet18 model utilised here was trained with a classification dataset that contained images from multiple existing datasets. A total of three classes were included in the classification dataset: person, car and *miscellaneous background* (misc). Images of people were acquired from the PETA [51] and MIO-TCD [52] datasets. Car class was represented by randomly sampling images from the MIO-TCD car and van categories. Misc consisted of images from the MIO-TCD background class, as well as all classes from CIFAR100 [53] excluding classes listed under people or vehicles. CIFAR100 was included in the misc class since it provided a number of images containing different objects with varying shapes, which assisted the learning of ResNet18 in correct recognition of people and cars. In the complete classification dataset, each class was relatively equally presented in terms of number of samples, as detailed in Table 2 Throughout the training and all tests presented in this paper, the scaled channel statistics of the training set were used for performing Z-score normalisation on the images fed to the ResNet18. The scaled RGB channels of the training images had means of 0.4786, 0.4712, and 0.4665, and standard deviations of 0.2352, 0.2317 and 0.2367, respectively.

The ResNet18 was trained on the training set with stochastic gradient descent for 60 epochs with an initial learning rate of 0.1, and a step-based learning rate decay of 0.1

every fifteen epochs. A batch size of 128 was utilised, and a weight decay of $5{\cdot}10^{-4}$ was applied for regularisation.

During training, several common image augmentation methods were also applied to induce variance in the training batches. All images were first randomly added Gaussian blur with a chance of 0.8. The sigma value for the blur was a randomly chosen integer ranging from one to five. The resulting images were padded with one eight of their height and width, resized to squares with black bars, and then randomly cropped to 48x48 pixels. The saturation, brightness, contrast and hue were each randomly adjusted with a multiplier from 0.9 to 1.1 for each image. Images were also randomly flipped horizontally with a chance of 0.5. All the applied image augmentation methods were noted to positively impact the detection performance of MoDeCla.

The detection benchmarks presented here were performed on novel manually labelled datasets gathered from cameras overlooking urban roads at Aalto University campus in Espoo, Finland. Video data were recorded with a two surveillance cameras at 30 fps with a resolution of 1280x720 pixels. H264 encoding was applied in the video feeds, as is common with surveillance cameras. The first camera overlooked an urban traffic intersection, and the second camera monitored a road leading to the same intersection. To achieve maximal variance in captured video data, the cameras were used to record datasets during different times of year. With the first camera, a dataset was recorded during summertime, capturing video clips throughout the day at fixed ten minute intervals for approximately 3 weeks. The second camera was utilised to capture a dataset during wintertime, recording video clips in a similar manner for two days. These datasets gathered with the first camera and the second camera are here referred to as Summer benchmark set and Winter benchmark set, respectively. Captured video clips were 30 s long, and the 800th frame of each video was manually labelled. The labelled frames were used for evaluating accuracies of the detectors in the benchmark. Only a single frame was labelled to achieve maximal variance between benchmark samples. Sample images of the Summer benchmark set are provided in Fig. 2, and samples of the Winter benchmark set are

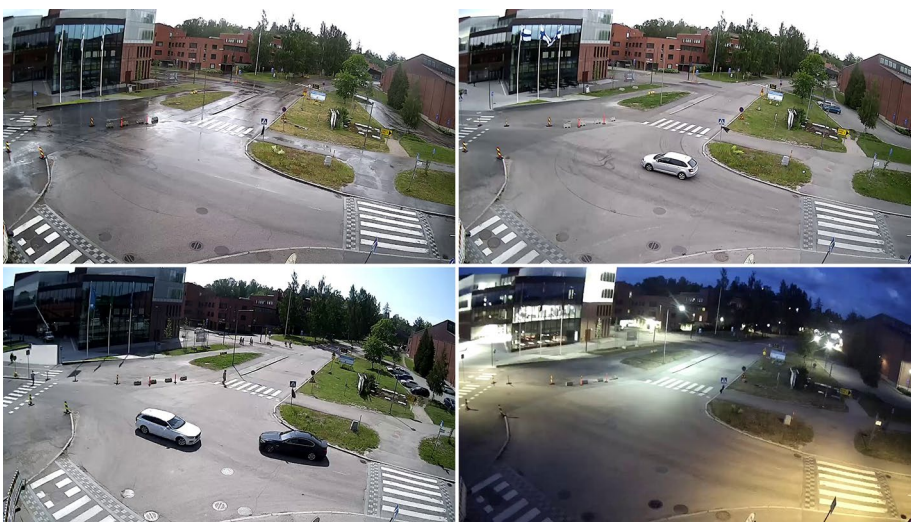

Fig. 2 Samples of the images in the Summer benchmark set, depicting the varying conditions

provided in Fig. 3. Samples of example detections made with MoDeCla on the recorded data are visualised in Figs. 4 and 5. Statistics for the collected benchmark sets are provided in Table 3. The Winter benchmark set has fewer images, since summer conditions were emphasised in the analysis due to the fact that these conditions are more common globally.

Instead of using public established datasets for benchmarking, novel datasets were required due to the characteristics of MoDeCla. Due to performing motion detection, MoDeCla requires a stationary camera, as well as video footage instead of individual images. No suitable existing datasets that would have included labelled pedestrians as well as cars were identified in the literature. MoDeCla is also only capable of detecting moving road users, and consequently the detection benchmark images do not contain stationary road users, excluding cars in parking areas. The coordinates of visible parking areas have been manually defined in the images, and do not contain any ground truth bounding boxes. Consequently, no detections made in these areas are recorded in the benchmark. These rules were justified based on the typical requirements of ITS applications, in which stationary road users and parking lots are inherently of low interest. Similar exclusion procedure was followed for detections and ground truths in the horizon of the Winter benchmark set images, as road users in the horizon were not clearly

Fig. 3 Samples of the images in the Winter benchmark set, including significant snowfall and typical dark winter conditions

Table 1 Parameter values used in Gaussian mixture model based background subtraction

Parameter	Value
History parameter	500
Threshold parameter	8
Maximum number of mixtures	10
Background ratio	0.8

Table 2 Number of images of each class in the training set of the ResNet18 classifier

Class	Number of Images
Person	22,736
Car	24,216
Misc	22,777
Total	69,729

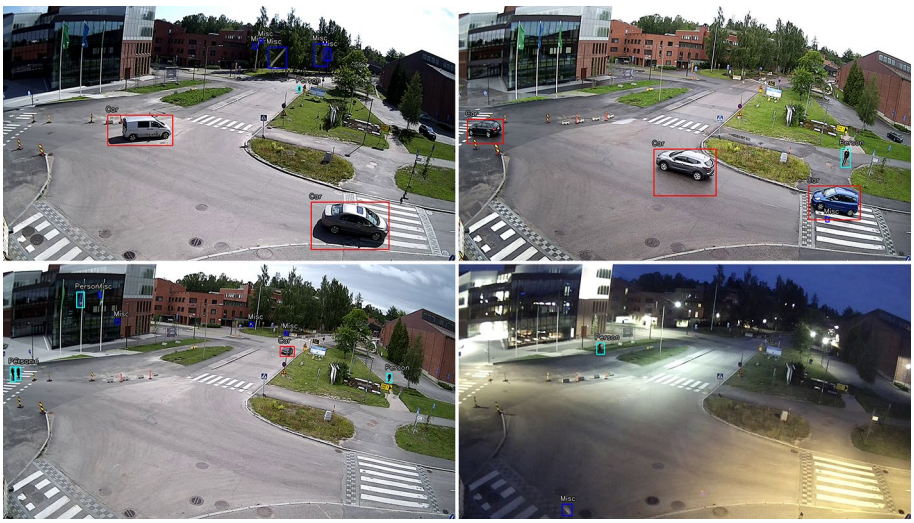

Fig. 4 Samples of detections made with MoDeCla in summer conditions

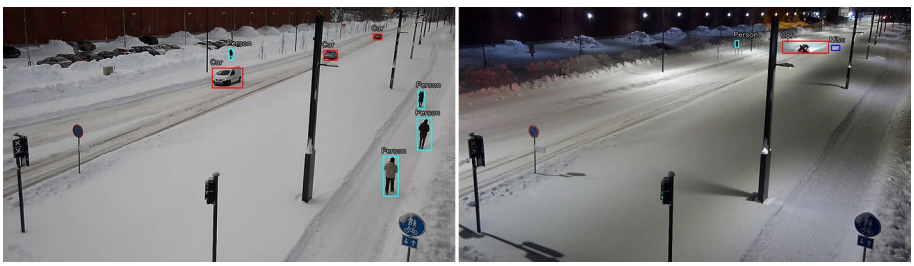

Fig. 5 Samples of detections made with MoDeCla in winter conditions

visible even to the human eye. Furthermore, the images in the datasets were chosen to only contain the road users analysed here: person and passenger car. This was to avoid false positives caused by road users not taught to the detectors, such as cyclists.

In addition to MoDeCla, several state-of-the-art CNN object detectors were benchmarked on the datasets for comparison. The detectors were chosen based on their frequent appearance in previous literature as well as open source availability. YOLOv4 [26], YOLOv3 and the lightweight TinyYOLOv3 [25], M2Det with VGG-16 backbone [30], and EfficientDet-D1 [31] were all evaluated on the data. YOLOv4, YOLOv3, TinyYOLOv3, M2Det and EfficientDet-D1 were benchmarked with input sizes of 608x608 pixels, 608x608 pixels, 416x416 pixels, 512x512 pixels, and 640x640 pixels, respectively. The detectors were applied with their original weights trained on the COCO dataset. The weights learned on the COCO dataset should fit the task of detecting people and cars well, as these are some of the most common objects in the dataset. However, COCO dataset contains objects other than person and car, and during testing it was noticed that the detectors misclassified passenger cars as trucks from time to time. This is likely due to COCO containing images of pick-up trucks which can resemble larger passenger cars. Since this classification mistake is not impactful in the presented application, truck

detections made by the detectors were translated to car detections to improve their results.

All the presented detectors were also benchmarked in terms of detection speed on an Nvidia Jetson Nano [7] SBC. TensorRT optimised models of YOLOv3 and TinyYOLOv3 [54], as well as YOLOv4 [55] were used in the benchmarking. Similarly, the ResNet18 PyTorch implementation of MoDeCla was optimised with TensorRT. For M2Det and EfficientDet-D1 TensorRT optimised models were not available, and therefore their detection speeds were benchmarked with regular PyTorch implementations [56, 57]. The Jetson Nano was available for roughly 100$ at the time of writing. The machine learning-oriented SBC contains a 128-core Maxwell GPU, ARM Cortex-A57 Quad-Core CPU and 4 GBs of LPDDR4 memory. In raw computational power, the unit has performance capabilities of 472 giga floating point operations per second. Benchmarking was conducted in maximum power mode with a power supply providing 5 V and 4 A, and with the GPU and CPU cores at their maximum frequencies of 921 MHz and 1.43 GHz, respectively. The Jetson Nano was chosen for the benchmark to highlight the possibilities of using the detectors in embedded road user detection solutions with local computation.

Accuracy metrics

Detection accuracies of the analysed detectors are reported as precision-recall curves as well as interpolated average precision (AP) and mean average precision (mAP) values. The *Intersection over Union* (*IoU*)-threshold was set at the common choice of 0.5. However, MoDeCla has an operation characteristic of grouping multiple overlapping road users into a single detection, due to the bounding box proposals being generated from unified contours of motion. This detection can at most be matched to only a single ground truth, which causes a significant decrease in precision and recall scores on the detection benchmark datasets, since people often tend to move in groups at the campus area. Grouping overlapping road users of the same type into a single detection is not necessarily considered disadvantageous for many applications of road user localisation, as overall the information of the presence and location of the road user type can be more essential than their exact number. Therefore, an application-specific update is presented for determining true and false positives when matching the detections to the ground truth labels. Nevertheless, in some applications distinguishing the exact number of road users might be necessary, and therefore results are also provided with the original metric. The precision-recall curves acquired with the Traditional matching procedure are here referred to as Traditional precision-recall. The results acquired with the proposed Cluster matching are referred to as Cluster precision-recall. This update allows matching multiple ground truths to a single detection, given that they pass certain conditions.

To define these conditions, the traditional definition of *IoU* is expanded here. The traditional definition computes an example detection d_e and an example ground truth g_e, which are nonempty sets of pixels, as

$$IoU(g_e, d_e) = \frac{|g_e \cap d_e|}{|g_e \cup d_e|}, \tag{1}$$

where intersection is denoted with the ∩-operator and union is denoted with the ∪ -operator. This definition is expanded to consider sets of ground truths by computing an example detection d_e and an example set of ground truths G_e as

$$IoU(G_e, d_e) = \frac{|(\bigcup G_e) \cap d_e|}{|(\bigcup G_e) \cup d_e|}, \tag{2}$$

where the $\bigcup$-operator denotes the union of all members of G_e.

Cluster matching follows Traditional matching by analysing all acquired detections in order of confidence, from highest to lowest. For an analysed detection, the goal is to match a subset of ground truths that have not yet been matched to any previous detections. Of the remaining ground truths, the one that has the maximum *IoU* with the detection is always included in the subset. Other remaining ground truths that have common area with the detection are added in the subset in such a way that the subset has maximal *IoU* with the detection. However, an added ground truth must not be the maximum *IoU* ground truth of a lower confidence detection, in case their *IoU* is above the selected *IoU*-threshold. True positives are marked for each of the matched ground truths if the *IoU* of the subset and the detection passes the *IoU*-threshold. Previously analysed detections and matched ground truths are always removed from the pool of available matching options.

Defining Cluster matching in mathematic notation, an analysed detection is denoted by d, the set of other remaining detections with lower confidence is marked as O. The set of remaining ground truths is denoted by G. For the other detections O, the single maximum *IoU* ground truths that pass the *IoU*-threshold t are mapped into set S as

$$S = \{\underset{g \in G}{\operatorname{argmax}}\, IoU(g, o) \mid o \in O \wedge \max_{g \in G} IoU(g, o) \geq t\}. \tag{3}$$

A set of ground truth candidates C is acquired for the detection, defined as

$$C = \{g \mid g \in G \wedge IoU(g, d) > 0 \wedge g \notin S\} \cup \underset{g \in G}{\operatorname{argmax}}\, IoU(g, d). \tag{4}$$

From this candidate set, the set of final matches M_{final} are chosen as

$$M_{final} = \underset{M \subseteq C}{\operatorname{argmax}}\, IoU(M, d). \tag{5}$$

If the *IoU* of the final matches and the detection surpasses the defined *IoU* threshold t, true positives are recorded for each of the members of the final matches, and the final matches are removed from the set of remaining ground truths. If the IoU threshold is not passed, the detection is marked as a false positive. In the end, the detection is discarded from the analysis, and the process is repeated for the next detection. Grouping overlapping road users into a single detection is not considered disadvantageous for the presented application of road user localisation The full Python-implementation of Cluster matching is open source and provided in the repositories linked to this paper.

Results

A Jetson Nano SBC was utilised to benchmark the detection speeds of MoDeCla and the other detectors included for comparison. These tests provide tangible information regarding cost-effective application of the detectors in widescale ITS employment. Furthermore, the detection accuracies of all the detectors were evaluated on the Summer benchmark and Winter benchmark sets. Traditional precision-recall curves as well as Cluster precision-recall curves were plotted for each of the detectors on both datasets. From these curves, the AP and mAP values were also computed for the detectors to allow for further quantitative analysis.

Detection speed

To analyse the applicability of the detectors in real ITS applications, the detection speeds of the detectors were clocked on a Jetson Nano. TensorRT optimised models of YOLOv4, YOLOv3 and TinyYOLOv3 were applied in the testing. The ResNet18 used for classification in MoDeCla was also TensorRT optimised with a maximum batch size of 100. TensorRT optimised models of M2Det and EfficientDet-D1 were not available, so their detection speeds were clocked with regular PyTorch models. All detectors were run with the CNNs using 32-bit floating-point arithmetic operations. Detection speeds are reported in Table 4 as mean of 500 detections.

Presented results demonstrate that MoDeCla was capable of operating in real-time with minimal hardware. Compared to the other analysed detectors, MoDeCla was able to perform detection in a notably shorter time. All other detectors except TinyYOLOv3 were an order of magnitude slower. TinyYOLOv3 reached a detection speed of roughly 60% of that of MoDeCla. However, the detection accuracy benchmarks demonstrated that MoDeCla greatly surpassed TinyYOLOv3 in terms of accuracy.

Detection accuracy

The detection accuracies of MoDecla, M2Det, EfficientDet-D1, YOLOv4, YOLOv3 and TinyYOLOv3 were benchmarked on the datasets characterised in Table 3. Detections were matched to ground truths with both the Traditional matching, as well as the presented Cluster matching. Traditional precision-recall curves generated from detections on the Summer benchmark set are presented in Fig. 6 for both classes present in the data: person and car. Cluster precision-recall curves are provided for the

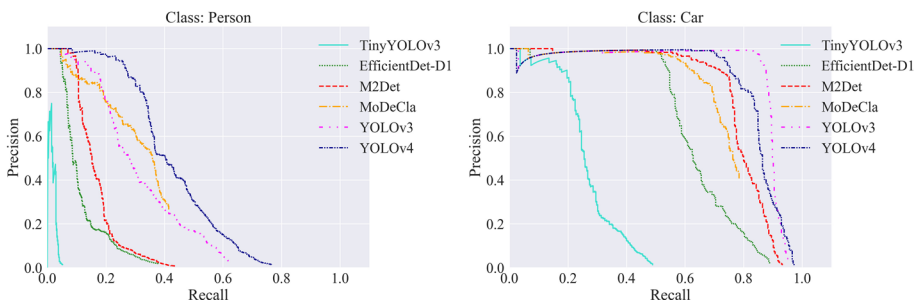

Fig. 6 Traditional precision-recall curves for the detectors on the Summer benchmark set

Table 3 Characteristics of the gathered detection benchmark data

Characteristic	Summer benchmark set	Winter benchmark set
Number of images	853	129
Number of people	610	154
Number of cars	325	100
Person height mean (std)	36 (18) px	38 (26) px
Person width mean (std)	16 (13) px	17 (14) px
Car height mean (std)	57 (34) px	28 (24) px
Car width mean (std)	95 (61) px	42 (44) px

Table 4 Detection speeds of the algorithms benchmarked on a Jetson Nano

Detector	Detection speed (fps)
M2Det*	0.630
YOLOv4	1.42
YOLOv3	1.45
EfficientDet-D1*	1.78
TinyYOLOv3	21.3
MoDeCla	**35.6**

*Not TensorRT optimised

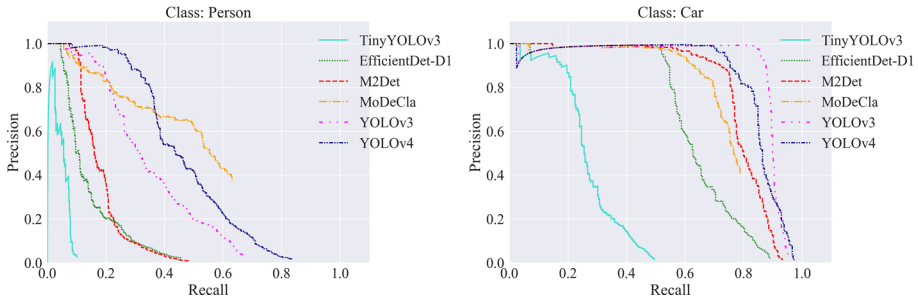

Fig. 7 Cluster precision-recall curves for the detectors on the Summer benchmark set

Summer benchmark set in Fig. 7. Furthermore, Traditional and Cluster precision-recall curves for the detectors on the Winter benchmark set are provided in Figs. 8 and 9, respectively. The provided curves allow in-depth analysis of the detection characteristics of the detectors. It is worth noting that MoDeCla did not reach as low precisions as the other detectors, as it does not provide as many low confidence detections due to only processing areas with apparent motion. For a convenient numerical comparison of the detectors, the AP and mAP values of the detectors calculated from the Traditional and Cluster precision-recall curves are reported in Tables 5 and 6 for the Summer benchmark set and Winter benchmark set, respectively.

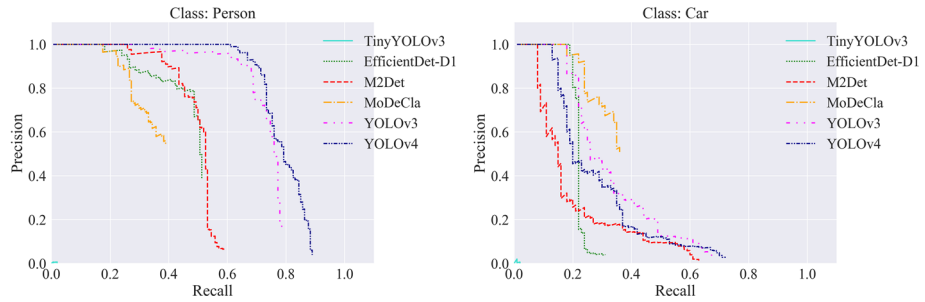

Fig. 8 Traditional precision-recall curves for the detectors on the Winter benchmark set

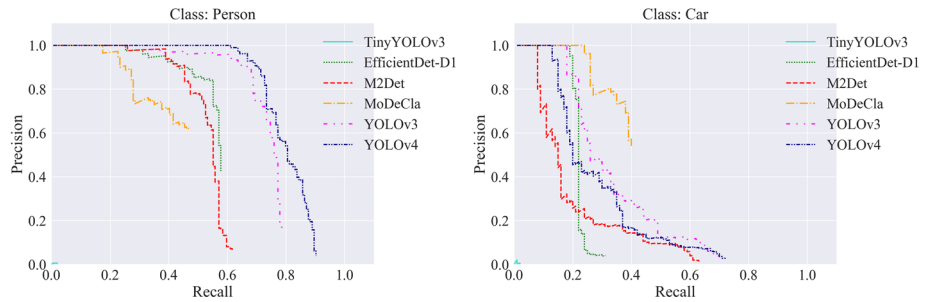

Fig. 9 Cluster precision-recall curves for the detectors on the Winter benchmark set

Analysing the mAP values for the Summer benchmark set in Table 5, the acquired results demonstrate that YOLOv4 and YOLOv3 were overall the most accurate detectors on this benchmark. MoDeCla was the third most accurate detector, with M2Det reaching a slightly lower mAP value. EfficientDet-D1 achieved a notably lower mAP score than the aforementioned detectors, and TinyYOLOv3 did not clearly succeed in the task as well as the other detectors. Observing the AP values of the person class, most of the detectors clearly had trouble correctly detecting people in the images. YOLOv4, YOLOv3 and MoDeCla were notably more successful than the other detectors when it came to detecting people. The Cluster matching evaluation greatly benefited the AP value of MoDeCla on the person class, ranking its accuracy nearly as high as that of YOLOv4. As for the car class, all detectors had an easier time detecting them correctly, as the reached AP values were much higher. This was likely due to cars appearing larger in the images than people. Compared to the other detectors, MoDeCla did not rank as high in detecting cars as it did in detecting people. In addition to YOLOv4 and YOLOv3, M2Det managed to reach a higher AP value on the car class than MoDeCla. EfficientDet-D1 also nearly acquired an equivalent AP value as MoDeCla. Cluster matching did not notably affect AP values of the detectors on the car class, as overlapping cars were rarer in the data than overlapping people.

The detectors performed fairly differently on the Winter benchmark set, as indicated by the results presented in Table 6. YOLOv4 and YOLOv3 were again overall

Fig. 10 Encountered difficulties with headlights and dark shadows in motion detection

Table 5 AP values computed from the Traditional and Cluster precision-recall curves on the Summer benchmark set

Detector	Traditional AP			Cluster AP		
	Person	Car	mAP	Person	Car	mAP
M2Det	16.8	80.0	48.4	18.3	80.0	49.2
YOLOv4	43.3	86.0	64.7	47.2	86.0	66.6
YOLOv3	31.5	89.7	60.6	35.2	89.7	62.5
EfficientDet-D1	11.8	65.4	38.6	14.4	65.4	39.9
TinyYOLOv3	1.99	27.3	14.7	5.23	27.6	16.4
MoDeCla	30.2	72.9	51.6	46.3	73.2	59.8

the highest performing detectors, achieving notably higher mAP scores than the other detectors. However, MoDeCla did not achieve such high accuracy, falling just slightly behind M2Det and EfficientDet-D1 in terms of mAP score. TinyYOLOv3 showcased the most drastic change in performance, as the detector was effectively not able to operate in the winter conditions, reaching by far the lowest AP scores for both classes. Contrary to the Summer benchmark, all detectors struggled to detect cars, yet had an easier time detecting people. Out of all the detectors, excluding TinyYOLOv3 due to its subpar performance, MoDeCla reached the lowest AP values on the person class. On the car class however, MoDeCla reached the second highest Traditional AP value, just barely surpassed by YOLOv3. The Cluster matching evaluation again improved the results of the detectors, notably increasing the AP values reached by MoDeCla on both classes. In fact, on the Cluster AP metric, MoDeCla achieved the highest accuracy of all the detectors on the car class. Overall, considering the previously presented processing speed results, MoDeCla seemed to offer a favourable combination of accuracy and speed on both benchmark sets.

Table 6 AP values computed from the Traditional and Cluster precision-recall curves on the Winter benchmark set

Detector	Traditional AP			Cluster AP		
	Person	Car	mAP	Person	Car	mAP
M2Det	50.1	20.4	35.3	53.4	20.4	36.9
YOLOv4	79.3	28.5	53.9	80.3	28.5	54.4
YOLOv3	73.2	33.5	53.4	73.2	33.8	53.5
EfficientDet-D1	46.6	22.2	34.4	54.6	22.2	38.4
TinyYOLOv3	0.0138	0.0264	0.0201	0.0138	0.0264	0.0201
MoDeCla	34.3	32.2	33.3	41.1	36.6	38.9

Discussion

As indicated by the presented results, MoDeCla showcased overall impressive performance on the gathered benchmark data. The precision and recall values achieved by MoDeCla are nearly on par with those of state-of-the-art general object detectors, while processing images notably faster. During testing, the accuracy of MoDeCla was noted to mostly suffer from slightly incorrect bounding box placement, which caused many detections to not quite pass the 0.5 *IoU* threshold. With an *IoU* threshold of 0.3 or 0.4, the results for MoDeCla would have likely improved, whereas the other detectors tend to leave objects completely undetected rather than placing a misrepresenting bounding box. Additional problems in the bounding box proposals of MoDeCla were noticed to emerge during night-time, and during sunrises and sunsets. As presented in Fig. 10, the headlights of cars as well as long dark shadows caused such drastic changes in pixel values that the motion detection registered much larger areas for the bounding boxes than intended. These problems were observed on both datasets, the Summer benchmark set and the Winter benchmark set. Furthermore, additional problems in bounding box proposals were noted to occur when road users moved directly away or towards the camera, as in these scenarios the perceived motion was naturally not as apparent. This was especially experienced on the Winter benchmark set, as pedestrians on the other walkway moved in a fairly straight line away or towards the camera. Future improvements to the bounding box proposal procedure could likely improve the results of MoDeCla greatly.

As for the general applicability of the acquired results, it is worth noting that the gathered image data only contained images from two different viewpoints overlooking two different areas. This limitation can skew the results, and operation in different scenarios may vary. The presence of objects that are visually undergoing constant change, such as reflective surfaces, bright lights or flora swaying in the wind, can trigger false positives in MoDeCla. Similarly, the accuracy of the other detectors may vary in different areas. Differences were already witnessed in operation of all the detectors between the benchmarking sets gathered from different camera views. This can however be largely attributed to the significantly different weather conditions as well. The performed benchmarks highlight that MoDeCla is capable of operating in significantly varying weather and lighting conditions, indicating that utilisation of the algorithm would be feasible in ITS applications. Still, further testing in varying traffic environments is required in order to benchmark the detection accuracies more exhaustively.

In addition to benchmarking with an increased amount of data, MoDeCla could be compared to computer vision algorithms specifically designed for detecting traffic users. In this paper, comparison to state-of-the-art general object detectors was made due to their wide reputation, as well as excellent accuracy and speed in a multitude of applications. Considering the person-class is by far the most common class in the COCO dataset used for training the CNN detectors, and the car-class is among the five most common classes, the detectors are arguably well-suited for the task evaluated here. More definitive results for the comparison of the detectors would have possibly been acquired if the detectors had been trained with the same data. However, since MoDeCla utilises classification data for training, whereas the other detectors require object detection data with bounding box coordinates, the training processes are inherently different. Furthermore, EfficientDet, M2Det, YOLOv3, and YOLOv4 all feature multi-scale processing, which makes the detectors exceptional for detecting small objects. This is ideal, as road users appear fairly small in the roadside camera view. Comparison against the detectors also has the benefit of the models being openly available for fair comparison. Similar to these detectors, MoDeCla can also be trained to detect any number of varying classes due to the classification stage being performed with a CNN. Many traffic-related computer vision methods presented in the literature are designed for detecting a single class, either pedestrians or vehicles. In the presented application of road user mapping, multiple classes must be detected, which would result in significant modifications in the detectors aimed for detecting specific types of road users. Nevertheless, for a more accurate benchmark, other detection approaches should be considered as well.

Regarding the suitability of the benchmarking metrics utilised here, the presented Cluster precision-recall may be biased towards the operation characteristics of MoDeCla. This metric allows clustering multiple ground truths to a single detection, given that they passed certain conditions. The Cluster precision-recall computation was implemented due to the bounding box proposal step of MoDeCla grouping overlapping road users into a single bounding box. In some road user detection applications this is not desirable behaviour, and therefore the Traditional precision-recall curves were also provided for the detectors. However, the presented safety-related application acting as motivation for this study does not benefit from differentiating individual road users from a tight group of road users. This information may not be crucial for many other applications as well. Therefore, this operation characteristic was defined acceptable, justifying the development of a modified evaluation metric. Analysing the results presented in Tables 5 and 6, the mAP of all the detectors can be seen to increase with the new Cluster precision-recall approach. As expected, the mAP of MoDeCla experienced the largest improvement. Other detectors did not benefit from the modified metric as much, since they attempt to find every single road user in a tight group, instead of placing a large bounding box which contains all the members of the group. This typically leads to detecting most road users in the front of a group as individuals, leaving the road users in the back undetected. This behaviour is less beneficial on the Cluster precision-recall metric, since a large bounding box can better cover mostly concealed objects in the back. Yet from the perspective of many applications, the detections should be assessed equally, since the behaviour of identifying only the front-most road users provides virtually the same information as placing a large bounding box on the entire group.

Due to the motion detection bounding box proposals utilised in MoDeCla, the algorithm is not as generally applicable as the other detectors. By definition, MoDeCla is only capable of detecting moving road users. A registry of some sort is required to keep track of road users that become stationary in the view. This type of additional processing may cause detection errors in camera views where road users make frequent stops. Additionally, usage of MoDeCla may be limited in camera views which commonly feature large numbers of overlapping road users, as overlapping road users are grouped into a single detection. This can be common in high traffic areas, where cars form tightly spaced ques and pedestrians move as an apparent unified crowd. However, proper camera placement can have a notable impact on the detection capabilities of MoDeCla, as well as the other detectors. Detecting overlapping objects is commonly a difficult task in computer vision applications. Basic tracking methodologies can greatly alleviate this problem, as earlier information of road user positions and trajectories assists in detecting cases of temporary overlap.

For many traffic-related applications, real-time detection with minimal hardware is crucial, and therefore the detection speeds of the detectors were benchmarked. As shown in Table 4, MoDeCla achieved a detection speed notably higher than that of any of the other detectors. The detection speed of TinyYOLOv3 was to some extent comparable to that of MoDeCla, yet its detection accuracy was notably poorer. It is worth noting that M2Det and EfficientDet-D1 were not evaluated with TensorRT optimised models, therefore limiting their performance. Optimised models would have processed images faster, yet the speed increase would not have likely been significant enough to alter the overall indications of the results. Furthermore, the compared detection speeds were acquired with using 32-bit floating-point arithmetic operations in the inference of all of the CNNs. Running inference with 16-bit floating-point or 8-bit integer representations has lately become a popular method for improving inference speed, typically at the cost of slightly reduced accuracy. Applying reduced precision here would have likely improved the speeds of all detectors, yet the improvement in the speeds of detectors other than MoDeCla would have been more notable due to the methods being end-to-end CNNs. The computational workload of MoDeCla is divided between the motion detection as well as CNN classifier, and therefore reduced inference speed of the CNN would be less impactful. Nevertheless, 32-bit precision was applied here due to the precision traditionally providing more reliable operation as well as being commonly used when reporting results in the literature. Applying a lower precision would not have likely changed the results drastically, leaving the the essence of the results intact.

Conclusion

A Motion Detection and Classification algorithm, MoDeCla, was proposed in this paper for road user detection. The algorithm was successfully developed to be ultra-fast while maintaining a reasonable balance of processing speed and detection accuracy. MoDeCla operates on the assumption of a stationary camera, applying well-established background subtraction methods for motion detection. Moving areas in images are captured as bounding box proposals, which are classified with a CNN to different types of road users or miscellaneous background. Detection speed benchmarks demonstrated that MoDeCla runs in real-time on minimal hardware, an order of magnitude faster than the

compared detectors. In the presented experiments, the classification stage of MoDeCla was trained to detect people, cars and miscellaneous background utilising existing datasets. The resulting detector was benchmarked on datasets gathered from two different cameras, one dataset gathered during summer and one dataset gathered during winter. On these datasets, MoDeCla achieved accuracies comparable to those of state-of-the-art object detectors. Analysing the detection performance of MoDeCla, its accuracy was noticed to be mostly hindered by incorrectly placed bounding boxes.

Future work will focus on improving the bounding box proposals of MoDeCla. Identified problems with bounding box proposals ranged from slight misplacements to more severe issues caused by car headlights and long dark shadows. These problems could possibly be solved by adding more complexity in the CNN that currently only classifies the proposals acquired with motion detection. The CNN architecture and training could be extended to additionally provide corrections for the bounding box coordinates. Furthermore, the CNN could be modified to consider multiple instances of objects in a single bounding box proposal, fitting a new bounding box for each. These improvements should not considerably reduce detection speed, since the CNN would not necessarily have to be significantly deeper to carry out the task properly. Described improvements would enhance the performance of MoDeCla and allow more general application. Currently the behaviour of grouping overlapping objects into a single bounding box may limit the areas where the detector can be applied.

Overall the results highlight that in terms of accuracy MoDeCla is comparable to state-of-the-art computer vision methods for road user detection, while running in real-time with the minimal hardware of a Jetson Nano. These features allow employment of the algorithm in many practical traffic user detection applications. Local computation and low hardware cost enable widespread adoption of intelligent infrastructure based sensors in road networks. As presented in related work, surveillance cameras could be utilised for improving safety in traffic intersection areas, providing information of obstructed and hidden road users to connected vehicles. The implementation of the algorithm used here is published as open source in the provided repositories to cultivate other creative applications or improvements to the algorithm.

Acknowledgements
Not applicable.

Authors' contributions
RO developed the research idea, implemented the algorithm, designed and executed the experiments, and drafted the manuscript. JV conceptualised the idea, structured the manuscript, defined the research questions, and verified the results. KT provided general research guidance, managed the research workflow, and revised initial versions of the manuscript. All authors read and approved the final manuscript.

Funding
This work was funded by Henry Ford Foundation Finland and Academy of Finland.

Availability of data and materials
The datasets generated and/or analysed during the current study are not publicly available due to data protection statements concerning the research, but are available from the corresponding author on reasonable request. The code implementations presented in the paper are available at the following open source repositories: https://github.com/ojalar/modecla and https://github.com/ojalar/map-calculator.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 3 November 2021 Accepted: 18 February 2022
Published online: 07 March 2022

References

1. WHO: The Top 10 Causes of Death. World Health Organization (WHO). World Health Organization (WHO). https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death. 2018. Accessed 12 May 2020.
2. National Highway Traffic Safety Administration: TRAFFIC SAFETY FACTS 2017. National Highway Traffic Safety Administration. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812806. 2019. Accessed 09 Apr 2021.
3. European Road Safety Observatory: Annual Accident Report 2018. European Road Safety Observatory. https://ec.europa.eu/transport/road_safety/sites/roadsafety/files/pdf/statistics/dacota/asr2018.pdf. 2018. Accessed 09 Apr 2021.
4. Ojala R, Vepsäläinen J, Hanhirova J, Hirvisalo V, Tammi K. Novel convolutional neural network-based roadside unit for accurate pedestrian localisation. In: IEEE Transactions on Intelligent Transportation Systems; 2019.
5. Buch N, Velastin SA, Orwell J. A review of computer vision techniques for the analysis of urban traffic. IEEE Trans Intell Transport Syst. 2011;12(3):920–39.
6. Atev S, Arumugam H, Masoud O, Janardan R, Papanikolopoulos NP. A vision-based approach to collision prediction at traffic intersections. IEEE Trans Intell Transport Syst. 2005;6(4):416–23.
7. NVIDIA: Jetson Nano Developer Kit. 2021. NVIDIA. https://developer.nvidia.com/embedded/jetson-nano-developer-kit. Accessed 09 Apr 2021.
8. Kim C, Lee J, Han T, Kim Y-M. A hybrid framework combining background subtraction and deep neural networks for rapid person detection. J Big Data. 2018;5(1):1–24.
9. Zhang Y, Yao D, Qiu TZ, Peng L. Scene-based pedestrian safety performance model in mixed traffic situation. IET Intell Transport Syst. 2014;8(3):209–18.
10. Zhou Z, Peng Y, Cai Y. Vision-based approach for predicting the probability of vehicle-pedestrian collisions at intersections. IET Intell Transport Syst. 2020;14(11):1447–55.
11. Pustokhina IV, Pustokhin DA, Vaiyapuri T, Gupta D, Kumar S, Shankar K. An automated deep learning based anomaly detection in pedestrian walkways for vulnerable road users safety. Safety Sci. 2021;142:105356.
12. Zhou Y, Liu L, Shao L, Mellor M. Fast automatic vehicle annotation for urban traffic surveillance. IEEE Trans Intell Transport Syst. 2017;19(6):1973–84.
13. Zhang B, Zhang J. A traffic surveillance system for obtaining comprehensive information of the passing vehicles based on instance segmentation. In: IEEE Transactions on Intelligent Transportation Systems; 2020.
14. Sharma B, Kumar S, Tiwari P, Yadav P, Nezhurina MI. Ann based short-term traffic flow forecasting in undivided two lane highway. J Big Data. 2018;5(1):1–16.
15. Viola P, Jones M, et al. Rapid object detection using a boosted cascade of simple features. CVPR (1). 2001;1(511–518):3.
16. Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1; 2005. pp. 886–93.
17. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, p. 1097–105; 2012.
18. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016; 770–778.
19. Luebke D. Cuda: Scalable parallel programming for high-performance scientific computing. In: 2008 5th IEEE International Symposium on Biomedical Imaging: from Nano to Macro, 2008. p. 836–8.
20. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018. p. 4510–4520.
21. Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. arXiv preprint arXiv:1602.07360 2016.
22. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC. Ssd: Single shot multibox detector. In: European Conference on Computer Vision. New YorK: Springer; 2016. p. 21–37.
23. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016. p. 779–88.
24. Redmon J, Farhadi A. Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017. p. 7263–71.
25. Redmon J, Farhadi A. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 2018.
26. Bochkovskiy A, Wang C-Y, Liao H-YM. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 2020.
27. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014. p. 580–7.
28. Girshick R. Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448, 2015.
29. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. New York: Springer; 2014. p. 740–55.
30. Zhao Q, Sheng T, Wang Y, Tang Z, Chen Y, Cai L, Ling H. M2det: A single-shot object detector based on multi-level feature pyramid network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019. p. 9259–66.

31. Tan M, Pang R, Le QV. Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. p. 10781–90.
32. Du X, El-Khamy M, Lee J, Davis L. Fused dnn: A deep neural network fusion approach to fast and robust pedestrian detection. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV). 2017. p. 953–61.
33. Dollár P, Wojek C, Schiele B, Perona P. Pedestrian detection: A benchmark. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. 2009. p. 304–11.
34. Wang L, Lu Y, Wang H, Zheng Y, Ye H, Xue X. Evolving boxes for fast vehicle detection. In: 2017 IEEE International Conference on Multimedia and Expo (ICME). 2017. p. 1135–40.
35. Wen L, Du D, Cai Z, Lei Z, Chang M-C, Qi H, Lim J, Yang M-H, Lyu S. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking. Computer Vision Image Understand. 2020;193:102907.
36. Ferraz PAP, de Oliveira BAG, Ferreira FMF, da Silva Martins CAP. Three-stage rgbd architecture for vehicle and pedestrian detection using convolutional neural networks and stereo vision. IET Intell Transport Syst. 2020;14(10):1319–27.
37. Benenson R, Omran M, Hosang J, Schiele B. Ten years of pedestrian detection, what have we learned? In: European Conference on Computer Vision. New York: Springer; 2014. p. 613–27.
38. Sivaraman S, Trivedi MM. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. IEEE Trans Intell Transport Syst. 2013;14(4):1773–95.
39. Benenson R, Mathias M, Timofte R, Van Gool L. Pedestrian detection at 100 frames per second. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012. p. 2903–10.
40. Dollár P, Tu Z, Perona P, Belongie S. Integral channel features 2009.
41. Viola P, Jones MJ, Snow D. Detecting pedestrians using patterns of motion and appearance. Int J Computer Vision. 2005;63(2):153–61.
42. Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. J Computer Syst Sci. 1997;55(1):119–39.
43. Gupte S, Masoud O, Martin RF, Papanikolopoulos NP. Detection and classification of vehicles. IEEE Trans Intell Transport Syst. 2002;3(1):37–47.
44. Bai H, Wu J, Liu C. Motion and haar-like features based vehicle detection. In: 2006 12th International Multi-Media Modelling Conference; 2006. p. 4.
45. Zhang Z, Cai Y, Huang K, Tan T. Real-time moving object classification with automatic scene division. In: 2007 IEEE International Conference on Image Processing, vol. 5, p. 149, 2007.
46. Zivkovic Z, Van Der Heijden F. Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognit Lett. 2006;27(7):773–80.
47. Suzuki S, et al. Topological structural analysis of digitized binary images by border following. Computer Vision Graphics Image Processing. 1985;30(1):32–46.
48. Bradski G. The OpenCV Library. Dr. Dobb's Journal of Software Tools. 2000.
49. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L et al. Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, 2019. p. 8024–35.
50. NVIDIA: NVIDIA TensorRT. NVIDIA. https://developer.nvidia.com/tensorrt. Accessed 09 Apr 2021.
51. Deng Y, Luo P, Loy CC, Tang X. Pedestrian attribute recognition at far distance. In: Proceedings of the 22nd ACM International Conference on Multimedia. 2014. p. 789–92.
52. Luo Z, Branchaud-Charron F, Lemaire C, Konrad J, Li S, Mishra A, Achkar A, Eichel J, Jodoin P-M. Mio-tcd: A new benchmark dataset for vehicle classification and localization. IEEE Trans Image Process. 2018;27(10):5129–41.
53. Krizhevsky A, Hinton G, et al. Learning multiple layers of features from tiny images. Citeseer: Technical report; 2009.
54. Franklin D. Deep Learning Inference Benchmarking Instructions. NVIDIA. NVIDIA. 2019; https://forums.developer.nvidia.com/t/deep-learning-inference-benchmarking-instructions/73291. Accessed 09 Apr 2021.
55. Jung J. TensorRT Demos. https://github.com/jkjung-avt/tensorrt_demos. 2021. Accessed 21 Sept 2021.
56. Qijie Z. M2Det. https://github.com/qijiezhao/M2Det. 2019. Accessed 21 Sept 2021.
57. Yet Another EfficientDet Pytorch. https://github.com/zylo117/Yet-Another-EfficientDet-Pytorch. 2020. Accessed 21 Sept 2021.

Publisher's Note