MIO-TCD: A New Benchmark Dataset for Vehicle Classification and Localization

Zhiming Luo, Frédéric Branchaud-Charron, Carl Lemaire, Janusz Konrad, *Fellow, IEEE*, Shaozi Li, *Member, IEEE*, Akshaya Mishra, Andrew Achkar, Justin Eichel, and Pierre-Marc Jodoin

Abstract—**The ability to train on a large dataset of labeled samples is critical to the success of deep learning in many domains. In this paper, we focus on motor vehicle classification and localization from a single video frame and introduce the "Miovision traffic camera dataset" (MIO-TCD) in this context. MIO-TCD is the largest dataset for motorized traffic analysis to date. It includes 11 traffic object classes such as cars, trucks, buses, motorcycles, bicycles, and pedestrians. It contains 786 702 annotated images acquired at different times of the day and different periods of the year by hundreds of traffic surveillance cameras deployed across Canada and the United States. The dataset consists of two parts: a "localization dataset," containing 137 743 full video frames with bounding boxes around traffic objects and a "classification dataset," containing 648 959 crops of traffic objects from the 11 classes. We also report the results from the 2017 CVPR MIO-TCD Challenge that leveraged this dataset and compare them with the results of the state-of-the-art deep learning architectures. These results demonstrate the viability of deep learning methods for vehicle localization and classification from a single video frame in real-life traffic scenarios. The top-performing methods achieve both accuracy and Kappa score above 96% on the classification dataset and mean-average precision of 77% on the localization dataset. We also identify the scenarios in which the state-of-the-art methods still fail, and we suggest avenues to address these challenges. Both the dataset and detailed results are publicly available online.**

Index Terms—**Traffic analysis, deep learning, vehicle localization, vehicle classification.**

I. Introduction

The localization and classification of vehicles in the field of view of a traffic camera is the very first step in most traffic surveillance systems (e.g., car counting, activity recognition, anomaly detection, tracking, post-event forensics). Although subsequent processing may be different in each case, one often has to start by localizing foreground objects and then identifying what these objects are (trucks, cars, bicycles, pedestrians, etc.).

To the best of our knowledge, except for pedestrian applications, most traffic monitoring systems rely on motion features such as optical flow, motion detection, and vehicle tracking [2]–[4]. Motion features are then used to count the number of vehicles on the road, estimate traffic speed, and recover global motion trajectories. But these traffic monitoring systems all share a fundamental limitation: in order to function properly, they need high frame-rate videos so motion features can be reliably extracted [5]. Also, reliable object tracking is still a challenge for cluttered scenes, especially when vehicles are partly occluded [6].

Unfortunately, low frame-rate videos are common as many large-scale camera networks cannot stream and store high frame-rate videos gathered by thousands of cameras. Instead, cameras are often configured to send one frame every second or so to the server. This is especially true for cameras transmitting over a cellular network whose bandwidth may vary over time making the throughput unpredictable. In such cases, one can only analyze ultra low frame-rate videos out of which no motion features can be accurately extracted. Also, those cameras often have low resolution which makes localization and classification difficult.

To date, many object localization algorithms have been developed, and even more articles have been written on the topic. With the rise of deep learning methods (mostly convolutional neural nets), an exponential number of publications have been devoted to deep architectures implementing object recognition and localization methods [7]–[10]. As a result, error rates on very challenging datasets, such as Pascal VOC [11], ImageNet [12] and Microsoft COCO [13], have decreased at a steady pace. Among other things, what makes

Manuscript received July 14, 2017; revised March 30, 2018 and May 25, 2018; accepted June 9, 2018. Date of publication June 18, 2018; date of current version July 19, 2018. This work was supported in part by the National Natural Science Foundation of China under Grants 61572409, U1705286, and 61571188, in part by the Fujian Province 2011 Collaborative Innovation Center of TCM Health Management and the Collaborative Innovation Center of Chinese Oolong Tea Industry-Collaborative Innovation Center (2011) of Fujian Province, in part by the Fund for Integration of Cloud Computing and Big Data, Innovation of Science and Education, in part by FQRNT under Grant B1 208594, in part by Mitacs Acceleration under Grant IT08995, and in part by NSERC under Grant 210836151. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Junsong Yuan. *(Corresponding author: Shaozi Li.)*

Z. Luo is with the Postdoc Center of Information and Communication Engineering, Xiamen University, Xiamen 361005, China, and also with the Computer Science Department, University of Sherbrooke, Sherbrooke, QC J1K 2R1, Canada (e-mail: zhiming.luo@usherbrooke.ca).

F. Branchaud-Charron, C. Lemaire, and P.-M. Jodoin are with the Computer Science Department, University of Sherbrooke, Sherbrooke, QC J1K 2R1, Canada (e-mail: frederic.branchaud-charron@usherbrooke.ca; carl.lemaire@usherbrooke.ca; pierre-marc.jodoin@usherbrooke.ca).

J. Konrad is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215 USA (e-mail: jkonrad@bu.edu).

S. Li is with the Cognitive Science Department, Xiamen University, Xiamen 361005, China (e-mail: szlig@xmu.edu.cn).

A. Mishra is with the Systems Design Engineering Department, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: akshaya.mishra.kumar@gmail.com).

A. Achkar and J. Eichel are with Miovision Technologies Inc., Kitchener, ON N2G 4X8, Canada (e-mail: aachkar@miovision.com; jeichel@miovision.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2848705

deep learning so successful is the availability of large and well-annotated datasets. Unfortunately, despite the large number of publications devoted to traffic analytics, no such traffic dataset has been released to date. The lack of such a dataset has a number of implications, the most important one being that deep architectures trained on non-traffic oriented datasets such as ImageNet and COCO do not generalize well to traffic images.

Recognizing the importance of labeled datasets for the development of traffic analysis methods, we introduce the *MIOvision Traffic Camera Dataset* (MIO-TCD). It contains a total of 786,702 images: 137,743 high-resolution video frames with multiple vehicles in each frame and 648,959 vehicles cropped out of full frames. These images have been captured by hundreds of cameras deployed in urban and rural areas taking pictures at different periods of the year, at different times of the day, with different camera orientations and with various traffic densities. Many people have participated in the process of hand-annotating each image to provide a bounding box around each distinguishable object.

Leveraging this dataset, we demonstrate how well-trained, state-of-the-art deep learning methods can be used to localize and identify moving objects with high precision without having to rely on motion features. With classification accuracies of 96% and localization mean-average precision of 77%, we show that one can localize and recognize vehicles regardless of their orientation, scale, color and environmental conditions.

We believe that this work will have a substantial impact as there exists an urgent need for traffic monitoring systems working on still 2D images. We hope the MIO-TCD datset will have similar impact on the video surveillance community as the Caltech [14] and INRIA [15] datasets had on the pedestrian detection community, or ImageNet and COCO datasets – on the deep learning community.

The MIO-TCD dataset was the foundation of the Traffic Surveillance Workshop and Challenge held in conjunction with CVPR 2017. There were three primary goals to this challenge:

1) Provide the scientific community with a rich dataset to compare and test new methods (the dataset will be regularly revised and expanded in the future and will maintain a ranking of methods).
2) Identify and rank the best traffic-oriented object localization and classification algorithms to date in various real-life conditions.
3) Identify remaining critical challenges in order to provide focus for future research.

In what follows, we elaborate on these goals, describe the dataset and the challenge results, and discuss unsolved challenges while offering avenues for future work.

II. Previous Datasets

Today, we are witnesses to an exponential growth in the number of surveillance cameras are deployed along the roads of almost every country in the world. However, the images acquired by those cameras are the sole property of traffic management departments and are rarely released to the public. Consequently, few datasets have been made public for traffic analysis research. The most widely used datasets can be roughly divided into 3 categories, namely: (1) datasets of images taken by on-board cameras and mainly aimed at autonomous driving, (2) datasets of vehicle images taken by non-surveillance cameras and mainly oriented towards automatic recognition of high-resolution images from the Internet and (3) datasets of vehicle images taken by surveillance cameras. Here, we present a brief survey of existing vehicle detection and localization datasets.

A. On-Board Camera Datasets

1) KITTI Benchmark Dataset [16][1]: This is a large dataset collected by an autonomous driving platform which addresses several real-world challenges, including: stereo vision calculation, optical flow estimation, visual odometry/SLAM, 3D object detection and 3D object tracking. This dataset consists of video frames captured by cars traveling both in rural areas and on highways.

2) Cityscapes Dataset [17][2]: This dataset focuses on the semantic segmentation of urban street scenes. It comes with 5,000 fully-annotated images and 20,000 weekly-annotated images. The annotated objects span 30 classes including eight different types of vehicles. Unfortunately, this dataset is limited to images acquired in urban areas (50 cities) and during summer daytime.

3) Tsinghua-Tencent Traffic-Sign Dataset [18][3]: This is a large traffic-sign dataset containing 100,000 images with 30,000 traffic-sign instances. Images in this dataset cover various illumination and weather conditions but do not contain any labeled vehicles.

B. Vehicles Captured by Non-Surveillance Cameras

1) Stanford Car Dataset [19][4]: This dataset contains 16,185 high-resolution images of 196 classes of cars. The vehicle classes include the brand, the model, and the year (e.g. 2012 Tesla Model S or 2012 BMW M3 coupe). This dataset is divided into 8,144 training images and 8,041 testing images. In addition to the large variety of vehicles, all pictures have excellent resolution and have been captured in good lighting conditions. However, none of the pictures were taken in a top-down orientation which is usually the case with surveillance cameras.

2) Comprehensive Cars Dataset (Web) [20][5]: This is one of the largest car dataset currently available. It comes with a total of 136,727 images showing entire cars and 27,618 images showing car parts. The dataset comprises 1,716 vehicle models as well as five different attributes (maximum speed, displacement, number of doors, number of seats, and type of car). Unfortunately, this dataset has the same limitations as the Stanford Car Dataset as its images were taken in a context far different than that of a surveillance camera (arbitrary orientation and filming 24/7).

[1] http://www.cvlibs.net/datasets/kitti/index.php

[2] https://www.cityscapes-dataset.com/

[3] http://cg.cs.tsinghua.edu.cn/traffic-sign/

[4] http://ai.stanford.edu/ jkrause/cars/car_dataset.html

[5] http://mmlab.ie.cuhk.edu.hk/datasets/comp_cars/index.html

C. Traffic Datasets From Surveillance Cameras

1) Comprehensive Cars Dataset (Surveillance) [20]: This dataset contains 44,481 images of cars captured by frontal-view surveillance cameras. The ground truth provides the model and the color of each vehicle. The main limitation of this dataset comes from the frontal view of the pictures which makes it hard to generalize to an arbitrary camera orientation. Furthermore, this dataset focuses on cars, mini vans and pickup trucks, and does not contain any large articulated trucks, buses, motorcycles and pedestrians.

2) BIT-Vehicle Dataset [21][6]: This dataset consists of 9,850 vehicle images. This is one of the most realistic datasets with images coming from real surveillance cameras. Vehicles are divided into six categories, namely bus, micro-bus, minivan, sedan, SUV and truck. Unfortunately, those images were all taken in a top-frontal view during daytime and clear weather, thus limiting the diversity needed for general traffic analysis applications.

3) Traffic and Congestions (TRANCOS) Dataset [22][7]: This dataset is used to count the number of vehicles on highly-congested highways. It consists of 1,244 images with 46,796 annotated vehicles, most of which are partially occluded. Images were captured by publicly available video surveillance cameras of the Dirección General de Tráfico of Spain. Unfortunately, no vehicle type is provided.

4) GRAM Road-Traffic Monitoring (GRAM-RTM) Dataset [23][8]: This is a benchmark dataset for multi-vehicle tracking. It consists of video sequences recorded by surveillance cameras under different conditions and with different platforms. Every vehicle has been manually annotated into different categories (car, truck, van, and big truck). Each video contains around 240 different objects.

Clearly, several publicly-available vehicle datasets contain images that do not come from surveillance cameras. The *KITTI benchmark dataset*, the *Cityscapes Dataset* and the *Tsinghua-Tencent Traffic-Sign Dataset* contain images captured by on-board cameras that can hardly be used to train traffic surveillance methods. The *Stanford Car Dataset* and the *Comprehensive Car Dataset (web)* contain high-resolution pictures of vehicles mainly taken in frontal and side view and rarely in top-down view as is usually the case with traffic surveillance applications. Images from these datasets are usually applied to fine-grained vehicle analysis (counting the number of doors, identifying the vehicle brand and year, etc.). As for the *Comprehensive Cars Dataset (surveillance)*, although it is one of the largest publicly-available surveillance dataset, its images come from a well-aligned frontal-view camera and show only one vehicle per image. It also shows images in daylight and good weather. Also, none of the datasets contains more than a couple of thousand images.

As for the *BIT-Vehicle Dataset*, it has up to two vehicles per frame but contains less than 10,000 images. The *TRANCOS Dataset* contains traffic jam images in which vehicles are too small to be categorized while the *GRAM-RTM Dataset* has only three video sequences.

III. MIO-TCD: Our Proposed Dataset

A. Dataset Overview

The MIO-TCD dataset contains a total of 786,702 images, 137,743 being high-resolution video frames showing multiple vehicles and 648,959 lower-resolution images of cropped vehicles. These images were acquired at different times of the day and different periods of the year by hundreds of traffic cameras deployed across Canada and the United States. Those images have been selected to cover a wide range of challenges and are typical of visual data captured in urban and rural traffic scenarios. The dataset includes images: (1) taken at different times of the day, (2) with various levels of traffic density and vehicle occlusion, (3) showing small moving objects due to low resolution and/or perspective, (4) showing vehicles with different orientations, (5) taken under challenging weather conditions, and (6) exhibiting strong compression artifacts. This dataset has been carefully annotated by a team of nearly 200 people to enable a quantitative comparison and ranking of various algorithms.

The MIO-TCD dataset aims to provide a rigorous facility for measuring how far state-of-the-art deep learning methods can go at classifying and localizing vehicles recorded by traffic cameras. The dataset consists of two components: the *classification dataset* and the *localization dataset*. The classification dataset is used to train and test classification algorithms whose goal is to predict the kind of vehicle located in a low-resolution image patch. The localization dataset may be used to train and test algorithms whose goal is to localize and recognize vehicles located in the image.

1) MIO-TCD–Classification Dataset: The classification dataset contains 648,959 low-resolution images divided into 11 categories: *Articulated Truck*, *Bicycle*, *Bus*, *Car*, *Motorcycle*, *Non-Motorized Vehicle*, *Pedestrian*, *Pickup Truck*, *Single-Unit Truck*, *Work Van* and *Background*. As shown in Fig. 1, each image contains one dominant object located in the middle of the image. The objects pictured in that dataset come in various sizes and were recorded in different parts of the year, different times of the day and from different viewing angles. The dataset was split into 80% training (519,164 images) and 20% testing (129,795 images). Note that the *Car* category contains vehicles of type sedan, SUV and family van.

The number of images in each category is listed in Table I. Since these images come from real footage, the number of samples per category is highly unbalanced as certain types of vehicles are more frequent than others. As such, almost half the images in the dataset fall into the *car* category, while the *Bicycle*, *Motorcycle* and *Non-Motorized Vehicle* categories contain roughly 2,000 training images and 500 testing images. We also randomly sampled 200,000 images to construct a background category.

2) MIO-TCD–Localization Dataset: The localization dataset contains 137,743 images of which 110,000 are intended for training and 27,743 for testing. These images have various resolutions, ranging from 720 × 480 to

[6] http://iitlab.bit.edu.cn/mcislab/vehicledb/

[7] http://agamenon.tsc.uah.es/Personales/rlopez/data/trancos/

[8] http://agamenon.tsc.uah.es/Personales/rlopez/data/rtm/

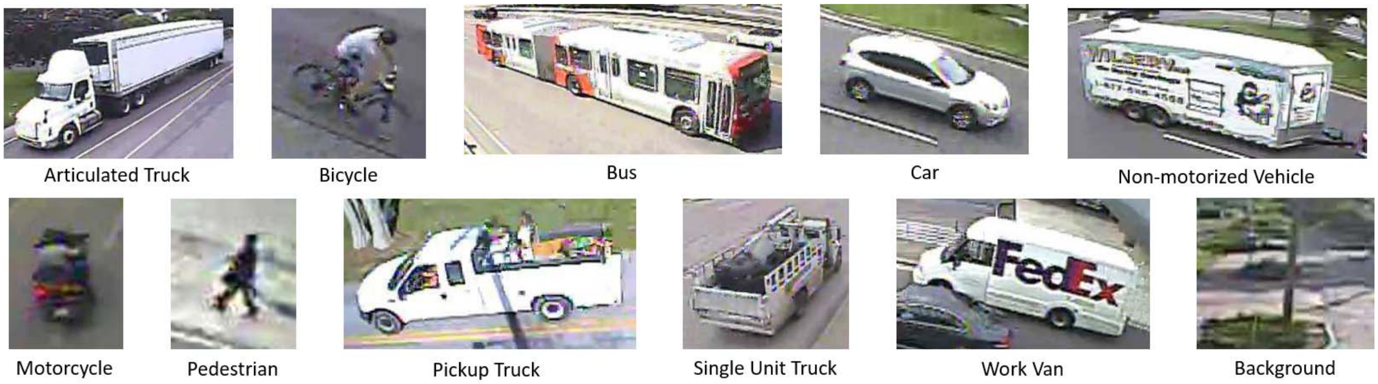


Fig. 1. Sample images from the 11 categories of the classification dataset.

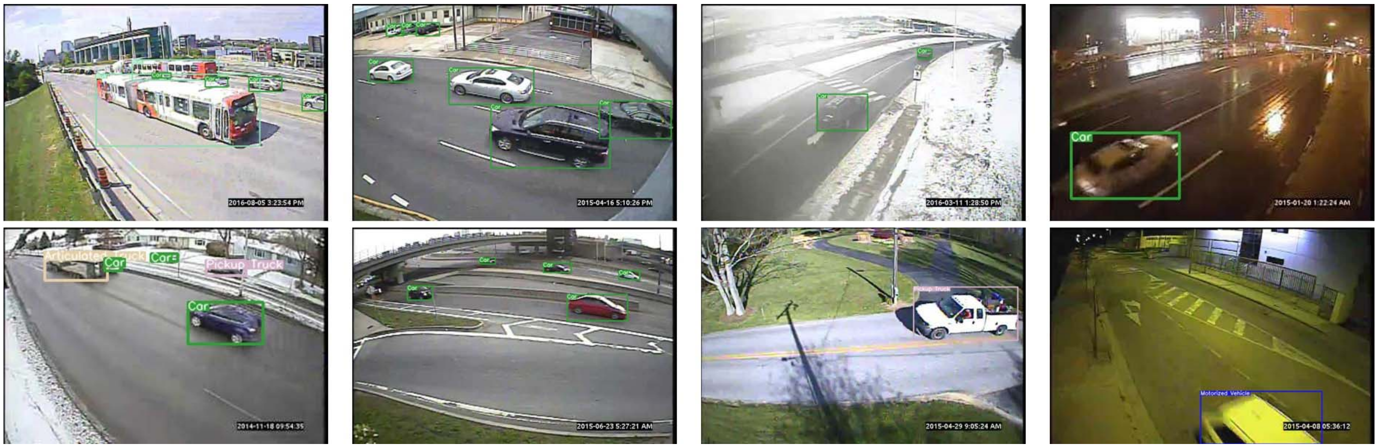


Fig. 2. Sample images from the MIO-TCD localization dataset.

TABLE I
SIZE OF EACH CATEGORY IN THE CLASSIFICATION DATASET

Category	**Training**	**Testing**
Articulated Truck	10,346	2,587
Bicycle	2,284	571
Bus	10,316	2,579
Car	260,518	65,131
Motorcycle	1,982	495
Non-Motorized Vehicle	1,751	438
Pedestrian	6,262	1,565
Pickup Truck	50,906	12,727
Single-Unit Truck	5,120	1,280
Work Van	9,679	2,422
Background	160,000	40,000
Total	**519,164**	**129,795**

342 × 228. A total of 416,277 moving objects have been manually annotated with a bounding box and a category label. Except for *Background*, the same category labels as those in Table I have been used. However, due to the fact that localizing and recognizing vehicles in a full video frame is more difficult than classifying images of already localized vehicles, we added a new *Motorized Vehicle* category which is unique to the localization dataset. This category contains all vehicles that are too small (or occluded) to be labeled into a specific category. Because of this category overlap, the classification dataset can be leveraged to improve the accuracy of localization methods.

Similarly to the classification dataset, images of the localization dataset came from hundreds of real traffic surveillance cameras. The category labels are thus unbalanced in similar proportions as those in Table I (see Fig. 2 for some examples).

B. Evaluation Metrics

1) Classification: Three metrics have been implemented to gauge performance of classification methods. The first metric is the overall accuracy (Acc) which is the proportion of correctly classified images in the whole dataset:

$$Acc = \frac{TP}{\text{total number of images}} \tag{1}$$

where TP is the total number of correctly-classified images regardless of their category. TP can also be seen as the trace of a confusion matrix such as the one in Fig. 3.

Since the classification dataset is highly unbalanced, large categories such as *Car* and *Background* have an overwhelming influence on the calculation of the overall accuracy. We thus implemented three metrics which account for this imbalance, namely the mean recall (mRe), the mean precision (mPr) and the Cohen Kappa Score ($Kappa$) [24].

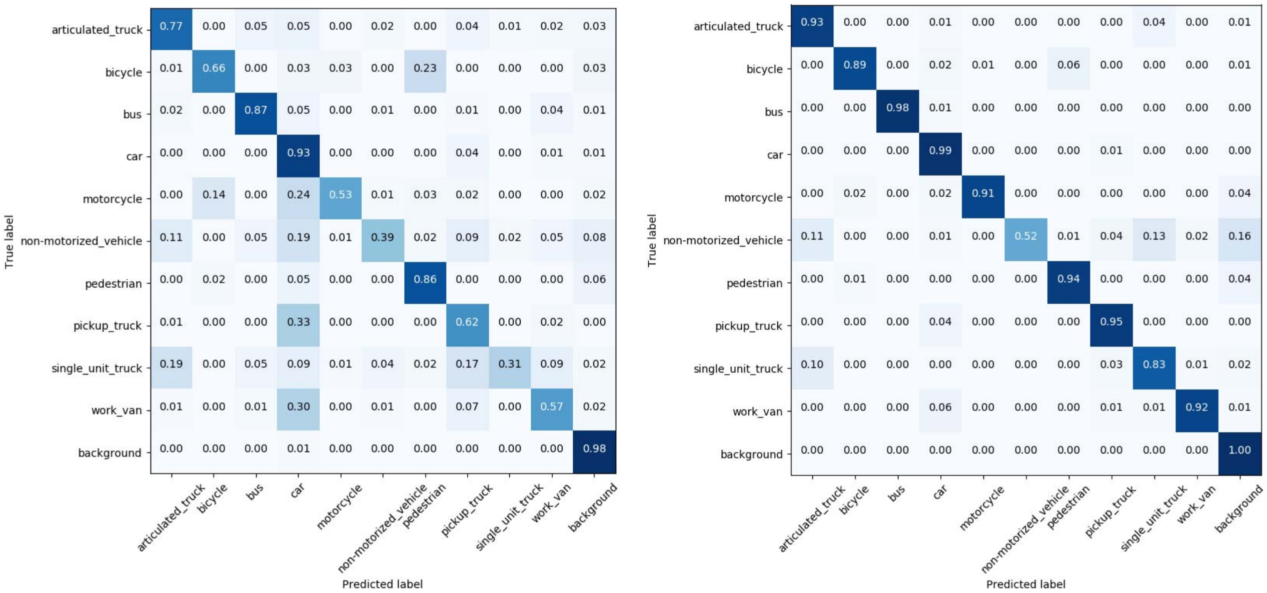


Fig. 3. Confusion matrices obtained on the classification dataset with pre-trained ResNet-50 features + SVM on left and the ensemble model by Jung *et al.* [39] on right.

The mean recall and mean precision are obtained by averaging the recall and the precision of each category thus giving an equal weight to each category. This is done as follows:

$$mRe = \frac{\sum_{i=1}^{11} Re_i}{11} \qquad mPr = \frac{\sum_{i=1}^{11} Pr_i}{11}$$

where $Re_i = TP_i/(TP_i + FN_i)$ and $Pr_i = TP_i/(TP_i + FP_i)$ are the recall and precision for category i, and TP_i, FN_i, FP_i are the numbers of true positives, false negatives and false positives for the i-th category, respectively.

$Kappa$ is a measure that expresses the agreement between two annotators. In our case, the first annotator is a method under evaluation and the second annotator is the ground truth. The Cohen Kappa Score is defined as [24]:

$$Kappa = \frac{Acc - P_e}{1 - P_e} \tag{2}$$

where Acc is the accuracy (1) and P_e is the probability of agreement when both annotators assign random labels. $Kappa$ values are in the $[-1, 1]$ range where $Kappa = 1$ means that both annotators are in complete agreement, while $Kappa \leq 0$ means no agreement at all.

2) Localization: Following the Pascal VOC 2012 object detection evaluation protocol, we report localization results *via* precision/recall curves and use the average precision (AP) as the principal quantitative measure for each vehicle category. A detection is considered a true positive when the overlap ratio between the predicted bounding box and the ground-truth bounding box exceeds 50%; otherwise, it is considered a false positive. We also report the *mean-average precision* (mAP) which is the mean of AP across all categories. We invite the reader to refer to the Pascal VOC development kit document[9] for more details on the AP metric.

The main difference between our localization challenge and other localization challenges is the occurrence of the *Motorized Vehicle* category. This category is used to label vehicles that are too small (or too occluded) to be correctly assigned a specific category. For example, a car seen from far away and whose size does not exceed a few pixels would be labeled as a *Motorized Vehicle*. In order not to penalize methods which would incorrectly label those small objects, every *Motorized Vehicle* labeled by one of following categories: *Articulated Truck, Bus, Car, Pickup Truck, Single-Unit Truck* and *Work Van*, is considered a true detection. In our implementation, we first identify every predicted bounding box whose overlap ratio with a *Motorized Vehicle*'s ground-truth bounding box exceeds 50%, and re-assign its category label to *Motorized Vehicle*. Then, we compute the AP for each category.

IV. Methods Tested

One of the overarching objectives of this paper is to identify how far state-of-the-art machine learning methods can go at classifying and localizing vehicles pictured by real traffic surveillance cameras. As mentioned earlier, this implies the analysis of low resolution 2D images with strong compression artifacts, recorded during daytime/nighttime, in different seasons, under diverse weather conditions, and with various camera positions and orientations. As such, one can expect that some methods might be robust to those challenges while others may not. In order to identify solved and unsolved issues, we carefully benchmarked a series of state-of-the-art deep learning methods. In this section, we describe the methods that we tested on the MIO-TCD classification and localization datasets. Some methods have been published before the release of the MIO-TCD dataset while others have been designed specifically for this dataset's challenges [1].

[9] http://host.robots.ox.ac.uk/pascal/VOC/voc2012/devkit_doc.pdf

A. Vehicle Classification

1) Pre-Trained CNN Features + SVM: The first series of methods aim at gauging how "different" is the statistical content of traffic images in the MIO-TCD dataset from other datasets, such as ImageNet. We did so by training a linear SVM classifier on CNN features obtained from ImageNet pre-trained CNN models. This is inspired by Razavian *et al.* [25] who demonstrated that features obtained from deep CNN models could be the primary choice of a large variety of visual recognition tasks. We did so with the following six pre-trained deep models: AlexNet [26], InceptionV3 [27], ResNet-50 [10], VGG-19 [28], Xception [29] and DenseNet [30]. The layer we used to extract the features from is: ReLU of FC-7 in AlexNet and VGG-19, and the global pooling layer in InceptionV3, ResNet-50, Xception and DenseNet. Their corresponding feature dimensions are 4096, 4096, 2048, 2048, 2048, 1920. The python *scikit-learn* library[10] was used to train the linear SVM with $C = 1$.

2) Retrained CNN Models: Although features from the first layers of a CNN model are independent from the dataset it was trained on (mostly Gabor-like filters [31]), we retrained end-to-end all six CNN models on our classification dataset. Those models were all initialized with ImageNet pre-trained weights and were trained with the loss function proposed in the corresponding original papers. We used the Adam [32] optimizer with a learning rate of 10^{-3} that we empirically found more effective than other optimizers such as RMSprop or SGD. Training was done for a maximum of 50 epochs with a validation-based early stopping criteria with a patience of 10 epochs to prevent over-fitting. The batch size was adjusted to each model so it could fit on our 12GB Titan X GPUs. Please note that we included a series of batch normalization layers [33] in AlexNet and VGG-19 to speed up training. All models but DenseNet were implemented with the Keras library [34]. DenseNet was implemented with PyTorch [35].

We also implemented the following four training configurations to further improve results on our dataset:

- The first configuration is a basic training with normal sampling of the data.
- The second configuration involves data augmentation using horizontal flipping and randomized shearing and zooming to enrich the training dataset.
- Since the dataset is highly unbalanced (see Table I), we used data augmentation with uniform sampling. That is, at each epoch we used an equal number of images from each class to prevent large classes, such as *Car*, *Pickup Truck* and *Background* from gaining too much importance over smaller classes, such as *Motorcycle*.
- As suggested by Havaei *et al.* [36], we implemented a two-phase training procedure. In the first phase, we used data augmentation with uniform sampling. In the second phase, we froze the entire network except for the last layer which was retrained with data augmentation and normal sampling.

TABLE II

EVALUATION METRICS FOR SIX PRE-TRAINED MODELS USED WITH LINEAR SVM CLASSIFIERS ON THE CLASSIFICATION DATASET

	Acc	mRe	mPr	$Kappa$
AlexNet	0.82	0.49	0.55	0.72
Inception-V3	0.84	0.57	0.64	0.75
ResNet-50	**0.89**	**0.69**	0.74	**0.83**
VGG19	0.83	0.66	0.59	0.75
Xception	0.87	0.54	0.76	0.78
DenseNet	0.86	0.51	**0.82**	0.78

3) MIO-TCD Classification (Ensemble Models): In the wake of the 2017 CVPR MIO-TCD Challenge [1], several methods have been designed for the sole purpose of classifying traffic images. Interestingly, all of those methods involve a combination of several deep learning models. Kim and Lim [37] proposed a bagging system where several CNN models are trained on a random subset of the MIO-TCD dataset. Their final result is obtained with a weighted majority vote to compensate for the unbalanced nature of the dataset. Lee and Chung [38] proposed an ensemble method which combines 3 convolutional nets (AlexNet, GoogleNet and ResNet18) trained on 18 different sets of data. GoogleNet was trained on 12 subsets of the dataset (aka the local nets) and AlexNet, GoogleNet and ResNet18 were trained on the entire dataset but with different image sizes (aka the global nets). At the test time, the networks are selected with a gating function and combined with a *softmax* layer. Jung *et al.* [39] proposed an ensemble model according to which several deep residual networks are jointly trained. The main novelty of their method lies within its loss function which allows to train every ResNet simultaneously. Theagarajan *et al.* [40] also proposed an ensemble of ResNet models. The authors implemented a weighted loss function to account for the unbalanced nature of the dataset. They also implemented a patch-wise logical reasoning process to disambiguate classes that are close to each other like trucks and buses.

B. Vehicle Localization

1) Recent CNN-Based Methods: Recently, CNN-based localization methods established state-of-the-art performance on several object localization datasets. In this paper, we evaluated Faster R-CNN [9], SSD-300, SSD-512 [41], YOLO [42] and YOLO-v2 [43].

Faster R-CNN is an improved version of Girshick *et al.*'s R-CNN [7] and Fast R-CNN [8] methods. Unlike R-CNN and Fast R-CNN, that use a selective search [44] to generate object bounding box proposals, Faster R-CNN has a region proposal network that can directly estimate proposals based on the CNN feature maps thus making it end-to-end trainable. Liu *et al.* [41] improved upon the Faster R-CNN model with their SSD (Single Shot MultiBox Detector) framework which generates object proposals with feature maps from multiple layers. As for YOLO (You Only Look Once) [42], it takes a 448×448 image as input and outputs object detections within a 7×7 grid. Later on, Redmon and Farhadi [43] integrated anchor boxes (for computing potential bounding boxes) into their YOLO model. We refer to this model as YOLO-v2.

[10] scikit-learn.org/

TABLE III

EVALUATION METRICS FOR RETRAINED CNN MODELS ON THE CLASSIFICATION DATASET IN FOUR DIFFERENT CONFIGURATIONS. 'N' STANDS FOR NORMAL SAMPLING, 'U' FOR UNIFORM SAMPLING, 'D' IS FOR USING DATA AUGMENTATION AND 'T' IS A TWO-PHASE TRAINING PROCEDURE

	Acc				mRe				mPr				$Kappa$			
	N	D+N	D+U	D+U(T)	N	D+N	D+U	D+U(T)	N	D+N	D+U	D+U(T)	N	D+N	D+U	D+U(T)
AlexNet	0.869	0.892	0.820	0.883	0.631	0.560	0.826	0.417	0.546	0.692	0.631	0.899	0.796	0.832	0.743	0.807
Inception-V3	0.947	0.962	0.929	0.959	0.809	0.828	0.888	0.872	0.792	0.848	0.763	0.889	0.918	0.941	0.892	0.937
ResNet-50	0.938	0.967	**0.959**	**0.969**	0.795	0.871	0.892	**0.877**	0.734	0.858	0.865	0.903	0.902	0.948	**0.936**	**0.952**
VGG-19	0.941	0.956	0.887	0.940	0.773	0.824	0.654	0.817	0.794	0.853	0.838	0.841	0.909	0.932	0.831	0.907
Xception	0.958	**0.976**	0.941	0.961	0.846	**0.908**	0.822	0.819	0.794	0.906	**0.900**	0.836	0.935	**0.963**	0.910	0.939
DenseNet	**0.970**	0.973	0.952	0.954	**0.889**	0.870	**0.896**	0.825	**0.886**	**0.916**	0.850	**0.908**	**0.953**	0.958	0.927	0.929

Similarly to the comparison of pre-trained CNN features and retrained CNN models for classification, we trained all localization models with and without updating the weights inherited from an ImageNet pre-trained model to observe the efficiency of ImageNet features on vehicle localization (i.e. we only trained the layers that weren't initialized with the ImageNet weights). We trained Faster R-CNN end-to-end for 300,000 iterations with code provided by the authors.[11] As recommended by the original paper, we used VGG-16 as a pre-trained model. Similarly, for SSD-300 and SSD-512 (i.e., SSD with input images resized to 300 × 300 and 512 × 512, respectively) we used the code released by the authors.[12] The SSD-300 model was trained for 120,000 iterations with a batch-size of 32, and the SSD-512 model was trained for 240,000 iterations with a batch-size of 16. YOLO was trained using the Darknet deep learning toolbox.[13] Both YOLO and YOLO-v2 were trained for 80,000 iterations with a batch-size of 64. As YOLO-v2 needs pre-clustered anchor bounding boxes, we evaluated two kinds of anchor boxes: boxes computed on the Pascal VOC datasets (referred to as YOLO-v2(P)) and on our localization dataset (referred to as YOLO-v2(M)).

2) MIO-TCD Localization: We report results from two localization methods designed specifically for the MIO-TCD dataset. The first one is from Wang *et al.* [45] which improves methods such as Faster R-CNN and SSD by leveraging the scene context. The idea is to combine the *softmax* score of these methods with a context term which the authors model with a *k*-NN algorithm. The second method is from Jung *et al.* [39] which combines the results computed from multiple R-FCN models [46] trained with different backbones (ResNet-50 and ResNet-101) together with a joint non-maximal suppression step to localize vehicles.

V. EXPERIMENTAL RESULTS

In this section, we report experimental results obtained on the MIO-TCD classification and localization datasets with the deep learning methods presented earlier.

A. Vehicle Classification

1) Pre-Trained CNN Features + SVM: Results obtained with SVM trained on features extracted from six pre-trained CNN models are shown in Table II. All six models have an accuracy of more than 80%, which is surprisingly high considering the different nature of ImageNet and MIO-TCD datasets. However, careful analysis reveals that these methods have relatively low mean recall, mean precision and Kappa score. To understand this situation, one has to consider the confusion matrix of ResNet-50 shown in Fig. 3. While the *Car* and *Background* categories have an accuracy of more than 90%, others, such as *Non-Motorized Vehicle*, *Motorcycle* and *Single-Unit Truck* categories, suffer from very low accuracy.

These numbers can be explained by the unbalanced nature of the dataset, as the *Car* and *Background* categories contain more than 80% of all images. In this case, large classes strongly influence the decision boundaries to the detriment of smaller classes. This also explains why several categories have a large proportion of samples wrongly classified as *Car*. It is the case for *Work Van* for which 30% of its images are classified as *Car*. A detailed analysis revealed that features trained on ImageNet do not discriminate family vans (labeled as *Car* in the dataset) from *Work Vans*. Confusion also happens between *Bicycle* and *Pedestrian*, as well as within the group of *Articulated Truck*, *Pickup Truck* and *Single-Unit Truck*.

Based on these results, we conclude that SVM trained on ImageNet CNN features is accurate at classifying large classes but this does not generalize well to smaller classes.

2) Retrained CNN Models: Table III reports evaluation metrics for the six models trained in four different training configurations. As can been seen, there is a substantial performance increase in comparison to the SVM results from Table II. Results show that data augmentation improves the Kappa score of every model, especially ResNet-50. Note that although data augmentation may reduce the mean recall of a certain method, this is compensated by a larger increase of the mean precision. However, the use of uniform sampling (+U) with and without the two-phase training procedure (T) does not improve results over the normal sampling (+N).

A careful inspection reveals that uniform sampling has a positive impact on small categories (such as *Motorcycle*, for example), but decreases the performance for large categories.

Overall, Xception and DenseNet with data augmentation and normal sampling are the best methods with very similar performance. VGG-19, ResNet-50 and Inception-V3 also get very accurate results with Kappa scores above 0.93.

3) MIO-TCD Classification (Ensemble Models): Table IV reports results submitted to the 2017 CVPR MIO-TCD Challenge. These are ensemble methods which combine the outputs of different models. As can be seen, all these methods have

[11] https://github.com/rbgirshick/py-faster-rcnn

[12] https://github.com/weiliu89/caffe/tree/ssd

[13] https://pjreddie.com/darknet/

TABLE IV

EVALUATION METRICS FOR ENSEMBLE MODELS FROM THE 2017 MIO-TCD CLASSIFICATION CHALLENGE

	Acc	mRe	mPr	$Kappa$
Kim and Lim [37]	0.9786	0.9041	0.9355	0.9666
Lee and chung [38]	0.9792	0.9024	0.9298	0.9675
Jung *et al* [39]	**0.9795**	0.8970	**0.9530**	**0.9681**
Theagarajan *et al* [40]	0.9780	**0.9190**	0.9439	0.9658

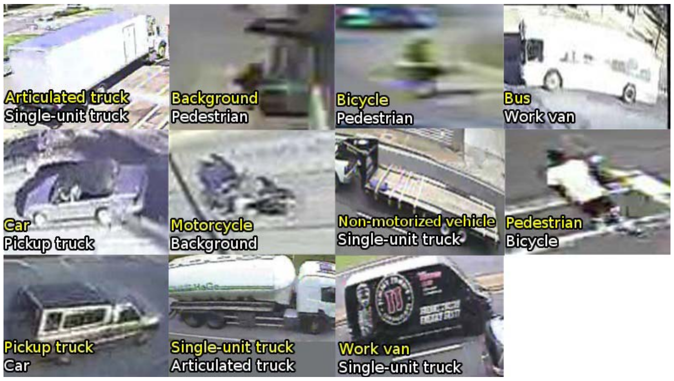


Fig. 4. Examples of failure cases from top-performing methods for every class where the yellow label (top) is the ground truth and the white label (bottom) is the predicted class.

similar performance with accuracy of about 0.98 and Kappa score of almost 0.97. With these results being only marginally better than those obtained with Xception and DenseNet, we conclude that the combination of several models does not bring much for a dataset such as MIO-TCD.

4) Error Analysis: Results from Tables III and IV reveal that despite large illumination variations between images, compression artifacts, arbitrary vehicle orientation, poor resolution and inter-class similarities, the classification of traffic vehicles seems almost solved. However, in-depth analysis of the top-performing methods reveals some unsolved issues. In Fig. 3, we show the confusion matrix for the method by Jung *et al.* [39] whose performance is globally similar to that obtained by other top performing methods. As one can see, *Non-Motorized Vehicles* are poorly handled. Images of *Non-Motorized Vehicles* in our dataset include a wide variety of trailers pulled by a vehicle, typically a car or a pickup truck. As shown in Fig. 4 (second row, third column), *Non-Motorized Vehicles* are often wrongly classified as a single-unit or an articulated truck, as their shapes are very similar.

Without much surprise, methods also get confused between categories with similar visual characteristics, such as the *Work Van* class and the *Car* class (more specifically, family vans considered to belong to the *Car* class) or the *Articulated Truck* and the *Single-Unit Truck*. They also get confused with vehicles that look unusual. For example, in Fig. 4, the blue car with a black top (second row, first column) gets wrongly classified as a pickup truck and the pickup truck with a cap (third row, fist column) is wrongly classified as a car. Also, classes with small objects such as *Pedestrian*, *Bicycle* and *Motorcycle* often suffer from heavy compression artifacts and are thus more likely to be mis-classified.

B. Vehicle Localization

1) Recent CNN-Based Methods: The average precision of localization for Faster R-CNN, SSD-300, SSD-512, YOLO-v1, YOLO-v2(P) and YOLO-v2(M) methods is shown in Table V.

Methods followed by “(w/o)” were trained without updating the weights inherited from a pre-trained ImageNet model. Note that when training Faster R-CNN with frozen layers, we found that it failed to converge. This behavior (also reported in [47]) is mainly due to gradient issues when using RoI pooling layers. For the other models, we see that training them in full is highly beneficial, with a very significant boost of mAP (24.1–42.5% improvement).

From this point, we only discuss results obtained with fully-trained models. As can be seen, the SSD methods outperform both Faster R-CNN and YOLO, with SSD-512 being the best-performing method with a mAP of 77.3%.

Results for SSD-300 and SSD-512 show that increasing input image resolution from 300 × 300 to 512 × 512 improves the mAP by 4%. Also, YOLO-v2 has the mAP 10% higher than YOLO-v1 thus showing that anchor boxes are useful features. Furthermore, results for YOLO-v2(P) and YOLO-v2(M) show that anchor boxes computed on our localization dataset marginally improve mAP over anchor boxes pre-computed from the Pascal VOC dataset.

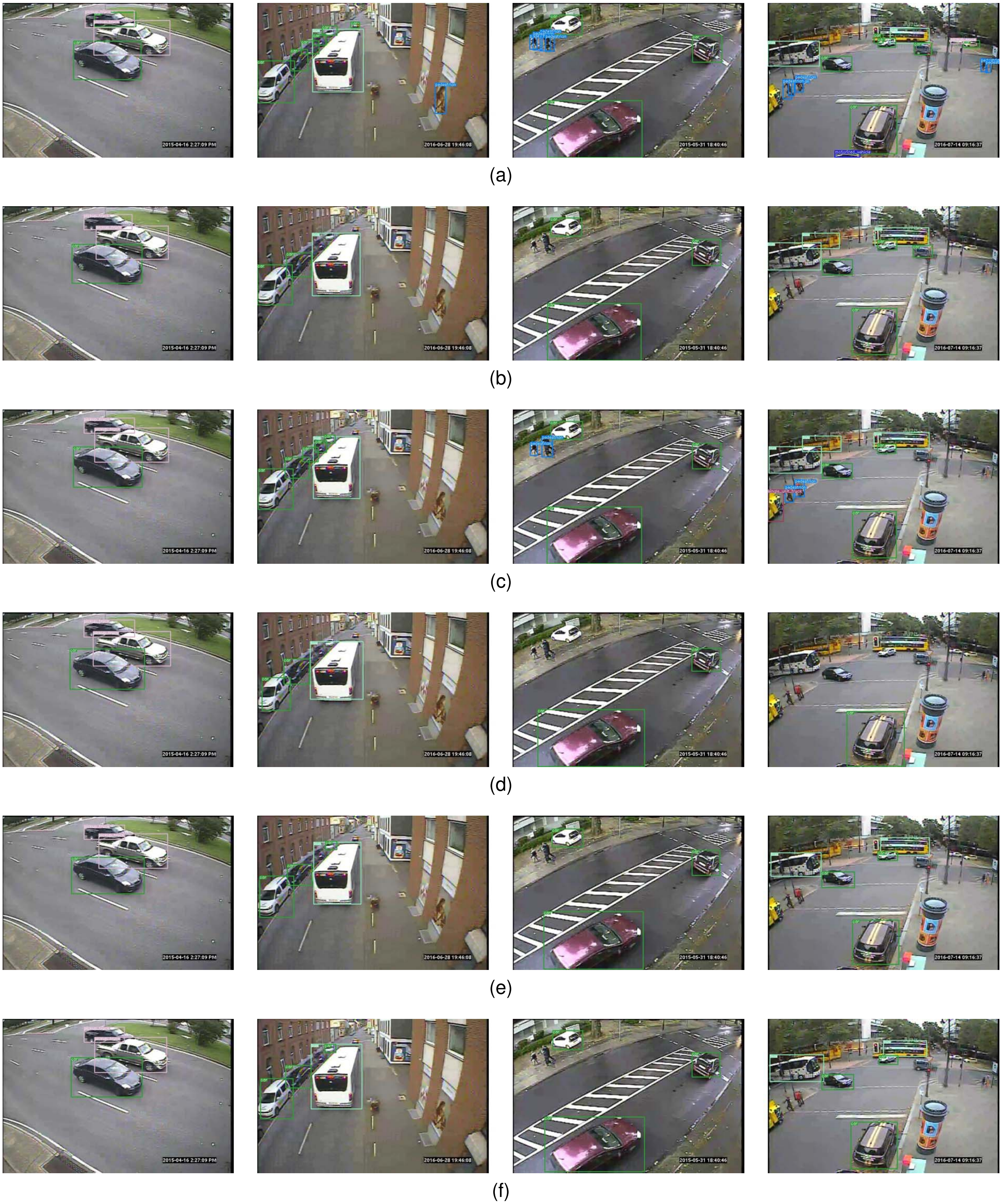
Here again, the largest classes, namely *Car*, *Pickup Truck*, *Articulated Truck* and *Bus* get the best results with an average precision above 80% for almost every method, while *Motorized Vehicle*, *Non-Motorized Vehicle* and *Pedestrian* are the three categories with the lowest average precision. The main challenge with *Motorized Vehicle* and *Pedestrian* classes stems from the small size of vehicles that are likely to be confused with the *Bicycle* and *Motorcycle* categories. As for the *Non-Motorized Vehicle*, similarly to the classification dataset, it is often confused with the *Articulated Truck* and *Single-Unit Truck* classes.

In Fig. 5, we show some detection results for different methods. While most methods can accurately localize large and well-contrasted vehicles, we can see that Faster R-CNN is prone to false detections while YOLO-v1 and YOLO-v2 suffer from mis-detections of small objects (typically, pedestrians).

2) MIO-TCD Localization: At the bottom of Table V, are shown localization metrics for methods by Wang *et al.* [45] and Jung *et al.* [39] submitted to the MIO-TCD Challenge. Both methods attain excellent performance, with the Jung *et al.* method achieving the best mAP of 79.2% and outperforming state-of-the-art CNNs for almost all vehicle classes.

3) Detailed Analysis: We now thoroughly analyze the influence of object scale on the performance of localization methods as well as the nature of false detections. We do so *via* the Microsoft COCO’s evaluation procedure [13] and the object detectors’ protocol by Hoeim *et al.* [48].

a) Scale: Every object has been classified as belonging to one of 3 scales: small objects with bounding box area below 32^2, medium objects with the area between 32^2 and 96^2, and large objects with the area larger than 96^2. The average precision for each of these scales is reported in Table. VI.



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 5. Detection examples on the localization dataset for Faster R-CNN, SSD-300, SSD-512, YOLO, YOLO-v2 (Pascal VOC) and YOLO-v2 (MIO-TCD). We only show detections with probability scores higher than 0.6. (a) Faster R-CNN. (b) SSD-300. (c) SSD-512. (d) YOLO-v1. (e) YOLO-v2 (Pascal VOC). (f) YOLO-v2 (MIO-TCD).

As can be seen, all methods in the table are ill-suited for detecting small objects, in our case *Pedestrian*, *Bicycle*, *Motorcycle* classes as well as vehicles seen from a distance (see Fig. 2 for examples of small vehicles due to perspective). Furthermore, when increasing the overlap ratio for correct detections from 0.5 to 0.75, we find that the AP of Faster

TABLE V

AVERAGE PRECISION (AP) OF LOCALIZATION FOR FASTER R-CNN, SSD, YOLO AND TWO METHODS SUBMITTED TO THE MIO-TCD CHALLENGE ON LOCALIZATION. ("W/O": WITHOUT UPDATING THE WEIGHTS INHERITED FROM AN IMAGENET PRE-TRAINED MODEL)

	mAP	Articulated Truck	Bicycle	Bus	Car	Motorcycle	Motorized Vehicle	Non-Motorized Vehicle	Pedestrian	Pickup Truck	Single-Unit Truck	Work Van
Faster R-CNN (w/o)	6.3	4.5	2.0	3.0	5.5	3.7	6.6	0.9	0.4	10.0	2.4	2.6
SSD-300 (w/o)	36.4	44.1	52.2	68.0	55.0	38.5	29.4	3.2	10.7	55.6	19.3	24.7
SSD-512 (w/o)	34.8	45.2	25.6	71.0	61.9	30.0	37.0	0.8	12.2	61.0	15.8	22.0
YOLO-v1 (w/o)	31.3	48.2	19.2	78.5	50.6	15.4	17.3	6.3	2.3	61.8	12.7	31.9
YOLO-v2(P) (w/o)	46.3	54.6	53.5	82.8	61.4	56.8	26.8	20.5	9.9	68.8	30.2	43.5
YOLO-v2(M) (w/o)	47.7	60.9	54.2	85.9	62.0	60.7	27.1	19.2	8.8	69.9	32.3	43.7
Faster R-CNN	70.0	85.9	78.4	95.2	82.6	81.1	52.8	37.3	31.3	89.0	62.5	73.6
SSD-300	74.0	90.6	78.3	95.7	91.5	78.9	51.4	55.2	37.3	90.7	69.0	75.0
SSD-512	**77.3**	**92.1**	**78.6**	**96.8**	**94.0**	**82.3**	**56.8**	**58.8**	**43.6**	**93.1**	**74.0**	**80.4**
YOLO-v1	62.7	82.7	70.0	91.6	77.2	71.4	44.4	20.7	18.1	85.6	58.3	69.3
YOLO-v2(P)	71.5	86.7	78.4	95.2	80.5	80.9	52.0	56.5	25.7	84.6	70.0	75.7
YOLO-v2(M)	71.8	88.3	**78.6**	95.1	81.4	81.4	51.7	56.6	25.0	86.5	69.2	76.4
Wang *et al.*[45]	77.2	91.6	79.9	96.8	**93.8**	83.6	56.4	58.2	42.6	**92.8**	73.8	79.6
Jung *et al.*[39]	**79.2**	**92.5**	**87.3**	**97.5**	89.7	**88.2**	**62.3**	**59.1**	**48.6**	92.3	**74.4**	**79.9**

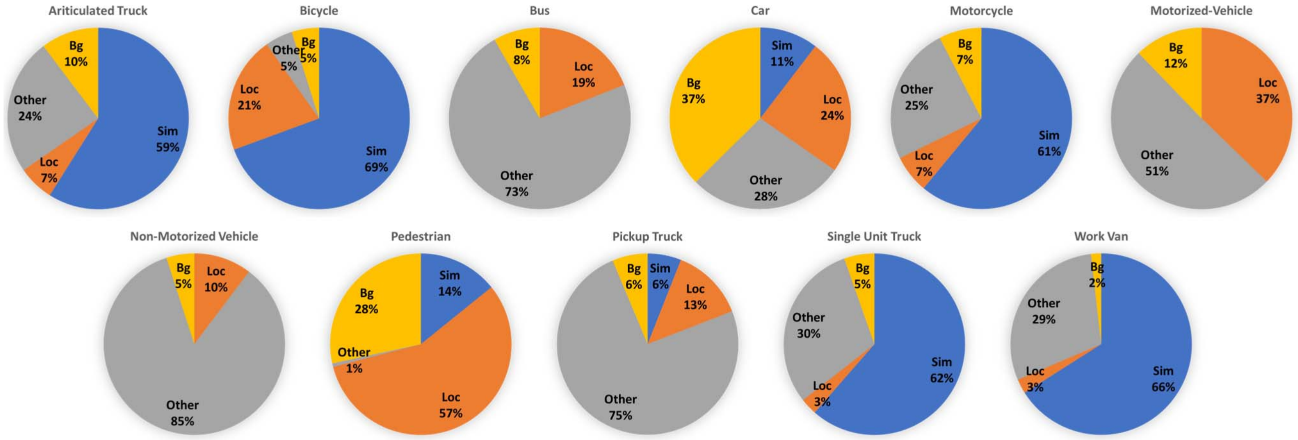


Fig. 6. Analysis of false detections by the SSD-300 method. Each pie-chart shows the fraction of top-ranked false positives of each category due to poor localization (Loc), confusion with similar categories (Sim), confusion with other categories (Other), or confusion with background or unlabeled objects (Bg).

TABLE VI

AVERAGE PRECISION OF LOCALIZATION COMPUTED USING MICROSOFT COCO'S EVALUATION PROTOCOL

	Average Precision				
	Overlap		Scale		
	0.5	0.75	small	medium	large
Faster R-CNN	70.0	38.5	14.3	40.2	55.1
SSD-300	74.3	57.1	21.5	53.3	69.0
SSD-512	**77.6**	**61.9**	**28.2**	**57.3**	**72.5**
YOLO-v1	62.6	34.0	11.3	32.4	52.7
YOLO-v2(P)	71.3	42.7	15.7	41.3	59.3
YOLO-v2(M)	71.8	43.0	16.0	41.7	61.3
Wang *et al.* [45]	77.4	**59.9**	**26.6**	**55.6**	60.7
Jung *et al.* [39]	**79.3**	58.8	26.5	54.9	**69.3**

R-CNN and YOLO decreases by almost 30%, while for SSD it decreases by around 15%. This means that the bounding boxes estimated by the SSD method are tighter around the ground-truth bounding boxes.

b) False positives: To examine the nature of false positives, we follow the methodology of Hoiem *et al.* [48] according to which each prediction is either correct or wrongly classified into one of the following errors:

- **Localization**: the predicted bounding box has a correct label but is misaligned with the ground-truth bounding box (0.1 < Overlap < 0.5).
- **Similarity**: the predicted bounding box has Overlap > 0.1 with a ground-truth bounding box but its predicted label is incorrect. However, the predicted label belongs to one of three similarity classes (groups of similar classes), namely: {*Articulated Truck, Pickup Truck, Single-Unit Truck*}, {*Bicycle, Motorcycle, Pedestrian*}, and {*Car, Work Van*}.
- **Other**: the predicted bounding box has a Overlap > 0.1 with a ground-truth bounding box but its predicted label is incorrect and does not fall within a group of similar classes.
- **Background**: all other false positives are classified as background, mainly confused with unlabeled objects.

Fig. 6 shows the frequency of occurrence of each type of error for the SSD-300 method. For the similarity class {*Articulated Truck, Pickup Truck, Single-Unit Truck*}, the *Articulated Truck* and *Single-Unit Truck* classes are likely to be confused with each other, while the *Pickup Truck* is confused with other classes, mostly *Car*. As for the {*Car, Work Van*} similarity class, we find that 66% of *Work Van* false positives are wrongly classified as *Car* for the reason mentioned before (*Work Van* is often confused with a family van). As for *Car* false detections, the confusion is mostly with *Background*. For the {*Bicycle, Motorcycle, Pedestrian*} similarity class, the *Bicycle* and *Motorcycle* classes are often confused with each other, while *Pedestrian*, due to small scale, suffers from localization errors. As for the *Bus*, *Motorized Vehicle* and *Non-Motorized Vehicle* classes, they all suffer from confusion with other categories.

VI. Discussion and Conclusions

In this paper, we introduced the *MIOvision Traffic Camera Dataset (MIO-TCD)*, the largest dataset to date for motorized traffic analysis. The dataset consists of two parts: a "localization dataset", containing full video frames with bounding boxes around traffic objects, and a "classification dataset", containing crops of 11 types of traffic objects.

We evaluated several state-of-the-art deep learning methods on the MIO-TCD dataset. Results show that well-trained models reach impressive accuracy and Kappa scores of more than 96% on the classification dataset and a mean-average precision of 79% on the localization dataset.

While Xception and DenseNet with data augmentation are the best classification methods, VGG-19, RestNet-50 and Inception-V3 attain very good results as well. As for the ensemble models, they reach, for all practical purposes, the same scores as Xception and DenseNet. A careful inspection of results reveals that *Non-Motorized Vehicles* is the only problematic class with a precision below 80%. Other errors in the top results can be explained by the confusion between classes with similar visual characteristics such as *Single-Unit Truck* and *Articulated Truck*.

As for localization methods, the method by Jung *et al.* [39] gets the best scores (mAP=79.2%) but is closely followed by SSD-512 (mAP=77.3%). A detailed analysis reveals that errors often happen between similar classes or are due to a misalignment of the predicted bounding box (overlapping ratio below 0.5).

In light of these results, we may conclude that state-of-the-art deep learning methods exhibit a capacity to localize and recognize vehicles from single video frames without the need for dynamic features captured by video, as was required to date. This opens the door to new, low-frame-rate video analytics applications such as traffic statistics, traffic density estimation, car counting, and anomaly detection.

Although deep models achieve very promising performance, a number of challenges remain, among them: similarly-looking vehicles, unbalanced data, false detections and small vehicles. We conclude the paper by discussing these challenges below.

1) *Similarly-looking vehicles:* To differentiate vehicles with similar appearance, such as bicycles and motorcycles, it may be necessary to identify semantic features in addition to global features.
2) *Unbalanced data:* Experiments show that ensemble models can deal with this issue to some extent, at the cost of significantly more computations. It would be beneficial to develop algorithms that leverage the benefits of ensemble models at reduced computational load.
3) *False detections:* The appearance and position of different vehicles is highly correlated with scene layout, such as road direction. It would be preferable to embed the layout information into the localization model as a means of enhancing detection and reducing false positives.
4) *Small vehicles:* The localization results indicate that large vehicles are more easily detected than small vehicles. Adversarial training or metric learning methods projecting vehicles of various sizes into the same feature space could be an avenue towards improving the localization and classification accuracy of small vehicles.

References

[1] Z. Luo *et al.* (2017). *Traffic Surveillance Workshop and Challenge at CVPR-2017*. [Online]. Available: http://tcd.miovision.com

[2] B.-F. Wu, C.-C. Kao, J.-H. Juang, and Y.-S. Huang, "A new approach to video-based traffic surveillance using fuzzy hybrid information inference mechanism," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 485–491, Mar. 2013.

[3] T. Zhang, S. Liu, C. Xu, and H. Lu, "Mining semantic context information for intelligent video surveillance of traffic scenes," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 149–160, Feb. 2013.

[4] H. Liu, S. Chen, and N. Kubota, "Intelligent video systems and analytics: A survey," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1222–1233, Aug. 2013.

[5] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," *Image Vis. Comput.*, vol. 21, no. 4, pp. 359–381, 2003.

[6] W. Luo *et al.* (2014). "Multiple object tracking: A literature review." [Online]. Available: https://arxiv.org/abs/1409.7618

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, Jun. 2014, pp. 580–587.

[8] R. Girshick, "Fast R-CNN," in *Proc. ICCV*, Apr. 2015, pp. 1440–1448.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, 2015, pp. 91–99.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Jun. 2016, pp. 770–778.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[12] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[13] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.

[14] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, Jun. 2005, pp. 886–893.

[16] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. CVPR*, Jun. 2012, pp. 3354–3361.

[17] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. CVPR*, Jun. 2016, pp. 3213–3223.

[18] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proc. CVPR*, Jun. 2016, pp. 2110–2118.

[19] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. CVPRW*, Jun. 2013, pp. 554–561.

[20] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *Proc. CVPR*, Jun. 2015, pp. 3973–3981.

[21] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2247–2256, Aug. 2015.

[22] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Oñoro-Rubio, "Extremely overlapping vehicle counting," in *Proc. ICPRIA*, 2015, pp. 423–431.

[23] R. Guerrero-Gómez-Olmedo, R. J. López-Sastre, S. Maldonado-Bascón, and A. Fernández-Caballero, "Vehicle tracking by simultaneous detection and viewpoint estimation," in *Proc. WCIBNAC*, 2013, pp. 306–316.

[24] J. Cohen, "A coefficient of agreement for nominal scales," *Educ. Psychol. Meas.*, vol. 20, no. 1, pp. 37–46, 1960.

[25] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. CVPRW*, Jun. 2014, pp. 512–519.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. CVPR*, Jun. 2016, pp. 2818–2826.

[28] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[29] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. CVPR*, Jun. 2017, pp. 1800–1807.

[30] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proc. CVPR*, Jun. 2017, pp. 4700–4708.

[31] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. NIPS*, 2014, pp. 3320–3328.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–15.

[33] S. Ioffe and C. Szegedy. (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167

[34] F. Chollet *et al.* (2015). *Keras*. [Online]. Available: https://github.com/fchollet/keras

[35] A. Paszke *et al.* (2016). *Pytorch*. [Online]. Available: https://github.com/pytorch/pytorch

[36] M. Havaei *et al.*, "Brain tumor segmentation with deep neural networks," *Med. Image Anal.*, vol. 35, pp. 18–31, Jan. 2017.

[37] P.-K. Kim and K.-T. Lim, "Vehicle type classification using bagging and convolutional neural network on multi view surveillance image," in *Proc. CVPRW*, Jul. 2017, pp. 914–919.

[38] J. T. Lee and Y. Chung, "Deep learning-based vehicle classification using an ensemble of local expert and global networks," in *Proc. CVPRW*, Jul. 2017, pp. 920–925.

[39] H. Jung, M.-K. Choi, J. Jung, J.-H. Lee, S. Kwon, and W. Y. Jung, "ResNet-based vehicle classification and localization in traffic surveillance systems," in *Proc. CVPRW*, Jul. 2017, pp. 934–940.

[40] R. Theagarajan, F. Pala, and B. Bhanu, "EDeN: Ensemble of deep networks for vehicle classification," in *Proc. CVPRW*, Jul. 2017, pp. 906–913.

[41] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.

[42] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, Jun. 2016, pp. 779–788.

[43] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. CVPR*, Jun. 2017, pp. 7263–7271.

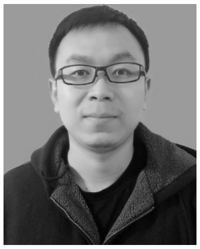
[44] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[45] T. Wang, X. He, S. Su, and Y. Guan, "Efficient scene layout aware object detection for traffic surveillance," in *Proc. CVPRW*, Jul. 2017, pp. 926–933.

[46] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. NIPS*, 2016, pp. 379–387.

[47] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "DSOD: Learning deeply supervised object detectors from scratch," in *Proc. ICCV*, Oct. 2017, pp. 1919–1927.

[48] D. Hoiem, Y. Chodpathumwan, and Q. Dai, "Diagnosing error in object detectors," in *Proc. ECCV*, 2012, pp. 340–353.



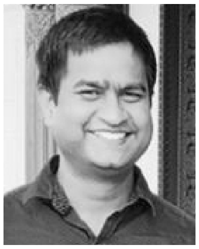
Zhiming Luo received the Ph.D. degree in computer science from Xiamen University, Xiamen, China, and the University of Sherbrooke, Sherbrooke, QC, Canada. He is currently a Post-Doctoral Researcher at Xiamen University. His current research interests include traffic surveillance video analytics, computer vision, and machine learning.

Frédéric Branchaud-Charron is currently pursuing the master's degree with the University of Sherbrooke, Sherbrooke, QC, Canada. His current research interests include complexity measures, object localization, computer vision, and machine learning.

Carl Lemaire is currently pursuing the master's degree in computer science at the University of Sherbrooke, Sherbrooke, QC, Canada. His current research interests include object classification and localization in images, architecture search and pruning of neural networks and variational Bayesian methods.

Janusz Konrad (M'93–SM'98–F'08) received the Ph.D. degree from McGill University, Montreal, QC, Canada. He is currently a Professor at Boston University, Boston, MA, USA. His current research interests include image and video processing, stereoscopic and 3-D imaging and displays, visual sensor networks, and human-computer interfaces. He was a member-at-large of the IEEE Signal Processing Society Conference Board. He was a co-recipient of the 2001 Signal Processing Magazine Award, from 2004 to 2005 and the EURASIP Image Communications Best Paper Award and AVSS-2010 Best Paper Award. He was the Technical Program Co-Chair of the ICIP-2000 and AVSS-2010 and the General Chair of the AVSS-2013. He is a Senior Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and an Area Editor of the *EURASIP Signal Processing: Image Communications* journal.

Shaozi Li (M'18) received the B.S. degree from Hunan University, Changsha, China, and the M.S. degree from Xi'an Jiaotong University, Xi'an, China, and the Ph.D. degree from the National University of Defense Technology, Changsha. He currently serves as the Chair and a Professor of the Cognitive Science Department, Xiamen University, Xiamen, China, the Vice Director of the Technical Committee on Collaborative Computing of China Computer Federation (CCF), and the Vice Director of the Fujian Association of Artificial Intelligence. His research interests include artificial intelligence and its applications, moving objects detection and recognition, machine learning, computer vision, and multimedia information retrieval. He has directed and completed over twenty research projects, including several National 863 Programs, the National Nature Science Foundation of China, and the Ph.D. Programs Foundation of Ministry of Education of China. He is a Senior Member of the ACM and CCF.



Akshaya Mishra received the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, and the M. Tech and B.E. degrees in electrical engineering from the Indian Institute of Technology. He was a Principal Research Scientist at Miovision Technologies. He is currently a Senior Research Scientist for Epson Canada and holds an adjunct faculty position at the University of Waterloo. His research interests include solving industrial problems using deep learning, statistical modeling, image processing, and pattern recognition methods.

Andrew Achkar received the Ph.D. degree from the University of Waterloo, Canada. He is currently a Senior Data Research Scientist at Miovision Technologies, involved in areas, including computer vision, machine learning, data pipelining, distributed systems, and software development.

Justin Eichel received the Ph.D. degree from the Vision and Image Processing Laboratory, Systems Design Engineering, University of Waterloo. He is the Technical Director at Miovision and an Adjunct Professor at the University of Waterloo and has previous experience as a Self-Employed Consultant related to multi-spectrum tracking and medical image processing research. He is skilled at statistical modeling, pattern recognition, and machine learning.

Pierre-Marc Jodoin received the Ph.D. degree from the Computer Science Department, University of Montreal, Canada, in 2007. He is currently a Full Professor at the Computer Science Department, University of Sherbrooke, Canada. He is the Co-Founder of Imeka Inc., served as the Director of the Research Center for Intelligent Environments from 2012 to 2016, he was the Co-Director of the Sherbrooke Image Processing and Visualization Service from 2011 to 2018. His research interest includes machine learning, video analytics and medical image analysis. He was a Program Committee Member of MICCAI 2017. He has chaired three CVPR and one ICPR Workshops. He was an Associated Editor of the IEEE TIP from 2013 to 2017 and a Guest Editor of the *Pattern Recognition* from 2013 to 2015 and the *Signal Processing* from 2013 to 2015.